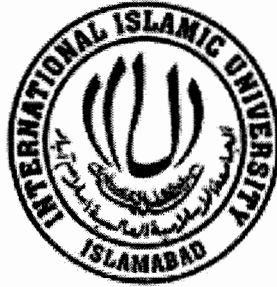


**SOFTWARE REQUIREMENTS PRIORITIZATION
TECHNIQUE FOR PROJECTS WITH MEDIUM/LARGE
SET OF REQUIREMENTS**

MSSE Thesis



By

Muhammad Imran Babar

Reg # 193-FBAS/MSSE/S08, DCS&SE, IIUI

Supervisor

Mr. Shahbaz Ahmed Khan Ghayyur

Assistant Professor, DCS&SE, IIUI

Co-Supervisor

Dr. Muhammad Ramzan

Assistant Professor, UAAR

Department of Computer Science & Software Engineering

Faculty of Basic & Applied Sciences

International Islamic University Islamabad, Pakistan



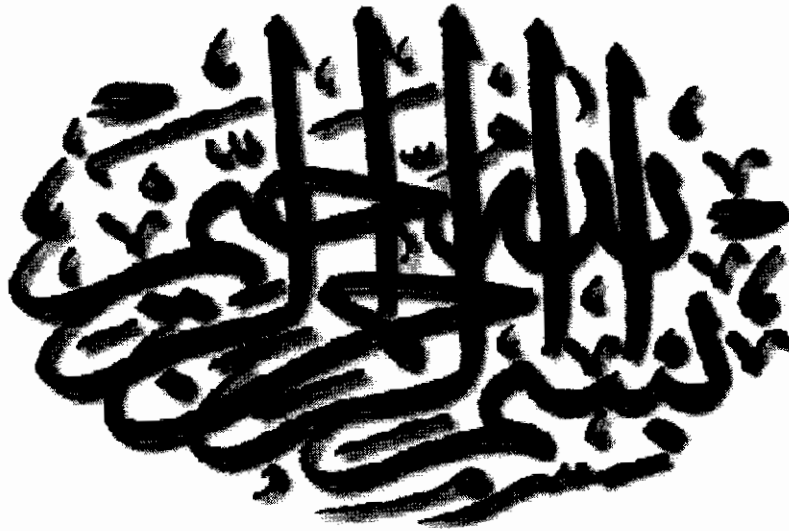
Accession No. TH-8626

MS
005.12
BAS

DATA ENTERED

W. J. ...
27/11/2012





*In the name of Allah the most beneficent and
merciful.*

*Department of Computer Science & Software Engineering
International Islamic University, Islamabad.*

Final Approval

It is certified that we have read the project report submitted by **Muhammad Imran Babar** Registration Number **193-FBAS/MSSE/S08**. It is our judgment that this project is of sufficient standard to warrant its acceptance by the International Islamic University, Islamabad for the Master of Science Degree in Software Engineering.

Committee

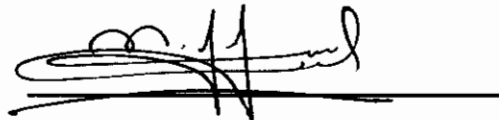
External Examiner

Dr. Muhammad Umer Khan
Assistant Professor, Department of Mechatronics,
Faculty of Engineering,
Air University, Islamabad,
Pakistan.



Internal Examiner

Mr. Imran Saeed
Assistant Professor, DCS&SE, FBAS
International Islamic University, Islamabad,
Pakistan



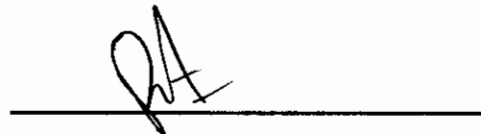
Supervisor

Mr. Shahbaz Ahmed Khan Ghayyur
Assistant Professor, DCS&SE, FBAS
International Islamic University, Islamabad,
Pakistan.



Co-Supervisor

Dr. Muhammad Ramzan
Assistant Professor
University Institute of Information Technology,
University of Arid Agriculture, Rawalpindi,
Pakistan.



Dedicated To

HAZRAT MUHAMMAD (SAWW)

& To

My Loving Parent & Family

*Who are a source of courage ,honesty and love,
without their prayers it was not possible for me to
complete this research*

& To

Precious Friendship

*That has made us laugh, held us when we cried
and always, always, be among us*

*A Thesis Submitted to the
Department of Computer Science & Software
Engineering, Faculty of Basic and Applied Sciences,
International Islamic University, Islamabad, Pakistan,
as a Partial Fulfillment of the Requirements for
the Award of the Degree of
MS in Software Engineering*

DECLARATION

I hereby declare and affirm that this thesis neither as a whole nor as part thereof has been copied out from any source. It is further declared that I have completed this thesis on the basis of my personal efforts, made under the sincere guidance of my supervisors. If any part of this report is proven to be copied out or found to be a reproduction of some other, I shall stand by the consequences. No portion of the work presented in this report has been submitted in support of an application for other degree or qualification of this or any other University or Institute of learning.

Muhammad Imran Babar (193-FBAS/MSSE/S08)

ACKNOWLEDGEMENTS

I bestow all praise to, acclamation and appreciation to Almighty Allah, The Most Merciful and Compassionate, The Most Gracious and the Beneficent, Whose bounteous blessings enabled me to pursue and perceive higher ideals of life, who bestowed me good health, courage and knowledge to carry out and complete my work. Special thanks to **His Holy Prophet Muhammad (SAWW)** who enabled me to recognize my Lord and Creator and brought me the real source of knowledge from **Allah**, the **Qur'an**, and who is the role model for me in every aspect of life.

I consider it a proud privileged to express my deepest gratitude and deep sense obligation to my reverend supervisor **Mr. Shahbaz Ahmed Khan Ghayyur** and co-supervisor **Dr. Muhammad Ramzan** that they kept my morale high by their kind suggestions and appreciations. Their motivation leads me to this success and without their sincere and cooperative nature and precious guidance I could never have been able to complete this task. I am also thankful to **Higher Education Commission (HEC) of Pakistan** for providing funding for this research.

It would not be out of place to express my profound admiration to **Mr. Flak Sher (Research Scholar, IIUD)**, **Mr. Shahid Ghauri (System Analyst, Bahria Town)**, **Mr. Farrukh (SQA Engineer, NETSOLACE IT PVT LTD)**, **Mr. Rehan Farooq (Senior Software Engineer, Bahria Town)** and **Mr. Muhammad Sohail (AM (Technical), NESCOM)** for their dedication, inspiring attitude, untiring help, and kind behavior throughout the project efforts.

Finally I must mention that it was mainly due to my family's moral support during the entire academic career which enabled me to complete my work in time. I once again would like to admit that I owe all my achievements to my most loving parent, who means most to me, for their prayers are more precious than any treasure on the earth. I am also thankful to my most loving brother, sisters, wife and daughters, friends and class fellows whose prayers have always been a source of determination for us.

Muhammad Imran Babar (193-FBAS/MSSE/S08)

ABSTRACT

Requirements Engineering is a critical discipline in software development life cycle. The major problem in software development is associated with the selection of the important requirements. The only way to select the requirements is associated with the phenomenon of prioritization. The current research is highlighting or analyzing the issues of existing software requirements prioritization techniques. One of the major issues of software requirements prioritization is that the current techniques handle only toy projects or software projects with very few requirements. Currently there is not an evidence of a single technique which is suitable for prioritization of large number of requirements in projects where requirements may grow in hundreds or in thousands. The current research is focusing on Value Based Intelligent Requirements Prioritization (VIRP) technique. In this research an extension is proposed in VIRP in the form of a neural network. The neural network is proposed for the second level of VIRP for reliable results. The extension will help to make the technique more efficient with respect to time and intelligent.

TABLE OF CONTENTS

CONTENTS		PAGE#
1.	Introduction	1
2.	Literature Survey	6
3.	Problem Statement	17
3.1.	Research Question	18
3.2.	Proposed Solution	18
3.3.	Intended Output	20
4.	(VIRP): Expert Driven Fuzzy Logic Based Requirements Prioritization Technique	21
4.1.	(VIRP): An Overview of Value based Intelligent Requirements Prioritization (VIRP) Technique	22
4.2.	Requirement Elicitation & Stakeholder Level Prioritization	23
4.3.	Expert Level Prioritization	24
4.3.1.	Project Specific RCFs	25
4.3.2.	Requirement Specific RCFs	25
4.4.	Fuzzy Logic Based Requirement Prioritization	26
5.	Research Methodology	27
5.1.	Methodology	27
5.1.1.	Research Plan	28
6.	Artificial Neural Networks	31
6.1.	Proposed Solution	32
6.2.	Training/Validation Data	33
6.3.	Create/Train the Network Stage 1	37
6.3.1.	Algorithm Stage1	37
6.3.2.	Results Stage1	38
6.3.3.	Training Graph/Window with TRAINLM Stage1	41
6.4.	Validation of Neural Network Stage 2	42

CONTENTS	PAGE#
6.4.1. Algorithm Stage2	42
6.4.2. Results Stage2	47
6.4.3. Graphs Stage2	50
6.4.4. Graphical User Interface (GUI Prototype)	53
7. Conclusion	55
8. References	58
9. List of Publications	66
Figures	
Figure 2.1: Average Time Consumed by Different Techniques	13
Figure 3.1: Flow graph of the Fuzzy Based Intelligent Requirements Prioritization Process with proposed extension work with respect to Neural Networks	20
Figure 4.1: Flow graph of VIRP process	22
Figure 6.1: Training Graph with TRAINLM	41
Figure 6.2: Untrained Network Output	51
Figure 6.3: Linear Fit of Validation Data with 25 Data values	52
Figure 6.4: Linear Fit of Validation Data with 500 Data values	53
Figure 6.5: Graphical User Interface for Neuro Computed RV	54
Tables	
Table 2.1: Inherent characteristics of the prioritizing methods	8
Table 2.2: Objective measures during the evaluations	8
Table 2.3: Subjective measures after the evaluations	9
Table 2.4: Fundamental scale used for pair-wise comparisons in AHP	9
Table 4.1: Project Specific Requirements Classification Factors (pRCFs)	25
Table 4.2: Requirements Specific Requirements Classification Factors (rRCFs)	26

Chapter # 1

INTRODUCTION

1. INTRODUCTION

Presently the software industry leaders are busy in the development of innovative systems in order to cope with the market demand of the customers. Innovation is the cause of complexity in the design of such systems. The very complexity is the result of unclear user requirements and objectives. The design and development of innovative systems mainly depends upon clear stakeholders' requirements or objectives, so due to lack of firm and clear requirements it is very difficult to design and develop such applications. The only way to eliminate these complexities is based on the right selection of the user or stakeholders' requirements in a professional way. The selection of the stakeholders' requirements is based on the importance or value of the requirements. The importance or requirement's value is calculated using a suitable requirements prioritization technique. For decision making the prioritization of requirements is also considered as important (Aurum, 2003) in order to eliminate the complexities induced due to the lack of clear user objectives or requirements. The process of requirements prioritization is also a very complex decision making process (Carlshamre, 2002, Karlsson, 1996, Lehtola et al. 2004, Moisiadis, 2002) so in order to apply different software requirements prioritization techniques the experts must have sound knowledge of the domain and professional skill set (Karlsson et al. 2004).

Requirements prioritization is a process in which requirement engineers find the most important stakeholders' requirements in order to develop a system (Somerville I., 1996). The process of requirements prioritization is used to discover or select the core or most important requirements in order to develop an innovative system. These core

requirements must be implemented within the defined constraints, of time, resources, cost and quality, in order to satisfy the customers (Abran et al., 2001, Karlsson, J., 1998, Somerville I., 1997, Wiegers K., 1999, Yeh A. C. 1992). Though the software requirements prioritization process is not only used to discover the least important requirements but it also paves the way to find out conflicts among various requirements and ultimately helps in the resolution of these conflicts and also provides future road map or plan (Wiegers K., 1999). Ruhe et al. have said that "The challenge is to select the 'right' requirements out of a given superset of candidate requirements so that all the different key interests, technical constraints and preferences of the critical stakeholders are fulfilled and the overall business value of the product is maximized" (Ruhe et al, 2002). So an innovative system can only be in conformance with the quality if the right user requirements are selected and implemented. If the set of requirements, which is selected for a particular system, will not be right, then it will result in the increase of cost incurred in modification of the system as and when required (Boehm, 1981) and the wrong requirements will also affect the quality of the system.

In order to gain the advantage in the market, the process of software requirements prioritization is also considered as vital and it helps in understanding the up and down of the market in terms of loss and profit (Aurum, 2003). The satisfaction of the stakeholders is taken into account when a quality software system fulfils all the requirements of stakeholders or users (Bergman, 2003, Schulmeyer et al. 1999). In software industry the consideration of all requirements, in order to develop a software system, is not possible because of the constraints like time to market, budget, and other resources so instead of

considering all the user requirements in a single release the consideration of important requirements is taken into account (Karlsson, 1997, Siddiqi, J. 1997). Software requirements have different features like risk (Arum, 2003, Moisiadis, F., 2002, Wiegers K., 1999, Somerville I., 1997), importance (Lehtola et al. 2004), volatility (Moisiadis, F., 2002), cost, penalty, time (Wiegers K., 1999) and dependencies of requirements with other requirements based on cost, technical importance or value, customers and change in the requirements (Dahlstedt et al. 2003, Ruhe et al. 2003). The requirement prioritization process is performed on the basis of these different features of requirements which are stated above. On the basis of these aspects the unimportant requirements are not given importance rather they are totally rejected and value added requirements are added in the requirements set in order to develop a system of high quality. Only value added requirements can add value to the system and this value may be in terms of profit (Sebastian et al., 2008), efficiency, good performance, correct data and fulfillment of right user needs. Different software requirements prioritization techniques are used in the software industry in order to prioritize the most important requirements and there is not a mature process to prioritize the requirements (Aurum, 2003). In software industry the software engineers are using multiple techniques on the basis of the factors like cost, time, and relevant importance, but such an approach is the cause of certain conflicts which initiate due to the affect of one aspect on the other aspect. The prioritization of the requirements with respect to cost may affect the priority of a given requirement e.g. if the cost of the given requirement will be high along with its high importance then it is possible that the stakeholder or customer will change its mind about that requirement because of its

high cost. Such a change in the mind of the customer will also result in the change of the priority of that very requirement (Lausen, 2002). During software requirements prioritization the requirements' attributes or aspects are directly affected by the role of customers or stakeholders involved in the development process of the innovative system. Donald Firesmith is of the view that the "fundamental problem with prioritizing requirements is that the phrase "prioritizing requirements" can have very different meanings to different stakeholders (Firesmith, 2004)".

In order to develop a successful innovative software system there is the need to select right and relevant user requirements. For right selection of user or stakeholders' requirements there exist different prioritization approaches which are discussed in detail in the next chapter.

Chapter # 2

LITERATURE SURVEY

1. LITERATURE SURVEY

Researchers have presented different software prioritization methods which are used to select the right set of user requirements in order to develop software applications of high end quality. Some of the most prominent software requirements prioritization techniques are Analytical Hierarchical Process (AHP), Binary Search Tree, Ranking, Numerical Assignment, Top Ten Requirements, Cumulative Voting or Hundred Dollar Test and the researchers are busy in developing different software requirements prioritization techniques in order to cope with the industry challenges. Different methods or ways are used to prioritize the requirements e.g. stakeholders have to prioritize requirements on the basis of factors like which requirement is mandatory, desirable or essential one and which one is not (Brackett, 1990) while some have to adopt quantitative ranking system for requirements. Kent Beck presented the planning game method to prioritize the requirements in extreme programming (Beck, 2000). Wiegers presented a technique in which quantitative ranking from 1-9 was used and for ranking the factors like cost, risk and importance were taken into account (Wiegers K., 1999). According to Laurent et al. "Although such techniques are relatively simple to implement, they provide little support for higher level goal-setting and negotiation and furthermore do not scale well for managing requirements in large and complex projects" (Laurent et al., 2007).

The maturity level of software requirements prioritization process is not still very high even in the presence of different proposed prioritization techniques. Industry leaders have less awareness of different software requirements prioritization methods and they have insufficient knowledge of assigning priority values to software requirements (Lubars

et al., 1993). Karlsson is also depicting the same concept that the domain of requirements engineering got a brisk growth but still the professionals are in lack of reliable and effective prioritization approaches (Karlsson, Ryan, 1997).

The existing prioritization techniques are tested thoroughly and factual results are obtained about their performance. On the basis of the results obtained from different prioritization techniques the new techniques came into being. In a research conducted by Karlsson et al. different requirements prioritization techniques were examined (Karlsson et al., 1998) and they recorded different results of all these requirements prioritization techniques. The inherent characteristics, objective and subjective measures of the prioritization methods are shown in the following tables.

Table 2.1: Inherent characteristics of the prioritizing methods (Karlsson et al., 1998)

Evaluation Criteria	AHP	Hierarchy AHP	Spanning tree	Bubblesort	Binary search	Priority groups
Consistency Index (yes/no)	Yes	Yes	No	No	No	No
Scale of Measurement	Ratio	Ratio	Ratio	Ordinal	Ordinal	Ordinal

Table 2.2: Objective measures during the evaluations (Karlsson et al., 1998)

Evaluation Criteria	AHP	Hierarchy AHP	Spanning tree	Bubblesort	Binary search	Priority groups
Required number of decisions	78	26	12	78	29,33,38	34,35,36
Total time consumption (Ordinal scale 1-6)	6	2	1	3	5	4
Time consumption per decision (Ordinal scale 1-6)	2	4	5	1	6	3

Table 2.3: Subjective measures after the evaluations (Karlsson et al., 1998)

Evaluation Criteria	AHP	Hierarchy AHP	Spanning tree	Bubblesort	Binary search	Priority groups
Ease of use	3	4	2	1	5	6
Reliability of results	1	3	6	2	4	5
Fault tolerance	1	3	6	2	4	5

The above statistics is derived or obtained using the following scale.

Table 2.4: Fundamental scale used for pair-wise comparisons in AHP (Karlsson et al., 1997)

Relative Intensity of Importance	Definition	Description
1	Of equal value	Two requirements are of equal value
3	Slightly more value	Experience slightly favors one requirement over another
5	Essential or strong value	Experience strongly favors one requirement over another
7	Very strong value	A requirement is strongly favored and its dominance is demonstrated in practice
9	Extreme value	The evidence favoring one over another is of the highest possible order of affirmation
2,4,6,8	Intermediate values between two adjacent judgments	When compromise is need

Reciprocals: If requirement i has one of the above numbers assigned to it when compared with requirement j , then j has the reciprocal value when compared with i

Karlsson et al. have applied the different software requirements prioritization techniques “toprioritize13 well-defined quality requirements on a small telephony system” (Karlsson et al., 1998). The statistics of the results obtained from different prioritization

techniques depict that Bubblesort and AHP are more reliable than other techniques. The results provided by most of the techniques are not correct. It is also noted that Bubblesort and AHP are more reliable but the time consumption is very high due to the rising number of decisions in order to prioritize the software requirements (Karlsson et al., 1998). AHP is considered as a most reliable technique but its implementation is very complex and the pair wise comparisons consume too much time. The implementation of Hierarchy AHP is also very complex and its results are prone to errors so Hierarchy AHP is considered as an unreliable technique. Same is the case with Spanning Tree technique that it is also prone to errors and is not considered as a reliable one. The other prioritization techniques like Priority Groups, Binary Search are also prone to errors so the results generated by them are unreliable (Karlsson et al., 1998). Consistency index is considered as vital for reduction of human error but no consistency index exists in case of Bubblesort so the results are prone to error and the technique is unreliable (Karlsson et al., 1998). The different software requirements prioritization techniques are evaluated using few techniques and about AHP Karlsson et al. are of the view that “AHP may be problematic to scale up for larger projects (the same is true for Bubblesort)”(Karlsson et al., 1998). Laurent et al. are of the view that “AHP does not scale well because the number of comparisons grows exponentially with the number of requirements” (Laurent et al., 2007). About larger projects Laurent et al. describe that “Although these techniques offer useful solutions, for managing requirements in small or medium sized projects, they do not provide a feasible solution for managing larger projects” (Laurent et al., 2007). AHP method is only suitable for few requirements and when the number exceeds 20 then it no longer remains reliable due to the pair-wise

comparisons and the requirements cannot be managed easily (Lehtola & Kauppinen, 2004).

A new software requirements prioritization technique called “Requirements Triage” was presented by Laurent et al., 2007. Laurent et al. describe that “The limitations of the approach are closely related to the limitations of the underlying traceability, classification, and clustering algorithms, all of which are based upon data mining and information retrieval techniques that are probabilistic in nature and that therefore do not return perfect precision or recall in the results”(Laurent et al., 2007). There is an algorithm proposed by Peng Shao and this “technique allows customers to rank a relatively small set of items then combines these rankings over a large number of customers to determine an ordering for a large requirement set” (Peng Shao, 2008). The algorithm or technique is tested on limited number of stakeholders i.e. 10 in number and with limited number of items i.e. 10 in number. The proposed algorithm is only encompassing the limited set of requirements and is also lacking in providing results or statistical evidence. In the paper Peng Shao has said about algorithm or technique that “In the future, we will try to implement our algorithm to improve requirements prioritization” (Peng Shao, 2008). Mostly the researchers have presented different extensions in AHP in order to cope with the issue of scalability and all the techniques are just dealing with toy projects and not dealing with the projects with large number of requirements like the projects comprising of hundreds and thousands of requirements (Firesmith, 2004). In most of the experiments a small set of requirements is taken into account. M.A. Iqbal et al. have presented a market driven products model for requirements prioritization in order to provide a solution to issue of scalability and this

model is an extension of AHP (Iqbal et al. 2010). The market driven products model is implemented in the real scenarios without statistical evidence of the experimentation and the results provided are just opinions in order to support or strengthen the hypothesis. Though the market driven products model is solving the scalability problem but its implementation is not valid for each and every domain (Iqbal et al. 2010) and in its statistics it is also not mentioned that for which domain it is suitable one. "Further there is no evidence that at which extent the proposed model is solving the issue of scalability" (Babar et al. 2011). Muhammad Ramzan et al. have presented a value based fuzzy logic prioritization technique and the requirements are prioritized on the basis of the perceived values of the requirements (Ramzan et al., 2009). After experimentation Ramzan et al., have documented the results as:

The process is prone to error due to its heavy reliance on experts.

The process is very time consuming.

The element of bias was visible in some of the experiments conducted. (Ramzan et al., 2009)

Later another technique called "Value based Intelligent Requirement Prioritization (VIRP): Expert Driven Fuzzy Logic based Prioritization" was presented by the same authors and the technique was consisting of three major steps i.e. "Requirement Elicitation and Stakeholder Level Prioritization", the "Expert Level Prioritization" and the last one is "Fuzzy Logic Based Requirement Prioritization" (Ramzan et al., 2011). The "Fuzzy Logic based Requirement Prioritization" step of the technique is fully automated and is made intelligent using fuzzy logic and is totally free of biasness. In VIRP the second level i.e.

“Expert Level Prioritization” is a manual step and there is the need to fully automate this step and made it intelligent. The second step of the technique is the cause of too much time consumption and results in reduction of the performance of the technique because this second step is manual and not automated and the involvement of the experts is at high extent.

After making the second step, of Value based Intelligent Requirement Prioritization (VIRP): Expert Driven Fuzzy Logic based Prioritization technique, fully automated and intelligent it will help us in elimination of the involvement of software engineers or industry experts and overall performance of the technique will increase in terms of time consumption.

The researchers have performed different experiments in order to fully evaluate the VIRP technique and they compared it with other techniques in terms of time consumed by each prioritization technique or method. The time consumption by all the techniques including VIRP (Ramzan et al., 2011) is presented in the figure as:

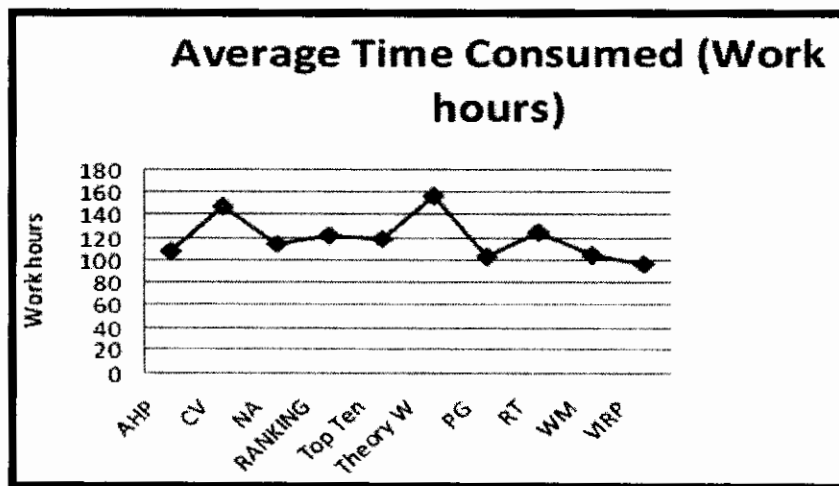


Figure 2.1: Average Time Consumed by Different Techniques (Ramzan et al., 2011)

There is no doubt that the technique is highly efficient but it can be made more efficient after making the second level of the technique, i.e. “Expert Level Prioritization”, fully automated and intelligent because it will help in elimination of role of experts which ultimately result in elimination of bias induced by the experts due to the heavy reliance of the technique on experts.

Sadiq et al. presented an approach to elicit the user requirements and AHP was used in order to prioritize the requirements (Sadiq et al., 2009). A toy project is used in order to develop mini software instead of handling a project with large number of requirements and it is claimed that “from this approach we can easily rank the requirements and can implement it on the basis of the ranking. In this paper we have developed the *mini software for numerical integration (MSNI)*” (Sadiq et al., 2009). The technique is in the form of an optimization algorithm which “is used to improve the effectiveness and efficiency of the freight transportation between source and destination using the optimization algorithm” (Babar et al., 2011). AHP is adopted as a prioritization method in order to prioritize the requirements which has certain limitations which are discussed previously and the technique is helpful in minimizing the cost of transportation so it is not a generic one rather its specific for freight transportation (Sadiq et al., 2010). Presently researchers are busy in introduction of new extensions in AHP in order to solve the issue of scalability and in elimination of other limitations of the existing techniques. But there is need of new techniques which are fully automated and intelligent that will prove fruitful for projects where requirements exist in large numbers i.e. beneficial for hundreds or thousands of the requirements. A requirements prioritization method presented

by Aaron K. Massey et al. in 2010 and it was about legal requirements. The focusing domain of the technique is healthcare and is implemented to solve the legal issues concerning requirements of healthcare domain. HIPPA (Health Insurance Portability and Accountability Act) is the focusing point of the technique which controls the Electronic Health Records (EHRs). The said technique is also specific and not for each and every domain.

“The literature shows that most of the techniques which are proposed are only suitable for few requirements and are not suitable when the requirements scaled up in case of the projects where requirements dataset is medium / large” (Babar et al., 2011). Qiao Ma has described in his dissertation that “to the author’s knowledge, no previous literature defines what number of requirements is medium and what number is large” (Qiao Ma, 2009). According to Hatton it is difficult for people to rank 15 or more requirements (Hatton, 2007). Qiao Ma says in his dissertation that “this research treats 1 to 14 (inclusive) requirements as a “small” number of requirements, 15 to 50 (inclusive) requirements as a “medium” number of requirements, and more than 50 requirements as a “large” number of requirements” (Qiao Ma, 2009). The problems which are associated with the current requirements prioritization methods are described as (Babar et al., 2011)

1. The existing techniques do not provide a scalable solution when the requirements scaled up in case of large number of requirements.
2. The existing techniques do not provide sufficient automation and are not intelligent enough.
3. Most of the techniques are time consuming.

4. The results are faulty or error prone.
5. Results do not recall.
6. Mostly solve the issues of small scale requirements but with errors.

I am focusing in my research the issue of time, automation and intelligent requirement prioritization. There is the need to provide a new requirements prioritization technique which would help in prioritization of large scale requirements.

Chapter # 3

PROBLEM STATEMENT

1. PROBLEM STATEMENT

Value based Intelligent Requirement Prioritization (VIRP): Expert Driven Fuzzy Logic based Prioritization Technique is comprised of three stages or level. The second level of the technique is used to calculate the RV of a requirement using equation 3.1. In the second level of the technique there is too much involvement of the software experts or engineers in order to calculate the RV of the requirements. The RV of requirements is calculated manually at second level and takes too much time. The manual calculation of the RV reduces the overall performance of the technique and also induces biasness of the software engineers. There is the need to fully automate the second level of the technique in order to make it more efficient in terms of time and in order to reduce the impact of biasness induced by the software experts.

3.1 Research Question:

What can be an efficient and cost effective mechanism to prioritize requirements in an environment where requirement dataset is medium / large and volatility is high?

3.2 Proposed Solution:

The prioritization of the functional and non-functional requirements of the software applications must be taken into account “inline with the business compliance” (Babar et al., 2011). So “there is the need to prioritize the significant functional requirements inline with business compliance” (Babar et al., 2011).The previous techniques regarding requirements prioritization are not automated and intelligent so there is the need to present fully automated and intelligent requirements prioritization techniques in order to solve the issue of projects with large number of requirements. Currently I am proposing an extension

in the existing technique called “Value based Intelligent Requirement Prioritization (VIRP): Expert Driven Fuzzy Logic based Prioritization Technique” in order to enhance the efficiency of the technique. The prioritization technique is divided into three steps or levels i.e. “Requirement Elicitation and Stakeholder Level Prioritization”, “Expert Level Prioritization” and in the last “Fuzzy Logic Based Requirement Prioritization” (M. Ramzan et al., 2011). The level 1 of the technique i.e. “Requirement Elicitation and Stakeholder Level Prioritization” is fully automated in order to get the full performance of the technique. The second level of the technique i.e. “Expert Level Prioritization” is manual which results in too much time consumption because of the experts’ involvement in calculation of the RV. The involvement of experts is the cause of biasness and the element of biasness induces variance in the actual priority value of a given requirement. “The second step “Expert Level Prioritization” is the cause of too much time consumption because of experts’ opinions so there is the need to remove the role of experts or software engineers in order to make the technique more efficient with respect to time” (Babar et al., 2011). In order to eliminate the role of software experts there is the need to fully automate and make the level 2 i.e. “Expert Level Prioritization” fully intelligent for calculation of RV of a requirement. The last or the third level of the technique i.e. “Fuzzy Logic Based Requirement Prioritization” is made intelligent by involvement of fuzzy logic which helped in elimination of element of biasness. Following formula is used to calculate the RV or requirement value (Ramzan et al., 2011).

$$RV = 0.35 + 0.02 \left\{ \sum_{i=1}^5 pRCF_i + \sum_{i=1}^5 rRCF_i \right\} \text{ ————— Eq: 3.1}$$

The flow graph of the technique is shown in the figure:

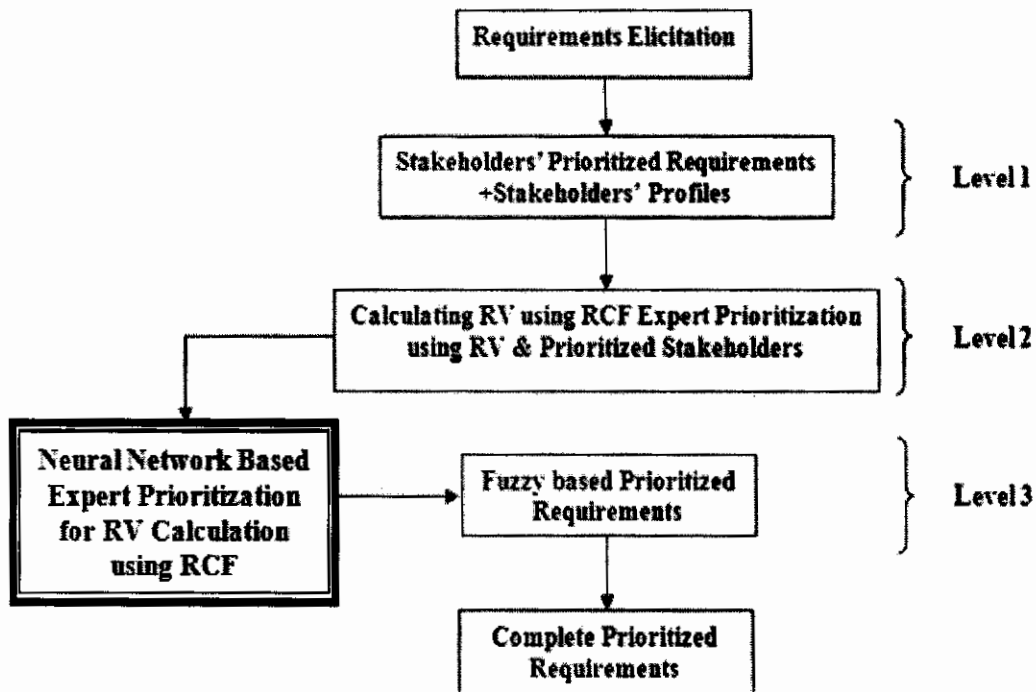


Figure 3.1: Flow graph of the Fuzzy Based Intelligent Requirements Prioritization Process with proposed extension work with respect to Neural Networks.

3.3 Intended Output:

- This should result in better understanding and translation of significant requirements and there would be less chances of error.
- The elimination of element of biasness.
- The technique will become more time efficient after making intelligent and fully automated at “Expert Level Prioritization” and the overall performance of the technique will also increase.

Chapter # 4

(VIRP): Expert Driven Fuzzy Logic based Requirements Prioritization Technique

4.1 (VIRP): AN OVERVIEW OF VALUE BASED INTELLIGENT REQUIREMENT PRIORITIZATION (VIRP) TECHNIQUE

Value based intelligent requirement prioritization technique (VIRP) (Ramzan et al., 2011) is comprised of different levels and is used for prioritization and classification of software requirements. The VIRP model is shown in the figure. As the technique is a multilevel technique and there is the involvement of stakeholders, experts and fuzzy logic at different levels of the techniques. VIRP is an iterative software requirements prioritization technique. The purpose of iteration is to evaluate the requirements iteratively with the help of different actors in order to achieve reliable results.

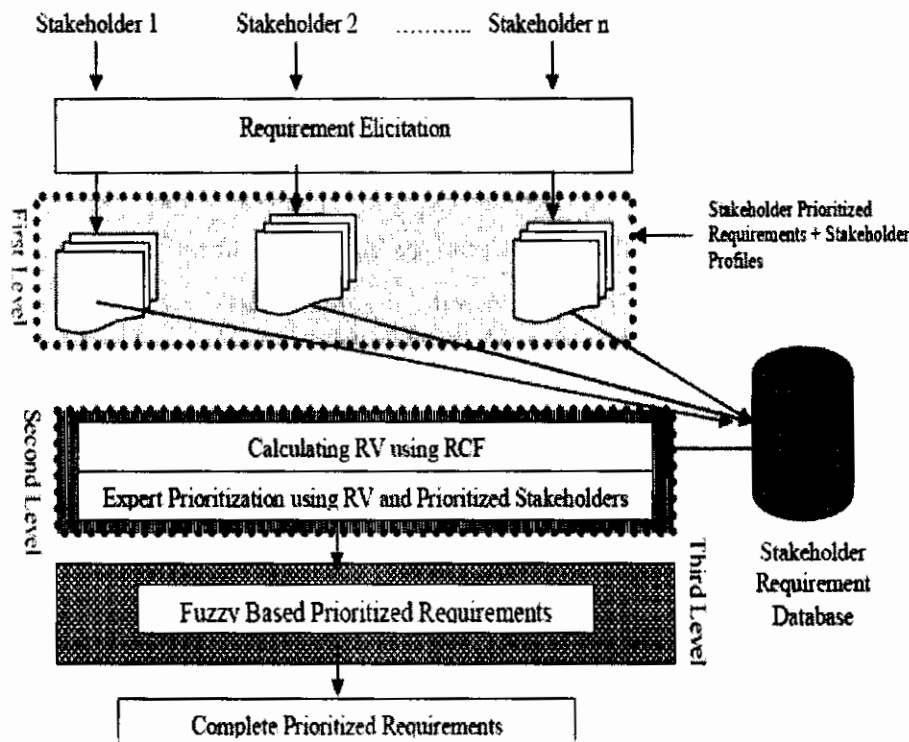


Figure 4.1: Flow graph of VIRP process (M. Ramzan et al, 2011)

The three stages of VIRP are as under:

1. Requirement Elicitation & Stakeholder Level Prioritization.
2. Expert Level Prioritization.
3. Fuzzy Logic based Requirement Prioritization.

4.2 Requirement Elicitation & Stakeholder Level Prioritization

Requirements elicitation process is an early stage requirement engineering process. Requirements elicitation process is considered as a standard process adopted in requirement engineering. During requirement elicitation the two new concepts are introduced in order to get prioritized requirements during its first stage.

- “The requirement elicitation is done electronically. All stakeholders submit their requirements on specially designed web scripts. Stakeholders submit their requirements according to their own priority” (Ramzan et al., 2011).
- “A new concept of stakeholder profiling is introduced. While submitting their requirements, stakeholders also give a brief description regarding themselves, their expectations about the system, system functionalities of their choice etc. Requirements engineers assigned to gather requirements also write down their observations regarding those stakeholders independently. Later both of these are combined to form a stakeholder profile. Stakeholder profile template is given in figure 3. These elicited requirements as well as stakeholder’s profile are stored in specially designed requirements database” (M. Ramzan et al., 2011).

4.3 Expert Level Prioritization

In this stage the requirements of stakeholders and their profiles are given to the experts. Experts have to perform three tasks in this stage of the technique.

The first task which is involved in the technique is the quick review of all the requirements given by all the stakeholders without taking into consideration the personal profiles of the stakeholders. The experts have to modify the existing priority of the requirements if required in case of a glaring problem.

The second task performed by the experts is related to the profiles of stakeholders. Experts have to review the profiles of stakeholders and they have to assign them a value in the range of 1-10. Experts have to assign this value on the basis of significance of stakeholders and the overall impact of the requirements, being posed by them, in order to make a project fully successful or working.

The third task being performed by the experts is related to the assignment of a value to each requirement in order to prioritize them properly. There are ten Requirements Classification Factors (RCFs) which are involved in assigning a value to each requirement. RCF_i is a requirement classification element and is defined as “one factor whose degree of presence or absence in a requirement can have a direct bearing on its value” (Ramzan et al., 2011). In the technique there are ten requirement classification factors which are identified by the authors and they are classified into two categories i.e. “Project Specific RCFs” and “Requirement Specific RCFs”.

4.3.1. Project Specific RCFs

Project Specific RCFs are taken into account for a requirement value which is elicited with respect to a project. Simply these RCFs are used to find out the value of a particular requirement. The pRCFs are feasibility, modifiability, urgency, traceability, and testability. Table-4.1 shows the list of Project Specific Requirements Classification Factors (pRCFs).

Table- 4.1: Project Specific Requirements Classification Factors (pRCFs)

Sr. No	Name	Description
01	Feasibility	The requirement is capable of being implemented within the constraints and resources
02	Modifiability	Requirement can undergo change to optimize the system without affecting the system adversely
03	Urgency	Degree of necessity of the requirement for system to be considered successful
04	Traceability	Requirement is such that subsequent function of the system can be traced to it. Requirements are less compound
05	Testability	Requirement can be tested and validated during testing phase. Independent test cases for the requirement can be generated.

4.3.2. Requirements Specific RCFs

Requirements Specific Requirements Classification Factors (rRCFs) are the attributes which play a vital role in adding worth to the value of a requirement in terms of “its description quality” (M. Ramzan et al., 2011). The rRCFs are completeness, consistency, understandability, within scope and non-redundancy. Table-4.2 shows the list of Requirements Specific Requirements Classification Factors (rRCFs).

Table-4.2: Requirements Specific Requirements Classification Factors (rRCFs)

Sr. No	Name	Description
01	Completeness	The requirement statement has enough information to proceed to the next development phase
02	Consistency	Requirement specifications use standard terminology and there are minimum conflicts due to statement and specifications
03	Understandability	Requirements are easy to describe and review. Requirements are grammatically correct with single and clear meaning
04	Within Scope	Requirement does not envisage something which is not described in original statement of scope
05	Non-redundancy	Requirement is not duplicated in complete or partially.

4.4 Fuzzy Logic Based Requirement Prioritization

In this part of the technique fuzzy c means method is used in order to make the technique fully automated and intelligent. “Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to be in the right position to two or more clusters” (Ramzan et al., 2011).

Chapter # 5
Research Methodology

5.1 METHODOLOGY

In this research Neural Networks (Artificial Intelligence) are applied in order to make the system more intelligent and it would help in the elimination of bias which may affect the performance and this bias is introduced because of the heavy reliance of technique on experts.

For the said research a controlled experiment is performed in the form of a simulation and the proposed controlled experiment is a most suitable method for this research. The simulation is done in MATLAB in order to implement Neural Network Algorithm. During experimentation the cause and effect relationship is measured in a controlled lab environment. The Neural Network is the most suitable way to find out the value of RV and this will help in the elimination of biasness being induced due to expert judgment. All other types of research methods, like benchmarking, survey questionnaire, case study, action research and systematic literature review, are not suitable for this research due to their uncontrolled environment.

5.1.1 Research Plan

➤ Hypothesis

H0: The technique is prone to biasness and is less efficient in terms of time.

H1: After automation or making the system more intelligent this should result in better understanding and translation of significant requirements and there would be less chances of error. It will result in elimination of element of biasness. The VIRP technique will become more time efficient after making intelligent and fully

automated at “Expert Level Prioritization” and the overall performance of the technique will also increase.

➤ **Definitions**

Controlled Experiment: An experiment that isolates the effect of one variable on a system by holding constant all variables but the one under observation.

Neural Network: Artificial neural networks works are like an “information-processing system” and these neural networks have some performance characteristics which are also common in biological neural networks (Fausett L., 2008).

➤ **Research Design Phases**

The research design is comprised of following four phases i.e.

1. Phase -1

In the first step the primary data is collected from research papers of reputed conferences and international journals in order to support the research problem.

2. Phase-2

In the second step a “Neural Network” is designed that applies heuristics to the technique at expert level prioritization step of VIRP technique in order to make it fully automated. The automation of this phase has enabled us to find out the overall performance of the technique.

3. Phase-3

In the third phase the values of pRCF and rRCFs of a large set of requirements are calculated with the help of industry professionals having experience more than 5 years.

4. Phase-4

In the last phase the Neural Network is simulated using the collected data set of pRCFs and rRCFs and the results are noted in order to calculate the RV of a requirement.

Chapter # 6

Artificial Neural Networks

6 NEURAL NETWORKS

Modern computational theories have provided solutions to highly complex problems. The application of computer is vivid in the domains like Biology, Chemistry, Mathematics, Statistics, Modeling and Simulation, Finance, Accounting, Education, Medical, and approximately in all engineering domains. Computational theories have enhanced the human performance in all the domains and we got the success in achieving error-free goals.

Neural Networks is a part of Artificial Intelligence and lies in the domain of computational theory. “The development of artificial neural networks began approximately 50 years ago, motivated by a desire to try both to understand the brain and to emulate some of its strengths” (Fausett L., 2008).

Artificial neural networks works like an “information-processing system” and these neural networks have some performance characteristics which are also common in biological neural networks (Fausett L., 2008).

6.1. *Proposed Solution*

In our case the RV of a software requirement is calculated manually which induces biasness and it takes too much time. The RV is calculated using neural network in order to remove or reduce the element of biasness from VIRP and in order to enhance the efficiency of the technique in terms of time. The eq. 3.1 is used to calculate RV.

$$RV = 0.35 + 0.02 \left\{ \sum_{i=1}^5 pRCF_i + \sum_{i=1}^5 rRCF_i \right\}$$

A back-propagation feed forward neural network is designed in order to calculate the RV of a requirement. In the first stage a neural network is designed and trained using the Input set P which is in the range of (0-50) and the training or target data T which is in the range of (0.35-1.35) so there are a total of 51 data classes. There are four steps given in the MATLAB 7 help to train a neural network which are stated as under:

1. Assemble the training data.
2. Create the network object.
3. Train the network.
4. Simulate the network response to new inputs (MATLAB 7 Help).

6.2. *Training/Validation Data*

In our case the training data is gathered using the 10 factors (Feasibility, Modifiability, Urgency, Traceability, Testability, Completeness, Consistency, Understandability, Within Scope, Non-Redundancy) described in VIRP. The values of the factors of pRCF and rRCF for each requirement are collected from industry professionals with experience more than 5 years in the industry (i.e. NESCOM, Bahria Town Automation Team, & NetSolace). In order to get the value of the factors of pRCF and rRCF for each requirement 20 projects are selected and from each project a set of 25 requirements is selected. The list of the projects is shown below.

1. Online Car Showroom
2. Online Evaluation and Management System
3. Hajians Filling Station Management System
4. Paraplegic Management System

5. Real Estate Management and Information System
6. Clinical Laboratory Management System
7. Student Project Management System
8. AV Alunched an Information and Entertainment System
9. Hardware Store Management System
10. Nadempiere CRM
11. Fly With US
12. BizTeller
13. RMS Restaurant Management System
14. Employee Management and Payroll Management System
15. Spare Part Management System
16. Books E-Books and Audio/Video Tutorial
17. Standard Automation Project of Kamal Laboratories
18. Online Multiplex System
19. Online Order Booking
20. Document Processing System

The values of the factors of pRCF and rRCF for each requirement are gathered with the help of following template.

PROJECT NAME

Functional/Non-Functional Requirements:

General:

1. The user shall be able to login to system.

pRCF

➤ Feasibility	0	1	2	3	4	5
➤ Modifiability	0	1	2	3	4	5
➤ Urgency	0	1	2	3	4	5
➤ Traceability	0	1	2	3	4	5
➤ Testability	0	1	2	3	4	5

rRCF

➤ Completeness	0	1	2	3	4	5
➤ Consistency	0	1	2	3	4	5
➤ Understandability	0	1	2	3	4	5
➤ Within Scope	0	1	2	3	4	5
➤ Non-Redundant	0	1	2	3	4	5

Assign a Requirement Value (RV) between 0.35-1.35

RV =

Industry professional will select a value of each of the factor in the range of (0-5) as given in front of each of the requirement. Suppose the professional selects the following

values i.e. 4 3 1 5 2 3 5 4 5 2 for the ten factors of pRCF and rRCF for the requirement “The cashier will be able to calculate the total sell”. So the net input to the function will be the sum of all ten factors i.e. 34 and the value of the RV will be 1.03 according to the formula. Suppose if a professional assign zero to all values then the net input to RV function will be zero and the calculated value of RV will be 0.35. If a professional assigns 0 value to 9 factors and value of 1 to one factor then the input to the RV function will be 1 as a sum of all value of the ten factors and the calculated value of RV will be 0.37. And if a professional assigns a value of 5 to all 10 factors then the sum of all ten factors will be 50 which will be treated as a net input to RV function and the calculated RV will be maximum in this case i.e. 1.35. So in this case there are 51 classes of data i.e. the input data P will be in the range of 0-50 and the target or training data T will be in the range of 0.35 to 1.35. The value of RV is also assigned, in the range of (0.35-1.35), by the professionals based on expert opinion without using the RV function. This thing induces the element of biasness because industry professionals have assigned the RV values to the requirements which are not right and the requirements priority is disturbed. Moreover a lot of time is spent by the professionals to assign the values to each and every factor of pRCF and rRCF. These two factors i.e. biasness and manual calculation of RV are reducing the overall efficiency of the VIRP technique. So in order to eliminate or reduce the impact of these two issues there is the need to fully automate and make the technique intelligent which would help in calculating actual RV value of a given requirement of software application.

6.3. Create/Train the Network Stage 1

In the first step network object is created and trained the network using given input and target data ranges with “TRAINLM” or “Levenberg-Marquardt” training function. The initial algorithm is shown below:

6.3.1 Algorithm Stage 1

```
%-Stage 1

clc

P=[0:1:50];          %-network input

fprintf('\n##### Start of Neural Net Simulation #####\n\n');
fprintf('\n##### Actual Target Values #####\n\n');

T=[0.35:0.02:1.35]          %-network target data

net = newff([0 50],[20 1],{'tansig' 'purelin'},'trainlm');    %-network object

%-network training using trainlm

net.trainParam.show=10;          %- Epochs between displays (NaN for no displays)

net.trainParam.showWindow=True; %- Show or make visible the training window

net.trainParam.showCommandLine=True; %- Generate the Command Line Output

net.trainParam.goal=1e-5;          %- Performance goal

net.trainParam.time=0.4;          %- Maximum time to train in

secondsnet.trainParam.max_fail=2;          %- Maximum validation failures

net.trainParam.mem_reduc=1;          %- Factor to use for memory/speed trade off.

net.trainParam.min_grad=1e-06;          %- Minimum performance gradient

net.trainParam.mu=0.001;          %- Initial Mu
```


%- In neural network, MU controls how much the weights are changed on each iteration. The value to use depends on the particular problem, being as low as 10^{-6} , or as high as 0.1.

(Steven W. Smith, <http://www.dspguide.com/ch26/4.htm>)

```
net.trainParam.mu_dec=0.1;           % - Mu decrease factor
net.trainParam.mu_inc=10;           % - Mu increase factor
net.trainParam.mu_max=1e+010;       % - Maximum Mu
net.trainParam.epochs =10;          % - Maximum number of epochs to train
net.performFcn='mse';               % - Performance Function
net = train(net,P,T);
fprintf('\n##### Neuro Computed Values #####\n\n');
Y = sim(net,P)                       % - Displaying the network output:
fprintf('\n##### Error Difference #####\n\n');
Z=T-Y
plot(P,Z,P,T);grid;
```

The TRAINLM or Levenberg-Marquardt training function is used and simulated the net in order to get the neuro-computed value of RV. The results given by the neural net are shown below.

6.3.2 Results Stage 1

The results generated by the neural network are divided into three parts i.e. actual target values against given data set of inputs then the neuro-computed values of RV and in the end the error-difference of neuro-computed values of RV from actual target values. The error-differences of the neuro-computed values of RV will help to find out that either the

results computed by the neural network are acceptable or not. Currently there is an acceptable error of ± 0.0042 which is negligible in order to prioritize the requirements based on the optimization of RV function.

Start of Neural Net Simulation

Actual Target Values

T =

Columns 1 through 9

0.3500 0.3700 0.3900 0.4100 0.4300 0.4500 0.4700 0.4900 0.5100

Columns 10 through 18

0.5300 0.5500 0.5700 0.5900 0.6100 0.6300 0.6500 0.6700 0.6900

Columns 19 through 27

0.7100 0.7300 0.7500 0.7700 0.7900 0.8100 0.8300 0.8500 0.8700

Columns 28 through 36

0.8900 0.9100 0.9300 0.9500 0.9700 0.9900 1.0100 1.0300 1.0500

Columns 37 through 45

1.0700 1.0900 1.1100 1.1300 1.1500 1.1700 1.1900 1.2100 1.2300

Columns 46 through 51

1.2500 1.2700 1.2900 1.3100 1.3300 1.3500

TRAINLM, Epoch 0/10, Time 4.0%, MSE 10.6387/1e-005, Gradient 591.604/1e-006

TRAINLM, Epoch 6/10, Time 11.7%, MSE 7.32483e-006/1e-005, Gradient 0.0456448/1e-006

TRAINLM, Performance goal met.

Neuro Computed Values

Y =

Columns 1 through 9

0.3508 0.3694 0.3891 0.4088 0.4286 0.4485 0.4685 0.4893 0.5086

Columns 10 through 18

0.5300 0.5487 0.5695 0.5909 0.6077 0.6324 0.6480 0.6698 0.6922

Columns 19 through 27

0.7068 0.7326 0.7489 0.7688 0.7934 0.8065 0.8314 0.8509 0.8673

Columns 28 through 36

0.8937 0.9074 0.9298 0.9530 0.9662 0.9924 1.0097 1.0281 1.0539

Columns 37 through 45

1.0666 1.0906 1.1122 1.1264 1.1533 1.1686 1.1884 1.2145 1.2261

Columns 46 through 51

1.2498 1.2749 1.2852 1.3094 1.3370 1.3438

Error Difference from Target Values

Z =

Columns 1 through 9

-0.0008 0.0006 0.0009 0.0012 0.0014 0.0015 0.0015 0.0007 0.0014

Columns 10 through 18

0.0000 0.0013 0.0005 -0.0009 0.0023 -0.0024 0.0020 0.0002 -0.0022

Columns 19 through 27

0.0032 -0.0026 0.0011 0.0012 -0.0034 0.0035 -0.0014 -0.0009 0.0027

Columns 28 through 36

-0.0037 0.0026 0.0002 -0.0030 0.0038 -0.0024 0.0003 0.0019 -0.0039

Columns 37 through 45

0.0034 -0.0006 -0.0022 0.0036 -0.0033 0.0014 0.0016 -0.0045 0.0039

Columns 46 through 51

0.0002 -0.0049 0.0048 0.0006 -0.0070 0.0062

6.3.3 Training Graph/Window with TRAINLM Stage 1

The performance training graph is shown below.

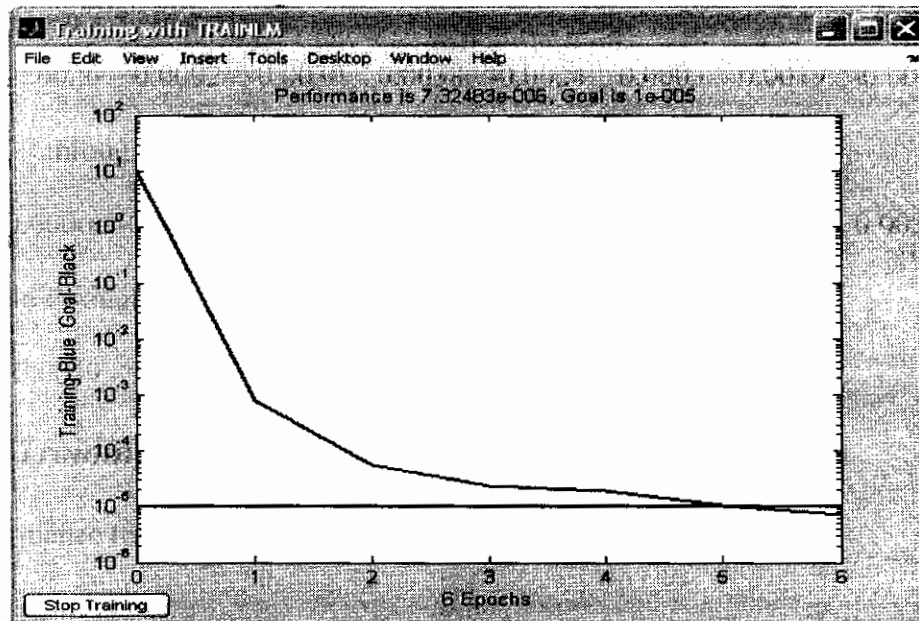


Figure 6.1: Training Graph with TRAINLM

6.4. Validation of Neural Network Stage 2

The neural network is validated using the dataset obtained from the industry experts as described in the training data section. The algorithm used to validate the neural network is shown below.

6.4.1. Algorithm Stage 2

% - Calculation of RV using Neural Net and elimination of role of experts or software engineers

% - Stage 2

clc % - Clear Command Window

clear % - Remove all Variables from the workspace

clear all % - Remove all variables, globals and functions from memory

p = [0:1:50]; % - Input Data

fprintf('\n##### Start of Neural Net Simulation #####\n\n');

fprintf('\n##### Actual Target Values #####\n\n');

t=(0.35:0.02:1.35) % - Target Data

fprintf('\n Validation starts...\n\n');

% - Load Validation Data

load 'RV.txt'; % - Load Data of RV assigned by industry expert

load 'TSample.txt'; % - Load Target Data

% For simulation of a set of 26 data values we may use the following syntax.

```
validation.P= RV(225:250);    % - Validation Test Inputs from column 225 to column 250
```

```
validation.T= TSample(225:250);    % - Validation Test Targets from column 225 to  
column 250
```

% For simulation of a set of 500 data values we may use the following syntax.

```
validation.P= RV(1:500);    % - Validation Test Inputs from column 1 to column 500
```

```
validation.T= TSample(1:500);    % - Validation Test Targets from column 1 to column  
500
```

```
tic    % - Start Timer
```

% - We can increase the data range according to our requirements even if the data is in thousands.

% - Initialization of Neural Network

% - Hidden layer is using 20 tansig functions and the output layer is using 1 purelin or activation function.

% - Levenberg-Marquardt training function 'trainlm' is used which is most efficient one.

```
net = newff(minmax(p),[20 1],{'tansig' 'purelin'},'trainlm');    % - Network Object
```

% - Plot the original data points (target values) and the untrained output

```
y = sim(net,p);
```

```
figure(1)
```

```
plot(p,t,'+',p,y,'o');  
grid;  
legend('FeededData','Untrained Output');  
title('Data and Untrained Network Output');  
  
% - Initialization of weights and biases.  
net.LW{2,1} = net.LW{2,1}*0.0000001;  
net.b{2} = net.b{2}*0.0000001;  
  
% - Network Training  
net.trainParam.show=10;           % - Epochs between displays (NaN for no displays)  
net.trainParam.showWindow=True;   % - Show or make visible the training window  
net.trainParam.showCommandLine=True; % - Generate the Command Line Output  
net.trainParam.goal=1e-5;         % - Performance goal  
net.trainParam.time=0.4;          % - Maximum time to train in seconds  
net.trainParam.max_fail=2;        % - Maximum validation failures  
net.trainParam.mem_reduc=1;       % - Factor to use for memory/speed trade off.  
net.trainParam.min_grad=1e-06;    % - Minimum performance gradient  
net.trainParam.mu=0.001;          % - Initial Mu  
  
%- "In neural network, MU controls how much the weights are changed on each iteration.  
The value to use depends on the particular problem, being as low as  $10^{-6}$ , or as high as  
0.1". (Steven W. Smith, http://www.dspguide.com/ch26/4.htm)
```

```

net.trainParam.mu_dec=0.1;           % - Mu decrease factor
net.trainParam.mu_inc=10;           % - Mu increase factor
net.trainParam.mu_max=1e+010;       % - Maximum Mu
net.trainParam.epochs =10;          % - Maximum number of epochs to train
net.performFcn='mse';                % - Performance Function

%NETWORK DESCRIPTION

%net - Network.
%p - Network inputs.
%t - Network targets.
%[] - Structure of validation vectors.
%[] - Structure of test vectors.

    %tr - Training record (epoch and perf).

[net,tr]=train(net,p,t,[],[],validation);    % - Network Validation

% - The validation data is:
fprintf('\n##### Neural Net Validation Input #####\n\n');
Input_Data=validation.P
fprintf('\n##### Neural Net Training Input #####\n\n');
Input_Training=validation.T

```



```
% - Displaying the network output:

fprintf('\n##### Neural Net Output #####\n\n');

Y=sim(net,validation.P)

% - Showing error in the original and feeded target values to trained net

fprintf('\n##### Error Difference #####\n\n');

Err=validation.T-Y

% - Sorting of the validation data in descending order

fprintf('\n##### Sorting of Validation Data#####\n\n');

ascend1 = sort (validation.P);

Sorted_Input = ascend1(length(ascend1):-1:1)

% - Sorting of the validation data and output in descending order which is shown on
workspace.

fprintf('\n##### Sorting of Output Data#####\n\n');

ascend2=sort(Y);

Sorted_Output = ascend2(length(ascend2):-1:1)

% - Plot the data and network output

figure(2)

plot(p,t,'*',validation.P,validation.T,'o',validation.P,Y,'+');

grid;
```

```

legend('Training Data','TestingData','Output Data');
title('Validation Data and Trained Network Output');

```

```

toc          % - Stop Timer

```

6.4.2. Results Stage 2

Start of Neural Net Simulation

Actual Target Values

t =

Columns 1 through 9

0.3500 0.3700 0.3900 0.4100 0.4300 0.4500 0.4700 0.4900 0.5100

Columns 10 through 18

0.5300 0.5500 0.5700 0.5900 0.6100 0.6300 0.6500 0.6700 0.6900

Columns 19 through 27

0.7100 0.7300 0.7500 0.7700 0.7900 0.8100 0.8300 0.8500 0.8700

Columns 28 through 36

0.8900 0.9100 0.9300 0.9500 0.9700 0.9900 1.0100 1.0300 1.0500

Columns 37 through 45

1.0700 1.0900 1.1100 1.1300 1.1500 1.1700 1.1900 1.2100 1.2300

Columns 46 through 51

1.2500 1.2700 1.2900 1.3100 1.3300 1.3500

Validation starts...

TRAINLM, Epoch 0/10, Time 11.8%, MSE 0.809166/1e-005, Gradient 130.001/1e-006

TRAINLM, Epoch 1/10, Time 19.5%, MSE 7.66908e-006/1e-005, Gradient 0.0020957/1e-006

TRAINLM, Performance goal met.

Neural Net Validation Input

Input_Data =

Columns 1 through 15

35 29 34 35 34 34 35 31 32 35 35 33 36 34 40

Columns 16 through 25

34 34 34 34 32 34 35 35 34 34

Neural Net Training Input

Input_Training =

Columns 1 through 9

0.3500 0.7500 0.7500 0.7500 0.7500 0.3500 0.7500 0.7500 0.7500

Columns 10 through 18

0.7500 0.7500 0.7500 0.7500 1.0000 0.3500 0.7500 0.7500 0.8000

Columns 19 through 25

0.3500 0.7500 0.7500 0.3500 0.7500 0.7500 0.7500

Neural Net Output

Y =

Columns 1 through 9

1.0536 0.9306 1.0279 1.0536 1.0279 1.0279 1.0536 0.9662 0.9934

Columns 10 through 18

1.0536 1.0536 1.0090 1.0665 1.0279 1.1538 1.0279 1.0279 1.0279

Columns 19 through 25

1.0279 0.9934 1.0279 1.0536 1.0536 1.0279 1.0279

Error Difference

Err =

Columns 1 through 9

-0.7036 -0.1806 -0.2779 -0.3036 -0.2779 -0.6779 -0.3036 -0.2162 -0.2434

Columns 10 through 18

-0.3036 -0.3036 -0.2590 -0.3165 -0.0279 -0.8038 -0.2779 -0.2779 -0.2279

Columns 19 through 25

-0.6779 -0.2434 -0.2779 -0.7036 -0.3036 -0.2779 -0.2779

Sorting of Validation Data#####

Sorted_Input =

Columns 1 through 15

40 36 35 35 35 35 35 35 35 34 34 34 34 34 34

Columns 16 through 25

34 34 34 34 34 33 32 32 31 29

```
##### Sorting of Output Data#####
```

```
Sorted_Output =
```

```
Columns 1 through 9
```

```
1.1538  1.0665  1.0536  1.0536  1.0536  1.0536  1.0536  1.0536  1.0536
```

```
Columns 10 through 18
```

```
1.0279  1.0279  1.0279  1.0279  1.0279  1.0279  1.0279  1.0279  1.0279
```

```
Columns 19 through 25
```

```
1.0279  1.0279  1.0090  0.9934  0.9934  0.9662  0.9306
```

```
Elapsed time is 1.797000 seconds.
```

6.4.3. Graphs Stage 2

The following graph shows the untrained data i.e. input and training data and the untrained output is placed in the space in a non-linear fashion. While the target or training data is shown in a linear fashion. This graph is depicting that before training of the network the output data is non-linear without any results or with wrong results. The untrained data is shown in the form of green circles i.e. O while the target data is shown in the form of blue positive sign i.e. + which is drawn linearly in the graph.

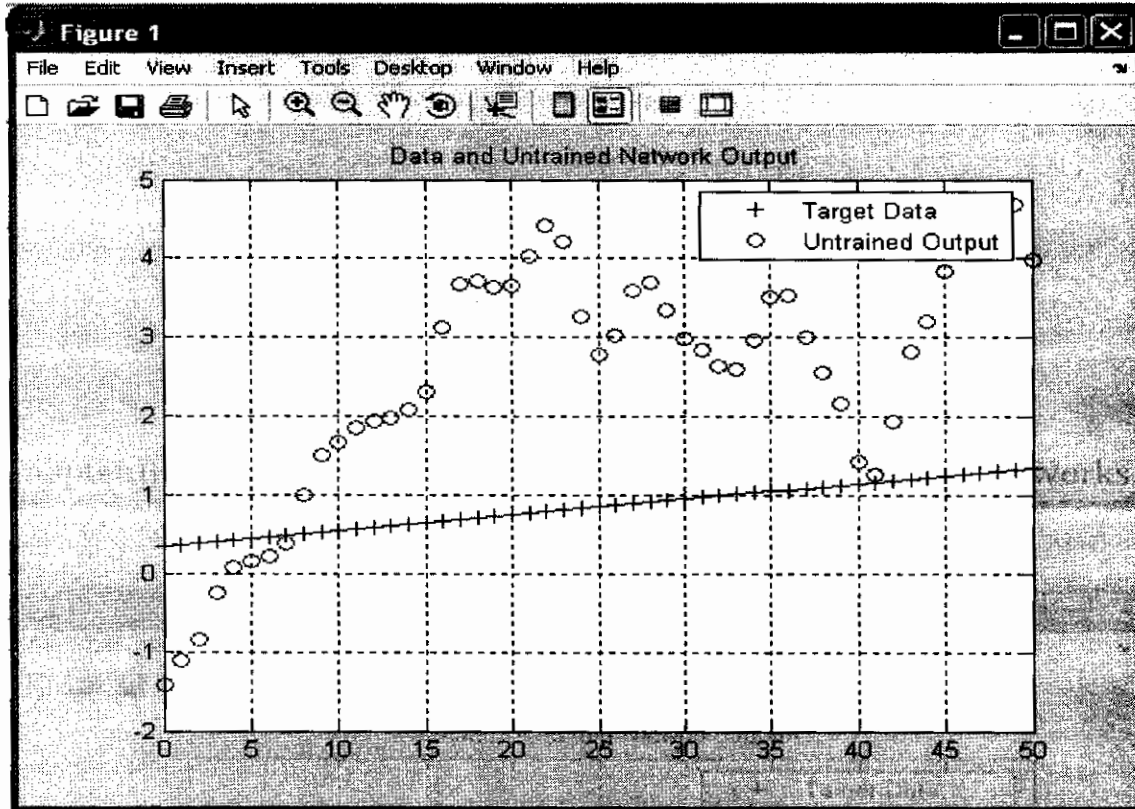


Figure 6.2: Untrained Network Output

The next graph is showing a linear fit of 25 data inputs which are selected from a data set of 500 requirements. The output data is mapping on the training data which is obvious in the graph. From experimentation it is observed that when the neural network simulates there may come an error of ± 0.0042 which is almost negligible. The elapsed time for calculation of RV for 25 data values is 1.797000 seconds which is very efficient but it may vary depending upon the machine. From the graph it is evident that the neural network is working up to its maximum performance and providing the results which are almost acceptable and these results are helping us in total elimination of the element of biasness in calculation of RV for a given requirement. It also helped in elimination of role software engineer in calculation of RV of a given requirement.

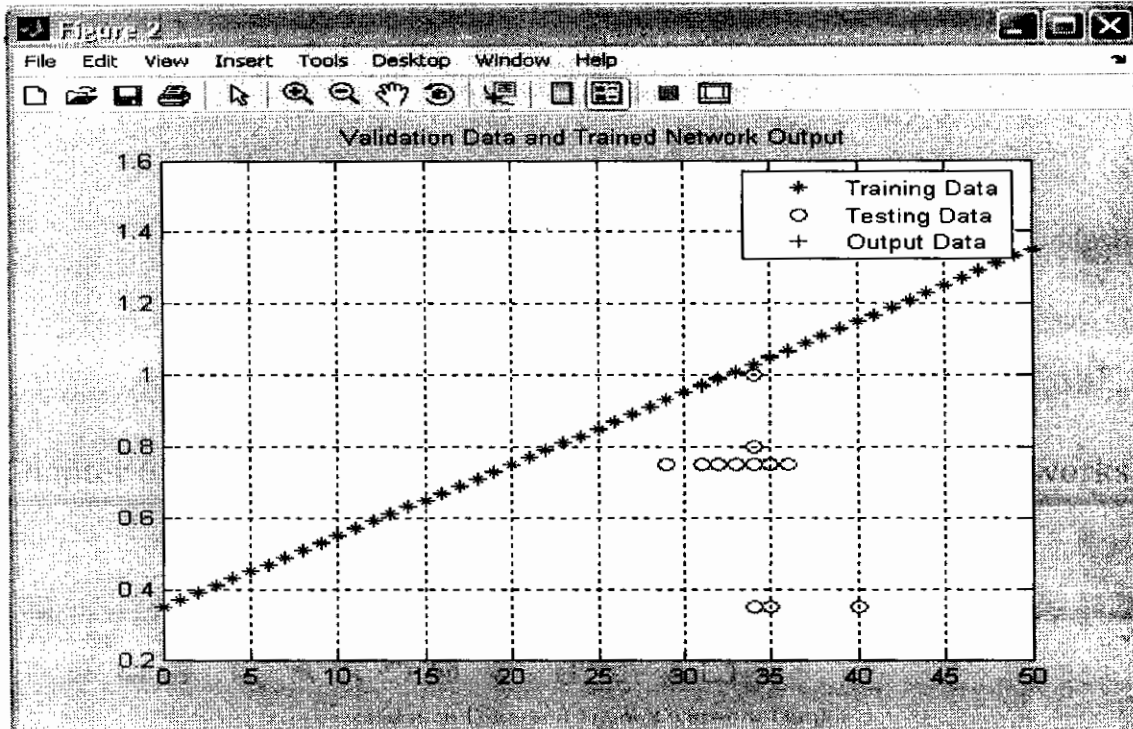


Figure 6.3: Linear Fit of Validation Data with 25 Data values.

In order to validate the RV calculation for large set of requirements i.e. ranging from hundreds to thousands the neural network is simulated comprising of a data set of 500 requirements the neural network is working with the same efficiency for the said data set as with small range of data set. The total time taken by the neural network is 2.688000 seconds which shows the effectiveness of the neural network and it may vary depending upon the machine. The automation at second level of the VIRP technique has reduced the effect of element of time consumption and biasness. The extension made in the technique VIRP is providing high performance in calculation of RV for a given set of requirements in terms of time and biasness. The technique will provide accurate and quick results without any time delays.

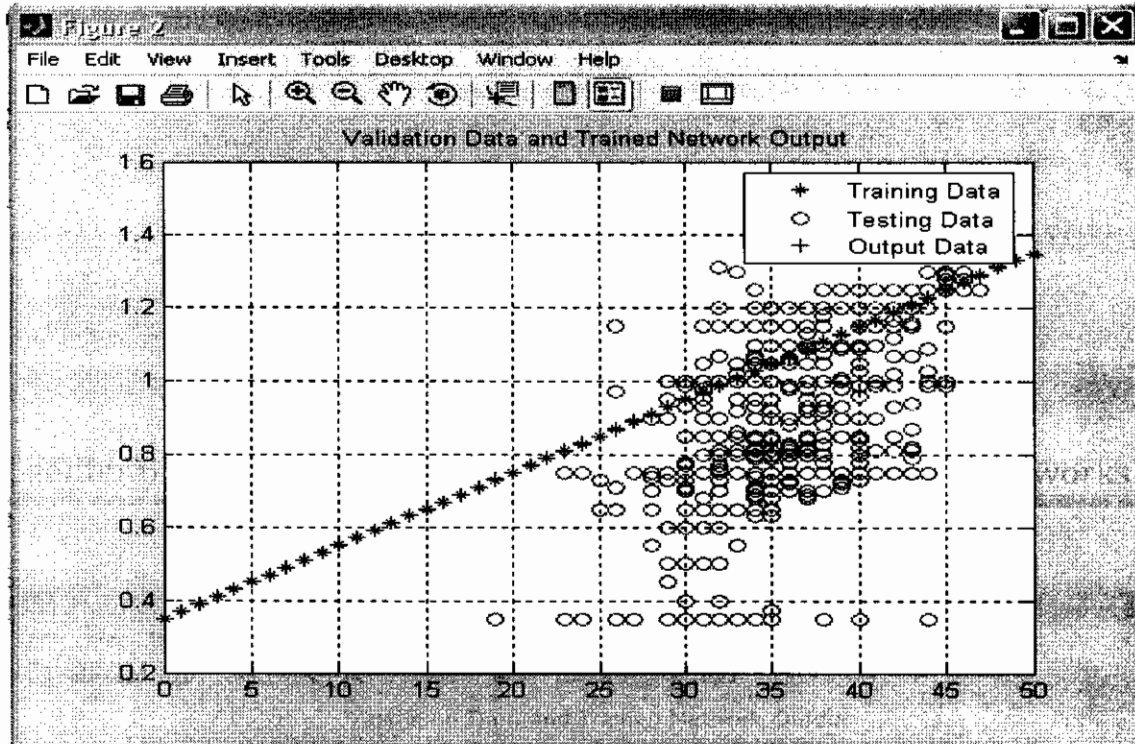


Figure 6.4: Linear Fit of Validation Data with 500 Data values.

6.4.4. Graphical User Interface (GUI Prototype)

The graphical user interface (GUI) is designed in Matlab 7 and it's a sort of prototype in order to test the performance of the designed neural network. The GUI is designed to calculate the RV of a given requirement using the 10 requirement classification factors i.e. the project specific RCFs and requirement specific RCFs. The user will select the values of these factors in the range of 0-5 and will press the start simulation button and as a result the GUI will return both the values i.e. calculated using the RV function which is given in the VIRP requirement prioritization technique and the second one will be neuro computed RV which is quite similar to the actual value computed by the function. The GUI designed for calculation of RV is user friendly and helps in enhancement of the overall performance

of VIRP technique. One can easily calculate or find out the RV of a given requirement by assigning values to 10 RCFs as shown in the GUI.

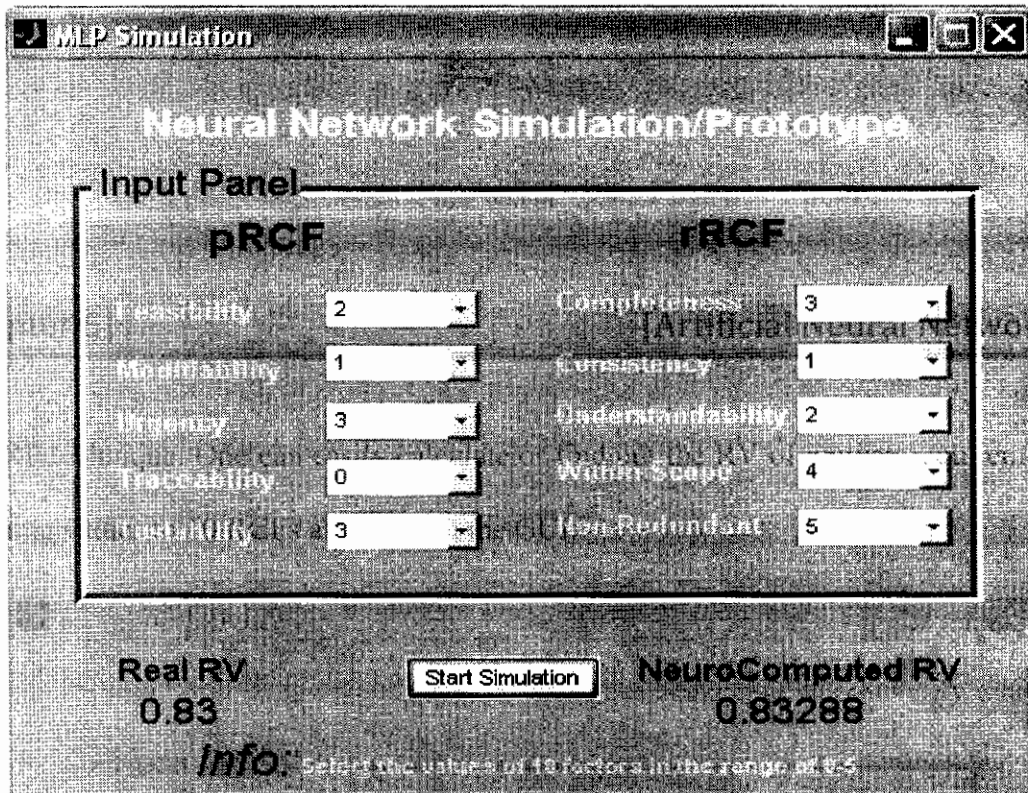


Figure 6.5: Graphical User Interface for Neuro Computed RV

Chapter # 7

Conclusion

7.1 CONCLUSION

The prioritization of the software requirements is a very complex and brainstorming phenomenon and it is also evident from literature. Researchers have proposed many techniques in order to tackle with the issues of prioritization but still there are glaring issues which are in the way of software engineers. Each and every technique is suffering from some drawback as described i.e. “The existing techniques do not provide a scalable solution when the requirements scaled up in case of large number of requirements. The existing techniques do not provide sufficient automation and are not intelligent. Most of the techniques are time consuming. The results are faulty or error prone. Results don’t recall. Mostly solve the issues of small scale requirements but with errors” (Babar et al., 2011). About the value of a requirement it is stated that “It is not easy to find out the value of a requirement in its true spirit because of the readily changeable nature of the requirements” (Babar et al., 2011). So far I have proposed an extension in VIRP model for software requirements prioritization in order to solve the issues of scalability and time consumption. In this research work a neural network based extension in VIRP model is proposed which is used to calculate the RV of a given requirement. The results obtained from the research work show that the proposed extension is quite helpful in understanding and translating the most critical requirements of a software during ‘analysis & design’ and development phases. The proposed neural network or solution also solved the issue of biasness in calculation of RV which is induced due to the heavy involvement of experts or software engineers. The technique may be used to calculate the RV of a given set of requirements i.e. ranging from hundreds to thousands after addition of neuro computed RV

in the second phase of the technique. So the VIRP model is now more scalable and time efficient in terms of calculation of RV (Requirement Value).

For future work I am doing my research in the area of software requirements prioritization which is a domain based research in which a standard set of requirements will be gathered in different domains like medical, education, engineering, etc. and each domain will be subdivided into sub domains in order to get standard requirements. And this research is based on standardization of the requirements for different domains which are stated above.

8. REFERENCES

- Abran, A. Moore, J. Dupuis, R. (2001): "*SWEBOK Guide to the Software Engineering Body of Knowledge*," IEEE.
- Aaron, K. Massey, Paul N. Otto, and Annie I. Anton. (2010): "*Prioritizing Legal Requirements*". 2009 Second International Workshop on Requirements Engineering and Law (relaw'09). IEEE
- Aurum, A., Wohlin, C. (2003): "*The Fundamental Nature of Requirements Engineering Activities as a Decision-Making Process*", Information and Software Technology, 45(14), pp. 945-954.
- Babar M. I., Ramzan M., Shahbaz A.K. Ghayyur. (2011): "*Challenges, Issues and Future Trends in Software Requirements Prioritization*". International Conference on Cyberspace, Networks and Information Technology, IEEE.
- Beck, K. (2000): "*Extreme Programming Explained: Embrace Change*", Addison-Wesley
- Berander, P. Jonsson, P. "*Hierarchical Cumulative Voting (HCV)—Prioritization of Requirements in Hierarchies*," International Journal of Software Engineering and Knowledge Engineering.
- Bergman, B., Klefsjö, B. (2003): *Quality from Customer Needs to Customer Satisfaction*, Student literature AB, Lund, Sweden.
- Boehm, B. W. (1981): "*Software Engineering Economics*", Prentice Hall, Englewood Cliffs, NJ.
- Brackett, J.W. (1990): "Software Engineering", Proceedings of *Software Engineering Institute*, 19(1.2), Carnegie Mellon University, Pittsburgh, PA.

- Carlshamre, P. (2002): "*Release Planning in Market-driven Software Product Development - Provoking an Understanding*". Requirements Engineering Journal 7 (3), pp. 139-151
- Dahlstedt, A., Persson, A. (2003): "*Requirements Interdependencies - Moulding the State of Research into a Research Agenda*", Proceedings of the Ninth International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ '03), pp. 71-80.
- Davis, A. M. (1993). "*Software Requirements, Objects, Functions and States.*" Prentice-Hall International, Inc., Englewood Cliffs, New Jersey.
- Ecklund, E. F., Delcambre, L. M. L., Freiling, M. J. (1996): "*Change Cases: Use Cases that Identify Future Requirements*". Proceedings of the 11th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA '96), pp. 342-358.
- Fausett L. (2008). "*Fundamentals of Neural Networks Architectures, Algorithms, and Applications*". 3rd Ed. Pearson Education, Inc.
- Firesmith, D. (2004): "*Prioritizing Requirements*", Journal of Object Technology, 3(8), September-October 2004
- Hatton, S. (2007). "*Early prioritization of goals*". In Advances in conceptual modeling – Foundations and applications (pp. 235-244).
- Iqbal M. A. Zaidi A. M., Murtaza S. (2010): "*A New Requirements Prioritization Model for Market Driven Products Using AHP*". International Conference on Data Storage and Data Engineering. IEEE.

- Karlsson, J. (1996): "*Software Requirements Prioritizing*". In: Proceedings of the 2nd IEEE International Conference on Requirements Engineering (ICRE1996), Colorado, USA, pp. 110-116
- Karlsson, J., Ryan, K. (1996): "Supporting the Selection of *Software Requirements*". In: Proceedings of IWSSD-8. IEEE.
- Karlsson, J., Ryan, K. (1997): "*A Cost-Value Approach for Prioritizing Requirements*", *IEEE Software*, 14(5), pp. 67-74.
- Karlsson, J., Olsson, S., Ryan, K. (1997): "*Improved Practical Support for Large-Scale Requirements Prioritizing*", *Requirements Engineering*, 2(1), pp. 51-60.
- Karlsson, J. (1998): "*A Systematic Approach for Prioritizing Software Requirements*", Linköping Studies in Science and Technology, Doctoral Dissertation No. 526, Department of Computer and Information Science, Linköping Institute of Technology, Linköping, Sweden.
- Karlsson, L., Regnell, B., Wohlin, C. (1998): "*An Evaluation of Methods for Prioritizing Software Requirements.*" *Information and Software Technology* 39, pp 939-947.
- Karlsson, L., Berander, P., Regnell, B., Wohlin, C. (2004): "*Requirements Prioritization: An Experiment on Exhaustive Pair-Wise Comparisons versus Planning Game Partitioning*". In: Proceedings of Empirical Assessment in Software Engineering (EASE2004), Edinburgh, Scotland.
- Laurent, P., Cleland-Huang, J., Duan, C. (2007) "*Towards Automated Requirements Triage*". 15th IEEE International Requirements Engineering Conference

- Lausen, S. (2002): *Software Requirements - Styles and Techniques*, Pearson Education, London, UK.
- Lehtola, L., Kauppinen, M., Kujala, S. (2004): “*Requirements Prioritization Challenges in Practice*”. Proceedings of 5th International Conference on Product Focused Software Process Improvement, Kansai Science City, Japan, pp. 497-508
- Lehtola L, Kauppinen M (2004) “*Empirical evaluation of two requirements prioritization methods in product development projects*”. In: Proceedings of the European Software Process Improvement Conference (EuroSPI 2004), Springer-Verlag, Berlin Heidelberg, pp.161-170
- Lubars, M., Potts, C., Richter, C. (1993): “*A review of the state of the practice in requirements modeling*”. Proceedings of IEEE Symposium on Requirements Engineering (RE'93), IEEE Computer Society Press.
- Maciaszek, L, A. (2001): “*Requirements Analysis and System Design -Developing Information Systems with UML*”, Addison Wesley, London, UK.
- Martin T. Hagan, Howard B. Demuth, Mark Beale. “Neural Network Design”
Matlab 7 Help.
- Moisiadis, F. (2002): “*The Fundamentals of Prioritizing Requirements*”. Proceedings of Systems Engineering/Test and Evaluation Conference (SETE2002).
- Qiao Ma. (2009): “*The Effectiveness of Requirements Prioritization Techniques for a Medium to Large Number of Requirements: A Systematic Literature Review*”. A Dissertation Submitted to Auckland University of Technology

Ramzan M., Jaffar M. A., Iqbal M. A., Anwar S., Arshad A. Shahid. (2009): “*Value Based Fuzzy Requirement Prioritization and its Evaluation Framework*”. 2009 Fourth International Conference on Innovative Computing, Information and Control. IEEE.

Ramzan M., Jaffar M. A., Arshad A. Shahid. (2011): “Value based Intelligent Requirement Prioritization (VIRP): Expert Driven Fuzzy Logic based Prioritization Technique”, Accepted in International Journal of Innovative Computing, Information and Control (IJICIC).

Ruhe, G., Eberlein, A., Pfahl, D. (2002): 'Quantitative WinWin - A New Method for Decision Support in Requirements Negotiation', *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE'02)*, pp. 159-166.

Ruhe, G. (2003): “*Software Engineering Decision Support - A New Paradigm for Learning Software Organizations*”, Advances in Learning Software Organization, Lecture Notes In Computer Science Vol. 2640, Springer-Verlag, pp. 104-115.

Ruhe, G., Eberlein, A., Pfahl, D. (2003): “*Trade-off Analysis for Requirements Selection*”, International Journal of Software Engineering and Knowledge Engineering, 13(4), pp. 345-366.

Sadiq M., Shahid M. (2009): “*Elicitation and Prioritization of Software Requirements*”. International Journal of Recent Trends in Engineering, 2(3), November 2009

Sadiq M., Ahmed J., Asim M., Qureshi A., Suman R. (2010): “*More on Elicitation of Software Requirements and Prioritization using AHP*”. 2010 International Conference on Data Storage and Data Engineering. IEEE.

- Schulmeyer, G. G. McManus, J. I. (1999): "*Handbook of Software Quality Assurance*", 3rd Edition, (Prentice Hall, Upper Saddle River, NJ).
- Sebastian Barney, Aybuke Aurum, Claes Wohlin. (2008): "*A product management challenge: Creating software product value through requirements selection*". *Journal of Systems Architecture* 54, pp. 576–593
- Shao. P.(2008): "*Sample Selection: An Algorithm for Requirements Prioritization*". ACM.
- Sommerville, I. (1996): "*Software Engineering*". 5th ed. Addison-Wesley, Wokingham, England.
- Sommerville I., Sawyer P. (1997): "*Requirements Engineering - A Good Practice Guide*", John Wiley and Sons, Chichester, UK.
- Wieggers, K. (1999): "*Software Requirements*", Microsoft Press, Redmond, WA.
- Wieggers, K. (1999): "*First Things First: Prioritizing Requirements*", Software Development.
- Yeh, A., C. (1992): "*REQUIREMENTS Engineering Support Technique (REQUEST) - A Market Driven Requirements Management Process*", *Proceedings of Second Symposium of Quality Software Development Tools*, pp. 211-223.

9. LIST OF PUBLICATIONS

Following is the list of publications that produced during the course of MS (Software Engineering).

Muhammad Imran Babar, Shahbaz A.K. Ghayyur. “A Review of Value-based Requirement Engineering for Value Webs”. International Journal of Reviews in Computing. Vol. 1. Dec. 2009. www.ijric.org.

Younas Wahab, **Muhammad Imran Babar**, Shahbaz Ahmed. “Single Repository for Software Component Selection (SRSCS): A Reusable Software Component Selection Technique”. Journal of Theoretical and Applied Information Technology, 15th April, 2011. Vol. 26 No. 1. www.jatit.org (SCOPUS)

Muhammad Imran Babar, Muhammad Ramzan, Shahbaz A.K. Ghayyur. “Challenges and Future Trends in Software Requirements Prioritization”. IEEE, ICCNIT 2011.