REVIEW ARTICLE

Requirements Elicitation in Transition: A Review of Conventional and Contemporary Approaches

Asma Akhtar[1,*] **and Samia Akhtar**[1]

[1] Department of Computer Science, Virtual University of Pakistan, Lahore 54000, Pakistan

Abstract

Requirements elicitation is one of the most important steps in the software development process. It involves understanding what users and stakeholders need from a system before it is built. Traditionally, this has been done using methods like interviews, questionnaires, document reviews, and direct observation. These approaches work well in structured environments but often fall short when dealing with large, fast-changing, or agile projects. In recent years, software development has shifted toward more flexible and fast-paced practices. This change has also affected how requirements are gathered. New techniques now include collaborative tools, user feedback from online platforms, and the use of artificial intelligence (AI) and natural language processing (NLP) to extract requirements from text automatically. This paper presents a comparative narrative review based on recent literature and practical insights. It presents both traditional and modern requirements elicitation methods, comparing them in terms of how they work, where they are most useful, and what challenges they present. A detailed comparison highlights key differences such as level of user interaction, tool support, scalability, and suitability for ongoing development cycles. By reviewing recent research and real-world practices, this paper identifies current trends, challenges, and open areas for future work. The goal is to help researchers, software engineers, and project teams choose the most suitable elicitation methods based on their specific project needs. In the end, this review supports the idea that a flexible, hybrid approach—blending old and new techniques—may be the most effective way forward in today's evolving software engineering landscape.

Keywords: requirements elicitation, software engineering, agile development, artificial intelligence, stakeholder analysis, requirements extraction.

1 Introduction

Requirements elicitation lies at the heart of successful software development [1]. It is the process through which software engineers, analysts, and stakeholders collaboratively discover, understand, and define what a software system should do. The ultimate goal is to ensure that the final system meets user expectations, business goals, and technical constraints [2]. As a core phase of requirements engineering, elicitation impacts every subsequent stage of the development

Academic Editor:
Summair Raza

Submitted: 13 July 2025
Accepted: 29 July 2025
Published: 18 August 2025

Vol. 1, **No.** 1, 2025.
10.62762/JSE.2025.862549

***Corresponding author:**
✉ Asma Akhtar
asmaakhtarjanjua@gmail.com

Citation
Akhtar, A., & Akhtar, S. (2025). Requirements Elicitation in Transition: A Review of Conventional and Contemporary Approaches. *ICCK Journal of Software Engineering*, 1(1), 32–45.

© 2025 by the Authors. Published by Institute of Central Computation and Knowledge. This is an open access article under the CC BY license (https://creativecommons.org/licenses/by/4.0/).

lifecycle — from design and implementation to testing and deployment. Errors or omissions made during this stage often propagate, leading to costly revisions, project delays, or even complete system failure.

1.1 Background and Historical Context

In the early days of software development, projects typically followed rigid, linear development models such as the Waterfall methodology. These projects were often executed in well-defined environments with limited user interaction during development [3]. Accordingly, requirements elicitation was treated as a one-time, upfront activity, completed during the initial phases of the project. Analysts conducted structured interviews, distributed questionnaires, and analyzed organizational documents to collect requirements from stakeholders. Use case diagrams, requirement specification documents, and functional models were created and often considered "final" once approved. The assumption was that user needs could be fully captured before any design or development began. These traditional methods were sufficient in stable and predictable domains, such as military or manufacturing software, where requirements remained static for long periods. The emphasis was on documentation and formal validation, with minimal flexibility to adapt to changes. Tools were limited, face-to-face communication was prioritized, and requirements gathering was viewed as more of a procedural task than an ongoing conversation.

1.2 Evolution of Development Practices and Its Impact

However, as software systems grew in scale, complexity, and interactivity, this one-time approach began to show its limitations [4]. The rise of Agile methodologies, DevOps culture, cloud-based deployments, and user-centered design dramatically changed the pace and structure of software engineering. In this new landscape, requirements were no longer static [5]. Stakeholders expected rapid updates, personalized features, and continuous delivery — and development teams needed elicitation methods that could adapt accordingly [6]. Today, requirements elicitation is seen as a continuous and iterative activity, woven into each development sprint or product release cycle [7]. Teams no longer assume that all requirements can be fully known in advance. Instead, they actively seek feedback during development, embracing change and evolution as constants. This shift has led to the adoption of modern techniques such as:

- **Collaborative modeling tools** like Jira, Confluence, or Trello, which allow teams to continuously gather and update requirements in real time
- **Crowdsourcing and online user feedback** via app store reviews, support tickets, and social media monitoring
- **Natural language processing (NLP)** and **machine learning** algorithms that analyze user feedback and generate requirement suggestions
- **Prototyping and wireframing tools** that encourage early and frequent stakeholder input
- **Agile ceremonies** such as sprint planning, daily stand-ups, and retrospectives, which keep requirements discussions active

These contemporary methods are designed to be lightweight, scalable, and collaborative, addressing the weaknesses of traditional elicitation approaches in dynamic environments [8].

1.3 The Elicitation Shift: Then vs. Now

The transition from conventional to contemporary elicitation methods marks a significant transformation in software engineering culture. Table 1 below summarizes some of the key differences:

While traditional methods emphasized formal structure, traceability, and control, modern techniques prioritize agility, speed, and inclusivity [9]. However, this shift also introduces new challenges: over-reliance on tools, difficulty managing unstructured input, and potential loss of deep stakeholder insights [10]. Figure 1 shows the evolution of requirement elicitation techniques over the years.

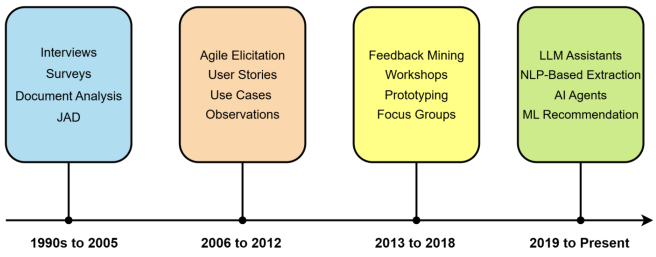


Figure 1. Timeline illustrating the evolution of requirements elicitation techniques from traditional methods to modern AI-driven approaches.

1.4 Motivation for This Review

Given the rapid changes in software engineering practices and the growing variety of elicitation

Table 1. General differences between conventional and contemporary elicitation approaches.

Aspect	Traditional Approaches	Modern Approaches
Process Timing	One-time, early-stage activity	Continuous and iterative
Stakeholder Involvement	Limited to initial meetings	Ongoing engagement
Techniques Used	Interviews, surveys, documents	Crowdsourcing, AI tools, feedback mining
Tool Support	Minimal or document-based	Collaborative platforms and automated tools
Suitability	Stable, large-scale projects	Agile, fast-paced, evolving systems
Focus	Documentation and completeness	Responsiveness and adaptability

techniques, it becomes essential to evaluate the current landscape systematically [11]. Many teams today operate in hybrid environments, mixing aspects of Agile, DevOps, and traditional models [12]. As a result, no single elicitation method fits all contexts, and understanding the trade-offs between approaches is crucial. Despite the increasing use of modern tools and AI-based methods, many organizations still rely on conventional elicitation techniques, especially in regulated industries like healthcare, finance, and aerospace [13–15]. This coexistence of old and new methods calls for a comprehensive, comparative study to guide researchers, educators, and practitioners in making informed decisions. This paper is designed to fill that gap. It reviews and contrasts both traditional and modern requirements elicitation techniques based on existing academic literature, case studies, and practical implementations. It provides a comparative framework based on key factors such as:

- Stakeholder involvement
- Degree of automation
- Tool dependence
- Adaptability to change
- Accuracy and completeness
- Suitability for Agile, DevOps, or hybrid environments

By analyzing the strengths and weaknesses of each approach, this paper aims to answer a fundamental question: *How should modern software teams elicit requirements in a world of constant change?*

1.5 Objectives and Scope

The primary objectives of this paper are as follows:

- To classify and explain key traditional and modern requirements elicitation techniques.
- To compare these techniques across various criteria relevant to today's development workflows.
- To highlight the benefits and challenges associated with each method.
- To identify gaps in current research and practice.
- To propose areas for future exploration, particularly in the use of AI, hybrid models, and tool integration.

While the paper does not propose a new elicitation methodology, it provides a strong foundation for further innovation by synthesizing what has been done, what is being done, and what could be done better.

2 Related Work

The field of requirements elicitation has changed significantly in recent years. Traditional practices have now been enhanced—or in some cases, replaced—by newer methods powered by artificial intelligence, machine learning, and data-driven tools. This section reviews the existing literature in two main areas: (1) classical approaches that formed the foundation of elicitation, and (2) recent research that highlights emerging trends. The section ends with a discussion of the research gap and how this paper contributes to the field.

2.1 Classical Foundations

Requirements elicitation has always been a key part of the software development process. Traditional techniques include interviews, questionnaires, observations, document reviews, and Joint Application Development (JAD) sessions. These methods were commonly used in structured models like the Waterfall and V-Model, where all requirements were gathered early in the project.

[16] proposed a structured approach in *Software Requirements: Objects, Functions and States*, emphasizing formal specification and traceability. He classified

requirements into object, function, and state views to improve clarity and analysis.

[17] introduced a well-structured and planned approach to elicitation in their book *Requirements Engineering: A Good Practice Guide*. They emphasized systematic planning, stakeholder identification, and the use of traditional techniques such as structured interviews, questionnaires, and document analysis.

Building on this foundation, [18] offered a detailed classification of elicitation techniques into two major categories: stakeholder-driven techniques—including interviews, focus groups, and workshops—and artifact-driven techniques such as document analysis, scenarios, and prototyping.

With the rise of Agile methodologies, [19] emphasized the shift away from heavy upfront documentation toward continuous, lightweight elicitation. He advocated for the use of user stories, maintained by product owners, as a means of capturing requirements in iterative cycles.

[20] further explored how Agile teams adopt iterative elicitation strategies. They highlighted techniques such as on-site customer collaboration, daily stand-ups, and incremental feedback loops that prioritize informal communication and evolving requirements.

While these classical approaches are still widely used—particularly in government, healthcare, and safety-critical projects—they face challenges in fast-moving, large-scale, or highly automated development environments.

2.2 Recent Literature

The past five years have brought a surge in AI, natural language processing (NLP), and machine learning (ML) methods that support or automate parts of the elicitation process. These technologies improve scalability, reduce manual effort, and make it possible to handle complex stakeholder needs and large data sources.

2.2.1 AI and NLP-Based Elicitation

[21] reviewed a number of studies to assess how NLP techniques are used in requirements engineering. They found a shift from older rule-based models to transformer-based models such as BERT, which offer better performance in classifying and extracting requirements from text.

[22] conducted a systematic review focused on automating requirement formalization through NLP and ML. They noted that while traditional heuristics are still in use, there is growing interest in deep learning due to its ability to handle contextual and ambiguous language.

[23] proposed RECOVER, a transformer-based tool that generates requirements directly from stakeholder conversations. The tool showed strong results in completeness and clarity compared to manual methods.

[24] introduced Elicitron, a simulation framework powered by large language models (LLMs). This system mimics stakeholder interviews using AI agents and consistently generated richer and more diverse requirements than human-led sessions.

2.2.2 ML and Recommendation Systems

[25] reviewed 92 papers to explore how machine learning supports requirements elicitation. They found ML useful for requirement classification, clustering, and recommendation. Their work proposed a framework linking ML methods to specific elicitation tasks.

[26] investigated how applying SMOTE (Synthetic Minority Over-sampling Technique) and feature-selection strategies can enhance the accuracy of ML models used for selecting requirements elicitation techniques. Their experiments demonstrated that balancing imbalanced datasets and choosing key input features significantly improved model performance, reducing overfitting risks.

[27] conducted a systematic literature review on recommendation systems in software requirements elicitation. They highlighted how collaborative and content-based filtering approaches assist in stakeholder identification, prioritization, and technique selection.

2.2.3 Feedback Mining and Automation

[28] conducted an empirical study evaluating app review mining tools for requirements engineering, focusing on feature-specific opinion mining and feature-based search. They found that existing tools often underperformed compared to earlier claims, revealing a significant gap between reported and real-world effectiveness.

[29] present a comprehensive framework for automated processing of user feedback in requirements engineering. They detail techniques—from extraction and clustering to summarization and sentiment analysis—using ML,

NLP, and LLMs to handle both volume and quality of feedback. Their work also highlights pipelines and benchmark resources to guide practitioners in implementing scalable, trustworthy feedback-mining solutions.

2.3 Research Gap and Contribution of this Paper

Although several papers have reviewed traditional techniques and others have focused on modern, AI-based methods, few studies offer a comparative review that brings both together. Similarly, foundational papers mainly focus on legacy techniques without mentioning newer innovations. Another gap lies in the lack of evaluation across diverse software contexts, such as Agile, DevOps, hybrid, or safety-critical systems. It is unclear how practitioners can mix and match techniques to meet the specific demands of each environment.

This paper addresses these gaps by:

- Providing a comparative analysis of traditional and contemporary elicitation techniques;
- Classifying techniques based on interaction style, tool support, adaptability, and stakeholder involvement;
- Highlighting advances such as large language models, NLP-based extraction, and machine learning–assisted recommendation systems;
- Proposing a flexible framework to guide the selection of elicitation methods according to context, complexity, and project goals.

By offering this balanced perspective, the paper helps both researchers and practitioners understand the transition in elicitation approaches and choose the best combination of techniques for modern software development.

3 Comparative Overview of Requirements Elicitation Techniques

This section presents a clear comparison between traditional and modern requirements elicitation techniques. We conducted a narrative review using IEEE Xplore, ACM Digital Library, and Google Scholar. Selection was based on relevance, citation impact, and practical significance. Keywords included "requirements elicitation," "traditional methods," and "modern techniques." Classification dimensions were adapted from prior studies and refined through thematic analysis. Techniques were chosen for their prevalence, relevance, and diversity across development settings. Traditional methods focus on human involvement, face-to-face interaction, and manual documentation. In contrast, modern approaches use artificial intelligence, automation, and data-driven tools to improve speed, scalability, and coverage. Figure 2 provides an overview of the techniques discussed in this study.

3.1 Traditional Techniques

Traditional elicitation methods have been the backbone of software engineering for decades [30]. These approaches rely on direct stakeholder input, structured processes, and manual analysis. While still widely used, they often face challenges in large-scale or fast-paced environments. A few important techniques are discussed below and Table 2 shows advantages and disadvantages of each.

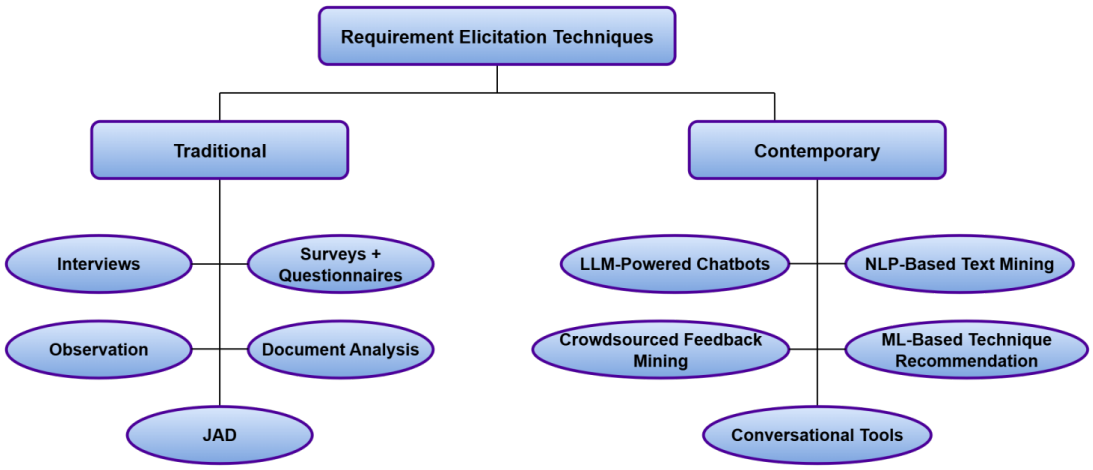


Figure 2. Flowchart categorizing traditional and contemporary requirements elicitation techniques.

3.1.1 Interviews

Interviews are one of the most widely used techniques for gathering requirements [31]. Analysts meet with stakeholders individually or in small groups to ask open-ended or structured questions. Interviews help explore business goals, user needs, and constraints through direct communication. They allow clarification of ambiguities and discovery of undocumented insights. Interviews are especially useful during early project phases or when working with domain experts [32]. However, the quality of results depends on the interviewer's skill, and the process can be time-consuming. Scheduling interviews and interpreting subjective responses can be challenging, especially in large or distributed teams.

3.1.2 Surveys and Questionnaires

Surveys are used to collect structured feedback from a wide range of users [33]. They consist of closed- or open-ended questions, distributed online or in print. Surveys allow stakeholders to respond at their convenience, making them ideal for remote teams and large user bases. They are cost-effective, provide quick statistical summaries, and support early requirement validation. However, they lack the depth of face-to-face methods, and poor question design can result in misleading or incomplete responses. Surveys are best suited for confirming known requirements rather than exploring new ones, and they often fail to capture nuanced user expectations or behavior [34].

3.1.3 Document Analysis

Document analysis involves examining existing sources such as business reports, policy manuals, technical specifications, and project documentation. This technique helps analysts understand past decisions, stakeholder expectations, and legacy systems [35]. It is particularly valuable when stakeholders are unavailable or when historical context is needed. Document analysis requires minimal stakeholder interaction and can be performed independently. However, documents may be outdated, incomplete, or inconsistent. Analysts must also interpret the content carefully, as documents rarely capture full stakeholder intent. Despite its limitations, this method is often used to supplement other techniques and reduce the need for repetitive stakeholder input.

3.1.4 Observation

Observation requires the analyst to watch stakeholders perform their tasks within a real work setting. This technique helps uncover tacit knowledge—information users don't realize they need to communicate. It is useful in understanding workflows, interface usability, and pain points in current systems [36]. Observation supports the discovery of real-world behaviors and edge cases that may not emerge in discussions. However, it demands time, planning, and ethical sensitivity. Users may behave differently under observation, and rare scenarios might not occur during the observed period. Despite this, it remains an effective method to validate assumptions and supplement interview findings [37].

3.1.5 Joint Application Development (JAD)

Joint Application Development (JAD) involves structured workshops where end users, developers, and facilitators collaborate to gather and refine requirements. These sessions create a shared understanding of system goals and ensure quick feedback. JAD improves stakeholder buy-in by giving everyone a voice during requirement definition [38]. It reduces documentation time by consolidating multiple meetings into a single, focused session. However, successful JAD sessions require skilled moderation, stakeholder availability, and proper planning. Without clear objectives or balanced participation, discussions can become unproductive. JAD is especially helpful for aligning technical and business teams, particularly in time-constrained environments that demand rapid consensus.

3.2 Contemporary Techniques

Contemporary techniques for elicitation use emerging technologies to address the limitations of traditional methods [39]. These approaches are well-suited to modern software environments that demand speed, automation, and the ability to process large volumes of data. Major techniques are discussed below and pros and cons of each technique are provided in Table 3.

3.2.1 LLM-Powered Chatbots

LLM-powered chatbots, based on large language models like GPT, simulate interviews and collect requirements by interacting with stakeholders through natural language [40]. These AI agents ask dynamic questions, adapt to context, and log responses in structured form. They are scalable, operate 24/7, and reduce reliance on human analysts. Chatbots can be integrated into websites, project management

Table 2. Pros and Cons of Traditional Elicitation Techniques.

Technique	Benefits	Limitations
Interviews	• Deep exploration of stakeholder needs • Allows clarification in real-time • Builds trust and rapport	• Time-consuming to schedule and conduct • May be biased by interviewer • Difficult to scale in large projects
Surveys/Questionnaires	• Can reach many stakeholders quickly • Quantifiable and easy to analyze • Cost-effective for remote input	• Lacks depth and context • Low response rate risk • Misleading if poorly designed
Document Analysis	• Uses existing internal knowledge • Low-cost and requires no scheduling • Supports historical insight	• May contain outdated/inaccurate information • Can be hard to interpret intent • Ignores undocumented requirements
Observation	• Captures actual user behavior • Uncovers tacit knowledge • Helps validate assumptions	• Time-intensive setup and analysis • Observer bias possible • Users may change behavior when watched
JAD Sessions	• Encourages real-time collaboration • Reduces rework through consensus • Increases stakeholder buy-in	• Hard to schedule all participants • Risk of dominant voices leading decisions • Requires skilled facilitation

tools, or messaging apps. However, their performance depends on prompt design and model reliability. They may generate vague or incorrect answers, particularly without fine-tuning. Despite these limitations, they offer significant promise for projects with limited analyst availability or high communication demands.

3.2.2 NLP-Based Text Mining

Natural Language Processing (NLP) is used to extract requirements from unstructured text such as emails, reviews, bug reports, or transcripts. These tools help automate requirements discovery, reduce manual analysis, and highlight relevant patterns [41]. Techniques like named entity recognition, sentiment analysis, and topic modeling are commonly used. NLP is valuable for large-scale or fast-changing projects, especially in Agile environments. However, success depends on language model quality, domain-specific vocabulary, and preprocessing. Poorly formatted text, informal language, or inconsistent phrasing can limit effectiveness. Despite challenges, NLP-based mining improves efficiency and supports semi-automated elicitation in data-rich software projects [42].

3.2.3 Crowdsourced Feedback Mining

Crowdsourced feedback mining gathers requirements from user-generated content, including app store reviews, support forums, and social media. It captures real opinions and highlights issues not covered in formal sessions [43]. This method reveals trends, common complaints, and desired features across large user populations. It is especially useful for consumer-facing apps or platforms with active communities. However, data quality can be inconsistent, and filtering noise is critical. Automated tools are often used to classify and summarize comments. While not a complete replacement for direct interaction, feedback mining provides valuable context for decision-making and can shape product roadmaps effectively.

Table 3. Pros and Cons of Contemporary Elicitation Techniques.

Technique	Benefits	Limitations
LLM-Powered Chatbots	• 24/7 availability • Effortless cross-project scaling • Automates question flow	• Can produce hallucinated or vague outputs • Lacks emotional intelligence • Requires careful prompt engineering
NLP-Based Text Mining	• Extracts insights from large data volumes • Reduces human effort • Identifies hidden or indirect requirements	• Depends on data quality and preprocessing • May miss domain-specific language • Limited support for ambiguous text
Crowdsourced Feedback Mining	• Reflects real user pain points • Captures emerging trends in usage • Inexpensive to collect and analyze	• High noise-to-signal ratio • Data not always relevant to business goals • Needs strong filtering and cleaning tools
ML-Based Technique Recommendation	• Suggests best-fit elicitation methods • Learns from project patterns • Reduces planning bias	• Needs large training datasets • Lacks transparency in decision-making • May not adapt to unique project scenarios
Conversational Tools	• Auto-generates structured requirements • Saves documentation time • Supports traceability and versioning	• Dependent on clean conversation inputs • Prone to missing subtleties or tone • Requires validation by a human analyst

3.2.4 ML-Based Technique Recommendation

Machine Learning (ML) models can analyze historical project data to suggest suitable elicitation techniques based on variables like domain type, stakeholder roles, team size, and past outcomes [44]. These systems support more objective and data-driven planning. Recommendation engines reduce analyst bias, speed up preparation, and improve technique alignment with project context. However, their accuracy depends on the quality and quantity of available training data. Such systems may struggle to adapt in unique or new project types. Despite this, ML-based recommendation tools are emerging as helpful assistants in tailoring elicitation strategies to fit the needs of complex software environments [45].

3.2.5 Conversational Requirement Generation

Tools like RECOVER use transformer-based models to automatically convert stakeholder conversations into structured requirements. These tools listen to recorded or real-time discussions and extract key requirements, goals, and constraints. This approach reduces manual note-taking, improves traceability, and speeds up documentation. It is especially useful in Agile and DevOps settings where requirements evolve quickly. However, these tools depend on clear speech, accurate transcription, and context-aware interpretation. Errors may occur if the conversation is fragmented or informal [46–49]. Despite this, conversational generation tools are becoming popular for enhancing productivity in requirements workshops and remote stakeholder meetings.

4 Comparative Analysis and Discussion

Requirements elicitation techniques have matured significantly over time, moving from conventional, human-led approaches to advanced, technology-supported methods. This shift reflects

Table 4. Comparison of Traditional vs. Contemporary Elicitation Techniques.

Criteria	Traditional Techniques	Contemporary Techniques
Stakeholder Interaction	High, face-to-face or synchronous	Low to medium, asynchronous or automated
Scalability	Low—limited by time and resources	High—can scale across users and datasets
Speed of Execution	Slower, often requires scheduling	Faster, automates many tasks
Cost Efficiency	Moderate to high—human-intensive	Lower per-user cost after setup
Tool Dependency	Low—primarily manual methods	High—depends on AI, NLP, ML tools
Data Type Handled	Structured or verbal input	Unstructured text, voice, reviews, transcripts
Flexibility/Adaptability	Medium—requires process customization	High—adapts dynamically based on data and context
Risk of Misinterpretation	Subjective but manageable through discussion	High—depends on model accuracy and training
Best Fit For	Complex domains, small to medium teams, regulated industries	Large-scale apps, fast-paced teams, customer-facing platforms
Human Oversight Requirement	High—manual review and validation needed	Medium to high—automated but still needs validation
Traceability	Manual documentation and tracking	Automated logs, model-based traceability
Reusability of Artifacts	Limited—documents are case-specific	High—datasets, models, and patterns can be reused
Feedback Integration	Slower—feedback loops via meetings or reviews	Faster—automated feedback capture from users and systems
Training Requirement	Low to moderate—based on domain expertise	High—requires technical skills to manage tools/models
Error Handling	Direct discussion to clarify errors	Error-prone if training data is biased or incomplete
Stakeholder Inclusivity	Limited to reachable participants	Broader—can include remote, global, or passive users
Documentation Quality	Rich narrative, manually curated	Structured, machine-generated summaries or insights
Cognitive Load on Stakeholders	High—active participation needed	Low to medium—passive data sources can be used
Change Management	Rigid—requires renegotiation	Agile—can adapt to requirement changes dynamically
Ethical/Privacy Concerns	Low—clear boundaries and control	Higher—data-driven methods raise ethical/privacy risks

the growing complexity of software projects, the rise of agile and DevOps cultures, and the need to handle large volumes of user feedback in real time. Traditional methods such as interviews, surveys, document analysis, and JAD sessions are still widely used because they provide rich, contextual understanding of stakeholder needs. They are especially effective in domains where personal interaction, legal compliance, or domain-specific knowledge is critical—such as healthcare, defense, or government systems.

However, these methods tend to struggle in large-scale projects where scalability, speed, and automation are essential. Scheduling interviews or conducting JAD sessions in globally distributed teams can be difficult and time-consuming. Furthermore, traditional techniques often rely on subjective interpretation and can suffer from stakeholder bias,

Table 5. Recommended Elicitation Techniques for Different Project Types Using Traditional and Modern Approaches.

Project Type	Recommended Traditional Techniques	Recommended Modern Techniques
Government / Defense	Interviews, Document Analysis, JAD	NLP Mining for legacy docs, Conversational AI Tools
Startup / MVP	Informal Interviews, Surveys	LLM Chatbots, Feedback Mining, ML Technique Selection
Mobile Application	Surveys, Observation	Crowdsourced App Review Mining, Sentiment Analysis Tools
Enterprise System	JAD, Document Reviews, Structured Interviews	Conversational NLP, ML-Augmented Traceability Tools
AI/ML Product	Expert Panels, Goal-Oriented Interviews	LLMs, NLP Pipelines, Requirements Extraction Models
E-Commerce Platform	User Behavior Observation, Feedback Forms	Review Mining, Personalized Elicitation via Recommenders
Healthcare Systems	Stakeholder Workshops, Regulatory Document Study	NLP for Clinical Guidelines, Semi-Automated Classification
Educational Portals	Focus Groups, Interviews with Teachers/Students	Chatbots for Learning Needs, Text Analysis of Feedback
Financial/Banking Apps	Risk Analysis Interviews, Compliance Checks	ML for Fraud Detection Requirements, Secure Chat Interfaces
Open-Source Projects	Forum Mining, Email/Commit Analysis	GitHub Issue Mining, Topic Modeling on Community Data

inconsistent documentation, or incomplete inputs. These limitations have fueled the development of modern techniques that leverage Artificial Intelligence (AI), Natural Language Processing (NLP), and Machine Learning (ML).

Modern techniques, such as LLM-powered chatbots, NLP-based text mining, and conversational requirement generators, are designed to address these limitations by automating parts of the elicitation process. They allow teams to gather feedback at scale, extract insights from unstructured data, and provide round-the-clock assistance. However, they are not without challenges. AI-driven tools can misinterpret ambiguous language, lack emotional intelligence, and require high-quality training data. Moreover, such tools still need human oversight to validate and refine the outputs, especially in safety-critical or ethics-sensitive domains.

In large-scale, user-facing applications like e-commerce or mobile platforms, modern techniques such as feedback mining and sentiment analysis consistently outperform traditional methods by enabling real-time, high-volume input processing. Conversely, in regulated domains like healthcare or government, traditional techniques such as document analysis and stakeholder interviews remain superior due to their interpretability, traceability, and compliance alignment. The hybrid use of LLMs for draft generation alongside human-led workshops has also shown to reduce stakeholder fatigue while improving requirement clarity. These differences underscore the importance of context-aware technique selection rather than one-size-fits-all adoption.

Despite the differences, traditional and contemporary techniques are not mutually exclusive. In fact, combining them often leads to better outcomes. For instance, interviews or JAD sessions can be used to frame the initial requirements, followed by NLP tools to process written feedback or refine documentation. Similarly, LLMs can act as assistants during stakeholder meetings, generating questions or summarizing outcomes in real time. A hybrid approach is especially useful in agile settings, where rapid iteration and frequent feedback cycles are expected.

Ultimately, the choice of elicitation technique depends on several factors, including project scale, domain complexity, team distribution, and tool availability. While traditional methods provide depth and trust, modern methods offer speed and scalability. A

thoughtful combination, tailored to the project's context, can lead to more complete, accurate, and actionable requirements. Table 4 provides the comparison between the discussed traditional and recent requirement elicitation techniques.

To better guide practitioners, we present a use-case mapping in Table 5, which aligns specific project types with the most effective elicitation techniques. This mapping considers project characteristics such as regulatory demands, team scale, and delivery speed. For instance, government and defense projects, which require traceability and extensive documentation, often rely on traditional techniques like JAD sessions and document analysis. On the other hand, fast-moving startups or mobile applications benefit more from modern approaches such as feedback mining or LLM-based assistance, where speed and automation are essential. This classification helps in selecting the most suitable strategies for varied development environments.

In another example, consider a mid-sized fintech startup launching a mobile banking app under strict regulatory compliance. The team combines structured interviews and compliance checks (traditional) with AI-driven feedback mining (modern) to balance legal accuracy with user-centered design. Table 5 helps identify this hybrid strategy by aligning project needs—security, speed, and usability—with suitable elicitation techniques.

While this review aims to provide a comprehensive comparison of traditional and modern elicitation techniques, it is subject to certain limitations. The literature selection may carry bias due to the focus on peer-reviewed sources, potentially overlooking valuable gray literature. Additionally, some modern techniques discussed are closely tied to specific technologies, which may limit generalizability across all software contexts. The classification framework, though grounded in recurring themes, is interpretive in nature and may evolve with future advancements.

5 Conclusion and Future Work

Requirements elicitation continues to play a pivotal role in shaping the success of software projects. Over the decades, the field has evolved from heavily manual and interaction-based practices to more automated, intelligent, and scalable solutions powered by recent advances in artificial intelligence and data mining. Traditional techniques such as interviews, document reviews, and JAD sessions remain valuable, especially in domains that require deep contextual understanding, regulatory compliance, or trust-building through personal interaction. On the other hand, modern techniques—including NLP-driven mining, LLM-powered chatbots, and machine learning-based recommendations—have addressed many of the scalability and efficiency limitations faced by traditional approaches. This paper presented a comprehensive review and comparison of both traditional and contemporary elicitation methods. Through structured analysis, use-case mapping, and feature-based comparison, it was observed that no single technique universally applies to all project types. Instead, the most effective elicitation strategies are those that balance human-centered insights with automated scalability, adapting techniques based on project scope, domain complexity, and stakeholder availability. This comparative synthesis offers a foundation for both academic and industry audiences for making informed decisions about elicitation practices in an era of rapid digital transformation.

Despite recent progress, several challenges remain in the practice and research of requirements elicitation. Future work should focus on refining hybrid elicitation models that seamlessly integrate traditional human-led methods with intelligent automation. Research into context-aware LLMs and domain-specific NLP models could further enhance the precision of automated requirement extraction. Another promising direction lies in the development of adaptive elicitation systems that learn from stakeholder behavior and dynamically switch between elicitation modes during a project's lifecycle. Moreover, there is a need to explore ethical concerns around AI-powered elicitation—particularly issues related to data privacy, consent, and bias in requirement formulation. Additionally, empirical studies should be conducted across diverse domains to evaluate the real-world effectiveness of modern techniques, especially in terms of stakeholder satisfaction, requirement completeness, and project success rates. Finally, greater emphasis should be placed on creating open-source datasets and benchmarks to enable consistent evaluation of elicitation tools and techniques. As the software industry continues to embrace agility, automation, and AI, requirements elicitation must evolve in parallel—remaining adaptive, inclusive, and intelligent in meeting the demands of tomorrow's software systems.

Data Availability Statement

Data will be made available on request.

Funding

This work was supported without any funding.

Conflicts of Interest

The authors declare no conflicts of interest.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Gobov, D., & Huchenko, I. (2021, February). Software Requirements Elicitation Techniques Selection Method for the Project Scope Management. In *ITPM* (pp. 1-10).

[2] Görer, B., & Aydemir, F. B. (2023, September). Generating requirements elicitation interview scripts with large language models. In *2023 IEEE 31st international requirements engineering conference workshops (rew)* (pp. 44-51). IEEE. [Crossref]

[3] Canedo, E. D., Calazans, A. T. S., Bandeira, I. N., Costa, P. H. T., & Masson, E. T. S. (2022). Guidelines adopted by agile teams in privacy requirements elicitation after the Brazilian general data protection law (LGPD) implementation. *Requirements Engineering, 27*(4), 545-567. [Crossref]

[4] Ronanki, K., Berger, C., & Horkoff, J. (2023, September). Investigating chatgpt's potential to assist in requirements elicitation processes. In *2023 49th Euromicro conference on software engineering and advanced applications (SEAA)* (pp. 354-361). IEEE. [Crossref]

[5] Wang, Z., Chen, C. H., Zheng, P., Li, X., & Khoo, L. P. (2021). A graph-based context-aware requirement elicitation approach in smart product-service systems. *International Journal of Production Research, 59*(2), 635-651. [Crossref]

[6] Zhang, K., Lin, K. Y., Wang, J., Ma, Y., Li, H., Zhang, L., ... & Feng, L. (2023). UNISON framework for user requirement elicitation and classification of smart product-service system. *Advanced Engineering Informatics, 57*, 101996. [Crossref]

[7] Alturaief, N., Aljamaan, H., & Baslyman, M. (2021, November). Aware: Aspect-based sentiment analysis dataset of apps reviews for requirements elicitation. In *2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)* (pp. 211-218). IEEE. [Crossref]

[8] White, J., Hays, S., Fu, Q., Spencer-Smith, J., & Schmidt, D. C. (2024). Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design. In *Generative AI for Effective Software Development* (pp. 71-108). Cham: Springer Nature Switzerland. [Crossref]

[9] Faik, I., & Sengupta, A. (2024). INCLUSION BY DESIGN: REQUIREMENTS ELICITATION WITH DIGITALLY MARGINALIZED COMMUNITIES. *MIS Quarterly, 48*(1). [Crossref]

[10] Gupta, S. (2022). Non-functional requirements elicitation for edge computing. *Internet of Things, 18*, 100503. [Crossref]

[11] Rahamathunnisa, U., Subhashini, P., Aancy, H. M., Meenakshi, S., & Boopathi, S. (2023). Solutions for software requirement risks using artificial intelligence techniques. In *Handbook of Research on Data Science and Cybersecurity Innovations in Industry 4.0 Technologies* (pp. 45-64). IGI Global. [Crossref]

[12] Arora, C., Grundy, J., & Abdelrazek, M. (2024). Advancing requirements engineering through generative ai: Assessing the role of llms. In *Generative AI for Effective Software Development* (pp. 129-148). Cham: Springer Nature Switzerland. [Crossref]

[13] Nazim, Z., Ishaq, K., Alvi, A., Rosdi, F., Nawaz, N. A., & Dogar, A. B. (2025). Toward a Gamification Framework for Requirement Elicitation: Insights From a Systematic Literature Review. *Human Behavior and Emerging Technologies, 2025*(1), 3255995. [Crossref]

[14] Aqeel, S., & Khan, N. A. (2025). Challenges and Issues in Requirements Elicitation for Based Systems: A Systematic Literature Review. *Bridging Global Divides for Transnational Higher Education in the AI Era*, 423-446. [Crossref]

[15] Ozkan, B., Jungerius, N., Adali, O. E., & Turetken, O. (2025). Value cocreation-oriented digital platform design: a method for requirements elicitation and platform assessment. *Production Planning & Control*, 1-19. [Crossref]

[16] Davis, A. M. (1993). *Software requirements: objects, functions, and states*. Prentice-Hall, Inc..

[17] Sommerville, I., & Sawyer, P. (1997). *Requirements engineering: a good practice guide*. John Wiley & Sons, Inc..

[18] Zowghi, D., & Coulin, C. (2005). Requirements elicitation: A survey of techniques, approaches, and tools. In *Engineering and managing software requirements* (pp. 19-46). Berlin, Heidelberg: Springer Berlin Heidelberg. [Crossref]

[19] Cohn, M. (2004). *User stories applied: For agile software development*. Addison-Wesley Professional.

[20] Paetsch, F., Eberlein, A., & Maurer, F. (2003, June). Requirements engineering and agile software development. In *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.* (pp. 308-313). IEEE. [Crossref]

[21] Sonbol, R., Rebdawi, G., & Ghneim, N. (2022). The use of nlp-based text representation techniques to support requirement engineering tasks: A systematic mapping review. *IEEE Access, 10*, 62811-62830. [Crossref]

[22] Kolahdouz-Rahimi, S., Lano, K., & Lin, C. (2023). Requirement formalisation using natural language processing and machine learning: A systematic review. *arXiv preprint arXiv:2303.13365*. [Crossref]

[23] Voria, G., Casillo, F., Gravino, C., Catolino, G., & Palomba, F. (2025). RECOVER: Toward Requirements Generation from Stakeholders' Conversations. *IEEE Transactions on Software Engineering*. [CrossRef]

[24] Ataei, M., Cheong, H., Grandi, D., Wang, Y., Morris, N., & Tessier, A. (2025). Elicitron: A large language model agent-based simulation framework for design requirements elicitation. *Journal of Computing and Information Science in Engineering, 25*(2), 021012. [Crossref]

[25] Cheligeer, C., Huang, J., Wu, G., Bhuiyan, N., Xu, Y., & Zeng, Y. (2022). Machine learning in requirements elicitation: A literature review. *AI EDAM, 36*, e32. [Crossref]

[26] Gobov, D., & Solovei, O. (2023, March). Approaches to improving the accuracy of machine learning models in requirements elicitation techniques selection. In *International Conference on Computer Science, Engineering and Education Applications* (pp. 574-584). Cham: Springer Nature Switzerland. [Crossref]

[27] Akram, F., Ahmad, T., & Sadiq, M. (2024). Recommendation systems-based software requirements elicitation process—a systematic literature review. *Journal of Engineering and Applied Science, 71*(1), 29. [Crossref]

[28] Dąbrowski, J., Letier, E., Perini, A., & Susi, A. (2023). Mining and searching app reviews for requirements engineering: Evaluation and replication studies. *Information Systems, 114*, 102181. [Crossref]

[29] Maalej, W., Biryuk, V., Wei, J., & Panse, F. (2025). On the automated processing of user feedback. In *Handbook on Natural Language Processing for Requirements Engineering* (pp. 279-308). Cham: Springer Nature Switzerland. [Crossref]

[30] Haq, I. U., Saddique, T., Basharat, M., & Butt, W. H. (2024, July). A Hybrid S/W Requirement Elicitation Approach to Improve Quality of Requirements. In *International conference on WorldS4* (pp. 75-85). Singapore: Springer Nature Singapore. [Crossref]

[31] Dunsin, D. BUSINESS ANALYSIS TECHNIQUES FOR EFFECTIVE PROJECT REQUIREMENTS GATHERING.

[32] Liu, K., Reddivari, S., & Reddivari, K. (2022, August). Artificial intelligence in software requirements engineering: State-of-the-art. In *2022 IEEE 23rd International Conference on Information Reuse and Integration for Data Science* (*IRI*) (pp. 106-111). IEEE. [Crossref]

[33] Ali, N., Hong, J. E., & Chung, L. (2021). Social network sites and requirements engineering: A systematic literature review. *Journal of Software: Evolution and Process, 33*(4), e2332. [Crossref]

[34] Ferrari, A., Spoletini, P., & Debnath, S. (2022). How do requirements evolve during elicitation? An empirical study combining interviews and app store analysis. *Requirements Engineering, 27*(4), 489-519. [Crossref]

[35] Shahzad, B., Javed, I., Shaikh, A., Sulaiman, A., Abro, A., & Ali Memon, M. (2021). Reliable requirements engineering practices for COVID-19 using blockchain. *Sustainability, 13*(12), 6748. [Crossref]

[36] Pei, Z., Liu, L., Wang, C., & Wang, J. (2022, August). Requirements engineering for machine learning: A review and reflection. In *2022 IEEE 30th International Requirements Engineering Conference Workshops* (*REW*) (pp. 166-175). IEEE. [Crossref]

[37] Sari, D. A. P., Putri, A. Y., Hanggareni, M., Anjani, A., Siswondo, M. L. O., & Raharjana, I. K. (2021, February). Crowdsourcing as a tool to elicit software requirements. In *AIP Conference Proceedings* (Vol. 2329, No. 1, p. 050001). AIP Publishing LLC. [Crossref]

[38] Khairat, M. I. S. B., Priyadi, Y., & Adrian, M. (2022, January). Usability measurement in user interface design using heuristic evaluation & severity rating (case study: Mobile ta application based on MVVM). In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference* (*CCWC*) (pp. 0974-0979). IEEE. [Crossref]

[39] Dalpiaz, F., Gieske, P., & Sturm, A. (2021). On deriving conceptual models from user requirements: An empirical study. *Information and Software Technology, 131*, 106484. [Crossref]

[40] Oleson, A., Solomon, M., Perdriau, C., & Ko, A. (2023). Teaching inclusive design skills with the cider assumption elicitation technique. *ACM Transactions on Computer-Human Interaction, 30*(1), 1-49. [Crossref]

[41] Villamizar, H., Escovedo, T., & Kalinowski, M. (2021, September). Requirements engineering for machine learning: A systematic mapping study. In *2021 47th Euromicro conference on software engineering and advanced applications* (*SEAA*) (pp. 29-36). IEEE. [Crossref]

[42] Olukoya, O. (2022). Assessing frameworks for eliciting privacy & security requirements from laws and regulations. *Computers & Security, 117*, 102697. [Crossref]

[43] Atoum, I., Baklizi, M. K., Alsmadi, I., Otoom, A. A., Alhersh, T., Ababneh, J., ... & Alshahrani, S. M. (2021). Challenges of software requirements quality assurance and validation: A systematic literature review. *IEEE Access, 9*, 137613-137634. [Crossref]

[44] Dar, H., Lali, M. I., Ashraf, H., Ramzan, M., Amjad, T., & Shahzad, B. (2018). A systematic study on software requirements elicitation techniques and its challenges in mobile application development. *IEEE Access, 6*, 63859-63867. [Crossref]

[45] Kiran, H. M., & Ali, Z. (2018). Requirement elicitation techniques for open source systems: a review. *International Journal of Advanced Computer Science and*

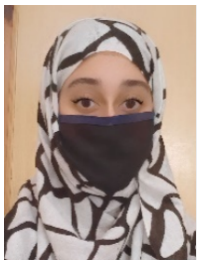
Applications, 9(1). [CrossRef]

[46] Saeed, S., Fatima, U., & Iqbal, F. (2018). A review of Requirement Elicitation techniques in OSSD. *Int. J. Comput. Sci. Netw. Secur, 18*, 86.

[47] Okesola, O. J., Okokpujie, K., Goddy-Worlu, R., Ogunbanwo, A., & Iheanetu, O. (2019). Qualitative comparisons of elicitation techniques in requirement engineering. *ARPN J. Eng. Appl. Sci, 14*(2), 565-570.

[48] Lim, T. Y., Chua, F. F., & Tajuddin, B. B. (2018, December). Elicitation techniques for internet of things applications requirements: A systematic review. In *Proceedings of the 2018 VII International Conference on Network, Communication and Computing* (pp. 182-188). [Crossref]

[49] Mishra, D., Aydin, S., Mishra, A., & Ostrovska, S. (2018). Knowledge management in requirement elicitation: Situational methods view. *Computer Standards & Interfaces, 56*, 49-61. [Crossref]



Asma Akhtar is a researcher who has received her degree in M.S. Computer Science from Virtual University of Pakistan, Lahore. Her research interests lie in the fields of Software Engineering and Machine learning. (Email: asmaakhtarjanjua@gmail.com)

Samia Akhtar received her M.S. degree in Computer Science from Virtual University of Pakistan, Lahore. Her research interests lie in the fields of Software Engineering, Machine Learning and Deep Learning. (Email: samiaakhtar9898@gmail.com)