International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
DOI: https://doi.org/10.17762/ijritcc.v11i10.8509
***Article Received:** 29 July 2023 **Revised:** 20 September 2023 **Accepted:** 10 October 2023*

An AHP-Based Framework for Effective Requirement Management in Agile Software Development (ASD)

[1]Kashif Asad, [2]Dr. Mohd. Faizan Farooqui, [3]Dr. Mohd. Muqeem
[1]Department of Computer Application, Integral University, Lucknow, India
kashif.asad007@gmail.com.
[2]Department of Computer Application, Integral University, Lucknow, India
ffarooqui@iul.ac.in
[3]Department of Computer Application, Integral University, Lucknow, India
muqeem@iul.ac.in

Abstract— In Agile changes may be made at any time throughout the project lifetime in agile projects. These changes, however, often lead to longer turnaround times and higher costs while having a substantial influence on the activities and quality of project management. The suggested framework's main goal is to increase the effectiveness and flexibility of the requirement engineering process by successfully managing requirements changes, particularly in contexts where Agile Software Development (ASD) is practiced. The Agile methodology has gained popularity as a strategy for developing software because it can adjust to changing requirements and deliver software gradually. To make sure that the software being produced satisfies stakeholder expectations and adds value to the firm, good requirement management is essential. Using the Analytic Hierarchy Process (AHP) to prioritize requirements based on their relative relevance and urgency, this article introduces a framework for requirement management in Agile Projects. The most important requirements are taken care of first thanks to this strategy, which enables a more organized and informed decision-making process. We demonstrate the actual use of our framework in real-world contexts and highlight its efficacy in solving the issues faced by Agile projects by including a case study and an accompanying table. The suggested framework also supports the three core tenets of the Agile approach—transparency, cooperation, and continuous improvement—to foster an environment of excellence and ongoing learning within the Agile team. By developing these fundamental ideas, our framework not only supports Agile teams' continual growth and development but also helps them manage and prioritize requirements more efficiently, which ultimately improves project results and increases organizational value.

Keywords- Requirement Management, Analytic Hierarchy Process, User Stories, Requirements Change, Requirement Prioritization, Agile Software Development (ASD).

I. INTRODUCTION

Since the software development business is so dynamic and because stakeholder demands are always changing, requirements change is unavoidable. Under the Agile process, change is not only allowed but welcomed as a way to satisfy customers by delivering functioning software frequently and quickly. In order to iteratively gather requirements, agile depends on stakeholder participation [3]. Most software development organizations adopt Agile Software Development (ASD) approaches because of the requirements' rapid changes [23]. But, with Agile projects, it's crucial to handle requirement changes well to keep the project on schedule and provide the required results. the tendency for requirements to alter over time in response to shifting demands from clients, stakeholders, companies, and the workplace [26]. Validating requirement statements and managing documents is the main goal of requirement management, which helps to ensure that the user is satisfied and that the product is delivered correctly [28]. The Analytic Hierarchy Process is one method for handling demand changes in Agile projects (AHP). AHP is a multi-criteria decision-making technique that enables decision-makers to assess and rank several factors. The foundation of AHP is the idea that decision-makers may divide complicated problems into smaller, more manageable components, then assess and contrast those components according to their relative significance [11]. AHP may be used to prioritize and manage requirement modifications in the context of Agile by empowering stakeholders to assess and contrast the effect of suggested changes based on a variety of factors, including business value, technical feasibility, user experience, and risk. Agile teams may choose which changes to prioritize, which changes to postpone, and which changes to reject with more knowledge if they apply AHP in requirement change management.

AHP may be used to enhance cooperation and communication amongst stakeholders in addition to assisting Agile teams manage requirement changes. By include all parties involved in the decision-making process and offering a formal framework for assessing and contrasting needs [12]. Agile methodology's using AHP in requirement change management offers a disciplined and methodical way for handling changes in a dynamic and quickly changing environment. AHP may assist Agile teams to make better decisions, enhance cooperation, and eventually produce software that satisfies the demands of all stakeholders by supplying a method of assessing and prioritizing changes based on various criteria. An alteration to an existing requirement or the addition of a new requirement that may or may not have an impact on existing requirements is referred to as a requirement change. As requirements change often in Agile Software Development (ASD), it is necessary to track and manage these changes and dependencies [20]. Due to the changing demands of system stakeholders and changes in the business environment, requirements frequently change (Kotonya & Sommerville, 2002). Because changes are inevitable, it has also been a significant problem during the software development process (Sommerville & Sawyer, 2000). Requirements can be changed by adding new ones, removing old ones, or improving existing requirements [14].

Key contribution of this research is as below-:
1. Evaluate the data and literature to find patterns and trends linked to Agile methodology's management of changing needs. Provide guidelines and suggestions from this study to help agile teams handle changing project needs successfully.
2. Proposed a framework for the Agile methodology's handling of requirement changes.
3. To implement the Analytic Hierarchy Process (AHP) in change requirement management in Agile methodology, the Agile team can follow a series of steps, such as identifying criteria, assigning weights, developing a hierarchy, assessing pairwise comparison, calculating weighted scores, validating results, and re-prioritizing as needed.

The outline of this research paper is as follows:
Section I explains the necessity for a comprehensive framework covering requirement management and change management procedures is introduced in the introduction. Section II conducts a review of the existing literature on requirement prioritizing strategies and identifies gaps in the methodologies that are already in use. Section III presents the proposed framework and its components, with a particular emphasis on (A) the Requirement Management Process, (B) the Change Requirement Management Process, and (C) the incorporation of the Analytic Hierarchy Process (AHP). Section IV explains the AHP algorithm as a critical component for prioritizing requirements inside the proposed framework. Analytic Hierarchy Process (AHP). Section V evaluates the performance of the proposed framework using case studies, analyzing its usefulness as well as its limits and suggested future avenues for research.

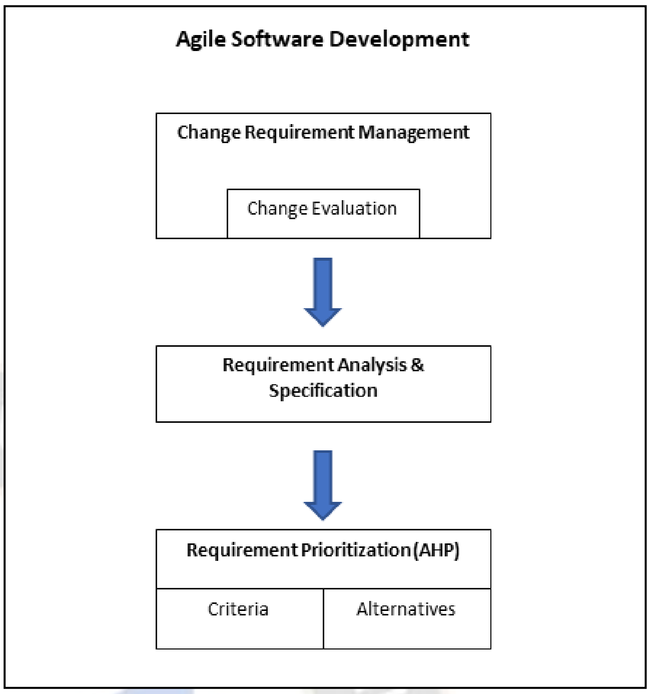
Agile Software Development
Change Requirement Management
Change Evaluation
Requirement Analysis & Specification
Requirement Prioritization (AHP)
Criteria
Alternatives

Figure 1. Requirement Management in Agile Software Development (ASD)

II. RELATED WORK

In each of the large-scale agile projects that Kasauli et al. [1] examined, there were about 30 sub-teams. These writers concluded that firms are struggling with issues like safety and security while attempting to handle QR. A. Perini [2] a machine learning-based strategy that blends stakeholder’s preferences with approximate calculated needs ordering has been developed. A framework was created by Brace and Cheutet [3] to provide a structured approach. They provided a model-driven strategy for determining requirements based on the methodology. Kautz [4] conducted a case study to look into user and client involvement in Agile Software Development. Although he makes no claim to have proposed a process model, the study's findings certainly show an implicitly used process model. A framework for the fusion of design knowledge reuse and requirements management was created by Baxter et al. [5]. The use of requirements management as a dynamic process is made possible by this framework. In their comprehensive analysis of the literature on Agile Requirement Change Management, Jayatilleke and Lai [6] looked at the connected causes of requirements changes as well as the procedures and methods that enable managing requirements change. The authors assessed how businesses manage the decision-making process while adopting requirements modifications and suggested new lines of inquiry for future study. The CS-Coefficients calculated from the Kano analysis are used in Chen and Chang's [22] technique to prioritize needs. As it stems from the Kano method, which made sure that customer-satisfying needs were extracted via a Kano survey, the goal is to provide prioritized requirements that fulfil consumers' expectations. Tahir Kamal et al [27] provide a thorough preparedness model for Agile Requirement Change Management process activities in a Global Software Development environment.

Quesf, A. [8] describes the issue with requirements traceability in agile software development, as well as the interconnection and effects between the traceability and refactoring processes. A practitioner study of issues and solutions for managing security needs in agile project management and agile software development projects was conducted by Terpstra et al. [9].

Hussain et al. [17] provided an overview of the process of change management. Throughout the process of change management, one of the most significant activities is to determine the effects that the proposed change will have on the organization and to estimate the amount of money that will be required to carry out the requested change. The consolidated structure of the change management process was established once these three processes were identified. According to Siponen et al. [24], a quantified risk assessment must be part of the requirements analysis step of an agile security requirements management approach. Security needs should be specifically mentioned at the design phase, monitored during the implementation phase, and tested during the testing phase. Imran Mahmud and Vito Veneziano [25] investigated how a particular cognitive and knowledge-management technique called "mind-mapping," along with its appropriately derived software tools, can facilitate requirements-related activities within an agile development project, how such technique could be reconciled with a sounder requirements management perspective, and, at the same time, mitigate most of agile criticalities and unwanted side-effects.

Mueller, C. [18] describes the effect that changing requirements has on development productivity and to determine whether or not there is a link between the amount of work put into development and changing requirements. Data on measurements were gathered from the various development teams. The effectiveness of a number of different project teams was measured using both time-tested and innovative techniques. One of the metrics that are being explored in this study is the number of test cases that have been passed. The results of the trials show that there is a strong connection between the number of test cases that were successfully completed and the level of productivity.

III. RESEARCH METHOD

The description of what the developer should implement is known as a requirement. Requirements could place restrictions on how the system is developed. Requirements are a statement of what the client expects from the software that will be supplied after the project is finished. Requirements have been outlined by several writers in various ways. The goal of this study is to analyze the problems and difficulties associated with managing change requirements in an agile framework and to pinpoint practical solutions. A mixed-methods strategy will be used for this study to collect both qualitative and quantitative data. A literature evaluation of prior studies on change requirement management in agile methodology will be undertaken in the first part of the study. Agile practitioners will be surveyed in the second and case studies of agile projects will be studied in the third phase.

Phase 1: Comprehensive review

A thorough search of academic databases and pertinent industry publications will be done for papers and research on change requirement management in agile methodology as part of the literature study. Keywords like "agile approach," "requirement management," "change management," and "software development" will all be included in the search. The publications and studies will be chosen based on their quality, applicability to the research question, and research technique. Two phases will be taken in doing the literature review. A preliminary search will be done in the first phase to find important research and publications on the subject. A more thorough search will be done in the second step to find more pertinent papers and research.

Phase 2: Surveys

Surveys will be done to learn more about how agile practitioners feel about change requirement management in the agile approach. The survey questions will include subjects such the frequency and impact of changing needs, the efficacy of existing requirement management procedures, and the tactics used to handle problems and difficulties. They will be developed based on the findings of the literature study.

Phase 3: Case Studies

To learn more specifically about the requirement management techniques used in agile projects, case studies will be undertaken. Project managers, developers, and stakeholders will all be interviewed in-depth for the case studies, which will also include issues like the requirement management process, the tools and techniques used, the difficulties encountered, and the solutions found.

RQ 1: How the Agile Approach handles the Requirement Management process.

RQ 2: How the changes in requirements are managed in Agile Software Development.

RQ 3: How the Requirement Prioritization takes place in Agile Software Development.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
DOI: https://doi.org/10.17762/ijritcc.v11i10.8509
Article Received: *29 July 2023* ***Revised:*** *20 September 2023* ***Accepted:*** *10 October 2023*

A. REQUIREMENT MANAGEMENT PROCESS

Customer input is provided to the product development team by professionals in sales, support, and field-based products. An item from a competing firm is evaluated by the technical specialists. One popular technique for competitor product analysis is to buy a rival's product and give it a close inspection. The requirements analysis of the process is connected to the product demand. The product is a part of a manufacturing procedure, and the particulars of that procedure include the chemicals employed and the required vacuum performance. The market needs specification is a key document that acts as the main source of the product target need. Agile methodology's top requirements management techniques. Many recommended practices may be used to get over requirement management's difficulties in agile methodologies. Some of the crucial behaviors include:

• User stories: In agile methods, user stories are a useful tool for collecting requirements. From the user's point of view, they offer a clear and obvious manner to describe needs, which can assist to prevent misunderstandings and misinterpretations.

• Continuous collaboration: To make sure that everyone is on the same page and that requirements are in line with customer demands, constant engagement between the development team, product owner, and stakeholders is crucial.

• Prioritization: To make sure that the team is working on the most crucial features first, it is imperative to prioritize needs based on their value to the client. Priorities should be reviewed and adjusted on a regular basis to assist minimize disputes and delays.

• Documentation: A certain amount of documentation is required to make sure that requirements are understood and expressed properly, even if the agile methodology prioritizes functioning software above thorough documentation. Agile teams should only produce the documentation required for the success of their project.

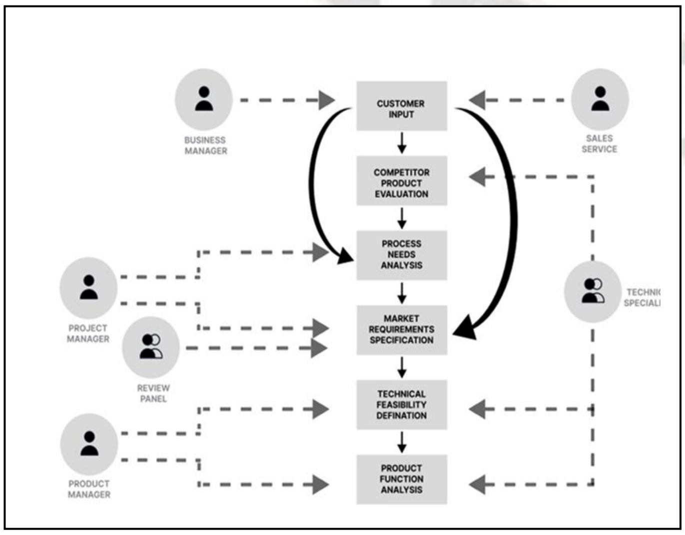

Figure 2. Requirement Management Process

In addition to competitive objectives based on price and performance of competing businesses, it contains information about customer requirements. Process requirements serve as the main source of limitations. The mention of viability confirms that the stated product is within the realm of already available capacity. The process requiring analysis is the main cause of constraints. When these two sources of requirements are integrated, the engineering specification, also known as the technical feasibility definition, is created. The target requirements are largely bound by the findings of the process needs analysis.

B. CHANGE REQUIREMENT MANAGEMENT PROCESS

A technique called Change Requirement Management (CRM) is used to gather, record, carry out, and manage changes to requirements. Modification requests include all pertinent information, including the type of change requested, the person who submitted the request, and the rationale for the change [10]. The needs of the client are his requirements; they guide what must be done but not how [21]. In the General Change Requirement Management process, we start by an initial change request which undergoes processing through the Evaluate Change Request. Here the possible errors are scanned and gauged. Evaluation being done right, it proceeds towards the decision for the final request. Here the change request may go under acceptance or denial. If rejected, the request goes back to the square one that is Initial Change Request. Following which the same process is repeated again from the initial request to evaluation to final decision making.

If acceptance is granted, the request proceeds further to Implementation where the request is applied. Implementation bring the cycle to a Close which brings the program to an end.

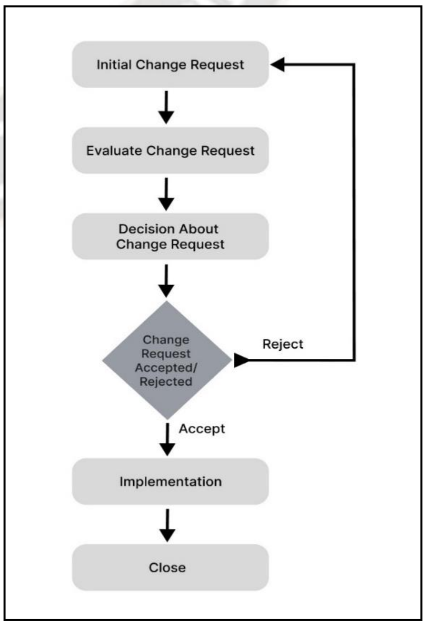

Figure 3. General Change Requirement Management Process[13]

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
DOI: https://doi.org/10.17762/ijritcc.v11i10.8509
***Article Received:** 29 July 2023 **Revised:** 20 September 2023 **Accepted:** 10 October 2023*

During the General Change Requirement Management process, a number of problems may appear, any of which might have an effect on how well the project is completed. These concerns include the following:

- Poor communication: If the members of the team, the stakeholders, and the decision-makers do not communicate well with one another, it may result in misunderstandings, delays, and inaccurate assumptions about the change request. This may lead to judgments that are not ideal and the execution of improvements that are not effective.
- Poor traceability: Lack of documentation and traceability makes it hard to track project changes. This may cause confusion, redundancy, and problems managing dependencies between modifications.
- Incomplete or unclear requirements: If the team receives change requests that are ambiguous or do not include adequate information, it may be challenging for the team to successfully analyze and execute such changes. This might result in the loss of resources, an increase in risk, as well as delays in the project.
- Resistance to change: Stakeholders or team members may reject the suggested changes for a variety of reasons, including personal preferences, fear of the unknown, or worries about more burden. The reasons for this resistance may be found in the previous sentence. This resistance may slow down the process of change management, which in turn can impair the success of the project.

C. PROPOSED FRAMEWORK

The client seeks a request for "Requirement Change Initiation" so that their needs may be met [13]. The next phase is when the "Analysis of Requirement" process takes place, which examines the requirement request to look for any probable errors. The "Analysis of Requirement" tool uses the information it gathers to determine whether or not the need can be put into practice. Here is where the ultimate decision on whether the requirement implementation will be accepted or denied is made.

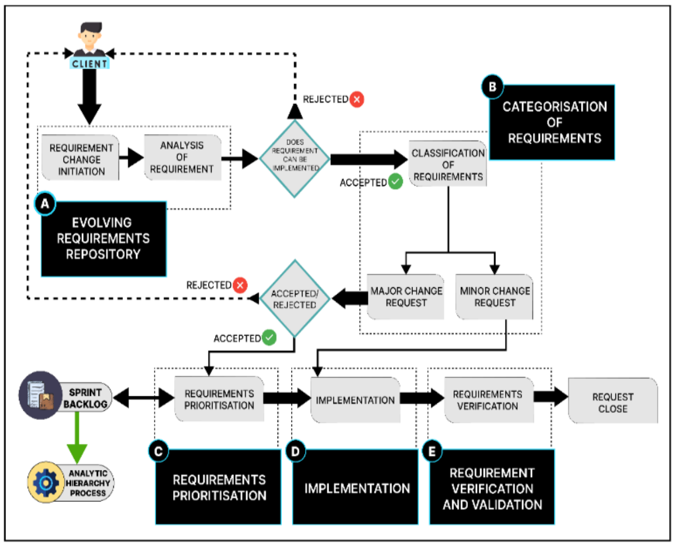
CLIENT
REJECTED
B
CATEGORISATION OF REQUIREMENTS
REQUIREMENT CHANGE INITIATION
ANALYSIS OF REQUIREMENT
DOES REQUIREMENT CAN BE IMPLEMENTED
ACCEPTED
CLASSIFICATION OF REQUIREMENTS
A
EVOLVING REQUIREMENTS REPOSITORY
REJECTED
ACCEPTED/ REJECTED
MAJOR CHANGE REQUEST
MINOR CHANGE REQUEST
ACCEPTED
SPRINT BACKLOG
REQUIREMENTS PRIORITISATION
IMPLEMENTATION
REQUIREMENTS VERIFICATION
REQUEST CLOSE
ANALYTIC HIERARCHY PROCESS
C
REQUIREMENTS PRIORITISATION
D
IMPLEMENTATION
E
REQUIREMENT VERIFICATION AND VALIDATION

Figure 4. A Framework of Change Requirement Management Process in Agile

If the request is denied, the process starts again from the beginning, which means it is sent back to the client. In the alternative, after approval has been given, it moves on to the "Categorization of Requirement." At this point, the requirements are classified as either "Major Change Request" or "Minor Change Request," depending on the scope of the change that is being requested. If the small modification need is satisfied, then the "Implementation" stage may be considered successfully completed. In the event that the proposal is not accepted, the programme will have to begin from the very beginning, which means returning to the "Requirement Change Initiation" stage performed by the client. After being approved, the next step is the Requirement Prioritization which stores the proceeds into the sprint backlog in which the needs are ranked in order of importance according to the preferences determined by the Analytic Hierarchy Process (AHP).

A Sprint backlog is a prioritized list of all work items envisioned for the programme and might include technical improvement, is where requirements sit in agile software development [29]. In the initial iteration, the top-priority project backlog needs will be taken into consideration for execution [31]. With its wealth of expertise, AHP organizes the prerequisites in a methodical manner according to the order of priority. Continuous requirement prioritization is required for efficiently handling changing needs in agile techniques. This approach also has a significant influence on the value delivered by software [15]. After this step, the order advances to the grade of "Implementation," whether change is applied. This further proceeds to the level of “Requirement Verification," in which any possible glitches are resolved. The implementation that was desired has now being carried out bringing the program to a conclusion.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
DOI: https://doi.org/10.17762/ijritcc.v11i10.8509
***Article Received:** 29 July 2023 **Revised:** 20 September 2023 **Accepted:** 10 October 2023*

IV. ANALYTIC HIERARCHY PROCESS

A framework for making decisions, to requirement engineering with Agile methodology is possible. AHP is a helpful tool for ranking needs and making judgements based on a variety of factors. It entails decomposing complicated judgements into a hierarchy of criteria and sub-criteria, assessing and weighing each criterion in accordance with its significance. AHP may be used in Agile approach to order requirements and guarantee that the most crucial features are provided first [11]. Early on in the development phase, this might assist teams in concentrating on providing value to the client. AHP may also be used to assess user stories and choose which ones should be given priority for the upcoming sprint or iteration. It lets us do a detailed analysis of the factors that led to a ranking, making the most of the different skills and knowledge of the people who made the decision [19]. This code may be modified and used into an Agile development process to enhance decision-making over which needs should be addressed first based on their relative relevance and influence on system performance. Prioritizing requirements for their upcoming iteration are a task for an Agile software development team.
Certainly, here's a mathematical expression for calculating the final priority score for a specific requirement i in the context of the given algorithm. Let

- n be the number of requirements.
- m be the number of criteria.
- criteriaWeights be an array storing the weights for each criterion.
- reqWeights be a matrix storing the weights for each requirement and criterion.
- reqPriorities be an array storing the final priority scores for each requirement.

Algorithm: Requirement Prioritization using Analytic Hierarchy Process (AHP)

```
    Input: n - number of criteria, criteria - array of criteria for
    prioritization, m - number of criteria
    Output: reqPriorities - array of final priority scores for each
    requirement
1   Initialize criteriaWeights[1..m] to store weights for each criterion
2   Initialize reqWeights[1..n, 1..m] to store weights for each requirement
    and criterion
3   Initialize reqPriorities[1..n] to store final priority scores for each
    requirement
4   for i = 1 to m do
5       weightVector ← performAHP(criteria)
6       normalizedWeightVector ← normalizeVector(weightVector)
7       criteriaWeights[i] ← normalizedWeightVector[i]
8   end for
9   for i = 1 to n do
10      for j = 1 to m do
11          weightVector ← performAHP([requirement, criteria[j]])
12          normalizedWeightVector←normalizeVector(weightVector)
13          reqWeights[i][j] ← normalizedWeightVector[1]
14      end for
15  end for
16  for i = 1 to n do
17      priorityScore ← 0
18          for j = 1 to m do
19              priorityScore ← priorityScore + reqWeights[i][j] *
                criteriaWeights[j]
20          end for
21      reqPriorities[i] ← priorityScore
22  end for
```

The mathematical expression for calculating the final priority score for requirement i can be represented as:

$$\mathbf{reqPriorities}[\mathbf{i}] = \sum_{j=1}^{m} (\mathbf{reqWeights}[\mathbf{i}][\mathbf{j}] \times \mathbf{criteriaWeights}[\mathbf{j}])$$

In this expression:
• i represents the specific requirement for which you want to calculate the priority score.
• j iterates over all criteria from 1 to m.
• reqWeights[i][j] represents the weight assigned to requirement i with respect to criterion j.
• criteriaWeights[j] represents the weight assigned to criterion j.
The expression calculates the final priority score for the given requirement by summing up the products of the requirement weights and the corresponding criterion weights.
This code may be modified and used into an Agile development process to enhance decision-making over which needs should be addressed first based on their relative relevance and influence on system performance. Prioritizing requirements for their upcoming iteration are a task for an Agile software development team.

They have determined the subsequent standards for prioritization:

- Business Value
- User Experience
- Technical Feasibility
- Risk

Step 1: Creation of a Hierarchy
The team developed a hierarchical structure of the criteria and sub-criteria, as shown below:
1. *Business Value*
    - Revenue Potential
    - Customer Retention

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
DOI: https://doi.org/10.17762/ijritcc.v11i10.8509
***Article Received:** 29 July 2023 **Revised:** 20 September 2023 **Accepted:** 10 October 2023*

2. *User Experience*
    - Ease of Use
    - Personalization
3. *Technical Feasibility*
    - Implementation Effort
    - Compatibility
4. *Risk*
    - Security
    - Compliance

Step 2: Develop a Pairwise Comparison Matrix

Using a scale of 1 to 9 to indicate the relative relevance of one criterion over another, the team created a pairwise comparison matrix for each criterion and sub-criterion. When comparing "Revenue Potential" with "Customer Retention," for instance, the team can conclude that "Revenue Potential" is three times more significant than "Customer Retention," awarding it a score of 3. This is a matrix for the "Business Value" criterion:

TABLE I. PAIRWISE COMPARISON MATRIX

Criteria/Sub-Criteria	Revenue Potential	Customer Retention
Revenue Potential	1	3
Customer Retention	1/3	1

Step 3: Calculate Weights

The team calculated the weights of each criterion and sub-criterion by deriving the eigenvalue and eigenvector of each matrix. The weights for each criterion and sub-criterion are shown in the table below:

TABLE II. CRITERIAN- WISE WEIGHTS

Criteria/Sub-Criteria	Weight
Business Value	0.323
User Experience	0.208
Technical Feasibility	0.208
Risk	0.203
Revenue Potential	0.153
Customer Retention	0.170
Ease of Use	0.091
Personalization	0.117
Implementation Effort	0.139
Compatibility	0.127
Security	0.088
Compliance	0.115

Step 4: Prioritize Requirements

If there are n total needs, then each level of the hierarchy in which this approach is used will do n(n-1)/2 comparisons [16]. An example of requirement prioritizing is shown in the table below:

TABLE III. REQUIREMENTS-WISE SCORE

Requirement	Business Value	User Experience	Technical Feasibility	Risk	Total Score
Requirement A	0.7	0.8	0.5	0.6	0.661
Requirement B	0.6	0.7	0.8	0.5	0.618
Requirement C	0.5	0.9	0.6	0.7	0.645
Requirement D	0.8	0.6	0.7	0.4	0.619
Requirement E	0.6	0.5	0.9	0.8	0.563

By multiplying each criterion's or sub-weight criterions by its corresponding score for each requirement, the team utilized the weights to prioritize the needs. The results were then added to get a final score for each requirement. The team would concentrate on implementing Requirement A first, followed by Requirements B, C, D, and E, in accordance with the priority list. In conclusion, the AHP-based strategy for prioritizing needs in the Agile methodology offers a solution that is methodical and clear. The number of requirements, the criteria used for prioritizing, the weights allocated to each criterion, the weight awarded to each requirement for each criterion, and the final priority score for each need are the variables that need to be declared first in this sample code. The next step is to utilize AHP to give each criterion a weight using a loop. We do pairwise comparisons for each criterion to generate a weight vector, normalize the weight vector, and then store the normalized weights in the criteriaWeights array. Then, we use a nested loop to give each criterion's need a weight.

We carry out pairwise comparisons for every requirement and criteria pair, normalize the weight vector, and save the normalized weight in the reqWeights array. Lastly, we calculate the weighted total of each requirement's weight for each criterion and store the result in the reqPriorities array to get the final priority score for each requirement. We may compare the findings with those obtained by using other prioritization factors or approaches in order to have a deeper comprehension of the usefulness of the AHP-based prioritization method. For the purpose of contrast, we will also consider two other frequent prioritization factors: the projected development time and the cost related with each demand. Now, let's prioritize the requirements based on development time and cost.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
DOI: https://doi.org/10.17762/ijritcc.v11i10.8509
***Article Received:** 29 July 2023 **Revised:** 20 September 2023 **Accepted:** 10 October 2023*

TABLE IV. REQUIREMENTS PRIORITIZED BY DEVELOPMENT TIME AND COST

Rank	Requirement	Development Time	Cost (In Dollars)
1	Requirement C	10	600
2	Requirement A	12	800
3	Requirement D	14	900
4	Requirement B	15	750
5	Requirement E	18	1000

We may design a table that includes the AHP-based prioritization method (Table III) and the development time and cost-based prioritization (Table IV) to visually compare them.

TABLE V. COMPARISON OF AHP PRIORIZATION BY COST AND TIME BASED PRIORITIZATION

Requirement	AHP-Based Rank	Development Time & Cost Rank
Requirement A	1	2
Requirement B	4	4
Requirement C	2	1
Requirement D	3	3
Requirement E	5	5

The comparison between the AHP-based ranking method and the ranking based on development time and cost is shown in the Table V.
AHP-based prioritization has the following benefits:

- Comprehensive decision-making: AHP looks at the decision-making process as a whole, considering different criteria and sub-criteria. This makes it a better way to rank needs in order of importance.
- Systematic method: AHP offers a structured, step-by-step way to make decisions, which can help teams stay on track and keep their eyes on their goals.
- Complex decision-making ability: AHP is ideal for projects with a lot of needs and different things to think about because it can handle complicated decisions with multiple criteria and sub-criteria.
- Consensus-building: AHP helps teams come to an agreement by letting each person add to the pairwise comparisons and general prioritization process. This helps people work together and gets parties on board.

Even though the AHP-based prioritization method has been shown to be a complete and useful tool for making decisions in a variety of situations, including putting requirements in software development in order of importance, it is important to think about its pros and cons and compare it to other prioritization methods.

V. RESULTS AND DISCUSSION

The table presents the needs that have been prioritized based on the criterion and sub-criteria weights that were generated in the processes that came before. The team used the weights to determine how much of a score to assign to each criterion, and then they added together all of those scores to determine the final total score for each criterion. The need with the highest ranking is A, which has a total score of 0.661. This requirement has a high score in the Business Value criteria, which is followed by the User Experience criterion and then the Risk criterion. This suggests that Requirement A is very useful to the company, provides a satisfying experience for the end user, and is not fraught with any significant hazards. The overall score for Requirement C is 0.645, making it the requirement with the second-highest score. It was given a high score in the User Experience and Risk criterion, which indicates that it provides a positive experience for users and has few dangers connected with it. On the other hand, in comparison to Requirement A, it was given a score that was lower for the Business Value criteria. A total score of 0.619 was awarded to Requirement D, which placed it in the third position. It achieved good scores in the Business Value and Technical Feasibility criterion, but a lower score in the Risk criterion.
This suggests that Requirement D is both technically achievable and very useful to the company, despite the fact that it may include certain potential dangers. Following that is Requirement B, which received a total score of 0.618. It received high scores in both the Technical Feasibility and User Experience categories, which indicates that it is very viable and provides a satisfying user experience. On the other hand, it scored poorly in terms of the criterion of Business Value and Risk. Last but not least, the score for Requirement E is the lowest of all the criteria, with a total score of 0.563. Nevertheless, it scored lower than expected in the other criteria, even though it obtained a high score in the Technological Feasibility criterion, which indicates that it is extremely viable. The comparison of the AHP-based prioritization method (Table III) with the development time and cost-based prioritization method (Table IV) is shown in Table V.
The findings show how different prioritization approaches rank requirements. Due to its comprehensiveness, the AHP-based prioritization method is better. AHP-based prioritization places Requirement A first, followed by C, D, B, and E. Requirement C is the most significant in the development time and cost-based prioritization, followed by A, D, B, and E. The AHP-based prioritization method's ability to weigh user wants, technological restrictions, and other considerations in assessing each requirement's relevance is a major benefit. This comprehensive approach balances project goals and ensures that all project factors are addressed when prioritizing requirements. The comparison time and cost-based

prioritization method, on the other hand, focuses largely on the effectiveness of achieving the requirements, disregarding solely the development time and cost. This strategy may optimize resource allocation, but it may not reflect the project's aims and objectives, leading to poor judgments. The AHP-based method also lets stakeholders assess considerations, tailoring the prioritization process to the project's requirements and goals. This flexibility makes the AHP-based prioritization method more applicable to many projects. The findings and analysis show that the AHP-based prioritization method is more thorough and robust for requirement prioritization.

VI. CONCLUSION

In order to ensure the success of software development projects, a framework for requirement management must be developed. Agile methodologies are excellent at handling shifting requirements and incremental software delivery, but they also necessitate efficient requirement management to guarantee that the product satisfies stakeholders' demands and adds value to the company. The Analytic Hierarchy Process (AHP) is used in the framework we've discussed here to rank needs according to their relative urgency and priority. Agile teams may make educated judgements and prioritize the most important needs first with the support of the AHP-based methodology, which offers a structured and systematic means of assessing requirements and allocating priority.

This research evaluated AHP-based and development time and cost-based requirement prioritization methodologies. While there was some alignment between the methods, Requirement A and Requirement C were ranked differently.

After thorough evaluation, the AHP-based prioritization method outperforms the development time and cost method. AHP-based prioritization considers user demands, technological restrictions, and other variables to better analyze each requirement's priority. This comprehensive approach helps project stakeholders prioritize requirements by considering all areas of the project. The development time and cost-based prioritization method prioritizes requirements based only on efficiency. This strategy optimizes resource allocation; however, it may not represent the project's aims and objectives, leading to poor judgments. Given these factors, the AHP-based prioritization method seems stronger and more thorough for requirement prioritization. It may assist project stakeholders make educated choices that consider all elements of the project, ensuring that critical requirements are satisfied and resources are allocated properly.

In conclusion, the AHP-based prioritization method considers many project success variables while prioritizing requirements. Stakeholders may make better choices and achieve a better project end by using the AHP-based method, ensuring that important requirements are addressed and resources are effectively allocated.

ACKNOWLEDGMENT

This work is acknowledged under Integral University manuscript No. IU/R&D/2023-MCN0001884.

REFERENCES

[1] R. Kasauli, G. Liebel, E. Knauss, S. Gopakumar, and B. Kanagwa, “Requirements Engineering Challenges in Large-Scale Agile System Development,” REFSQ 2018, pp. 6–8, 2018.

[2] A. Perini, A. Susi, and P. Avesani, “A machine learning approach to software requirements prioritization,” IEEE Transactions on Software Engineering, vol. 39, no. 4, pp. 445–461, 2013.

[3] W. Brace, V. Cheutet, A framework to support requirements analysis in engineering design. Journal of Engineering Design, 23(12) 2012, 873-901.

[4] Kautz, K. (2010) Participatory Design Activities and Agile Software Development. IFIP Working Conference on Human Benefit through the Diffusion of Information Systems Design Science Research, Perth, 30 March-1 April 2010, 303-316.

[5] D. Baxter, J. Gao, K. Case et al., A framework to integrate design knowledge reuse and requirements management in engineering design. Robotics and Computer-Integrated Manufacturing, 24(2008), 585- 593.

[6] Jayatilleke, S. and Lai, R. (2018). A systematic review of requirements change management. Information and Software Technology, 93:163–185.

[7] Minhas, N.M., Qurat-ul-Ain, Zafar-ul-Islam and Zulfiqar, A. (2014) An Improved Framework for Requirement Change Management in Global Software Development. Journal of Software Engineering and Applications, 7, 779-790.

[8] Quesf, A., 2010, Requirements Engineering in Agile Software Development, Journal of Emerging Technologies in Web Intelligence, Vol.2, No.3.

[9] Terpstra, E., Daneva, M., Wang, C.: Agile practitioners’ understanding of security requirements: insights from a grounded theory analysis. In: 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW), pp. 439–442. IEEE (2017).

[10] W. Puarungroj, N. Boonsirisumpun, S. Phromkhot and N. Puarungroj, "Dealing with Change in Software Development: A Challenge for Requirements Engineering," 2018 3rd Technology Innovation Management and Engineering Science International Conference (TIMES-iCON), Bangkok, Thailand, 2018, pp. 1-5.

[11] Sharma, A., & Bawa, R. K. (2016). A framework for agile development method selection using modified PROMETHEE with Analytic Hierarchy Process. International Journal of Computer Science and Information Security, 14(8), 846.

[12] Abusaeed, S., Khan, S. U. R., & Mashkoor, A. (2022). A Fuzzy AHP-based approach for prioritization of cost overhead factors in agile software development. Applied Soft Computing, 109977

[13] Shehzadi, Z., Azam, F., Anwar, M. W., & Qasim, I. (2019, August). A novel framework for change requirement management (CRM) in agile software development in Proceedings of the 9th

International Conference on information communication and management (pp. 22-26).

[14] Muqeem, M., Sultan, A., Nazeer, J., Farooqui, M. F., & Alam, A. (2022). Selection of requirement elicitation techniques: a neural network-based approach. International Journal of Advanced Computer Science and Applications, 13(1).

[15] Al-Ta'ani, R. H., & Razali, R. (2016). A framework for requirements prioritisation process in an agile software development environment: empirical study. International Journal on Advanced Science, Engineering and Information Technology, 6(6), 846-856.

[16] P. Berander and A. Andrews, "Requirements prioritization," Engineering and managing software requirements, vol. 11, no. 1, pp. 79–101,2005

[17] Minhas, N. M., & Zulfiqar, A. (2014). An improved framework for requirement change management in global software development. Journal of Software Engineering and Applications, 2014.

[18] Oyeyipo, E. O., & Mueller, C. J. (2011). Requirements Management in an Agile-Scrum.

[19] Kifetew, F., Munante, D., Perini, A., Susi, A., Siena, A., & Busetta, P. (2017, September). DMGame: a gamified collaborative requirements prioritisation tool. In 2017 IEEE 25th International Requirements Engineering Conference (RE) (pp. 468-469). IEEE.

[20] Wang, X., Zhao, L., Wang, Y., & Sun, J. (2014). The role of requirements engineering practices in agile development: an empirical study. In Requirements Engineering: First Asia Pacific Requirements Engineering Symposium, APRES 2014, Auckland, New Zealand, April 28-29, 2014. Proceedings (pp. 195-209). Springer Berlin Heidelberg.

[21] Alam, S., Aziz, M. A., Bhatti, S. N., & Alam, S. (2016). Analysis of Requirement Engineering Techniques in Agile Development Method. International Journal of Computer Science and Information Security, 14(11), 889.

[22] Chen, C., Chang, Y. 2011. Prioritizing 5S Activities by Kano Model for a Semiconductor Water Fabrication. Proceedings of the 4th WSEAS Int'l Conf. on Manufacturing Engineering, Quality and Production Systems, Barcelona, Spain, 15-17 Sept:52-56.

[23] Shameem, M., Kumar, C., Chandra, B., & Khan, A. A. (2017, December). Systematic review of success factors for scaling agile methods in global software development environment: A client-vendor perspective. In 2017 24th Asia-Pacific Software Engineering Conference Workshops (APSECW) (pp. 17-24). IEEE.

[24] Siponen, M., Baskerville, R., & Kuivalainen, T. (2005, January). Integrating security into agile development methods. In Proceedings of the 38th Annual Hawaii International Conference on System Sciences (pp. 185a-185a). IEEE.

[25] Mahmud, I., & Veneziano, V. (2011, December). Mind-mapping: An effective technique to facilitate requirements engineering in agile software development. In 14th International Conference on Computer and Information Technology (ICCIT 2011) (pp. 157-162). IEEE.

[26] Nurmuliani, N., Zowghi, D., & Powell, S. (2004, April). Analysis of requirements volatility during software development life cycle. In 2004 Australian Software Engineering Conference. Proceedings. (pp. 28-37). IEEE.

[27] Kamal, T., Zhang, Q., Akbar, M. A., Shafiq, M., Gumaei, A., & Alsanad, A. (2020). Identification and prioritization of agile requirements change management success factors in the domain of global software development. IEEE Access, 8, 44714-44726.

[28] Nadeem, R., Amir Latif, R. M., Hussain, K., Jhanjhi, N. Z., & Humayun, M. (2022). A flexible framework for requirement management (FFRM) from software architecture toward distributed agile framework. Open Computer Science, 12(1), 364-377.

[29] Heikkilä, V. T., Damian, D., Lassenius, C., & Paasivaara, M. (2015, August). A mapping study on requirements engineering in agile software development. In 2015 41st Euromicro conference on software engineering and advanced applications (pp. 199-207). IEEE.

[30] Alsaqaf, W., Daneva, M., & Wieringa, R. (2017, November). Agile quality requirements engineering challenges: First results from a case study. In 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) (pp. 454-459). IEEE.

[31] AL-Ta'ani, R. H., & Razali, R. (2013). Prioritizing requirements in agile development: A conceptual framework. Procedia Technology, 11, 733-739.