Towards Identification and Recognition of Trace Associations in Software Requirements Traceability

Hussain Saleem [*] **M. Zamin Ali Khan** [**] **Shiraz Afzal** [***]

[*] **Department of Computer Science,**
University of Karachi, Karachi, Pakistan

[**] **Department of Main Communications Network,**
University of Karachi, Karachi, Pakistan

[***] **Electronics Engineering Department,**
Sir Syed University of Engineering & Technology, Karachi, Pakistan

Abstract

In the field of Software Engineering, Requirements Traceability is becoming a dynamic area of research. Producing numerous artifacts is crucial activity to maintain and yield high quality software. These artifacts are created and generated throughout the lifetime of software development, and are highly interrelated where the impact of any change in either artifact imparts changes on all the relevant product outcomes. This research focuses on the identification of inherent relationships that occur and became existent in the software requirements engineering activity due to which new and hidden traces among the artifacts including those to be self-created or generated at a later juncture can be discovered as a Post-RS[1] activity. This results in the reduction of traces to be found at a later development period and offers faster identification and realization of traces among innumerable artifacts. As a consequence of this scheme, the complexity of generating traces among artifacts is reduced and the overall quality of the software improves.

Keywords: *Software Engineering, Requirement Traceability, Trace Dependency, Traceability Model, Software Traceability.*

1. Introduction

Traceability is a dynamic area of research [6]. The importance of Requirements Traceability (RT) has developed over the years since development trials became more and more iterative and interdependent where every time a new version is created with minor or major modification [7].

RT can be defined as:

"A software requirements specification is traceable if:

(i) the origin of each of its requirements is clear and if

(ii) it facilitates the referencing of each requirement in future development or enhancement documentation"

(ANSI/IEEE Standard 830-1984) [8].

Two significant types of requirements traceability are (i) Pre-RS traceability and (ii) Post-RS traceability.

Pre-RS traceability deals referentially to those facets of requirements life preceding to inclusion as RS and in the RS document, and Post-RS traceability deals referentially to those aspects of a requirements life that result since and after inclusion in the RS deliverable outcome [4]. Several policies have been anticipated to furnish the problem of traceability but much of the work has been projected in the Post-RS traceability [4].

This research focuses on the problems that are fashioned due to the inherent existent interrelationships in the software requirements. Surprisingly not much emphasis is laid down at RS phase. The Pre-RS phase comprises of informal approaches of data mining and creation of artifacts. Thus it is very difficult to find hidden relationships among requirements units and packets.

As a solution, we have proposed a framework to endow this problem. We have named the framework "IRTRR, Identification and Recognition of Traceability Relations within Requirements". The framework can be integrated with the various RT activities available as it helps in generating traces among artifacts more accurately & efficiently and at an earlier stage reducing the time required to generate traces among various artifacts and works as a post RS activity to discover the hidden relationships among requirements packets.

With the introduction in this section, the paper is organized as follows: In section-2, a discussion on traceability phenomena is presented in context to the requirement engineering. Traceability ontology is discussed in section- 3. In section-4, explanation for classifying inter-relationships for better traceability is discussed. In section-5, a proposed traceability framework to identify and recognize traceability associations is explained. The Application of the Framework with other Models is explained in section-6. The conclusion and future work is mentioned in section-7.

[1] RS: Requirement Specification

2. Discussion

According to Reference [13], "Requirement Traceability aims at understanding the complex relationship between different development artifacts. However the approach suffers from the enormous effort and complexity of creating and maintaining traceability information". We have identified the deficiency of traceability activity in the early phases of the software development. In actual sense, lack of determination of traceable links and relations being prior determinance may become one of the attributes that might cause merely a dense complexity in generating and maintaining traces among artifacts and objects.

As cited earlier in this paper that there exist inherent relationships among various packets of requirements in software, such relationships can be identified as post RS activity but most of the RT activities allow these relationships to be propagated into the design, code and testing phases. The requirements are then traced from and back to a baseline (RS) through a succession in which artifacts or object packets are scattered. Later on changes to the baseline are needed to be re-propagated through this chain to observe the impact [4]. This change activity can be reduced if some of these relationships are discovered between the requirements gathering and collection phase and the other software development activities that follow the requirements gathering phase.

3. Traceability Ontology

This is quite important to imagine the traceability occurrences, as to identify the impact of change in time varying updates in software development.

In traceability phenomena, according to the end-user perspective, traceability "of what item i.e. trace object" and "traceability in what way i.e. access and presentation of dependency link" depends on "the user… who wants it", "the task that is needed to be carried out" with "why" and "when" information, and upon the "project/query characteristics". Similarly according to the system perspective, traceability depends on "working practices of resources, time, support and ongoing cooperation & coordination", "the awareness of information required to be traceable", "the ability to obtain and document required information" and "the ability to organize and maintain required meta-trace information for end-users for restructuring or supporting change". Figure-1 describe the traceability ontology excellently with the above two perspectives [2][6].

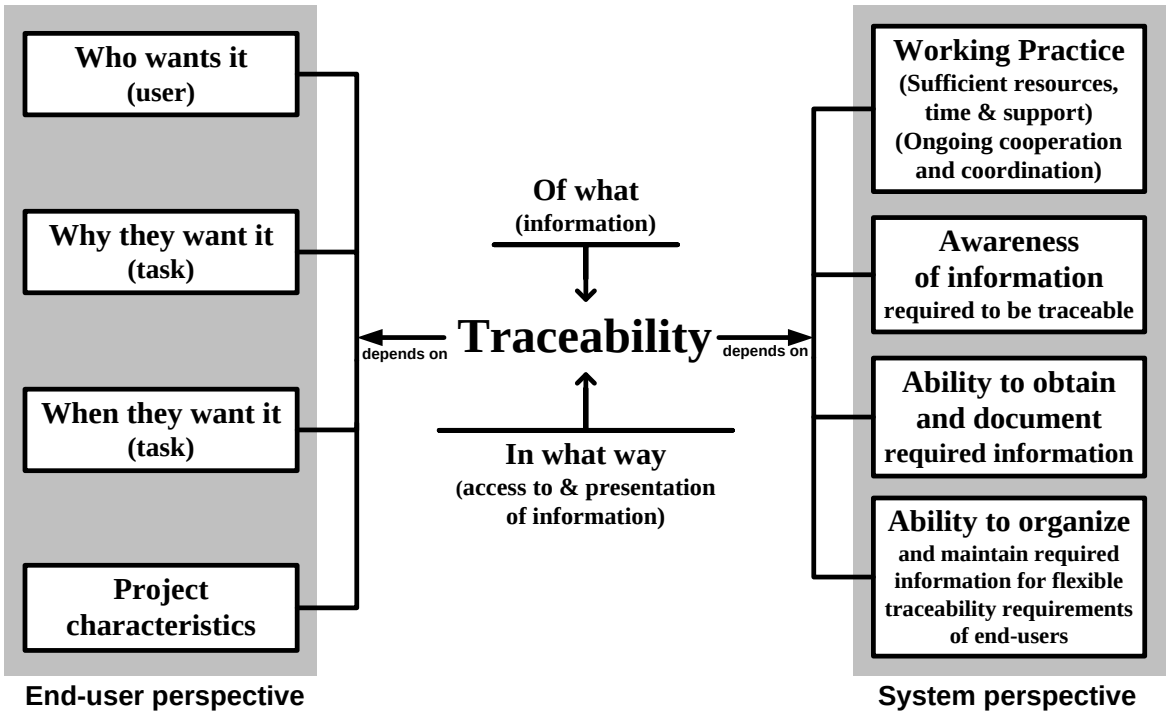


Figure-1: Traceability Ontology according to End-user and System perspective [6]

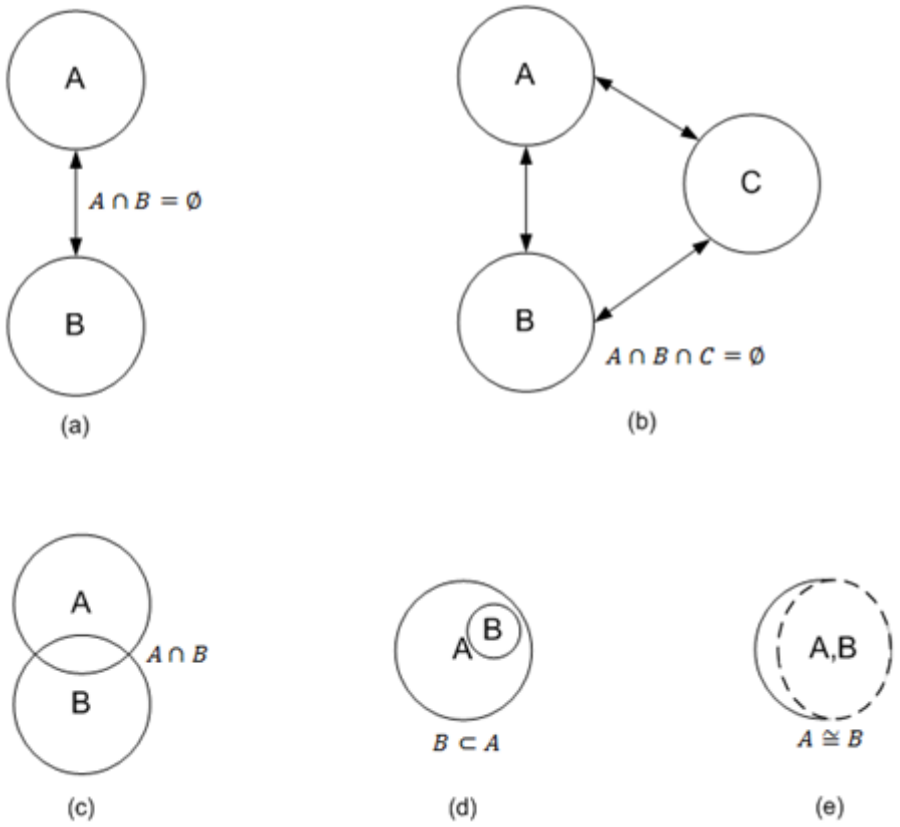


Figure-2: Views of Associations among Requirements Packets

4. Categorizing Interrelationships among Requirements Packets for Traceability

In order to analyze and develop trace links between packets parameters within the requirements, it is necessary to develop a healthier comprehension of the categorized distinct types of interrelationships that might exist in the requirements. The classification discussed here is borrowed from the set theory and mapped to the requirements engineering paradigm.

Figure-2(a) represents the disjoint but inter-linked relationship among the requirements packet explaining as nothing is common in set A and set B. Mathematically $A \cap B = \emptyset$. Figure-2(b) represents transitively connected relationship. Mathematically $A \cap B \cap C = \emptyset$. Figure-2(c) represents partially overlapped requirements, $A \cap B$. Figure-2(d) represents part-of (inscribed-circumscribed) relationship, $B \subset A$ and Figure-2(e) represents the relationship of approximately-equal-to, $A \cong B$.

5. The IRTRR Framework

The proposed framework is an activity to be performed immediately after completion of the RS phase. The framework is named as "IRTRR: Identification and Recognition of Traceability Relations within Requirements". Figure-3 best elaborate the framework. There are two key parts of the framework; A) the Identification of traceability link associations and B) the Recognition of traceability link associations. The identification part involves the steps i) Analyze and ii) Associate and the recognition part involves iii) Classify or Classification and iv) Filter. The identification part, as the name suggests, is concerned with the discovery of the hidden traces that exist within the RS document. Whereas the recognition part involves the validity and comprehension of traces generated in the identification part. Both these parts are equally important to identify and recognize the trace associations that exist inherently in software development life-cycle. The four stages of activities in IRTRR model are discussed concisely below.

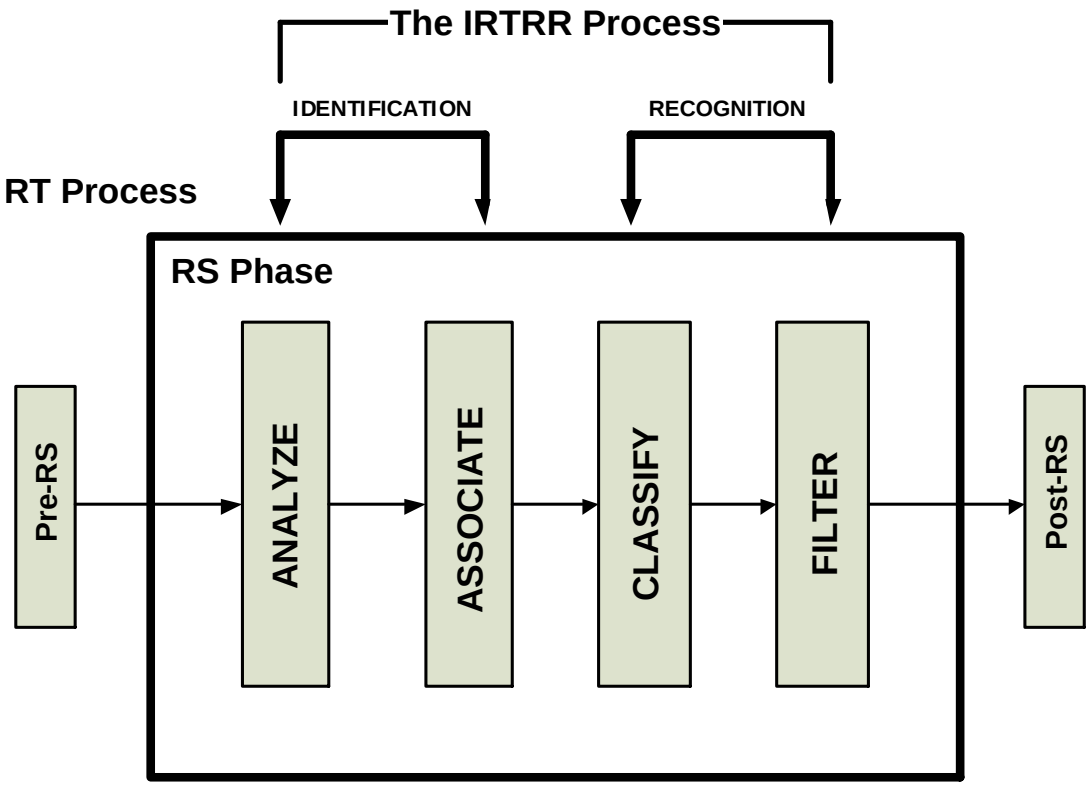


Figure-3: Requirement Traceability Process and the IRTRR Framework [5], [6]

5.1 Analyze

This step involves the detailed study and analysis of the requirement specification document and workflow prepared in the RS Phase. Analysis is the process of breaking a complex chain or substance into smaller parts to gain a better understanding of it mathematically or logically visualizing the data and footprint graph on screen (Figure-4). As a result of this activity the requirements are to be broken down into atomic product functions. This atomization is based on the product functions as described in the IEEE Recommended Practice for Software Requirements Specifications [8]. The output of this activity is a set of product functions that are atomic and cannot be further broken down.

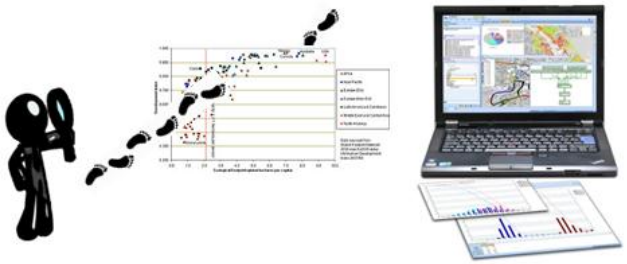


Figure-4: Software Analysis: Analyzing footprint traces of various objects through Graphs obtained from Categorical Data

Any representation can be used to represent this set such as a list but if a logical grouping of requirements is applied and the requirements are arranged in a hierarchical structure, this structure helps in developing associations among the various levels of requirements.

5.2 Associate

Once the requirements are atomized, it is important to develop association among the requirements. This association can be of one of the types described in section 4 of this paper. The association can be based on set theory as represented in figure-2. Association could be established either randomly or symmetrically between the objects as represented in figure-5. Techniques of RT can be applied to find these relationships such as “matrix sequences” [1], “RT matrices” [2], “key-phrase dependencies” [10] and “hypertext” [11] etc. The output of this activity presents a link association relationship among various requirements objects. Any appropriate representation can be chosen, for example generating a dependency graph to depict the relationships or using a meta-language.

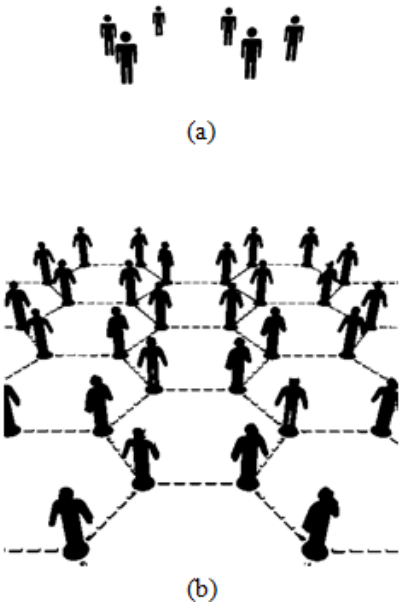


Figure-5: Traceability Association links: (a) Un-Associated objects. (b) Symmetrically Associated objects

5.3 Classify

The relationships generated as a result need to be restructured according to the product functions categories with a scheme from which the relationships are generated. Figure-6 explains the analogy to five distinct categorical classifications of object packets. This is important to comprehend the overall system and to develop a global perspective of the relationships that exist in the software system to be developed. More than one parameter can be used to classify the trace links. For N-tier application this classification can be Interface implementation, Business logic and Database management.

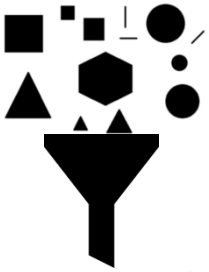
Figure-6: Analogy to five distinct categorical classifications of object packets

Mathematically, Set theory or Group theory can also be applied to determine the categories of objects. In mathematics, a classification theorem answers the classification problem "What are the objects of a given type, up to some equivalence?" It gives a non-redundant enumeration: each object is equivalent to exactly one class. A few related issues to classification are: (1) The equivalence problem is "given two objects, determine if they are equivalent". (2) A complete set of invariants, together with which invariants are realizable, solves the classification problem, and is often a step in solving it. (3) A computable complete set of invariants (together with which invariants are realizable) solves both the classification problem and the equivalence problem. (4) A canonical form solves the classification problem, and is more data: it not only classifies every class, but gives a distinguished (canonical) element of each class [15].
There exist many classification theorems in mathematics and could be applied as geometrical sketch, or algebraic sketch, or complex algebraic sketch through equations and algorithms.

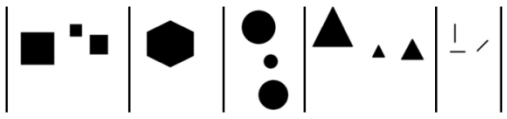
5.4 Filter

The process of developing relationships among requirements is a complex activity especially if anticipation is applied as a method of discovering relationships. Anticipation is a useful technique but may result in over anticipation. As a result some of the relationships identified might lie outside the scope of the software product to be developed. Some relationships might not mature into traces among the artifacts generated and thus can be filtered out. Figure-7 explains analogically the process of filter. The output of this activity is a set of filtered relationships among the requirements that are found from the RS artifact.

(a) Broadly scattered objects



(b) Filtering all determined objects of various variants



(c) Categorically placement

Figure-7: Categorically Filtration to recognize object groups

6. Application of the Framework with other Models

The activities described in the IRTRR framework are independent not only from the rest of the requirement engineering activities but also from the entire software engineering process. Thus it can be placed in any software-engineering model after the requirements gathering phase and before the design phase. In an iterative model such as the spiral or incremental model, where the requirement gathering activity and the design activity is repeated a number of times, the identification and recognition of trace relationships must also be repeated the same number of times. If it is viewed as RT activity, it can be attached ahead of the various techniques for traceability. Some of the techniques with which we have found the framework to be useful are "Cross Referencing Schemes" [3]; "Keyphrase Dependencies" [10]; "Templates" [9]; "RT Matrices" [2]; "Matrix Sequences" [1]; "Hypertext" [11]; "Integration Documents" [12]; "Assumption-Based Truth Maintenance Networks" and "Constraint Networks" [14] etc. The framework comply the techniques in developing traces among artifacts more efficiently and rapidly.

7. Conclusion and Future Work

We have proposed this framework and named "IRTRR". This framework can be used as a post RS activity which not only helps to reduce the time spent on the RT activity, but controls the change management process and improves the overall quality of the software. The trace links if not identified at an earlier phase of software development may result in inconsistencies and ambiguities in the software and badly affect the overall quality of the software with the impact of waste of time, cost and development effort. If identified at a later stage they may trigger a change management activity, which might impart its impact on all the relevant artifacts generated thus far.

The proposed framework presented here works as an add-on activity to the various RT techniques available to improve their utility. Future work directions include the study of a similar set of activities presented in this paper to be applied on, not only as a Post-RS activity but also as a continuous process of identification and recognition of requirement traces in all the phases of software development.

References

[1] Brown P.G., "QFD: Echoing the Voice of the Customer", AT & T Technical Journal, pp.21-31, March/April 1991.

[2] Davis A.M., "Software Requirements: Analysis and Specification", Prentice-Hall, Inc., 1990.

[3] Evans M.W., "The Software Factory: A Fourth Generation Software Engineering Environment", JW & Sons, 1989.

[4] Gotel O.C.Z. & Finkelstein A.C.W., "An Analysis of the Requirements Traceability Problem", Proc. 1st IEEE Intl. Conf. on Requirements Engineering, pp.94-102, Colorado Springs, April 1994.

[5] Hussain Saleem & *et. al.*, "Identification and Realization of Trace Relationships within Requirements", In Proc. of Intl. Conf. on Software Engineering (ICSE'06), Lahore, Pakistan, 14-15 April 2006.

[6] Hussain Saleem & *et. al.*, "Traceability Management Framework for Patient Data in Healthcare Environment", In Proc. IEEE Intl. Conf. on Computer Science & Information Technology, Chengdu, China, pp.264-8, 2010.

[7] Baxter, I., "Tools and Techniques for Maintaining Traceability during Design", IEE Colloquium, Computing and Control Division, Professional Group C1, Digest No. 1991/180, 1991.

[8] IEEE Standard 830-1998, "IEEE Recommended practice for Software Requirements Specifications", IEEE, New York, 1998.

[9] Interactive Development Environments, "Software through Pictures: Products and Services Overview", IDE Inc., 1991.

[10] Jackson J., "A Keyphrase Based Traceability Scheme", In IEE Colloquium on Tools and Techniques for Maintaining Traceability during Design, Computing and Control Division, Professional Group C1, Digest No. 1991/180, pp.2-1-2/4, 1991.

[11] Kaindle H., "The Missing Link in Requirements Engineering", ACM SIGSOFT Software Engineering Notes, Vol. 18, No.2, pp.30-39, 1993.

[12] Lefering M., "An Incremental Integration Tool between Requirements Engineering and Programming in the Large", Proc. IEEE Intl. Symposium on Requirements Engineering, San Diego, California, pp.82-89, Jan 4-6, 1993.

[13] Ramesh B., Stubbs L.C., and Edwards M. "Lessons Learned from Implementing Requirements Traceability", Crosstalk – Journal of Defense Software Engineering, 8(4):11-15, 1995.

[14] Smithers T., Tang M.X., & Tomes N., "The Maintenance of Design History in AI-Based Design", In IEE Colloquium, Computing and Control Division, Professional Group C1, Digest No. 1991/180, 1991.

[15] Wikipedia: The free encyclopedia, (Wikipedia.org) Web: http://en.wikipedia.org/wiki/Classification_theorem, accessed dated Friday, September 07, 2012 [16:21:05] PST.

Hussain Saleem is currently Assistant Professor and Ph.D. Research Scholar at Department of Computer Science, University of Karachi, Pakistan. He received B.S. in Electronics Engineering from Sir Syed University of Engineering & Technology, Karachi in 1997 and has done Masters in Computer Science from University of Karachi in 2001. He has also obtained Diploma in Statistics. With having vast experience of about 15 years of University Teaching, Administration and Research in various dimensions of Computer Science, Hussain is the Founder member cum Pioneer of High Speed Fiber-based LAN Establishment Project with wireless computing support at University of Karachi, which became the largest campus network of Asia. He is also founder member of the Main Communication Networks Department, University of Karachi. Moreover, he is Academic Head of Computer Application Compulsory courses at Faculty of Arts and Sciences at University of Karachi. He is the Author of several International Journal publications. His field of interest is Software Science, System Automation, Hardware design engineering, and Simulation & Modeling. He is member of Pakistan Engineering Council (PEC).

Dr. M. Zamin Ali Khan is a Head of Main Communication network department at University of Karachi. He has received B.E. (Electrical Engineering) from NED University, Karachi, Pakistan and MS (Electrical & Computer Engineering) from Concordia University, Montreal, Canada and PhD in Computer Science from University of Karachi. He has more than 18 years of experience of teaching and industry. He has worked in Victhom Human Bionics, Canada as an Engineer Scientist. Currently. He is a senior member of Pakistan Engineering Council, Canadian Engineering Council and IEEE. His research interest includes VLSI, Digital design, Digital signal processing and Analog front end of wireless devices.

Shiraz Afzal is a full time faculty member of SSUET. He received his B.E. degree in Electronics from Sir Syed University of Engineering and Technology Karachi, Pakistan and M.E degree in Electronics, specialization in Micro-System design from NED University of Engineering & Technology Karachi, Pakistan in 2006 and 2012 respectively. His research interest includes Microelectronic circuit design. He is also a member of PEC.