CHAPTER – 3

STAKEHOLDER REQUIREMENT CLUSTERING ALGORITHM (SRCA) FOR REQUIREMENT CLUSTERING

3.1 INTRODUCTION

The software requirement process incorporates the tasks of elicitation, analyzing, and specifying the behavioral and functional properties of a system, which denotes one of the major critical stages of the software development life cycle (Castro-Herrera, et al. 2009). The basic concern of a software model is, "How it meets the requirements for that it was built?" In general, the requirement engineering is the process of identifying the stakeholder’s requirements, purpose and significance of the system for software development.

The success or disaster of a system development work is based on the quality of the requirements. The methods included during the requirement elicitation influence the requirements quality. Hence, elicitation provides the need for the requirement engineers to interact with the stakeholders to collect the requirements that are needed to build the software. The success of the process is based on identifying the relevant stakeholders (customers, decision makers, developers and end users) and discovering their needs.

The stakeholders are usually from diverse backgrounds and have diverse goals. Hence, it is essential to include all the stakeholders during information collection; otherwise certain perspectives are never exposed. Various difficulties occur during the requirements elicitation phase that prevent in achieving the goal. It is vital to analyze all the relevant factors to result in better understanding of the system constraints, application domain, stakeholders and business needs.

This thesis is focused on coordinating and handling the large number of stakeholders’ requests and arranges the successive requirements into meaningful structures. The necessary features are extracted based on the fuzzy classification algorithm. Here, the Bayesian network formulation is utilized to negotiate the stakeholders as the nodes. The stakeholders are clustered based on the similarity among them. The similarity measure between the stakeholders is calculated based on the Jaccard similarity
equation. The relevant stakeholder’s clusters are used for software model generation. The flow of the proposed method is illustrated in figure 3.1.

Figure 3.1 Flow of the proposed method
3.2 REQUIREMENTS OF THE SOFTWARE PROCESS

During the implementation of the software process, it is necessary to define the right tools, activities, roles etc., in the process. The requirements on the process can be taken in two ways:

i. **Descriptively (how the process is currently performed):** It is also represented as base lining. It is a key activity during the initiation process

ii. **Prescriptively (how the process should be performed):** It is used to improve the previously defined process.

The critical feature of the software development cycle is receiving the appropriate requirements. However, most companies come up short; studies point to a failure rate of more than 60% for IT projects, with poor requirements as one of the top five reasons. Moreover, when requirements are stated early in the lifecycle, 80% of the functionalities are comparatively unwanted by the users and 45% of these structures are never used. These difficulties result in any one or more of the following:

1. Costly rework
2. Negotiated product quality
3. Overdue delivery on present projects and start of new ones
4. Lost credibility
5. Lowered revenue opportunities.

A small mistake in the requirement phase can be augmented to a major flaw during the development of the software product. Some of the typical requirement issues include:

- Incomplete or inadequate requirements that don’t clearly specify the need of the users
- Imprecise and ambiguous requirements that lead to failed scope, project rework, and analysis paralysis
- Struggling with what are necessities and who organizes them
- Underrating the value of spending time on requirements; providing only a small window to do the work
- Approving requirements practices to work efficiently with agile projects
- Requirements that don’t take into account the business needs and resource constraints
Banking on requirements tools that don’t help or trying to determine what requirements activities should be automated.

3.3 REQUIREMENT ENGINEERING PROCESS

The requirement engineering development process includes the feasibility study, requirement elicitation and analysis, requirement specification and requirement validation. The purpose of these steps is to identify the stakeholders, and their needs in the form that is cooperative to analyze and validate the documented requirement.

![Figure 3.2 Requirement Engineering Development Process](image)

**Figure 3.2 Requirement Engineering Development Process**

Figure 3.2 depicts the requirement engineering development process. It initiates from the feasibility study of the system.

3.3.1 FEASIBILITY STUDY

It is the initial step before continuing towards the requirement engineering process. This study is completed on the basis of the following factors:

- Technology and system feasibility
- Economic feasibility
- Schedule feasibility
• Operational feasibility and
• Legal feasibility

All the outcomes of the feasibility study are documented in the feasibility report.

3.3.2 REQUIREMENT ELICITATION

The Requirement elicitation is a major step in the requirement engineering process, which cannot be separated from the succeeding activities. It is utilized to discover the requirement of customers, users, developers, decision makers, and other stakeholders. Moreover, it is used to identify the system boundaries and specify the behavioral and functional properties of a system in order to achieve the objective for which the planned system is established.

Figure 3.3 Components of Requirement Elicitation

Figure 3.3 depicts the components of the requirement elicitation. The most common sources for this phase are:

• Customers, End users
• Customer requirements specifications
• Users of pre-existing systems
• Users of the new system
• Documentation related to pre-existing systems
Various difficulties are encountered to achieve the requirement elicitation objective. Hence, it is important for the analyst to consider the whole relevant factors in order to understand the application domain, business needs, system constraints, and stakeholders.

### 3.3.3 REQUIREMENT ANALYSIS

This stage analyzes and models the requirements that are taken in the requirement elicitation process. Requirement elicitation process is an input to requirement analysis phase and the output of this stage is reliable and comprehensive set of requirements. It is used to detect the missing and inconsistence requirements, which are delivered by the stakeholders for the determination of consistency, completeness, necessity and feasibility. Various techniques are used for requirement analysis, such as Joint Application Development (JAD) sessions, Prioritization, and Modeling. Figure 3.4 shows the interaction between the requirement elicitation and requirement analysis.

![Interaction between the requirement elicitation and analysis](image)

**Figure 3.4 Interaction between the requirement elicitation and analysis**

### 3.3.4 REQUIREMENT DOCUMENTATION

After a requirement is elicited, analyzed and modeled it must be documented in clear and unambiguous terms. Requirement analysis is an input given to the requirement documentation. The output of this process is a well-defined and structured specification. This documentation is used to interrelate among the various stakeholders’ requirements. A clear and understandable requirement document should be correct, complete, consistent
and feasible because it is used as a standard model for calculating the consequent progression of the system. The most common approaches used for requirement specification are:

- Graphical notation
- Design description language
- Natural language
- Formal specification
- Requirements specification language
- Structured natural language

### 3.3.5 REQUIREMENT VERIFICATION AND VALIDATION

This stage is used to simplify that the requirement documents are consistent, unambiguous and complete and that the stakeholders are satisfied with the concluding requirement specification. This step is used to evaluate that each stage of development process meets the standards with the process and the product meets user desires. Various methods such as requirement checklist, requirement inspection, and requirement testing are used to identify the defect to recover the quality of a necessity and to ensure that assured criteria meet the information elicited and indicated.

### 3.3.6 REQUIREMENT MANAGEMENT

This stage is used to organize, identify and track the complete changing requirements in software as well as the impact of these variations. It is a continuous procedure in order to ensure that the association meets the stakeholders’ expectation.

### 3.4 STAKEHOLDERS DATA EVALUATION

Any type of data, collected from heterogeneous stakeholders is greatly susceptible to be missed, inconsistent and noisy due to their large size. Low quality data will result in low quality software model. Hence, the data should be preprocessed before it is applied for classification. There are a various data preprocessing techniques such as data cleaning, data integration, data transformation and data reduction.

- *Data Cleaning:* It is also called Data Scrubbing, which deals with identifying and removing the errors and inconsistencies from the requirements to improve
the data quality. Data quality problems are present in data collections such as files and databases due to the data entry misspellings, invalid data or missing information. In order to present accurate and consistent requirements, the consolidation of diverse data in different representations and elimination of duplicate information becomes necessary.

- **Data Integration**: It is the problem of merging data residing at different sources, which provide the user with an integrated view of these data. The problem of manipulating the data integration systems is significant in current real world applications and it is categorized by a number of issues that are motivating from a theoretical point of view. It merges the data from different stakeholders into a coherent data store.

- **Data Transformation**: Normalization can be applied for data transformation. Normalization improves the accuracy and efficiency of the requirement engineering.

- **Data Reduction**: It reduces the data size by combining or eliminating the redundant features. It is not mutually exclusive and it can work together. Data cleaning may include transformations to precise the wrong data such as converting all the entries for a date field to a standard format.

### 3.5 FEATURE EVALUATION AND SELECTION

The stakeholder’s requirements are considered as the features and it is evaluated for selecting the necessary features for software model generations. Feature selection algorithms are designed with various evaluation measures and it is categorized into three models:

1. Filter Model
2. Wrapper Model
3. Hybrid Model

#### 3.5.1 FILTER APPROACH

The filter model depends on the general features of the requirements to evaluate and select the feature subsets of the stakeholder’s requirements without including any algorithm. For a given data set A, the algorithm begins the search from a subset $S_0$ and searches through the feature space based on a searching strategy. Each subset $S$ is
validated by an independent criteria I and compared with the previous optimal one. If it is identified to be optimal, it is observed as the present best subset. The algorithm yields the final current optimal subset $S_{\text{best}}$ as the final output by changing the search policies and evaluation criteria. The algorithm for the filter approach is described as follows:

**Algorithm 1: Filter Approach**

**Input:** $A (E_0, E_1, \ldots, E_n)$ is the training data set with N features

$S_0$ is the subset from which to initiate the search

$\alpha$ is the stopping criteria

**Output:** $S_{\text{best}}$ is the optimal subset

1: Begin
2: Initialize $S_{\text{best}} = S_0$
3: $\delta_{\text{best}} = \text{evaluate} (S_0, A, I)$; // Evaluates $S_0$ by an independent criteria I
4: do begin
5: $S = \text{generate} (A)$ // generate a subset for evaluation
6: $\delta = \text{evaluate} (S, A, I)$ // evaluate the current subset $S$ by I
7: if($\delta$ is better than $\delta_{\text{best}}$
8: $\delta_{\text{best}} = \delta$
9: $S_{\text{best}} = S$
10: End until ($\alpha$ is reached)
11: return $S_{\text{best}}$
12: End

Various individual algorithms within the filter approach can be designed. Subsequently, the filter approach applies independent validation measures without including the algorithm. It is computationally well-organized.
3.5.2 WRAPPER APPROACH

The wrapper model needs one fixed algorithm P and uses its routine as the evaluation criterion. It searches for features (requirements) that better fit with the algorithm directing to improve the searching performance. But, it also tends to be more computationally expensive than the filter model. It is very similar to the generalized filter algorithm except that it utilizes a predefined algorithm instead of an independent criteria \( I \) for subset evaluation. For each generated subset \( S \) of the requirements, it calculates its goodness by applying the algorithm P to the data with feature subset \( S \) and calculating the quality of selected results. Therefore, different algorithms will yield diverse feature selection results.

Algorithm 2: Wrapper Approach

**Input:** \( A (E_0, E_1, \ldots, E_{n-1}) \) is the training data set with \( N \) features

\( S_0 \) is the subset from which to initiate the search

\( \alpha \) is the stopping criteria

**Output:** \( S_{\text{best}} \) is the optimal subset

1: Begin
2: 
   Initialize \( S_{\text{best}} = S_0 \)
3: 
   \( \delta_{\text{best}} = \text{evaluate} \ (S_0, A, P) \); \ // Evaluates \( S_0 \) by an algorithm \( P \)
4:  
   do begin
5:   
      \( S = \text{generate} \ (A) \) \ // generate a subset for evaluation
6:   
      \( \delta = \text{evaluate} \ (S,A,P) \) \ //evaluate the current subset \( S \) by \( I \)
7: 
   if \( \delta \) is better than \( \delta_{\text{best}} \) 
8: 
   \( \delta_{\text{best}} = \delta \)
9: 
   \( S_{\text{best}} = S \)
10: End until (\( \alpha \) is reached)
11: return \( S_{\text{best}} \)
12: End
Changing the search approaches via the function \( \text{generate} \) \((A)\) and algorithms can result in diverse wrapper algorithms. Since, the algorithms are utilized to control the selection of feature subsets; the wrapper approach inclines to give a superior presentation as feature subsets found are well suited to the predetermined algorithm. Accordingly, it is also more computationally expensive than the filter model.

### 3.5.3 HYBRID APPROACH OF FILTER AND WRAPPER

The hybrid model takes the advantage of the two models by developing the different evaluation criteria in various search stages. The proposed method uses the Hybrid model of filter and wrapper approach for requirements evaluation and selection. The hybrid model has recently been proposed to handle the large stakeholder’s requirements. It makes use of an independent criteria and algorithm to estimate the feature subsets: it uses the independent criteria to choose the optimal subsets for certain cardinality and uses the algorithm to choose the concluding optimal subset of the requirements among the best subsets through different cardinalities.

**Algorithm 3: Hybrid Approach**

**Input:** \( A \) (\( E_0, E_1, \ldots, E_{n-1} \)) is the training data set with \( N \) features

\( S_0 \) is the subset from which to initiate the search

**Output:** \( S_{\text{best}} \) is the optimal subset

1: Begin

2: Initialize \( S_{\text{best}} = S_0 \)

3: \( l_0 = \text{card} \ (S_0) \) \ // calculate the cardinality of \( S_0 \)

4: \( \delta_{\text{best}} = \text{evaluate} \ (S_0, A, I) \) \ // Evaluates \( S_0 \) by an independent criteria \( I \)

5: \( \vartheta_{\text{best}} = \text{evaluate} \ (S_0, A, P) \) \ //Evaluates \( S_0 \) by an algorithm \( P \)

6: for \( l = l_0+1 \) to \( N \) begin

7: for \( i = 0 \) to \( N - l \) begin

8: \( S = S_{\text{best}} \cup \{ F_j \} \)

\ // generate a subset with cardinality \( l \) for evaluation

9: \( \delta = \text{evaluate} \ (S, A, I) \) \ // evaluate the current subset \( S \) by \( I \)
10: if (δ is better than δ_{best})
11: δ_{best} = δ
12: S_{best} = S
13: end
14: θ = evaluate (S'_{best}, A, P) // evaluate S'_{best} by P
15: if (θ is better than θ_{best})
16: S_{best} = S'_{best}
17: θ_{best} = θ
18: else
19: return S_{best}
20: end
21: return S_{best}
22: End

Ultimately, it starts the search from a given subset S₀ and repeats to identify the optimal subsets of requirements at each increasing cardinality. For each round for an optimal subset with cardinality ℓ, it searches through all possible subsets of cardinality ℓ+1 by calculating one feature from the enduring features of requirements. Each newly generated subset S with cardinality ℓ+1 is calculated by an independent criteria I and compared with the previous optimal one.

If S is better, it becomes the current optimal subset S_{best} at level ℓ+1. At the end of each iteration, an algorithm is applied on S_{best} at level ℓ+1 and the quality of the result θ is associated with that from the optimal subset at level ℓ. If S_{best} is better, the algorithm endures to identify the optimal subset of the requirements at the next level; otherwise, it breaks and outputs the current best subset as the concluding optimal subset. The excellence of results from an algorithm provides a natural stopping criterion in the hybrid approach.
3.6 DATA CLASSIFICATION BASED ON FCM

Fuzzy algorithm is the standardized families of clustering algorithms, which is based on the function optimization. The stakeholder’s requirements are needed to classify in order to group the similar stakeholder’s requirements. Fuzzy classification makes it easy for the decision makers to explore their ideas using the classification equations. The Fuzzy C-Means (FCM) algorithm belongs to that family. It tries to identify the fuzzy partitioning of a given set by reducing the objective function,

\[ O_r(S; P, M) = \sum_{i=1}^{X} \sum_{k=1}^{Y} (\mu_{ik})^r G_{ik} \]  

Here, \( P = [\mu_{ik}] \in R_{fcm} \), fuzzy partition matrix of S

\[ m_i = \frac{\sum_{k=1}^{Y} (\mu_{ik})^r s_k}{\sum_{k=1}^{Y} (\mu_{ik})^r} \]  

\( \mu_{ik} \) is defined as follows,

\[ \mu_{ik} = \frac{1}{\sum_{j=1}^{Y} (g_{ijk})^{(r-1)}} \]  

Where, \( g_{ijk} = \sqrt{G_{ik}} \) and \( r \) is called fuzzifier, which controls the fuzziness of the algorithm. The efficiency of the algorithm is greatly depends on the proper selection of the fuzzifier.

During the optimization of the function \( O_r(S; P, M) \), the constraints to be satisfied are as follows:

1. \( \sum_{i=1}^{X} \mu_{ik} = 1 \)
2. \( \mu_{ik} \in [0,1] \)

Generally, FCM normalizes the partition P. If the algorithm is shifted by one half cycle, then the initialization is done on M rather than on P. This kind of iteration is called as alternating optimization (AO). Here, the stopping criteria become\( \| (M_t - M_{t-1}) \| < \epsilon \). Finally, the FCM algorithm results the similar characteristics requirements.

The combination of supervised and unsupervised methods is the recent approach used in machine learning. Such a combination has diverse goals. On the other hand, it can be attempted to improve the classification accuracy of the relevant stakeholder’s requirements. It influences the classifier using the information coming from the
unsupervised process. The stakeholders are formulated and distributed in the Bayesian network, which uses the outcomes of the supervised classification process. It can be used as a way to combine the large amount of unlabeled stakeholder’s requirements in the supervised classification process.

3.7 BAYESIAN NETWORK FORMULATION

Bayesian networks are a well-known artificial intelligence technique suitable in handling decision-making problems involving ambiguity. In the proposed work, the stakeholder’s are considered as nodes, which are incorporated into the design of Bayesian Network. The optimal clusters are identified with the accurate stakeholder’s requirements in order to generate the effective software model. Bayesian network models have been found to be very robust in the sense that small alterations in the model do not affect the performance of the system dramatically. It includes the merits of having rich semantics and it can be interpreted by the stakeholders without the knowledge on statistics. From the user’s perspective, it provides a natural structure for relevance analysis and prediction procedure. Hence, a Bayesian network can be used in requirement confirmation which estimates whether a requirement specification attains enough quality to be considered as a software project baseline. Bayesian network basics have been developed through interactions with experts and various sources like standards and reports to inform whether the iterations can be stopped to describe the Software Requirement Specification (SRS) which is shown in figure 3.5.

To assess the goodness of the SRS, the fundamentals of the network uses the following variables:

- **Stakeholder’s expertise**: It is the degree of knowledge with expertise regarding the particular tasks about requirement engineering. The more experienced stakeholder may commit lesser errors.

- **Domain expertise**: The level of expertise is achieved by the development team regarding the project domain. If the developers and the stakeholders handle the same terminology, then the interaction will be more effective.

- **Reused requirements**: If the total number of requirements taken from the reusable libraries is high, then the entire specification of the requirements may not requisite new iterations.
• **Unexpected dependencies**: Generally, an unexpected dependency among the requirements includes a new revision of the specification of the requirements.

• **Specificity**: It is the number of requirements distributed with the same meaning for all stakeholders. A higher value of specificity infers lesser revision and a smaller process of negotiation in order to obtain a commitment.

• **Unclear cost/benefit**: The developers or stakeholder requirements cannot be clearly enumerated.

• **Degree of commitment**: A number of requirements that needs a negotiation in order to be accepted.

• **Homogeneity of the description**: A best SRS should be defined using the same level of the factor. If SRS contains no homogeneity, then the document needs to be revised.

• **Requirement completeness**: It indicates whether all the necessary requirements are specified and elicited.

• **Requirement variability**: It signifies whether the requirements have suffered changes. When a requirement specification alters, it requests an additional revision.

• **Degree of revision**: It is a value predicted by the fundamental Bayesian network.
The representation of requisites, models the dependence associations among the variables. Note that every node in the network is attached in its own conditional probability distribution given its parent nodes. In the qualitative way of requisites, unexpected dependencies, stakeholder expertise, domain expertise, specificity and reused requisites are not affected by any other variables. The degree of commitment and unclear cost benefit are associated, as the degree of commitment increases the level of specificity is reduced. If suppose the stakeholders have little knowledge in the requirement engineering processes then it is more probable to lead the requirements, which are imprecise in terms of benefits or cost.

The requirement homogeneity and completeness of the explanation are subjective by the knowledge of software and requirement engineers in the field of the project and by stakeholders in the processes or tasks of requirement engineering. If knowledge is great, then the specification will be broad and similar because developers have been able to define the requirements with the same level of detail and have exposed all requirements.

The requirement variability denotes the number of fluctuating requirements. The changes made in the requirements will be more probable to occur based on the following conditions:

- if unexpected dependencies are exposed
- if there are requirements that do not add any value to the end software
- if there are missing requirements
- if requirements have to be negotiated

### 3.8 CLUSTERING OF STAKEHOLDERS

There are various similarity measures used, but only efficient measures are utilized to use with the data. Some of the examples of the similarity measure are Jaccard coefficient, Cosine similarity, adjusted cosine similarity, Dice coefficient, Correlation based similarity, Extended Jaccard coefficient, and Mean squared difference. In the proposed system, the similarities among the stakeholders are determined based on the Jaccard similarity measure.

In Jaccard similarity, the similarity among the stakeholder’s ratings is calculated by the distance between the stakeholder’s ratings. A smaller estimated distance suggests a higher similarity among the ratings. For example, if there are two ratings from
stakeholders such as \( r_i \) and \( r_j \) for \( i \) and \( j \) for the same requirement, then the distance between them is given as

\[
d = |r_i - r_j|
\]  

(4)

The Jaccard coefficient is used to measure the data consisting of asymmetric binary attributes. It includes two values 1 and 0. 1 indicates the relevant measures and 0 indicates the irrelevant measures used to analyze the stakeholder requirements. The Jaccard coefficient is given as

\[
J = \frac{R_{11}}{R_{01} + R_{10} + R_{11}}
\]  

(7)

Here, \( R_{11} \) denotes the total number of attributes that both the objects have value is 1. \( R_{01} \) denotes the total number of attributes of one object is 0 and the object is 1. \( R_{10} \) denotes the total number of attributes of one object is 1 and the object is 0.

**Algorithm 4: Stake Requirement Clustering Algorithm (SRCA)**

Input: Classified Data (CR)

Output: Clustered requirement list

\[\text{SR}_n = \text{Parse(CR)};\]

\[\text{SS}_n = \emptyset;\]

\[\text{Sim}[n][n] = \emptyset;\]

\[\text{J}[n][n] = \emptyset;\]

\[i = 0;\]

while \( i < n \) then

\[\text{SS}_i = \text{WordNet (SS}_i);\]

\[i++;\]

end while

for \( i = 0 \ldots n-1 \) then
for j =0…n-1 then

    Sim[i][j]=Similarity(SS_i, SS_j);

    J[i][j]=Jaccard(SS_i, SS_j);

end for j

end for i

ClusterId=0;

for i=0…n-1 then

    if (SR_i. ClusterId==null)

        ClusterId++;

    for j =0…n-1 then

        if (Sim[i][j]>Ts&& J[i][j]<T_ja&& SR_j.ClusterId=null)

            SR_j.setClusterId(Clusterid);

        end for j

    end for i

return SR;

Where, SR denotes the stakeholder requirements, SS is the Synset for requirement, Ts is the similarity threshold, and J[n][n] is the Jaccard coefficient. Based on the SRCA algorithm, the required software model is generated. The resulting cluster of stakeholders’ requirements provides accurate and well-organized requirements to model the software.

3.9 SUMMARY

The software requirements are collected from different stakeholders and evaluated based on the hybrid approach of filter and wrapper approach. In the proposed method, the supervised and unsupervised learning approaches are combined to result an accurate software model generation. The proposed SRCA algorithm is used to group the similar
characteristics stakeholders. Moreover, Bayesian Network is used to formulate the stakeholder’s network to show the effective communication among the stakeholders. Here, the Jaccard Similarity measure is used to evaluate the similarity among the stakeholders to cluster the optimal groupings, which is finally used to generate a software model.