sustainability

MDPI

Article

A Comparison of Recent Requirements Gathering and Management Tools in Requirements Engineering for IoT-Enabled Sustainable Cities

Muhammad Asgher Nadeem [1], Scott Uk-Jin Lee [1,*] and Muhammad Usman Younus [2,3]

[1] Department of Computer Science and Engineering, Hanyang University, Ansan 15588, Korea; nadeem@hanyang.ac.kr
[2] Department of Computer Science and IT, University of Jhang, Jhang 35200, Pakistan; usman1644@gmail.com
[3] Ecole Doctorale Mathematiques, Informatique, Telecommunications de Toulouse, University Paul Sabatier, 31330 Toulouse, France
* Correspondence: scottlee@hanyang.ac.kr

Abstract: The Internet of Things (IoT) is a paradigm that facilitates the proliferation of different devices such as sensors and Radio Frequency Identification (RFIDs) for real-time applications such as healthcare and sustainable cities. The growing popularity of IoT opens up new possibilities, and one of the most notable applications is related to the evolving sustainable city paradigm. A sustainable city is normally designed in such a way to consider the environmental impact and a social, economic, and resilient habitat for existing populations without compromising the ability of future generations to experience the same, while the process of managing project requirements is known as requirements management. To design a high-quality project, effective requirements management is imperative. A number of techniques are already available to perform the requirement gathering process, and software developers apply them to collect the requirements. Nevertheless, they are facing many issues in gathering requirements due to a lack of literature on the selection of appropriate methods, which affects the quality of the software. The software design quality can be improved by using requirements capture and management techniques. Some tools are used to comprehend the system accurately. In this paper, a qualitative comparison of requirements-gathering tools using Artificial Intelligence (AI) and requirements-management tools is presented for sustainable cities. With all the tools and techniques available for capturing and managing requirements, it has been proven that software developers have a wide range of alternatives for selecting the best tool that fits their needs, such as chosen by the AI agent. This effort will aid in the development of requirements for IoT-enabled sustainable cities.

Keywords: sustainable cities; requirement-gathering tool; qualitative comparison; requirement engineering; software engineering; requirement-management tools

Citation: Nadeem, M.A.; Lee, S.U.-J.; Younus, M.U. A Comparison of Recent Requirements Gathering and Management Tools in Requirements Engineering for IoT-Enabled Sustainable Cities. *Sustainability* **2022**, *14*, 2427. https://doi.org/10.3390/su14042427

Academic Editors: Sheraz Aslam, Herodotos Herodotou, Nouman Ashraf and Wann-Ming Wey

Received: 13 December 2021
Accepted: 18 February 2022
Published: 20 February 2022

1. Introduction

The Internet of Things (IoT) is intended to improve quality of life, and it has significantly changed today's life through different standards [1,2]. The IoT allows different sensors, embedded systems, and intermediate devices to connect to the Internet and form a fully connected world by collecting and sharing data. An IoT network generally consists of different entities, including sensors and Radio Frequency Identification(RFIDs) forming a distributed network for real-time applications such as healthcare, sustainable cities, manufacturing, and supply chain [2]. A smart city is a term used for technologies in which information and communication technology (ICT)-driven technology improves the quality of services in a city, such as healthcare, transportation, business, and communications [3,4]. Currently, many countries are planning to adapt their cities to modern smart city concepts [5,6].

In the software development lifecycle, requirement gathering and managing are very important. Requirements can be defined as stakeholders' expectations from the system end product, functional abilities, or non-functionalities [7]. Requirement engineering is considered at an early stage in the software product development life cycle. The process of dealing with requirements started in 1960. Requirement gathering and analysis is meant to decrease the communication gap between developers and end-users. Researchers have been focusing and exploring requirement engineering (RE) for the past few years, but there is still a gap between research and practical implementation of RE processes.

The software industry has focused on the requirement management system for the effective handling of project requirements. There are two major categories of requirement gathering and management techniques: direct and indirect techniques. The client requirements define the underlying software system's purpose, objectives, working, and limitations [8]. Software engineers have the choice to either use the interview, brainstorming, and prototyping, arrange workshops, or use a passive storyboard for the requirement gathering from the user or client. This improves the software quality by selecting the best available tool for requirement gathering and management, and directly enhances the project success rate. Requirements engineering is recognized as a critical task for the success of any project [8,9].

The selection of appropriate requirement gathering tools depends on many factors, including the system type, client expectation from the system, present working conditions, and features currently available in the system. These systems work with the help of various requirement gathering and management tools.

In the past, the requirement gathering and management process was completed using the manual approach, and all the requirement analysis, testing, and documentation was done using the manual systems [10]. To date, the software development market has designed some tools for data gathering and management purposes [11]. These requirement management tools are the better option for the development teams. These tools help the development team and the overall organization of various phases of the project and align the project with the time, cost, and human efforts.

However, these tools are also used for risk assessment, testing of the requirement, and tracking the completion of the requirement over the time. Report generation and management are also one of the important features of these requirement management tools. The identification of system bugs is easily handled by using these tools. Many tools have the options of visual system designing and requirement quality analysis [12]. Thus, the aim of the study is a qualitative comparison of requirements collection and required requirements.

The rest of the paper is organized as follows. The literature review is presented in Section 2, the proposed methodology and discussion is given in Section 3. The conclusion and future work of this study are given in Section 4.

2. Literature Review

The software product is continuously improving and assisting human life. Many software engineering companies are presenting new software for the betterment of humans. On the other hand, this improvement also creates a problem for software engineers because the clients assume many nonfunctional aspects of the system [13]. Problems become more critical when the clients are not exactly sure what they need. Many times, the clients do not understand the technical details of the systems. From the perspective of the system designers, there are many problems faced by the engineers for the elicitation activities.

The study presented in [14] focused on this issue. They performed a grounded theory-based study involving the 17 software startups engineers. They interviewed them all. They proved that the software startup engineers initially do not follow a single requirement engineering tool or techniques; rather, they continuously change the tools depending on the client's expectations and requirements. They showed that the software engineers changed the selected tool over time based on the influence of developers, software project managers,

market, business model, and clients. They showed that 57% of software engineering must incorporate agile project development in any of their project development phases, but other activities vary and depend on the accessibility of the owner of the project [14].

Wellsandt et al. [15] selected eight tools for the requirement elicitation and checked the performance on the selected parameters. They performed a qualitative analysis focusing on the information quality and client satisfaction rate. Their results were presented in the form of net-graphs. Their analysis was based on client satisfaction parameters. Ikram, Siddiqui, and Khan [16] worked on issue-based information systems (IBIS) and misuse cases (MUC). They compared the system performance by checking the security parameters of the two selected techniques. Their controlled experimental setup was based on a 2×2 factorial design set. They showed that if we use the issue-based information systems and our clients is trained enough to use this mechanism, only then can the acceptance rate of the project be improved to ahigh level.

In the past years, it has been noticed that the field of software engineering is diverted towards the goal-oriented requirement engineering approach [17]. In this approach, the client and the software project developer initially agree on some goals for the modeling and analysis of the requirement. This approach helps the system developer in that now, he has to work on the pre-defined goals. It also minimizes the forthcoming confluent and discourages the clients from having any alternative expectations from the final project. Today, the goal-oriented approach is considered as important as the approaches of the aspect-oriented model, business intelligence model, agent orientation, and model-oriented development.

In [18], the authors evaluated different software engineering models; they classified the work based on framework type, adaptability, implementation platform, and goal-oriented models. In the field of software engineering, there is a concept that we have to select the best available requirement gathering technique for getting quality-based requirements from the clients.

Software engineers have a variety of requirement gathering and management tools. The study presented in [19] showed that software engineers hardly use these elicitation techniques. Rather, they work for the requirement completion of the clients by selecting a combination of multiple approaches. They showed that the individual, collaborative, and contextual aspects of the project influence the software project manager in the choice of tools and techniques [19].

The authors in [20] worked for product-service systems to integrate products and services together and achieve maximum offering value with high functional results. They proposed an approach to deal with incomplete requirements gathered from the clients. Many software engineers face this issue in that the clients are often unaware of answering the questions asked by the software project management team. They introduced a novel approach to handle the uncertainties in the client requirement and presented an easy-to-use model for both the clients and the software project management team [21].

Requirement gathering is always a difficult task for the software project team. This phase is more complex in the case of upcoming software startups. In [21], the authors worked on various collaborative practices that assist new entrepreneurs in the process of software startups and guide them with the available tools and techniques for the software project requirement gathering and management process [21]. In the age of emerging techniques used for the software engineering process, we have a dynamic environment full of many options in software startups; the important thing to handle in this domain is the choice of an appropriate software development approach. In [22], a detailed discussion is given of the importance of the client's requirements and its impact on the quality of the developed software product [22].

The application of AI techniques to SE challenges has recently sparked a surge of interest. The authors in [23] improved the quality and relieved the burden of the crowdsourcing software development (CSD) platform by using traditional AI planning approaches. They found that automatic planning significantly impacts a given task and is more efficient than manual planning. Researchers have dedicated their efforts to developing DeepCoders,

an AI-based auto-coding software [24]. DeepCoders can create a functioning program by consulting an extensive database of codes. Although the current version of this software only allows you to write a five-line mini-program, the developers have stated that this will be expanded in the following years.

Furthermore, the researchers can use publicly accessible software repositories using ML with a large amount of available data to learn and improve software reusability [25] continuously. For the reusability of components in software development, mining SE data has proven successful. SE tasks benefit from AI in a wide range of applications. For intelligent computing of SE tasks, intelligent knowledge discovery combines AI and data mining. Software intelligence is the result of combining AI and data mining for supporting software engineering applications. It promotes automated software reliability for software construction and overall software development. By analyzing several AI technologies, the potential research prospects of AI and software reuse in software engineering have been prospected.

To address the SE problem, many researchers have used AI-based techniques. The authors in [25] examined how AI approaches have been used for SE. Three types of techniques have received significant use: optimization and search, fuzzy reasoning, and learning [26]. The authors in [27] presented ongoing work from multiple groups, providing an overview of machine learning (ML) and SE in health informatics. It shows that the scale of clinical practice necessitates novel engineering techniques from both disciplines. This is only one example of how novel engineering approaches are needed in AI and SE for specific applications. Therefore, there is a need to tackle the challenges at a higher level of abstraction, as Harman [27] pointed out, because the abstraction would help establish "strategies for finding solutions rather than the solutions themselves."

The requirements phase serves as a core for the success of any software development project. Over the past years, researchers have suggested many techniques/tools for the various activities of the requirements phase of the Software Development Life Cycle (SDLC). SDLC is a process that includes different phases such as requirements analysis, design, coding, testing, implementation, and maintenance of a software system, as well as how these phases are carried out [28]. Still, there are undoubtedly new research problems and issues as well. Because the requirements phase is a conceptual phase of the SDLC, one of the most notable difficulties is the high level of human intervention. Integrating AI tools in the requirements stage is a vast and successful research area. Practitioners are conducting solid research in relevant domains, but there is still a lot of need for more study to create a lot of imperatives [29].

The design phase of the SDLC has great scope for incorporating AI approaches to enhance process efficiency. Furthermore, AI-based approaches may solve the issues by providing a variety of tools/techniques for automating specific operations to some level. However, the authors in [30] point out the challenges at each stage of the design phase and the potential for AI tools to address these concerns. Moreover, using a Venn diagram, proper mapping of the challenges associated with each of the design stages was achieved using suitable AI techniques.

According to [19], the process of RE is the initial step in software development and thus the essential stage of the SDLC. In [31], the authors emphasize the importance of requirements management as the most acutely knowledge-intensive activity for managing complex systems. Requirement elicitation enables users to pinpoint their expectations and prospects for the new system. Unfortunately, due to various difficulties, such as confusing and inconsistent requirements, requirements engineers believe that requirement elicitation is an intriguing and challenging work to obtain adequate requirements. However, addressing these issues to some extent is essential for effective software development. Therefore, the authors categorize the main problems of each requirement elicitation step and explore how AI technology is a viable technology to overcome these problems. This work also describes the connection between the identified issues and their potential AI explanations in many requirements elicitation techniques.

At first glance, the use of AI in SE appears to be a contradiction in terms since AI relies on "intelligent machines", while SE is a creative and knowledge-intensive activity that often requires the participation of human experts [32]. However, from another perspective, the creative process in SE can be effectively supported by machines, which are supported by self-optimizing algorithms that take over the optimization tasks. It is frequently noted that the elicitation methodologies used to acquire software requirements significantly impact the quality of elicited requirements. Therefore, a lot of elicitation strategies have been given in requirement engineering (RE). Still, they are rarely used in practice due to a lack of empirical and comparative evaluations to help the software industry decide which technique is best.

Another study also examines the application of AI approaches to SE processes to significantly reduce time to market and improve the overall quality of software systems. The pattern recognition and the acoustic-phonetic approach are combined in AI because of combining the ideas of pattern recognition methods with an acoustic-phonetic approach. AI plays an indispensable role in various speech recognition activities, including recognizing algorithms and the representation of appropriate inputs [33]. It is worth noting that AI is the most reliable and effective among all the speech recognition methods.

3. Proposed Methodology

In any project development life cycle, requirement gathering and management are the important phases. Requirement gathering requires a lot of mental attention. Many clients are unaware of how to properly describe their requirements. Sometimes, the project team makes irrelevant assumptions about the requirements, and they start focusing on how to "implement", instead of "what to implement". Sometimes, the given document template for the client's requirement gathering is not sufficient for the clear elicitation of the requirements. Initially, the customer requirement elicitation document that is the product of the first three meetings is not exactly implementable.

The software teams mainly gather the requirements by brainstorming, one-to-one meetings, and visiting the client workspace. The team uses a variety of tools and predefined templates for the requirement elicitations. The project management team must understand that all the techniques, tools, and software requirement gathering models are not totally fitted in all the situations. Many project management professionals have experienced better results after implementing the common tools and adopting various techniques for different projects. Some software engineering experts recommended the use of the toolbox technique for a better customer satisfaction rate. Roulette Wheel Selection is used for requirement gathering along with AI models. It has been observed that the learning models provide the best results. An AI-based selection model for requirement gathering is shown in Figure 1.

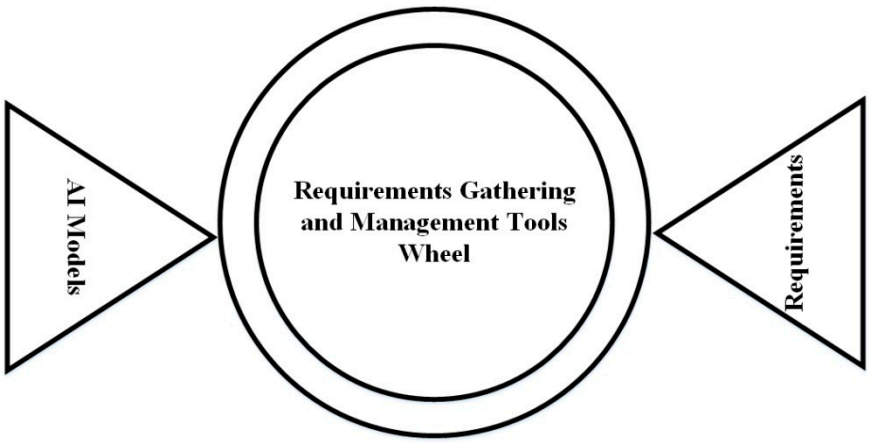

Figure 1. AI-based requirement gathering model.

3.1. Toolbox Technique

For the proper requirement gathering and management, currently, the project management team uses a combination of various tools and RE models. In many software developments companies, the use of multiple software development methodologies is encouraged. We have a choice between the waterfall, agile, spiral, or scrum project development methodology. It is obvious that if the client's project is something different, maybe in the context of working, implementation environment, and future use, different development model is adopted to handle that client and his requirements.

In the next section, we list the most useful tools for requirement gathering. We never use all the tools in every project. However, the choice of right tool for the relevant task can make our project more feasible, easier to manage, and increase the customer satisfaction rate. These tools are good for better software project requirement elicitation and give the team a much clearer insight for implementing client requirements into the software format.

3.1.1. Context Diagram

All the relevant entities, the surrounding environment of the used actors, and the system boundaries are cleared by the context diagram. This diagram acts as the first supporting tool for the requirement elicitation process. This diagram's output highlights the internal and external system's boundaries, the end-users of the system, and all the supporting team members either in-house or outsourced. It identifies which of the services the project team may have to get from a third-party service provider [34]. Figure 2 shows the context diagram model.

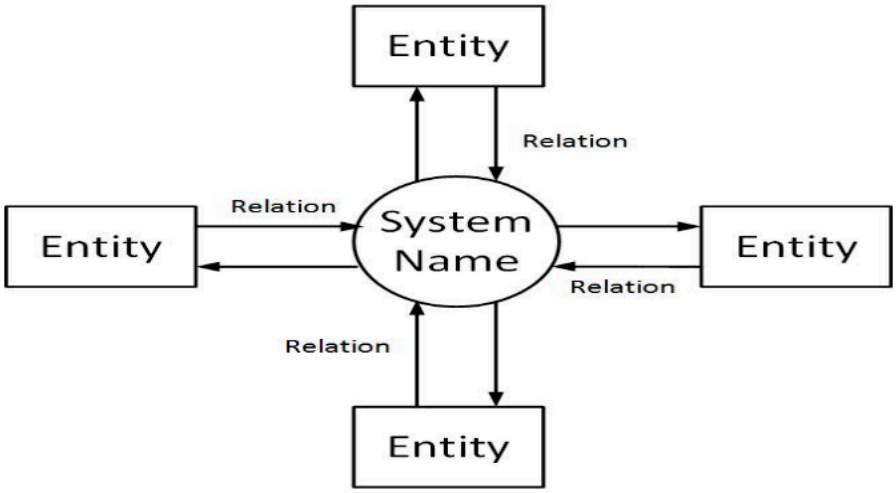

Figure 2. Context diagram.

This visual model of the system gives a visual understanding of the actors, their interaction, working modules, and communication among all. The major advantage of this context model is the clear understanding of the software development team and the customer. All the stakeholders know about the project's scope, what the included things are, and which things are out of the scope. This diagram also helps to validate the operational activities of the system. It is like the company chart that tells the clients about the major to minor processes, tasks, and activities.

3.1.2. Functional Decomposition

This diagram is the hierarchical representation of the working modules. All the major functions and the major use cases are listed in a top-down manner. The previous tool listed the major entities, functions, and interactions among all the components [35]. Now, the use of this tool gives us the functional decomposition and divides the major use cases into sub-tasks and activities in a hierarchy as shown in Figure 3.

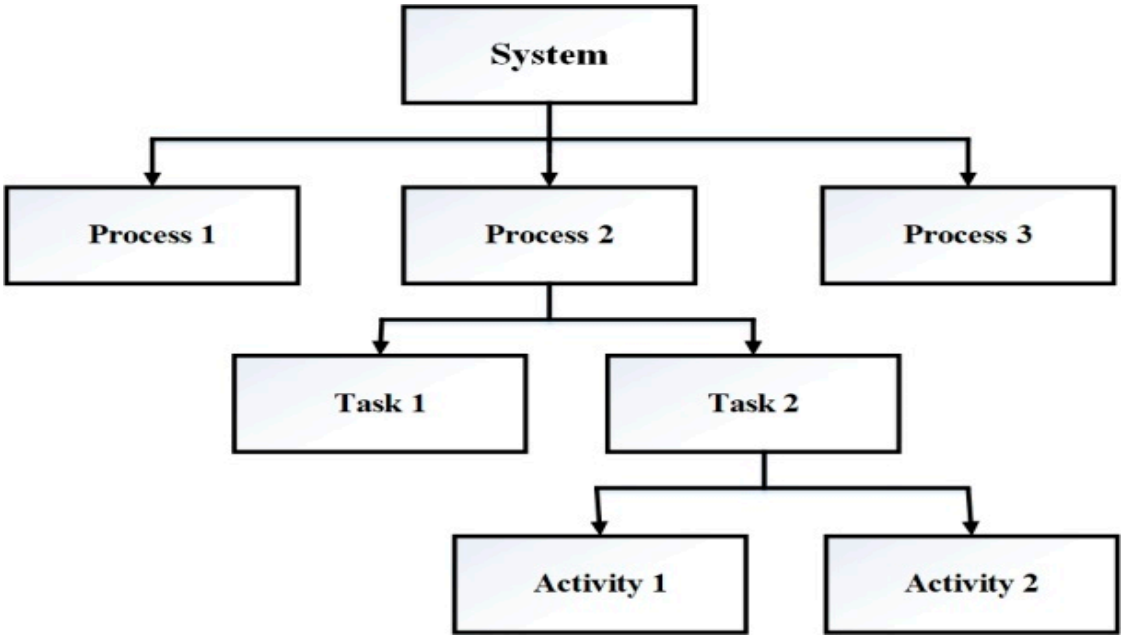

Figure 3. Functional decomposition.

This diagram also helps to validate the operational activities of the system. It is like the company chart that tells the clients about the major to minor processes, tasks, and activities.

3.1.3. AS-IS Model

This is used to elaborate the present workflow of the end-user environment graphically. It is very important for the team to design this diagram first for a more comprehensive system description. It gives all the complex parts of the project. After using this requirement elicitation tool, the project development team is now more satisfied about the clearance of the requirements [36]. They know the easy and complex parts of the project under development, as represented in Figure 4.

3.1.4. TO-BE Activity Model

The TO-BE activity model gives the organization state after the final process. It is the after-development business expectation, showing how the new project can benefit the client's company and what difference the under-development project could create for the owner's company. As we discuss the requirement elicitation or gathering tool, this is a very supportive tool for the initial requirement gathering phases. Now the project team member, the owner, and all the stakeholders clearly understand the project phases and activities involved [37].

3.1.5. User Stories

The use of user stories helps the software engineers in the identification of the requirements [38]. The physical or virtual cards are more significant and assist in the initial requirement gathering phase with the user stories. The role of user stories is vital in case of a complex system or working on the major iterations of the system. User stories make the system more understandable and reduce its complexity level.

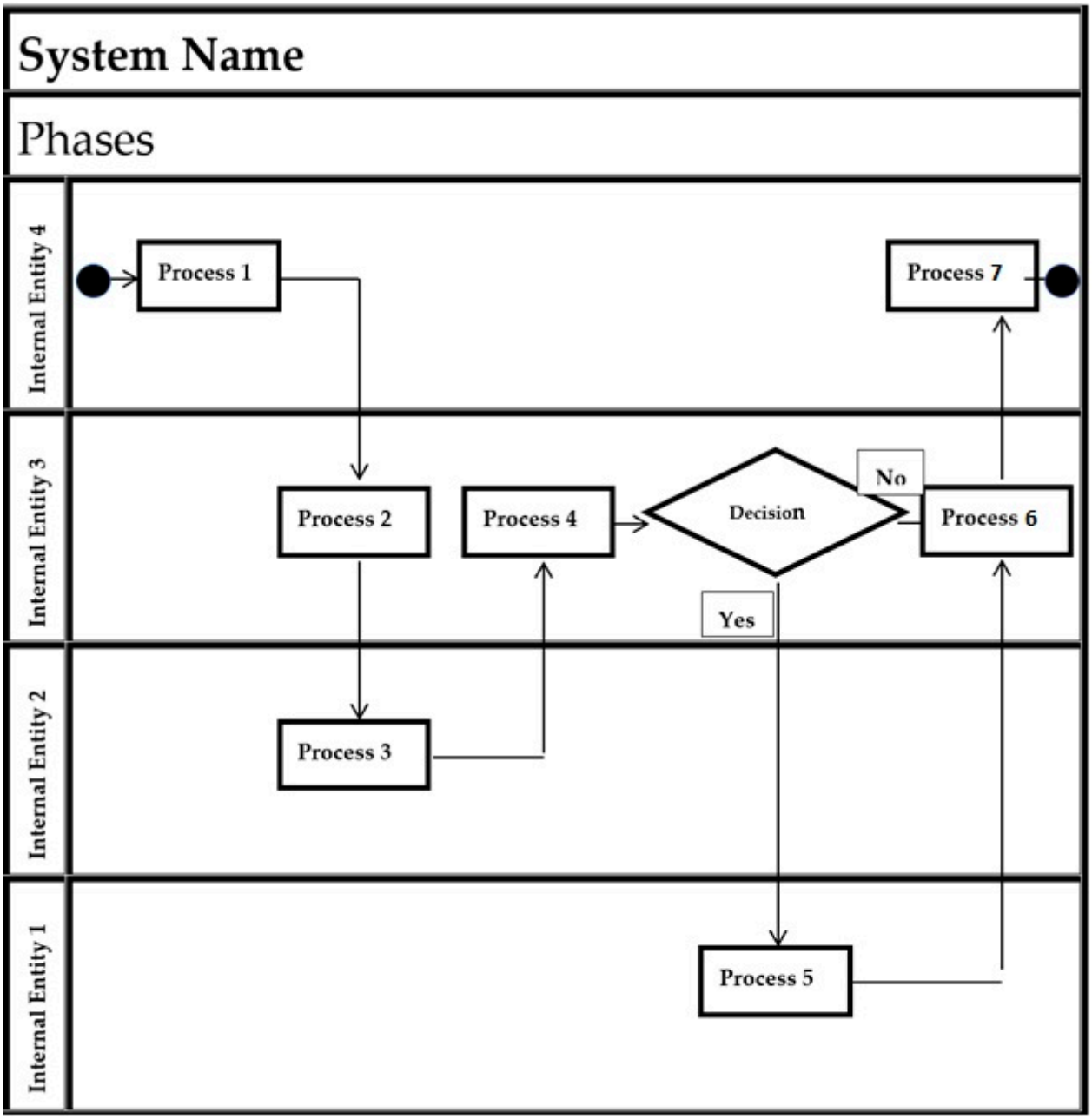

Figure 4. AS-IS model.

3.1.6. Mind Maps

The choice of tools for data gathering and management depends on many factors, such as the mind maps used by the software engineer to get the idea of the customers [39]. Often, the clients cannot make explicit the system requirements; at that time, the software engineer gets help from the mind map tool to elicit the idea from the client.

In some cases, the client does not know the technical details or is unable to express their expectations. In these conditions, the use of mind map is the better choice. There are many requirement gathering techniques other than this that are being practiced in the market. The reason for selecting these techniques in the underlying qualitative comparison is the fact that no previous work in the literature had been done on these techniques. We believe that this qualitative comparison encourages researchers to focus their attention on these techniques and the software engineers also find this study helpful in choosing the better requirement gathering technique in their run time situations, as shown in Figure 5.

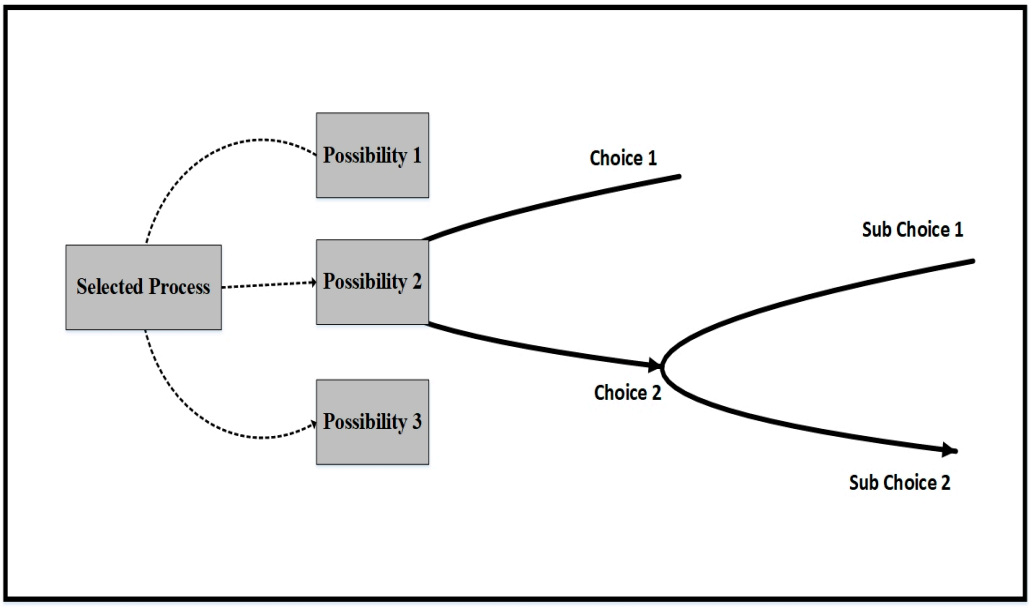

Figure 5. Mind map.

We use the following parameters for the qualitative comparison of the above-mentioned requirement gathering techniques. Details for these parameters are mentioned in the below section.

3.2. Evaluation Parameters for the Qualitative Comparison Using AI

3.2.1. User Involvement

User involvement is an important parameter in the selection of appropriate requirement gathering techniques. There are some techniques in which the client is also directly involved in the system development phase or even in the requirement elicitation activity. Software engineers argue that user involvement has crucial importance in getting a maximum satisfaction rate from the software. User involvement enhances software project success and reduces many aspects that cause conflicts among the client, software developer, and designer. Along with this, user involvement guarantees the fulfillment of maximum client expectations, as the client is directly involved in the designing and requirement elicitation phases. In light of the above-mentioned details, we find the choice of user involvement a significant parameter to consider for the evaluation parameter selection.

3.2.2. Ease in Requirement Elicitation

It is important to consider the difficulty level being experienced by the client and the requirement elicitation team. Many times, the client feels uncomfortable in the requirement elicitation phase. Therefore, we have to consider a technique that provides maximum ease in the requirement elicitation phase.

3.2.3. Technical Background of Client

Technical background matters a lot in the requirement gathering phase. A client having good technical knowledge of the project is a perfect source of giving the project requirements to the software project development team. They know the technical details of the system. The requirement elicitation phase runs smoothly with the person who knows the technical details of the system that is under development in the requirement elicitation phase.

3.2.4. Number of Client Meetings

The number of client meetings has its own impact. There are some techniques for requirement gathering in which we have a lower number of meetings with the clients. For

example, in the case of surveys, questionnaires, and requirement workshops, there is less chance to arrange these more frequently than map maps and interviews.

3.2.5. Operational Difficulty for Software Engineers

There are some techniques used for requirement gathering that are difficult to manage for the software development team. Many times, it creates problems to manage in the case of larger projects. In the case of brainstorming, if we have multiple stakeholders, it is difficult for the software project team member to have separate discussions. They may have different and slightly confusing requirements, and those requirements are difficult to build in a single software piece, such as if two or three requirements are contradictory to each other. Therefore, these are the parameters that we have selected for the qualitative comparison of the select requirement gathering techniques.

Figures 6–8 shows the qualitative comparison in the form of spider chart. For the design of these spider charts, we assigned numerical values to the high, medium, and low terms used in Tables 1 and 2. We assigned number three for high, two for medium, and one for low value.

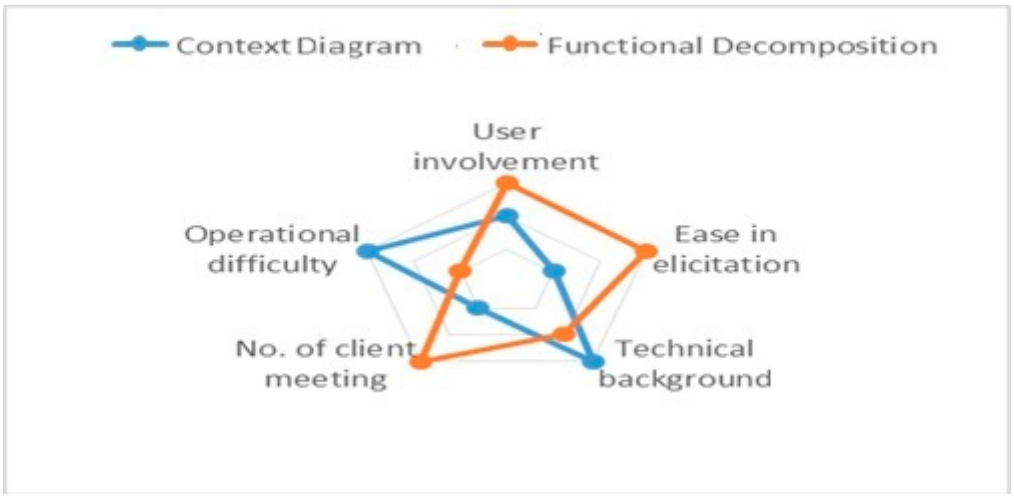

Figure 6. Spider chart for context diagram and functional decomposition.

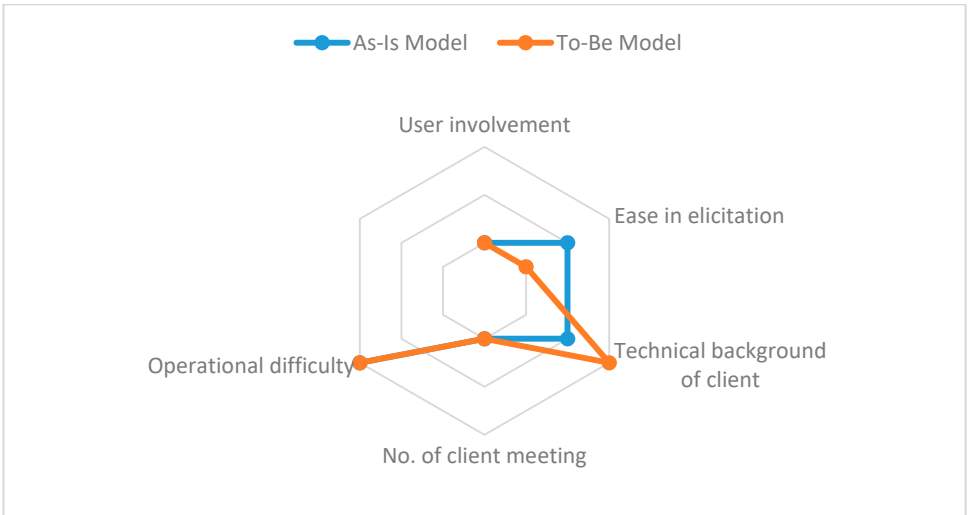

Figure 7. Spider chart for As-Is and To-Be model.

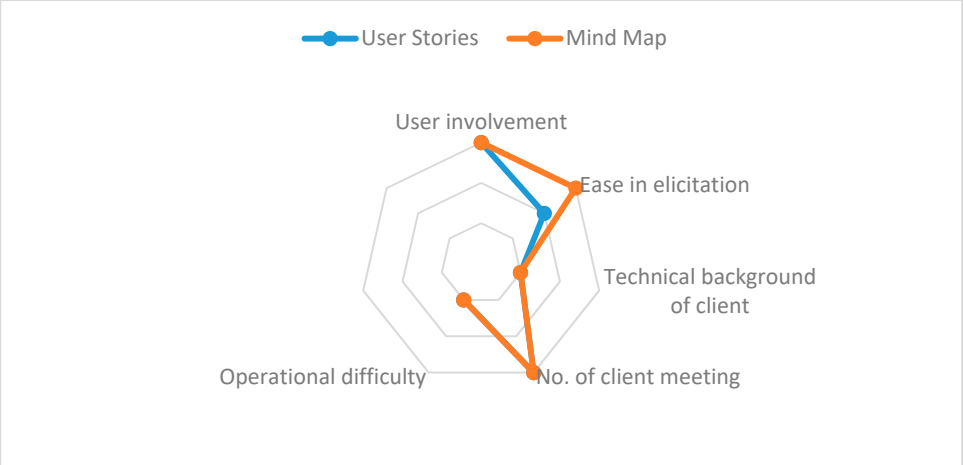

Figure 8. Spider chart for user stories and mind map.

Table 1. Qualitative evaluation of AI-based requirement gathering techniques.

Parameters	Context Diagram [34]	Functional Decomposition [35]	AS-IS Activity Model [36]	TO-BE Activity Model [37]	User Stories [38]	Mind Maps [39]
User involvement	medium	high	low	low	high	high
Ease in elicitation	low	high	medium	low	medium	high
Technical background of client	high	medium	medium	high	low	low
No. of client meetings	low	high	low	low	high	high
Operational difficulty	high	low	high	high	low	low
Total	2H, 1M, 2L	3H, 1M, 1L	1H, 2M, 2L	2H, 0M, 3L	2H, 1M, 2L	3H, 0M, 2L

Table 2. Qualitative comparison using AI for the requirement management tools.

Techniques		ReQtest [40]	ReqSuite [41]	Jama Software [42]	SpiraTeam [43]	Process Street [44]	Visure [45]	Aligned Elements [46]	IBM Rational DOORS [47]	Xebrio [48]	Borland Caliber [49]
Sr	Parameters										
1	Bug Tracking	High	Low	Low	Low	Low	High	High	High	Low	Low
2	Hierarchical requirement management	High	High	Low	High	Medium	Medium	Low	Low	High	Medium
3	Visual designing	High	Medium	Low	Medium	High	Low	Low	Low	Low	Low
4	Risk management	Low	Low	High	Medium	Low	High	Low	Medium	High	High
5	Test management	Medium	High	High	Low	Medium	High	High	Medium	High	Medium
6	Requirement testing	Medium	High	High	High	Low	High	Low	High	High	Medium
7	Requirement quality analysis	Medium	Medium	Medium	Low	High	High	Low	High	Low	Low
8	Requirement versioning	Low	Low	Low	High	Low	High	Low	Low	Low	Medium
9	Release management	Low	Low	Low	High	Low	Low	Low	Medium	Low	Medium
10	Comprehensive report writing	Medium	High	Low	Medium	Low	Medium	Medium	Low	Medium	Medium
	TOTAL	3H, 4M, 3L	4H, 2M, 4L	H, M, L	4H, 3M, 3L	H, M, L	6H, 2M, 2L	2H, 1M, 7L	3H, 3M, 4L	4H, 1M, 5L	1H, 6M 4L

Table 1 represents the results of the qualitative evaluation of requirement gathering techniques. In this table, we check the selected requirement gathering techniques for the

selected parameters and list the results in the form of high, medium, and low. At the end of the table, we collectively write the total number of high, medium, and low rates.

3.3. Requirement Management Tools

In this section, we highlight the major features of 10 selected tools used in the requirement management phase.

3.3.1. ReQtest

ReQtest is a requirement management tool [40]. Some of its distinct features are bugs tracing and linking between modules of the software development phase. There is the option for direct link generation between the client's requirements with the corresponding test cases. This module assists the system tester in tracing the possible bugs in the system. Along with this, it helps the user in the visual designing of the system and creates ahierarchical tree for the client requirement that finally turned into the context diagram.

3.3.2. ReqSuite

ReqSuite is the well-known requirement management software in the market [41]. Its configuration is simple and can be extended from a single client system to an enterprise-level software. Many software project management companies use ReqSuite for the management of their requirements. The hierarchical requirement management is best maintained using this tool. It is highly used for comprehensive report writing and requirement testing.

3.3.3. Jama Software

Jama tools are designed for requirement management, risk assessment among the client's requirements, and rapid software development [42]. These tools are commonly used for the software requirement of autonomous vehicles, healthcare, and defense industries.

3.3.4. Spira Team

Spira Team is the popular software requirement management tool developed by Inflectra (Washington, DC USA) [43]. This tool works best with the software requirement gathering tool that is "user stories". It gives the options for easy requirement management for all the software team members. The team members can work in collaboration using this tool. When you add the gathered requirements in Spira Team, it automatically determines the number of required test cases used to validate all the gathered requirements. It supports the hierarchical management of the gathered requirements. It also helps the software team in the assignment of the priority at the client's requirement.

3.3.5. Process Street

This is a user-friendly requirement management tool [44]. It is used to manage processes, activities, and tasks involved in each iteration. If the software designer interacts with the client by using this tool, it will assign the software designing in many dimensions. It is also an easy and time saving option for the software development team.

3.3.6. Visure

The requirements management toolkit provided by the Visure has many features, including bug tracking, requirement testing, requirement version maintenance, risk management, and comprehensive report generation tools integrated together in the single tool kit [45]. It provides the services for the requirement quality verification and interlinks the requirement with a suitable development module.

3.3.7. Aligned Elements

This tool helps the software development team develop, maintain, and track for requirement completion [46]. It is a flexible tool that saves much of the designer's time and helps in the management of various requirements.

3.3.8. IBM Rational DOORS

This tool helps for requirement gathering, requirement completion tracking, and analyzes the requirements [47].It gives a visual representation of the requirement and data for assistance to the software development team.

3.3.9. Xebrio

Xebrio is another tool used for requirement management in many software houses [48]. It links the user's requirement to the relevant task. It creates the milestones and tracks the completion of the project over the settled milestones. It is not just a requirement management tool; it provides complete project management services. Some of its features include the management of enterprise tasks, communication interface management, test management, and bug management.

3.3.10. Borland Caliber

This is a complete project management tool. It manages the client's requirements; testing, analyzing, and risk management can be easily performed in this tool [49]. The changing client's requirements can also be easily maintained in this tool.

Table 2 presents a qualitative comparison for the 10 selected requirement management tools. We checked these tools on the basis of the 10 selected parameters. The details of these parameters are presented below. Figures 9–11 represent the results in the form of spider charts.

3.4. Selected Parameters

3.4.1. Bug Tracking

When we are working on different iterations, there is a chance that a few modules are not working well. They create errors or bugs in the operational phase. It is important to identify the bugs in the system development. Some requirement management tools help the software engineers in the detection of these software bugs in the system.

3.4.2. Hierarchical Requirement Management

When the client is giving the system details, the software engineers often follow ahierarchical approach for the requirement gathering phase; with this hierarchical approach, they better understand the system or product being developed.

3.4.3. Visual Designing

Many times, the software designer wants to depict the system model. For this, they are looking for some tools that can manage the client's requirements, and at the same time, those tools assist them in the visual designing phase of the system.

3.4.4. Risk Management

Risk management is an important task in the project management phase. Many softwares help software engineers in the risk management phases.

3.4.5. Test Management

When the software developer is working on various interconnected iterations, he has to manage the testing of these iterations. They look for test management tools that are embedded into the software management toolkit.

3.4.6. Requirement Testing

Requirement testing parameters are also important to check before selecting the appropriate management tool.

3.4.7. Requirements Quality Analysis

The quality analysis of the requirement is very important. Some tools help the software engineers in this regard. That is why we selected this parameter for the qualitative assessment of the requirement management tools.

3.4.8. Requirement Versioning

Some clients give the requirements in different phases. In this situation, the software management team has to maintain the requirement version aligned with each client–designer meeting. Some requirement management tools have this option for the requirement version management.

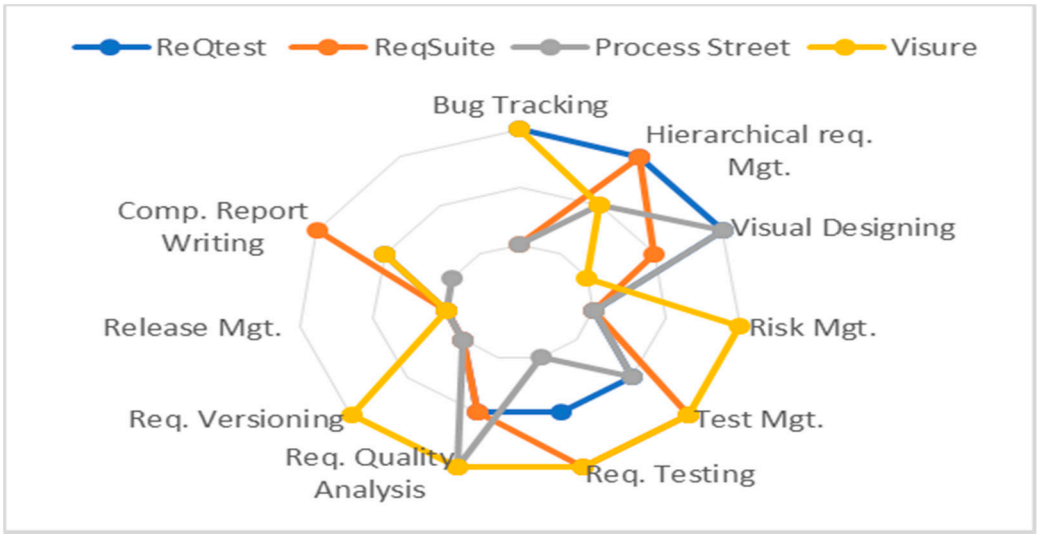

Figure 9. Spider chart for ReQtest, ReqSuite, Process Street, and Visure.

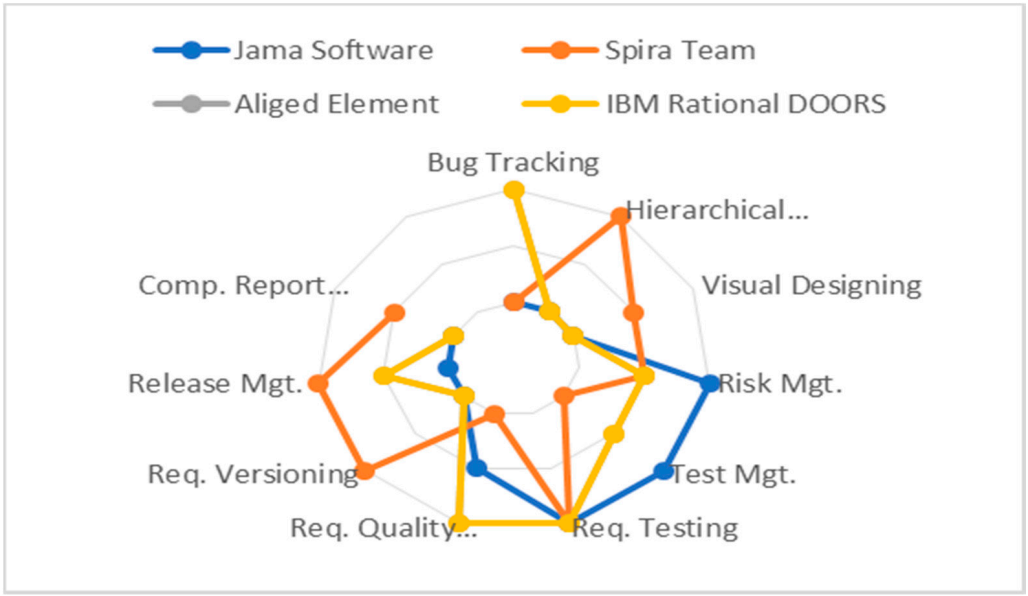

Figure 10. Spider chart for Jama Software, Spiral Team, Aligned Element, and IBM Rational DOORS.

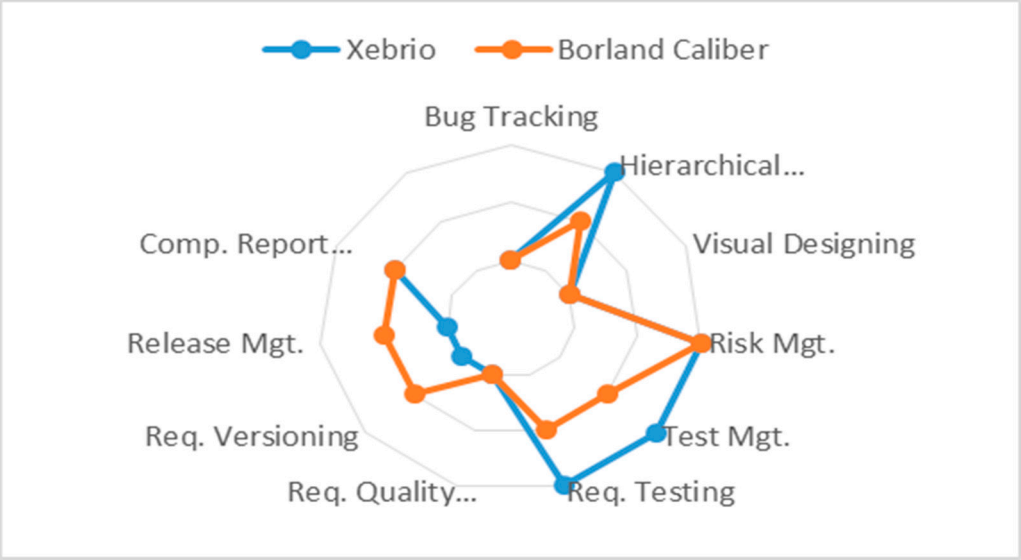

Figure 11. Spider chart for Xebrio and Borland Caliber.

3.4.9. Release Management

Sometimes, the software development team is working on the upgraded version of the software that was previously developed by the same team. In this situation, they have to work for the release management of the product.

3.4.10. Comprehensive Report Writing

The final working software needs comprehensive documentation. Other than this, some clients demand a report for the major modules of the system. Some requirement management tools help in this regard. Hence, we added this parameter for the qualitative comparison of the management tools. The software designing occurs in many dimensions, and it is an easy and time-saving option for the software development team.

3.5. Discussion

Requirement engineering is a subject area that is influenced by a variety of factors, from customer involvement to analyst experience. There are several tools used for requirements gathering and management in requirements engineering. However, choosing the right tool for the task at hand can increase project feasibility, ease management, and improve customer satisfaction. Therefore, in this study, several tools are comprehensively investigated, taking into account several (10) parameters to examine the feasibility of these tools. Parameters include bug tracking, hierarchical requirement management, visual designing, risk management, test management, requirement testing, requirements quality analysis, requirement versioning, release management, and comprehensive report writing. We also present the advantages and disadvantages of all the tools studied. New researchers or people from the industry can analyze any tool from this study before using it. Using appropriate tools helps a software developer to effectively manage user requirements, especially when these tools are used in improving the performance of the sustainable city paradigm.

4. Conclusions and Future Work

In this work, an AI technique is presented to compare requirement gathering and management tools in requirements engineering for IoT-enabled sustainable cities. In conclusion, none of the requirement gathering and management tools is completely suitable for all situations. All the tools have their strengths and weaknesses. The overall performance of selected requirement gathering tools is summarized in Tables 1 and 2. From the results of

the spider chart and the comparison performed, it is concluded that all the tools have low, medium, and high scores based on the parameters.

The future work of this study is to carry out case studies on the selected requirement management tools, analyze the impact on different projects, and measure the client's satisfaction level.

Author Contributions: Conceptualization, M.A.N. and S.U.-J.L.; methodology, M.A.N.; software, M.A.N.; validation, M.A.N., S.U.-J.L. and M.U.Y.; formal analysis, M.U.Y.; investigation, M.A.N. and M.U.Y.; resources, S.U.-J.L.; data curation, M.A.N. and M.U.Y.; writing—original draft preparation, M.A.N.; writing—review and editing, M.A.N. and M.U.Y.; visualization, M.A.N. and M.U.Y.; supervision, S.U.-J.L.; project administration, M.A.N. and S.U.-J.L.; funding acquisition, S.U.-J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-01343, Artificial Intelligence Convergence Research Center (Hanyang University ERICA)).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Aslam, S.; Michaelides, M.P.; Herodotou, H. Internet of ships: A survey on architectures, emerging applications, and challenges. *IEEE Internet Things J.* **2020**, *7*, 9714–9727. [CrossRef]
2. Zhang, X.; Manogaran, G.; Muthu, B. IoT enabled integrated system for green energy into smart cities. *Sustain. Energy Technol. Assess.* **2021**, *46*, 101208. [CrossRef]
3. Khan, A.; Aslam, S.; Aurangzeb, K.; Alhussein, M.; Javaid, N. Multiscale modeling in smart cities: A survey on applications, current trends, and challenges. *Sustain. Cities Soc.* **2022**, *78*, 103517. [CrossRef]
4. Shahzad, B.; Javed, I.; Shaikh, A.; Sulaiman, A.; Abro, A.; Memon, M.A. Reliable Requirements Engineering Practices for COVID-19 Using Blockchain. *Sustainability* **2021**, *13*, 6748. [CrossRef]
5. Aslam, S.; Herodotou, H.; Mohsin, S.M.; Javaid, N.; Ashraf, N.; Aslam, S. A survey on deep learning methods for power load and renewable energy forecasting in smart microgrids. *Renew. Sustain. Energy Rev.* **2021**, *144*, 110992. [CrossRef]
6. Yigitcanlar, T.; Mehmood, R.; Corchado, J.M. Green Artificial Intelligence: Towards an Efficient, Sustainable and Equitable Technology for Smart Cities and Futures. *Sustainability* **2021**, *13*, 8952. [CrossRef]
7. Jeong, J.; Kim, N. Does sentiment help requirement engineering: Exploring sentiments in user comments to discover informative comments. *Autom. Softw. Eng.* **2021**, *28*, 1–26. [CrossRef]
8. Singh, I.; Lee, S.-W. Requirement Engineering and Its Role in a Blockchain-Enabled Internet of Things. *Blockchain Technol. IoT Appl.* **2021**, 1–15. [CrossRef]
9. Singh, I.; Lee, S.-W. Self-adaptive and secure mechanism for IoT based multimedia services: A survey. *Multimedia Tools Appl.* **2021**, 1–36. [CrossRef]
10. Althar, R.R.; Samanta, D. The realist approach for evaluation of computational intelligence in software engineering. *Innov. Syst. Softw. Eng.* **2021**, *17*, 17–27. [CrossRef]
11. Morais, P.; Ferreira, M.J.; Veloso, B. Improving Student Engagement With Project-Based Learning: A Case Study in Software Engineering. *IEEE Rev. Iberoam. Tecnol. Aprendiz.* **2021**, *16*, 21–28. [CrossRef]
12. Tukur, M.; Umar, S.; Hassine, J. Requirement Engineering Challenges: A Systematic Mapping Study on the Academic and the Industrial Perspective. *Arab. J. Sci. Eng.* **2021**, *46*, 3723–3748. [CrossRef]
13. Perscheid, M.; Siegmund, B.; Taeumel, M.; Hirschfeld, R. Studying the advancement in debugging practice of professional software developers. *Softw. Qual. J.* **2017**, *25*, 83–110. [CrossRef]
14. Melegati, J.; Goldman, A.; Kon, F.; Wang, X. A model of requirements engineering in software startups. *Inf. Softw. Technol.* **2019**, *109*, 92–107. [CrossRef]
15. Wellsandt, S.; Hribernik, K.A.; Thoben, K.-D. Qualitative Comparison of Requirements Elicitation Techniques that are Used to Collect Feedback Information about Product Use. *Procedia CIRP* **2014**, *21*, 212–217. [CrossRef]
16. Ikram, N.; Siddiqui, S.; Khan, N.F.; Siddiqui, S. Security requirement elicitation techniques: The comparison of misuse cases and issue based information systems. In Proceedings of the 2014 IEEE 4th International Workshop on Empirical Requirements Engineering (EmpiRE), Karlskrona, Sweden, 25 August 2014; pp. 36–43.
17. Rasheed, A.; Zafar, B.; Shehryar, T.; Aslam, N.A.; Sajid, M.; Ali, N.; Dar, S.H.; Khalid, S. Requirement Engineering Challenges in Agile Software Development. *Math. Probl. Eng.* **2021**, *2021*, 6696695. [CrossRef]

18. Horkoff, J.; Aydemir, F.B.; Cardoso, E.; Li, T.; Maté, A.; Paja, E.; Salnitri, M.; Piras, L.; Mylopoulos, J.; Giorgini, P. Goal-oriented requirements engineering: An extended systematic mapping study. *Requir. Eng.* **2017**, *24*, 133–160. [CrossRef]
19. Okesola, O.J.; Okokpujie, K.O.; Goddy-Worlu, R.; Ogunbanwo, A.; Olamma, I. Qualitative comparisons of elicitation techniques in requirement engineering. *Qual. Comp. Elicitation Tech. Requir. Eng.* **2019**, *14*, 565–570.
20. Chanin, R.; Pompermaier, L.; Sales, A.; Prikladnicki, R. Collaborative practices for software requirements gathering in software startups. In Proceedings of the 2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), Montreal, QC, Canada, 27 May 2019; pp. 31–32.
21. Haber, N.; Fargnoli, M. Prioritizing customer requirements in a product-service system (PSS) context. *TQM J.* **2019**, *31*, 257–273. [CrossRef]
22. Hoff, J.D. Requirements practices in software startups. *Sch. Horiz. Univ. Minn. Morris Undergrad. J.* **2019**, *6*, 3.
23. Xie, T. Intelligent software engineering: Synergy between AI and software engineering. In Proceedings of the International Symposium on Dependable Software Engineering: Theories, Tools, and Applications, Beijing, China, 4–6 September 2018; pp. 3–7.
24. Harman, M. The role of artificial intelligence in software engineering. In Proceedings of the 2012 First International Workshop on Realizing AI Synergies in Software Engineering (RAISE), Zurich, Switzerland, 5 June 2012; pp. 1–6.
25. Wangoo, D.P. Artificial intelligence techniques in software engineering for automated software reuse and design. In Proceedings of the 2018 4th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 14–15 December 2018; pp. 1–4.
26. Bécue, A.; Praça, I.; Gama, J. Artificial intelligence, cyber-threats and Industry 4.0: Challenges and opportunities. *Artif. Intell. Rev.* **2021**, *54*, 3849–3886. [CrossRef]
27. Clifton, D.A.; Gibbons, J.; Davies, J.; Tarassenko, L. Machine learning and software engineering in health informatics. In Proceedings of the 2012 First International Workshop on Realizing AI Synergies in Software Engineering (Raise), Zurich, Switzerland, 5 June 2012; pp. 37–41.
28. Aljawarneh, S.A.; Alawneh, A.; Jaradat, R. Cloud security engineering: Early stages of SDLC. *Future Gener. Comput. Syst.* **2017**, *74*, 385–392. [CrossRef]
29. Sharma, S.; Pandey, S.K. Integrating AI Techniques in Requirements Phase: A Literature Review. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.667.4632&rep=rep1&type=pdf (accessed on 12 December 2021).
30. Sharma, S.; Pandey, S.K. Integrating AI Techniques in SDLC: Design Phase Perspective. In Proceedings of the Third International Symposium on Women in Computing and Informatics, Kerala, India, 10–13 August 2015; pp. 383–387.
31. Sharma, S.; Pandey, S.K. Integrating AI Techniques in Requirements Elicitation. Available online: https://ssrn.com/abstract=3462954 (accessed on 12 December 2021). [CrossRef]
32. Shankari, K.H.; Thirumalaiselvi, R. A survey on using artificial intelligence techniques in the software development process. *Int. J. Eng. Res. Appl.* **2014**, *4*, 24–33.
33. Ammar, H.H.; Abdelmoez, W.; Hamdi, M.S. Software engineering using artificial intelligence techniques: Current state and open problems. In Proceedings of the First Taibah University International Conference on Computing and Information Technology (ICCIT 2012), Madinah, Saudi Arabia, 12–14 March 2012; Volume 52.
34. Bleistein, S.J.; Cox, K.; Verner, J. Validating strategic alignment of organizational IT requirements using goal modeling and problem diagrams. *J. Syst. Softw.* **2006**, *79*, 362–378. [CrossRef]
35. Saputri, T.R.D.; Lee, S.-W. Addressing sustainability in the requirements engineering process: From elicitation to functional decomposition. *J. Softw. Evol. Process* **2020**, *32*, e2254. [CrossRef]
36. Truong, T.-M.; Lê, L.-S.; Paja, E.; Giorgini, P. A data-driven, goal-oriented framework for process-focused enterprise re-engineering. *Inf. Syst. E-Bus. Manag.* **2021**, *19*, 683–747. [CrossRef]
37. Pohl, K. *Requirements Engineering: Fundamentals, Principles, and Techniques*; Springer Publishing Company, Incorporated: Berlin/Heidelberg, Germany, 2010.
38. Raharjana, I.K.; Siahaan, D.; Fatichah, C. User Stories and Natural Language Processing: A Systematic Literature Review. *IEEE Access* **2021**, *9*, 53811–53826. [CrossRef]
39. Otaduy, I.; Diaz, O. User acceptance testing for Agile-developed web-based applications: Empowering customers through wikis and mind maps. *J. Syst. Softw.* **2017**, *133*, 212–229. [CrossRef]
40. ReQtest: Requirements, Test Management, Bug Tracking Tool. Available online: https://reqtest.com/ (accessed on 12 December 2021).
41. ReqSuite-The Smart Requirements Management Tool. Available online: https://www.osseno.com/en/requirements-management-tool/ (accessed on 12 December 2021).
42. Jama Software: Requirements Management Software. Available online: https://www.jamasoftware.com/ (accessed on 12 December 2021).
43. Requirements Management Tools & Software. Available online: https://www.inflectra.com/SpiraTeam/Highlights/Requirements-Management.aspx (accessed on 12 December 2021).
44. Simple Process and Workflow Management. Available online: https://www.process.st/ (accessed on 12 December 2021).
45. Visure Solutions: Requirements Management Tool. Available online: https://visuresolutions.com/ (accessed on 12 December 2021).
46. Requirements Management Software. Available online: https://www.aligned.ch/ (accessed on 12 December 2021).

47. Engineering Requirements Management DOORS. Available online: https://www.ibm.com/docs/en/ermd/9.7.2 (accessed on 12 December 2021).
48. Requirements Management Software and Tool. Available online: https://www.xebrio.com/requirements-management-software/ (accessed on 12 December 2021).
49. Borland Caliber. Available online: https://borland-caliber.software.informer.com/11.4/ (accessed on 12 December 2021).