A Framework of Software Requirements Quality Analysis System using Case-Based Reasoning and Neural Network

Hajar Mat Jani
College of Information Technology
Universiti Tenaga Nasional
Kajang, Malaysia
hajar@uniten.edu.my

ABM Tariqul Islam
College of Information Technology
Universiti Tenaga Nasional
Kajang, Malaysia
tariquex@yahoo.com

***Abstract*—In this paper, we propose a new approach to Software Requirements Specifications (SRS) or software requirements quality analysis process. We apply the Software Quality Assurance (SQA) audit technique in determining whether or not the required quality standards within the requirements specifications phase are being followed closely. Quality analysis of the SRS is performed to ensure that the software requirements among others are complete, consistent, correct, modifiable, ranked, traceable, unambiguous, and understandable. Here, a new approach that combines case-based reasoning (CBR) and neural network techniques in analyzing SRS quality is proposed. This approach is used in improving the process of analyzing the quality of a given SRS document for a specific project. The CBR technique is used to evaluate the requirements quality by referring to previously stored software requirements quality analysis cases (past experiences). CBR is an artificial intelligence technique that reasons by remembering previously experienced cases, and this technique will speed up the quality analysis process. Neural Network (Artificial Neural Network or ANN) is the type of information processing paradigm that is inspired by the way biological nervous systems (brain) process information. Neural network technique works well with CBR because it also uses examples to solve problems. The new approach proposed in this research aims at enhancing and improving existing methods in analyzing SRS quality. A framework of the proposed approach is the main outcome of this research study.**

***Index Terms*—Artificial neural network (ANN), case-based reasoning (CBR), quality analysis, software requirements specifications (SRS)**

I. Introduction

The Software Requirements Specifications (SRS) document states all those functions and capabilities a software system must provide, as well as states any required constraints by which the system must abide. By definition, a requirement is an objective that must be met, while a specification describes how the objective is going to be accomplished. In other words, a specification document describes how specific tasks are supposed to be done. A very critical part of the quality assurance role is proactive involvement during the system's requirements specifications phase. In the past, several studies have determined that companies will have to pay less to fix problems that are found early in any Software Development Life Cycle (SDLC) [1].

The requirements specifications phase of the SDLC is the main focus of this research study because if errors are found during this phase, then the cost of correcting the errors is going to be less.

In analyzing the quality of the prepared SRS, the Software Quality Assurance (SQA) audit technique is used to determine whether or not the required standards within the requirements specifications phase are being followed closely. The main objective of performing a quality analysis of the SRS is to ensure that the software requirements among others are complete, consistent, correct, modifiable, ranked, traceable, unambiguous, and understandable.

CBR is an artificial intelligence technique that reasons by remembering and recalling previously experienced cases, and it is quite similar to the way human beings reason when they are faced with new problems. It also focuses on how people generate new solutions to newly encountered problems based on their past experiences and at the same time learn new skills. This technique will speed up the SRS quality analysis process since many tasks that need to be performed are quite similar to tasks that have been experienced in the past and the solutions to these similar tasks can be retrieved from a knowledge base or case base that holds all past problems and solutions. Basically, the four main steps within a CBR cycle are *Retrieve*, *Reuse*, *Revise*, and *Retain*.

Neural Network or Artificial Neural Network (ANN) is the type of information processing paradigm that is inspired by the way biological nervous systems (brain) process information [2].

Based on several past research, CBR is best used under the following conditions [3][4]:

- There is no underlying model of the application domain.
- There exist exceptions and new cases or problems to be handled within the domain.
- Most of the cases or problems reoccur again and again.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 15,2025 at 04:25:49 UTC from IEEE Xplore. Restrictions apply.

- It is discovered that there is significant advantage in reusing and adapting past solutions to past experiences.
- Relevant past cases are available and are stored in a case base for future retrieval and usage.

Neural networks on the other hand have the following advantages:

- A neural network is good in handling unstructured problems with no specific models, which resembles problems suitable for CBR.
- When a portion or component of the neural network fails, it can still continue without any problem because of their parallel nature.
- A neural network automatically learns on its own and does not need to be reprogrammed.
- It can be implemented in almost any application without much hassle.

Based on the above reasons and advantages of CBR and ANN, it is decided that both can be combined in improving SRS quality analysis process.

II. Research Objectives

This research study attempts to achieve the following objectives:

- To propose a new approach for analyzing software requirements specifications (SRS) using case-based reasoning (CBR) and neural network techniques that improves SRS quality analysis process.
- To construct a framework of the proposed approach.

III. Related and Previous Works

Several papers have been written on software requirements quality analysis, and a few of them are discussed below.

In a previous work in [5], a simple online quality analysis system that measures the quality of the requirements specifications phase's results of the SDLC was developed. The following gives a list of steps (algorithm) that is used to measure software quality for the online quality analysis system that were applied in [5][6][7]. These steps will also be partially applied in this proposed approach. The only modification is that the ANN technique is used in measuring the similarity level of the new case with all existing cases in the case base. In addition, the weights that are also needed in the calculation will be randomly generated based on some available information about past experiences on SRS quality analysis.

Step 1: Select quality indicators categories to measure each software quality attribute.

Step 2: Select a weight w for each quality indicators category (usually 0 <= w <= 1; depends on the number of quality indicators categories that correspond to a particular software quality attribute).

Step 3: Select a scale of values for quality indicators categories scores (1 – 5, where 5 is the highest).

Step 4: Select minimum and maximum target values for each quality indicator category score (here, the value 3 is set as the minimum, and the value 5 as the maximum).

Step 5: Select minimum and maximum target values for the software quality attribute score (here, the value 3 is set as the minimum, and the value 5 as the maximum).

Step 6: Give each quality indicators category score (entered by the user).

Step 7: Compute a weighted sum.

Step 8: Compare the weighted sum with the preset min-max software quality attribute scoring range.

Step 9: If the weighted sum is outside the min-max scoring range, compare each individual quality indicators category score with the preset min-max criterion score range to direct software improvement activities (all this information will be displayed in the audit report).

The *weighting formulas* for each software quality attribute in the quality measurement framework have the form $w_1QI_1+w_2QI_2+ \ldots + w_nQI_n$, where $w_1,..,w_n$ are weights and $QI_1,\ldots,QI_n$ are quality indicators categories measurements. A weighting formula measures the aggregative effect of weighted quality indicators categories [5].

Table I shows the *Quality Attributes* and the relevant categories of *Quality Indicators* of the listed *Quality Attributes* along with their relationships. This table was also used in our previous works.

TABLE I. SRS Quality Attributes and Their Respective Categories of Quality Indicators

INDICATORS OF QUALITY ATTRIBUTES											
	Quality Attributes										
Categories of Quality Indicators	1. Complete	2. Consistent	3. Correct	4. Modifiable	5. Ranked	6. Testable	7. Traceable	8. Unambiguous	9. Understandable	10. Validatable	11. Verifiable
1. Imperatives	X			X			X	X	X	X	X
2. Continuances	X			X	X	X	X	X	X	X	X
3. Directives	X		X			X		X	X	X	X
4. Options	X					X		X	X	X	
5. Weak Phrases	X		X			X		X	X	X	X
6. Size	X					X		X	X	X	X
7. Text Structure	X	X		X	X		X		X		X
8. Spec. Depth	X	X		X			X		X		X
9. Readability				X		X	X	X	X	X	X

This approach [5] was further improved in another paper [6] that applied *case-based reasoning* (CBR) in analyzing SRS quality so that the analysis is more accurate and more efficient. In this paper [6], the CBR technique is used to evaluate the requirements quality by referring to previously stored software requirements quality analysis cases (past experiences). CBR is an AI technique that reasons by remembering previously experienced cases. Within this research study, the CBR is used to evaluate the requirements information (quality attributes & indicators) provided by the user, and give the corresponding quality analysis results along with a proposed most suitable

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 15,2025 at 04:25:49 UTC from IEEE Xplore. Restrictions apply.

solution to any problems related to the quality of the software requirements provided by the user. In CBR, there are four main steps (CBR cycle), which are *Retrieve*, *Reuse*, *Revise*, and *Retain* (Refer to Fig. 1). The CBR approach was enhanced and implemented to an online SRS quality analysis in [7].

In another previous work [8], fuzzy logic was applied to the *Retrieve* step of the CBR cycle with the intention of making the *Retrieve* step more efficient. Fuzzy logic was used to classify the cases (previous experiences or examples) into several main categories that assisted in enhancing the overall performance of the CBR's *Retrieve* step.

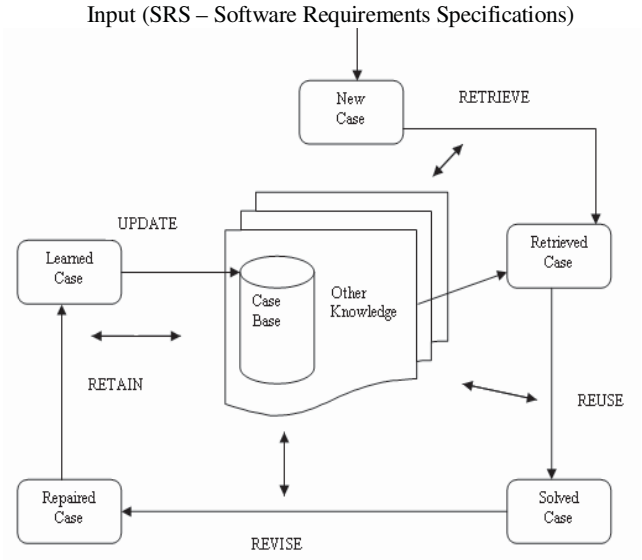

Fig. 1. The CBR cycle [6][7]

This research study intends to further improve the analysis approaches used in [6][7][8]. In this research, the neural network (artificial neural network) soft computing technique is going to be combined with the CBR technique in order to improve the performance of the proposed approach. This hybrid approach is a new approach and has never been applied to SRS quality analysis. As mentioned earlier, Artificial Neural Network (ANN) is the type of information processing paradigm that is inspired by the way biological nervous systems process information [2]. It is composed of a large group of interconnected processing elements that works towards solving a given set of problems. Neural network technique works well with CBR because it also uses examples to solve problems; it learns by example. A trained neural network can be considered as an "expert" in the domain that it has been assigned to analyze, and in this research, the domain is software requirements specifications (SRS).

Within the CBR cycle (Refer to Fig. 1), the neural network (ANN) is applied at the *Retrieve* step. It assists in finding the most suitable solution to the given problem (SRS quality analysis problem – a new case) by referring to existing examples (past experiences or cases), which are stored in a case base or a knowledge base. The main advantage of using the combination of CBR and neural network (ANN) is that this approach will promote adaptive learning in resolving the problems at hand, and this will directly improve the efficiency of the quality analysis approach.

Other related works by researchers in this domain include the following:

Knauss and El Boustani [9] defined their own model in assessing the software requirements specifications quality based on the Goal-Question-Metric (GQM) method. They analyzed more than 40 software projects using this method. Their main goal was to assess the quality of a Software Requirements Specifications (SRS) document and to connect it to the corresponding project success. Project success was measured based on the answers given to several questions related to the project's results.

Heck and Parviainen [10] presented a method called LSPCM (The LaQuSo Software Product Certification Model) that was developed for certifying software product quality [11]. Here, they described their experiences from using this method for analyzing requirements quality in different cases (systems), which are the Central Registration System, Counter Automation Solution, and Embedded Systems Case. They showed that the checks in LSPCM were able to discover inconsistencies in requirements specifications, regardless of the application domain.

Another research work on quality analysis of the SRS was conducted by Dang Viet Dzung and Atsushi Ohnishi [12] in which they established a method of checking a SRS using requirements ontology. A set of rules that were classified into general rules and domain specific rules were used in determining the correctness and completeness of the SRS.

IV. Benefits of Using CBR and ANN

Several benefits of using CBR and ANN in SRS quality analysis have been identified as follows:

A. Avoid the repetition of steps or tasks in order to arrive to a solution to the current analysis problem

SRS quality analysis is not a simple task and it takes many steps to be performed before the analysis's results can be produced. Many steps are similar to previous cases or problems that have been encountered before. A lot of time can be saved by eliminating all of the steps that have been carried out before.

B. Past experiences or cases can be used directly for any new case that happens to be exactly the same as one of the cases in the case base.

The *Reuse* step of CBR allows a case's solution to be used directly without performing any modification. ANN assists in determining the similarity level of the new case when compared with all past cases stored in the case base.

C. Ability to adapt an existing similar case or group of cases to the new case to be solved.

In SRS quality analysis, new cases usually have some similarity with one or more existing cases in the case base. The

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 15,2025 at 04:25:49 UTC from IEEE Xplore. Restrictions apply.

most similar case or a set of similar cases can be retrieved and modified accordingly. Both CBR and ANN play important roles here.

D. Ability to learn new cases or experiences

The *Retain* step of the CBR component assists in saving new cases and their solutions whenever new cases are encountered.

E. Can learn from past mistakes

Learning from mistakes and stop repeating the same mistakes will also reduce the time taken to solve new problems.

F. Allow the system that uses both CBR and ANN to solve problems that have no known or pre-determined solutions

ANN in one of the many soft computing components or tools that solves problems that naturally have a lot of uncertainty and imprecision in the available data or inputs.

V. The Proposed Framework

In order to understand the framework some basic explanations about CBR and how ANN is applied within the CBR component need to be given. The following gives brief descriptions of the steps [3]:

- Step 1 - *Retrieve*: Retrieve the most similar case or group of cases. Artificial neural network (ANN) is applied here to measure the similarity level of the new case when compared to all existing cases (past experiences) in the case base.
- Step 2 - *Reuse*: Reuse the information, knowledge, and solution in that case to solve the problem at hand if there is a perfect match. A perfect match occurs when the new case is exactly the same (100% similarity level) as an existing case or past experienced case.
- Step 3 - *Revise*: Modify and adapt the most similar case or group of cases as appropriate if a perfect match is not found.
- Step 4 - *Retain*: Retain or save the new experience or case for future retrievals and problem solving, and the case base is updated by saving the newly learned case.

In this paper, a case is an example of a previously experienced SRS quality analysis result that is stored in a file (knowledge base or case base). When a new case or problem (SRS quality analysis) is evaluated, previously stored cases are retrieved from the case base and are used to come up with the quality analysis results for the new case [6]. If the new case (entered case) has been experienced before, the previously stored similar (exactly the same) case is retrieved and reused directly (Step 1 and Step 2 of the CBR cycle). This will save a lot of time. If the new case has not been experienced in the past, the most similar case or a group of most similar cases is retrieved and revised (Step 3), and the newly experienced case is retained/stored (Step 4) inside the case base.

ANN is used to measure the similarity level between the new case and the existing cases stored in the case base. The similarity measurement needs to be intelligent and highly optimized [13] because the size of existing cases in the case base will grow overtime. It will be unwise to match a new case against a million existing cases that will result in serious performance drawback.

The system will assign scores to all existing cases based on their features (*Quality Attributes* and *Quality Indicators*) and group the cases based on the score. While performing similarity measurement, the system will only match the new case against groups rather than matching against all existing cases. The groups can be grouped at another level to narrow down the search space even further. ANN engine will be implemented on top of this grouped data structure to perform similarity measurement via pattern recognition (also known as classification [14] for the incoming case. The ANN engine will follow supervised learning paradigm, which is a process where ANN is trained with a training data set. Each example in the training data set is a pair comprising of sample input and output vectors. Learning from the sample data, ANN can perform efficiently when encountered with actual problems in real applications. In the proposed system, the case base will be the training data set for the ANN. So, when new cases are added to the case base, the ANN engine also needs to be retrained accordingly. Fig. 2 illustrates the flow of processes within the proposed SRS quality analysis framework.

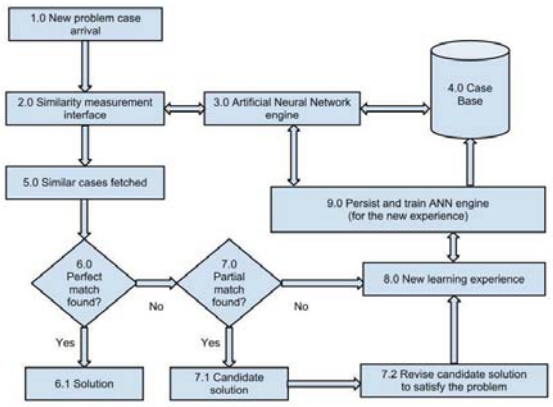

Fig. 2. The proposed framework showing a high-level process flow of the SRS quality analysis using CBR and ANN

In addition, Fig. 3 shows a basic diagram of the CBR's *Retrieve* step that accepts inputs entered by the user regarding the new case (SRS document) that needs to be analyzed in terms of its quality. Based on Table I, the number of scores or input values that describe the SRS under evaluation is 61 and all these input values are stored in a matrix or two-dimensional array that represents the structure given in Table I. Each case is represented by an array containing 61 items or values that represent the quality scores for each case.

Basically, neural networks are made up of many artificial neurons. An artificial neuron that tries to mimic the real neuron in human's brain is simply an electronically modeled biological neuron [15]. In the diagram below, the neuron accepts a number of inputs ($QI_1..QI_n$, where QI stands for

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 15,2025 at 04:25:49 UTC from IEEE Xplore. Restrictions apply.

Quality Indicator) and processes the inputs based on the given summation function and the calculated assigned weights ($w_1..w_n$), and gives an output, v_j. This summation output is put through an *Activation Function* that produces the final *Output*. This *Output* assists in measuring the similarity level of the new case with the cases in the case base (the case base holds previously experienced cases). The number of artificial neurons required to optimize the process of calculating the similarity level of the new case or SRS problem with each existing case in the case base depends on the tasks (application) at hand. This will be determined during the actual implementation of the proposed approach.

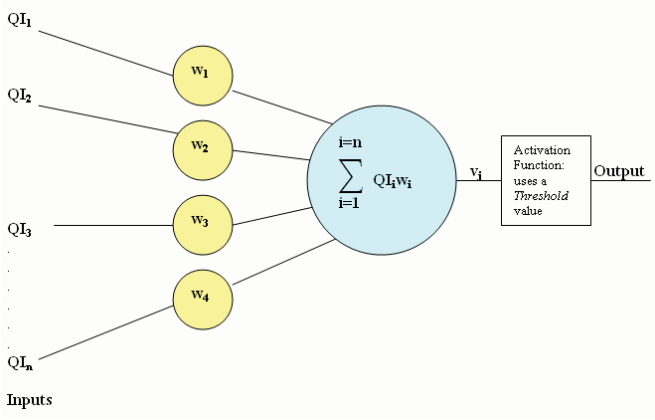

Fig. 3. A neuron of the ANN within the CBR Retrieve's step

VI. Future Work and Conclusion

Future work includes the implementation of the proposed framework within a web-based software requirements specifications (SRS) quality analysis system that applies artificial neural network (ANN) to CBR. This will produce an improved SRS quality analysis system that will eventually benefit many software project developers in producing better quality SRS document.

In conclusion, this research is believed to be able to contribute in improving the quality analysis process of Software Requirements Specifications (SRS) by ensuring that the produced SRS document meets certain standards. In addition, the time taken to perform the analysis is also reduced drastically with the use of both CBR and ANN. This saves time and money because of the fact that in general the software will cost less if errors are found at early stage of the software development life cycle, which is during the requirements specifications phase.

Acknowledgment

This research project is funded by the Ministry of Higher Education (MOHE) Malaysia under the Fundamental Research Grant Scheme (FRGS).

References

[1] http://www.philosophe.com/design/requirements.html. "Requirements and Specifications", Retrieved 21 Jan 2010.

[2] C. Stergiou and D. Siganos, "Neural Networks", Retrieved 26 Feb 2011, http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html.

[3] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approach," AI Communications, vol. 7, no. 1, 1994, pp. 39–59.

[4] J.L. Kolodner, Case-Based Reasoning, Morgan Kaufmann: San Mateo, CA, pp. 27–28. 1993.

[5] H. Mat Jani, and A. Lee, "Online Quality Analysis of the Requirements Specifications Phase of the Software Development Cycle", In Proc. 7th Annual SEAAIR Conference (SEAAIR 2007), 2007, pp 166-179.

[6] H. Mat Jani, "Applying Case-Based Reasoning to Software Requirements Specifications Quality Analysis System", In Proc. The 2nd International Conference on Software Engineering and Data Mining (SEDM2010), 2010, pp. 140-144.

[7] H. Mat Jani,, and S.A. Mostafa, "Implementing Case-Based Reasoning Technique to Software Requirements Specifications Quality Analysis", International Journal of Advancements in Computing Technology (IJACT), Vol. 3, No. 1, 2011, pp. 23-31.

[8] S.A. Mostafa, , and H. Mat Jani, "Applying Fuzzy Logic to Software Requirements Specifications Quality Analysis", In Proc. ICACT2011: The 2nd International Conference on Advancements in Computing Technology, Korea, 2011, pp. 1100 -1104.

[9] E. Knauss and C. El Boustani, "Assessing the Quality of Software Requirements Specification", 16th IEEE International Requirements Engineering Conference", 2008, pp. 341-342.

[10] P. Heck and P. Parviainen, "Experiences on Analysis of Requirements Quality", In Proc. Third International Conference on Software Engineering Advances, 2008, pp. 367-372.

[11] P. Heck and M. van Eekelen. The LaQuSo Software Product Certification Model, CS-Report 08-03, Technical University Eindhoven, 2008., Cited in Heck and Parviainen, "Experiences on Analysis of Requirements Quality".

[12] V.D. Dang and A. Ohnishi, "Improvement of Quality of Software Requirements with Requirements Ontology", Ninth International Conference on Quality Software, 2009, pp. 284-289.

[13] T.W. Liao, Z. Zhang, and C.R Mount,., "Similarity Measures for Retrieval in Case-Based Reasoning Systems", Applied Artificial Intelligence, 12, 1998, pp. 267-288.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 15,2025 at 04:25:49 UTC from IEEE Xplore. Restrictions apply.

[14] G.P. Zhang, “Neural Networks for Classification: a Survey”, IEEE Trans Syst Man Cybern Part C Appl Rev 30, 2000, pp. 451–462.

[15] http://www.learnartificialneuralnetworks.com/#Intro, Retrieved 11 Aug 2011.

Authorized licensed use limited to: China University of Petroleum. Downloaded on February 15,2025 at 04:25:49 UTC from IEEE Xplore. Restrictions apply.