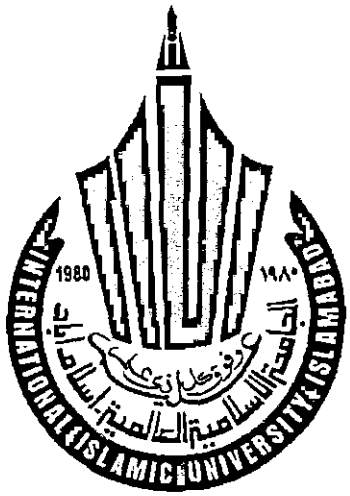
ARCHITECTURE REQUIREMENTS ENGINEERING ACCURACY AND ERROR-THE ANALYSIS METHOD (AREA-TEAM)



Muhammad Amir Aman
Muhammad Sulayman

Supervised By

Prof. Dr. Khalid Rashid

Department of Computer Science
Faculty of Applied Sciences
International Islamic University Islamabad
2006

In
the
Name
of

ALLAH

The Most Merciful
The Most Beneficent

INTERNATIONAL ISLAMIC UNIVERSITY, ISLAMABAD
DEPARTMENT OF COMPUTER SCIENCE

April 08, 2006

FINAL APPROVAL

It is certified that we have read the research thesis submitted by Mr. Muhammad Amir Aman 42-FAS/MSSE/F04 and Mr. Muhammad Sulayman 43-FAS/MSSE/F04 and it is our judgment that this thesis is of sufficient standard to warrant its acceptance by International Islamic University Islamabad for the MS in Software Engineering.

COMMITTEE

External Examiner

Dr. Hafiz Farooq Ahmad
Associate Professor,
NUST Institute of Information Technology,
H# 166-A, St. # 09, Chaklala Scheme III,
Rawalpindi-Pakistan

Internal Examiner

Dr. Sikandar Hayat Khiyal
Head,
Department of Computer Science,
International Islamic University,
Islamabad-Pakistan

Supervisor

Prof. Dr. Khalid Rashid
Dean,
Faculty of Applied Sciences,
International Islamic University,
Islamabad-Pakistan

A
Dissertation
Submitted as Partial Fulfillment
of Requirements for the Award of The Degree of
MS in Software Engineering

To
The Holiest man ever born,
Prophet Muhammad (Peace Be Upon Him)
&
Our Dear Parents & Family
Who are an embodiment of diligence and honesty,
Without their prayers and support
This dream could have never come true

DECLARATION

We hereby declare that this research neither as a whole nor as a part thereof has been copied out from any source. It is further declared that we have conducted this research and developed this software architecture analysis method, case study and accompanied thesis on the basis of our personal efforts, under the sincere guidance of our supervisor. If any artifact of this research is proved to be copied out from any source or found to be reproduction of someone else, we shall stand by the consequences.

Muhammad Amir Aman
42-FAS/MSSE/F04
Muhammad Sulayman
43-FAS/MSSE/F04

ACKNOWLEDGEMENTS

Countless gratitude to the Almighty Allah, who is Omnipotent and He, who blessed us with the ability to read and write. He blessed us with a chance and choice, health and courage to achieve this goal.

We would like to pay special regards to our parents Dr. Muhammad Tufail, Mrs. Naseem Tufail, Prof. Aman Ullah Khan and Mrs. Khalida Aman. Without their support and prayers we could have never completed this research work. We also want to thank Mrs. Asma Amir who supported the authors specially her husband during their degree program. Let us not forget the prayers of our brothers, sisters and beloved friends. They all always helped us in our down times and boosted our morals. We would like to pay them the very special thanks for their best wishes, encouragement and support in not only this research but throughout our lives without that we would have not been able to achieve anything worthwhile.

We would like to pay special thanks to our supervisor Prof. Dr. Khalid Rahsid , Dean, Faculty of Applied Sciences for his timely guidance, encouragement and personal interest to maintain the quality of work apart from his duties throughout the research. Thanks to Dr. Sikandar Hayat Khiyal, Head, Department of Computer Science and Dr. Sy. Afaq Hussain, Head, Department of Telecommunication & Computer Engineering, International Islamic University, Islamabad without theirproper guidance we could have never achieved this milestone.

We would also like to mention the support of HEC and PTCL who generously provided us R&D Funds for both the international conferences. We also owe a great deal to all the respected colleagues who extended their help whatsoever we needed.

Muhammad Amir Aman

Muhammad Sulayman

PROJECT IN BRIEF

Project Title:

Architecture Requirements Engineering Accuracy and Error- The Analysis Method (AREA-TEAM)

Objectives:

- Development of a software architecture analysis method
- Mapping of requirements on the software architecture
- Quantification of error factor present in the architecture
- To discover and explore the benefits associated with this new method
- To specify how the method satisfies the stakeholders

Undertaken By:

Muhammad Amir Aman
42-FAS/MSSE/F04
Muhammad Sulayman
43-FAS/MSSE/F04

Supervised By:

Prof. Dr. Khalid Rashid
Dean,
Faculty of Applied Sciences,
International Islamic University Islamabad

Started On:

September 2005

Completed On:

February 2006

Research Area:

Software Engineering, Software Architecture, Software Architecture Analysis, Software Requirements Engineering

ABSTRACT

Since the software architecture is a key artifact of the software development life-cycle, so architectural analysis is also a key practice involved in the process. Software architectures are usually complex and developed on the quality attributes based on preliminary requirements. Requirements are specified during requirements engineering phase and normally are incomplete and ambiguous and their effects are propagated in the software architecture. The devised method "Architecture Requirements Engineering Accuracy and Error-The Analysis Method (AREA-TEAM)", provides solid technical foundations for performing architectural analysis. It is based on the analysis of software architecture with respect to the requirements and finding the architecture accuracy factor and architecture error factor. The major contribution of AREA-TEAM is quantification of the error made by the architect as well as the stakeholder, generally the customer. The AREA-TEAM is a method that has been applied and tested on Islamabad Stock Exchange's software architecture of its data warehouse solution. AREA-TEAM is refined in practice over the research tenure.

TABLE OF CONTENTS

Ch. No.	Contents	Page No.
1	Introduction	1
	1.1 Background	3
	1.1.1 System Software Architecture	3
	1.1.2 Software Architecture Evaluation	4
	1.2 Research Paradigms on Software Architecture	5
	1.3 Research Agenda	6
	1.3.1 Objectives	6
	1.3.2 Approach	7
	1.4 Validation of the Research	7
	1.4.1 The Organization	7
	1.4.2 Data Warehouse Project of ISE	9
	1.5 Overview of Thesis	10
	1.5.1 Publications related to this thesis	11
2	Literature Survey	13
	2.1 The Quantified Design Space (QDS)	15
	2.2 Software Architecture Analysis Method (SAAM)	16
	2.3 Architecture Trade-Off Analysis Method (ATAM)	17
	2.4 Cost Benefit Analysis Method (CBAM)	18
	2.5 Summary of Research	19
3	Problem Definition	21
	3.1 Hypothesis	22
	3.2 Significance of the Research	23
	3.3 Problem Statement	23
4	Proposed Solution	26
	4.1 The Conceptual Model	27
	4.2 Development of the Method	28
	4.2.1 Participants in AREA-TEAM	29
	4.2.2 Outputs of the AREA-TEAM	31
	4.2.3 Phases of the AREA-TEAM	33
	4.2.4 Steps of the AREA-TEAM Phases	34
5	Case Study	41
	5.1 Start of the Evaluation	42
	5.2 Phase 1 of Evaluation	42
	5.3 Phase 2 of Evaluation	43
	5.4 Business Drivers	43
	5.5 Architecture Available for Evaluation	44
	5.6 Step-wise Evaluation of the Architecture	45

6 Results & Conclusion.....64
6.1 The Benefits.....65

References & Bibliography.....68

Appendix A- Publication # 1

Appendix B- Publication # 2

Appendix C- Islamabad Stock Exchange: Data Warehouse Project

Chapter 1

INTRODUCTION

1 Introduction

An architecture is the result of a set of business and technical decisions. There are many influences at work in its design, and the realization of these influences will change depending on the environment in which the architecture is required to perform. An architect designing a system for which the real-time deadlines are believed to be tight will make one set of design choices; the same architect, designing a similar system in which the deadlines can be easily satisfied, will make different choices. And the same architect, designing a non-real-time system, is likely to make quite different choices still. Even with the same requirements, hardware, support software, and human resources available, an architect designing a system today is likely to design a different system than might have been designed five years ago.

In any development effort, the requirements make explicit some—but only some—of the desired properties of the final system. Not all requirements are concerned directly with those properties; a development process or the use of a particular tool may be mandated by them, but the requirements specification only begins to tell the story. Failure to satisfy other constraints may render the system just as problematic as if it functioned poorly.

When a stage has reached that an architect has designed and documented an architecture. This leads to discuss how to evaluate or to analyze the architecture to make sure it is the one that will do the job. One of the most important truths about the architecture of a system is that knowing it will tell important properties of the system itself—even if the system does not exist yet. Architects make design decisions because of the downstream effects they will have on the system(s) they are building, and these effects are known and predictable. If they were not, the process of crafting an architecture would be no better than throwing dice: We would pick an architecture at random, build a system from it, see if the system had the desired properties, and go back to the drawing board if not. While architecture is not yet a cookbook science, we know we can do much better than random guessing.

Architects by and large know the effects their design decisions will have. Architectural tactics and patterns in particular bring known properties to the systems in which they are used. Hence, design choices—that is to say, architectures—are analyzable. Given an architecture, we can deduce things about the system, even if it has not been built yet.

Why evaluate an architecture? Because so much is riding on it, and because you can. An effective technique to assess a candidate architecture—before it becomes the project's accepted blueprint—is of great economic value. With the advent of repeatable, structured methods (such as the ATAM, presented in Chapter 2), architecture evaluation has come to provide relatively a low-cost risk mitigation capability. Making sure the architecture is the right one simply makes good sense. An architecture evaluation should be a standard part of every architecture-based development methodology.

It is almost always cost-effective to evaluate software quality as early as possible in the life cycle. If problems are found early, they are easier to correct—a change to a requirement, specification, or design is all that is necessary. Software quality cannot be appended late in a project, but must be inherent from the beginning, built in by design. It is in the project's best interest for prospective candidate designs to be evaluated (and rejected, if necessary) during the design phase, before long-term institutionalization.

1.1 Background

This research encompasses system software architecture and its analysis. Before going into further details of AREA-TEAM, we need to have sound understanding and knowledge of system software architecture and its analysis.

1.1.1 System Software Architecture

There are various schools of thought regarding the concepts of system software architecture. Since the evolution of the field of software engineering Software

Engineering Institute (SEI), Carnegie Mellon University has been evolved as the most profound source of knowledge. We also support SEI's concepts on software architecture.

The Software Engineering Institute (SEI) defines what software architecture is. This definition comprises of what constitutes a software architecture.

> *"The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them. [1] "*

Importance

Software architecture forms the backbone for building successful software-intensive systems [2]. Architecture largely permits or precludes a system's quality attributes such as performance or reliability. Architecture represents a capitalized investment, an abstract reusable model that can be transferred from one system to the next. Architecture represents a common vehicle for communication among system's stakeholders, and is the arena in which conflicting goals and requirements are mediated. The right architecture is the linchpin for software project success. The wrong one is a recipe for disaster.

1.1.2 Software Architecture Evaluation

An organization should analyze a software or system architecture, because it is a cost-effective way of mitigating the substantial risks associated with this highly important artifact [3]. Architectures are the blueprints for a system, and the carriers of the system's quality attributes.

Most complex software systems are required to be modifiable and have good performance. They may also need to be secure, interoperable, portable, and reliable. But for any particular system, what precisely do these quality attributes - modifiability,

security, performance, reliability - mean? Can a system be analyzed to determine these desired qualities? How soon can such an analysis occur? How do we know if a software architecture for a system is suitable without having to build the system first?

Experience has shown that the quality attributes of large software systems live principally in the system's software architecture. In such systems the achievement of qualities attributes depends more on the overall software architecture than on code-level practices such as language choice, detailed design, algorithms, data structures, testing, and so forth. It is therefore a critical risk mitigation measure to try to determine, before a system is built, whether it will satisfy its desired qualities.

The Software Engineering Institute (SEI) has developed several methods for analyzing system and software architectures — Active Reviews for Intermediate Designs (ARID), the Architecture Tradeoff Analysis Method (ATAM), and the Cost-Benefit Analysis Method (CBAM). These techniques can be used in combination to obtain early and continuous benefits. ARID can be used to evaluate early designs or portions of designs for their viability in satisfying stakeholder concerns. Once the software architecture is more fully developed, the ATAM can be used to reveal how well the architecture satisfies particular quality attribute requirements and the risks, sensitivities, and tradeoffs involved in satisfying those requirements. The CBAM guides system engineers and other stakeholders to determine the economic tradeoffs associated with the architectural decisions that result in the system's qualities.

1.2 Research Paradigms on Software Architecture

Over the past decade, software architecture research has emerged as the principled study of the overall structure of software systems, especially the relations among subsystems and components. From its roots in qualitative descriptions of useful system organizations, software architecture has matured to encompass broad explorations of notations, tools, and analysis techniques. Whereas initially the research area interpreted software practice, it now offers concrete guidance for complex software design and development.

We can understand the evolution and prospects of software architecture research by examining the research paradigms used to establish its results. These are, for the most part, the paradigms of software engineering. We advance our fundamental understanding by posing research questions of several kinds and applying appropriate research techniques, which differ from one type of problem to another, yield correspondingly different kinds of results, and require different methods of validation. Unfortunately, these paradigms are not recognized explicitly and are often not carried out correctly; indeed not all are consistently accepted as valid. This retrospective on a decade-plus of software architecture research examines the maturation of the software architecture research area by tracing the types of research questions and techniques used at various stages. We will see how early qualitative results set the stage for later precision, formality, and automation and how results build up over time. This generates advice to the field and projections about future impact. [4]

1.3 Research Agenda

Much work has been done in the area of software architecture analysis. Software Engineering Institute (SEI) at Carnegie Mellon University, USA is leader of the research relating to this area. Many software architecture analysis methods have been developed by SEI that has opened up many interesting areas for research. Turning the pages of recent SEI reports and Software Architecture conferences reveals a wealth of research directed towards architecture analysis, but the analysis of the architecture w.r.t. the requirements is missing. This thesis seeks the development of an architecture analysis method w.r.t. the software requirements.

1.3.1 Objectives

The objectives identified from the problem statement are given as follows:

- Development of a software architecture analysis method

- Mapping of requirements on the software architecture
- Quantification of error factor present in the architecture
- To discover and explore the benefits associated with this new method
- To specify how the method satisfies the stakeholders

1.3.2 Approach

The selection of appropriate research methodology is an important decision in a research project. Software engineering as a field offers many competing research paradigms and methodologies. The methodology employed in this thesis is to develop a new method for analysis or evaluation research paradigm [5]. Software engineering research answers questions about methods of development or analysis, about details of designing or evaluating a particular instance, about generalizations over whole classes of systems or techniques, or about exploratory issues concerning existence or feasibility. Methods of analysis or evaluation help software engineers and architects to answer, "How can I evaluate the quality/correctness of X?" or How do I choose between X and Y? We have designed a new software architecture analysis method.

1.4 Validation of the Research

For the validation of our method and prove the claims, it was decided to test the method on an industrial case study. For this purpose an industrial-academia liaison with Islamabad Stock Exchange (ISE) was established on Data Warehouse project. For analysis of the software architecture of Data Warehouse project of ISE, series of meetings, surveys and reviews with the software engineering team of Islamabad Stock Exchange were conducted.

1.4.1 The Organization

Islamabad Stock Exchange (Guarantee Limited) is the youngest stock exchange in Pakistan. It came into existence on October 1989. ISE started trading on 8th October,

1992. At that time there were very few listed companies and members with ISE. But with the passage of time ISE showed extensive improvements in terms of companies, members and business volumes. There are 101 corporate as well as individual members listed with ISE. There are 285 listed companies eligible to be traded by ISE.

Islamabad Stock Exchange (Guarantee Limited) is a place where second hand securities can be bought and sold. If a share or debenture is listed, its holder can sell it on the exchange of the desired bidder. Company must be listed if its shares are to be traded in the stock exchange. There are specified members of Islamabad Stock Exchange who can trade on behalf of their clients. As by law, only a member is allowed to conduct trading in the stock exchange. Since there is a bulk of transactions going on in a stock exchange so members themselves can't involve themselves directly into the business trading activity. They have their representatives called agents who perform the trading as ordered by the member and its client.

The stock exchange is one of the very important places for mobilizing national resources and broadening industrial ownership base to promote economic development of the country. The stock exchange, the world over has assumed a very important and vital place in the sphere of industrial and business finance. Because of its role in promoting investment climate and capital formation, it also ensures the maximum opportunities of equities participation for growth and expansion of small and medium size industries in the country. The stock exchange redirects the capital lying idle with the potential investor to industry and commerce.

Islamabad Stock Exchange is rightly considered to be a barometer of Pakistan's economy. If the economy and politics of the country are stable then the market will flourish and vice versa. Islamabad Stock Exchange satisfies itself that the company is substantial, its shares are legally issued and the company agrees to issue adequate, timely public notices for its financial position, for the closure of its books i.e. purpose of dividend, right and bonus issues.

Trading Scenario

All the stock holders meet at the same place and trading competition between them ensures that the best price is obtained. Only registered companies with the stock exchange can be traded in a stock exchange. The companies listed with the stock exchange are compelled to disclose more information that is demanded by law.

Overall Market Performance

Market performance is judged by the network index that is calculated at the end of each day. If the share prices increase then the index progresses and if the prices decrease then the index shows a downward trend.

Company Distributions

Registered companies distribute a certain profit to their clients depending upon their performance throughout the year. Company distribution plays a major role in the fluctuation of the stock prices. Similarly companies are required to conduct the board meetings annually and after regular intervals of time.

1.4.2 Data Warehouse Project of ISE

The project that was selected as a case study is Data Warehouse project of ISE. The system is designed and developed to meet the following objectives:

- ✓ To provide a full fledged and very less error prone online transaction processing system for a stock exchange. It includes real time transaction processing among different agents working under specific members of a stock exchange.

- ✓ To enhance the decision support for the stock exchange by developing a data warehouse for bulk storage and efficient retrieval of data. The operational database

cannot be used for deep data drilling and decision making, so a data warehouse can be used to store huge volumes of data i.e. up to Terabytes. It also includes the replication, loads and stores of data among distributed databases.

- ✓ To improve the overall business scenario of the stock exchange by retaining the customers. This will help in increasing the market capital and overall business activities. It will lead to gather maximum customers (investors) and to motivate the people to invest in the stock market.

- ✓ To develop software, that provides business intelligent data of a stock exchange on handheld Palm device. Synchronizing the Palm device with the data source for quick and efficient data retrieval.

For discussion about the modules of the Data Warehouse project please see Appendix C.

1.5 Overview of Thesis

This section provides an overview of the research presented over the following five chapters.

Chapter 2 presents a critical analysis of previous research relevant to this thesis. Some of the legacy techniques to analyze the software w.r.t. requirements are discussed. A number of existing software architecture analysis methods are also discussed. Limitations in prior research in the area of software architecture analysis w.r.t. the requirements are used to motivate the thesis research agenda and methodology. In particular the techniques of the existing software analysis methods to analyze the software architecture helped us a lot to design a new architecture analysis method w.r.t requirements. Due to this reason proposed method can easily be accepted in the current family of the architecture analysis methods.

Chapter 3 mainly describes the Problem Definition. Research Questions are identified which makes the focus of the research clearer to find answers. The research questions lead us to perform research activities towards how to analyze the architecture with respect to requirements, to find out the benefits from this method and to explore the advantages that stakeholders can get with the help of the newly developed method. This research uses a hypothesis that the degree to which a software system fulfils the requirements leads it towards success. Moreover there are inherent difficulties associated with the requirements elicitation phase. Software architecture is an emerging field in this era of software engineering and the software development companies are moving towards the design of the software architecture that provides a basis for a successful software system. If the architecture is analyzed at an earlier stage to check whether it meets all the user requirements then it can save the cost of development if some requirements are missing and identified at a later stage. Therefore it was decided that a new software architecture analysis method may be devised in order to analyze the software architecture with respect to requirements.

Chapter 4 discusses the newly devised architecture analysis method AREA-TEAM. The discussion encompasses the development of the method beginning from the conceptual model towards its final shape and structure.

Chapter 5 is the industrial case study that is used to validate the method and find out the associated benefits and drawbacks.

Chapter 6 concludes the thesis by summarizing the main findings, benefits and limitations of the method.

1.5.1 Publications Related to this Thesis

We have published two research papers relating to this thesis work. First paper presents the method itself [6]. The architecture analysis method AREA-TEAM is described in detail and a dummy case study is also part of the paper.

Second paper describes the use of the developed architecture analysis method AREA-TEAM on an industrial project [7]. The paper presents results found after analyzing the reference architecture by applying our analysis method.

Chapter 2

LITERATURE SURVEY

2 Literature Survey

Software architecture is a novel area with a lot of research potential. New questions arise with a rapid pace and researchers are investigating with new ideas. Software Engineering Institute (SEI) has developed many software architecture evaluation methods (discussed in the following sections). While studying these methods, the idea of evaluating the software architecture with respect to the requirements came in our mind.

The study introduces a way to analyze the software architecture with respect to the requirements gathered during the requirements engineering phase and as a result delivers the accuracy and error factors related to the architect and the stakeholders. The name was proposed, "The Architecture Requirements Engineering Error & Accuracy-The Analysis Method (AREA-TEAM). The method focuses on the requirements and error factor calculation during the architecture analysis process. The requirement engineering provides the appropriate mechanism for what customer wants, analyzing need, assessing feasibility, negotiating a reasonable solution, specifying requirement and their validation [8][9]. There is always some percentage of error involved on part of both the development team and the customers. A plus point in the AREA-TEAM is that it calculates both Error by the Architect (E_a) and Error by the Stakeholder (E_s) separately, which helps identifying the competency of the architect and the maturity of the customer.

The idea of analyzing different artifacts involved in various phases of software engineering with respect to the requirements is not new. Due to the popularity of modern notations to represent the software architecture, older methods of architecture analysis are outdated. Moreover a number of drawbacks were found in these methods. Other methods of architecture analysis with respect to requirements are unable to differentiate between the architect and the customer errors and have no proper measurement mechanism. The initial methods were unable to provide proper mapping from requirements to architecture because many architectural patterns were very complex and provided no provisions of mapping, for example, call-and-return architecture [10]. For that purpose structured design evolved, which stressed modularity [11], top-down design [12] and structured

program [13]. Myers and Stevens proposed dataflow oriented design translation from the architectural patterns but they were not able to measure neither the performance of dataflow diagrams nor the architectural competency and error responsible authority [31].

2.1 The Quantified Design Space (QDS)

The quantified design space method [14], a spread sheet model based on realization mechanisms and contribution analysis in the software architecture and the architectural decisions are prioritized. The Quantified Design Space (QDS) is supported by a software analysis tool. It combines the techniques of Quality Function Deployment (QFD) and Design Space to create a powerful software engineering tool for requirements elicitation, software design, and product evaluation [14]. QFD is a structured methodology and mathematical tool used to identify and quantify customers' requirements and translate them into key critical parameters [15]. QFD helps to prioritize actions to improve the process or product to meet customers' expectations. Design spaces provide a uniform, semi-formal way for describing and classifying both requirements and properties of software artifacts [16]. They are equally suited to capture functional as well as non-functional properties such as architectural issues or time and memory complexity. During the development of an application, a vast number of design decisions have to be taken. The *design space* for a specific application domain is intended to describe the entirety of these design choices along with the possible alternatives. In order to structure design space descriptions, it is possible to define *groups* of design space dimensions referring to some distinct part of the overall system functionality, e.g. fault tolerance or scalability aspects. This allows emphasizing that certain classification criteria are closely related, and thus facilitates understanding of the design space by the application developers. The concept of design spaces is useful for a variety of purposes. First, the dimensions in the requirements design space can be considered as a catalogue enumerating important criteria for the requirements specification, providing hints concerning the relevant criteria that have to be taken into account. By reusing this catalogue, the activity of requirements specification is considerably facilitated. Another effect is that the requirements

specifications of different systems belonging to the same application domain become comparable.

The Quantified Design Space (QDS) is therefore a useful method in development of high quality software that meets the requirements and satisfies the stakeholders. Its application on the software architecture has also been under discussion [14]. This method aids towards the development of software with good design and this is not an architecture evaluation method. The drawback is that the requirements' weights are neglected and no error finding mechanism is present for architecture measurement.

2.2 Software Architecture Analysis Method (SAAM)

While software architecture has become an increasingly important research topic in recent years, insufficient attention has been paid to methods for evaluation of these architectures. SAAM is a method proposed by SEI for analyzing the properties of software architectures [17]. It evaluates the architecture with respect to software quality. Evaluating architectures is difficult for two main reasons. First, there is no common language used to describe different architectures. Second, there is no clear way of understanding an architecture with respect to an organization's life cycle concerns--software quality concerns such as maintainability, portability, modularity, reusability, and so forth. SAAM is a five-step method for analyzing software architectures that discusses the above mentioned issues.

This method is based upon a common understanding and representation for architectures and an analysis of an organization's lifecycle requirements. SAAM permits the comparison of architectures within the context of an organization's particular quality requirements. This sort of comparison has been up till now quite difficult.

The SAAM places strong demands on an organization to articulate those quality attributes of primary importance. It also requires a selection of benchmark tasks with which to test those attributes. The purpose of the SAAM is not to criticize or commend a

particular architecture, but to provide a method for determining whether the architecture supports an organization's needs or not. This method aids in selection of the appropriate architecture.

2.3 Architecture Trade-Off Analysis Method (ATAM)

The ATAM is another method which focuses on quality attribute requirements [18]. Evaluating an architecture using the ATAM, goal is to understand the consequences of architectural decisions with respect to the quality attribute requirements of the system. A system is motivated by a set of functional and quality goals. For example, if a telephone switch manufacturer is creating a new switch, that system must be able to route calls, generate tones, generate billing information and so forth. But if it is to be successful, it must do so within strict performance, availability, modifiability, and cost parameters. The architecture is the key to achieve—or failing to achieve—these goals. The ATAM is a means of determining whether these goals are achievable by the architecture as it has been conceived, before enormous organizational resources have been committed to it. It evaluates the quality attributes by considering each quality attribute in isolation and identifies sensitivity of quality attributes by various architectural attributes, which may result in evaluating the architecture w.r.t. the quality scenarios [19]. Since the basic goal of ATAM is to find out the trade-offs made w.r.t quality attributes while making architectural decisions, therefore, it does not provide a proper way for mapping of requirements with the architecture. Similarly there are no proper metrics for architecture correctness percentage and error propagation authority. But for our method AREA-TEAM, ATAM is one of the major inspiration in terms of its technique to analyze the software architecture.

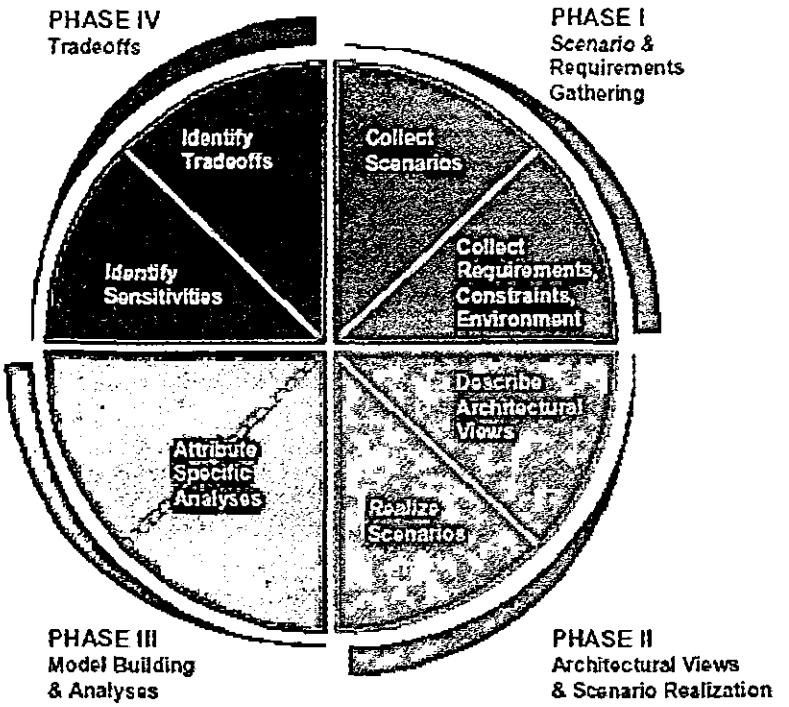


Figure 2.1: Steps of the Architecture Tradeoff Analysis Method (Adopted from [20])

2.4 Cost Benefit Analysis Method (CBAM)

The Cost Benefit Analysis Method (CBAM) is a method for economic modeling of software and systems, centered on an analysis of their architectures [21]. The CBAM builds on the ATAM to model the costs and benefits of architectural design decisions and to provide a means of optimizing such decisions [22]. A simple way to think about the objectives of this method is that we are adding money to the ATAM as an additional attribute to be traded off. We are showing how to make decisions in terms of benefits per dollar, as well as in terms of quality-attribute responses. The CBAM begins where an ATAM leaves off and depends on the artifacts that the ATAM produces as output, see Fig. 2.1.

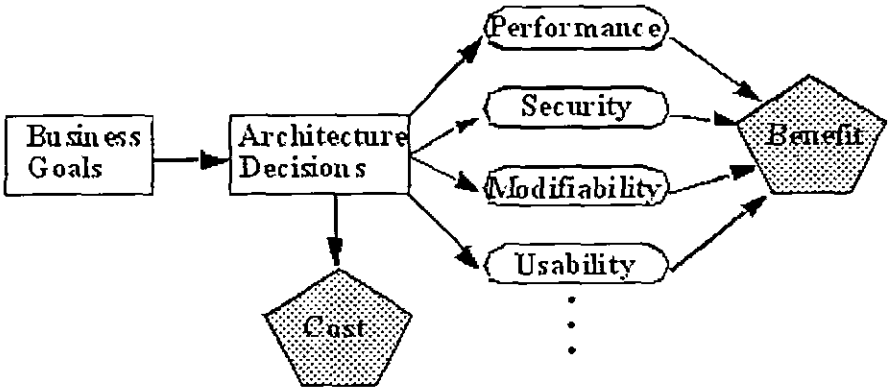


Figure 2.2: The Context for the CBAM (Adopted from [23])

The ATAM uncovers the architectural decisions that are made (or are being considered) for the system and links these decisions to business goals and QA response measures via a set of elicited scenarios. The CBAM builds on this foundation, as shown by the shaded pentagons in Figure 2.2, by enabling engineers to determine the costs and benefits associated with these decisions. Given this information, the stakeholders could then decide, for example, whether to use redundant hardware, checkpointing, or some other method to address concerns about the system's reliability. Or, the stakeholders could choose to invest their finite resources in some other QA—perhaps believing that higher performance will have a better benefit/cost ratio. CBAM focuses on cost and benefit in terms of money and resources. This question is addressed after the question of whether the architecture meets the requirements or not, which is answered by our method AREA-TEAM.

2.5 Summary of Research

There are a number of methods available for the analysis of software architecture. These methods are used to evaluate the architecture with respect to the software quality, trade-offs between the quality attributes and cost & benefit related to the architectural decisions made during the architecture design. These techniques can be used in combination to obtain early and continuous benefits. The popularity is gained by the methods proposed by the Software Engineering Institute (SEI) at Carnegie Mellon University, USA. Many of the organizations are taking benefits from this family of methods.

In remaining part of this report we will elaborate the steps of the AREA-TEAM, explain its foundations, and concludes with an extended example of applying the AREA-TEAM to a real system.

Chapter 3

PROBLEM DEFINITION

3 Problem Definition

In this chapter we shall specify the problem statement, specific objectives that will be addressed by the research which we are going to conduct and the approach that we shall use in problem solution. We shall try to answer the research questions given below.

- How the software architecture can be analyzed with respect to the requirements?
- How the architecture analysis with respect to the requirements can provide an early benefit?
- How the stakeholders can be satisfied about the architecture with the help of the analysis with respect to the requirements?

3.1 Hypothesis

The primary measure of success of a software system is the degree to which it meets the purpose for which it was intended. Broadly speaking, software systems requirements engineering (RE) is the process of discovering that purpose, by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation. There are a number of inherent difficulties in this process [24]. Stakeholders (including paying customers, users and developers) may be numerous and distributed. Their goals may vary and conflict, depending on their perspectives of the environment in which they work and the tasks they wish to accomplish. Their goals may not be explicit or may be difficult to articulate, and, inevitably, satisfaction of these goals may be constrained by a variety of factors outside their control.

The software architecture has emerged as a new field in the broader spectrum of software engineering. The software architecture may contain a factor of error due to the conflicting, misunderstood or unidentified requirements. Although a number of methods and techniques exist to analyze the low level design with respect to requirements. But no method or study exists to analyze the software architecture (the higher level design) with

respect to requirements. A number of benefits can be achieved by analyzing the architecture at an early stage with respect to requirements.

We shall devise a method for the software architecture evaluation with respect to the requirements. The error factor that exists in the architecture will be quantified. One of the major benefits of the method will be to save the cost if the requirements are changed during the software development or some new requirements arrive at a later stage. The study will also include discovering of some other benefits that we can get, by applying the method on an industrial case study.

3.2 Significance of the Research

No such method exists up to the time on which the research is in progress according to the best of our knowledge. Moreover the existing analysis methods may be applied only to the low level design to analyze the software with respect to the requirements.

Since no studies related to the software architecture have been conducted in the domain of requirements engineering. It is important to conduct research in order to devise a method to analyze the software architecture with respect to the requirements.

3.3 Problem Statement

In the light of discussion in the preceding sections of this thesis, the problem statement is written as follows:

> *"A method will be proposed to analyze the software architecture with respect to requirements, i.e., to analyze whether the architecture meets the requirements or not. The error factor present in the architecture due to the problems faced during requirement elicitation phase of requirements engineering will be quantified. The benefits of analyzing the architecture*

with respect to the requirements will also be discovered. The feedback of the stakeholders will also be collected to check how the method benefits them."

This problem statement provides us a base line for this research. Our devised method will provide a comprehensive solution for the remedy of the mentioned problem.

Chapter 4

PROPOSED SOLUTION

T- 132

4 Proposed Solution

The purpose of this research thesis is to describe the approach used behind the architecture requirements engineering accuracy and error-the analysis method (AREA-TEAM) and discuss its steps in practice, its implementation on a case study and its benefits. Software architecture of a program or computing system is a structure or structures of the system which comprise software elements, the externally visible properties of these elements and the relationships among them [25]. AREA-TEAM is important because it elaborates how well an architecture satisfies important requirements and finds the architecture accuracy factor and architecture error factor both in terms of errors of the architect and error by the stakeholder. Now the question arises, why do we need AREA-TEAM? We all know that architecture is a key ingredient in a successful software engineering process. The software architecture is developed based on the quality attributes identified from the key requirements [26]. It is a key to achieve or failure to meet the requirements [27].

AREA-TEAM is a way to analyze whether the requirements are achieved in the architecture. Such an analysis is significant because valuable organizational resources are allocated to the project and if at a later stage it is discovered that the architecture was not fulfilling the requirements, the resources may go astray. It also contributes to find the factor of error made by the architect and the customer. This method uses the standard architecture documentation techniques as prescribed by SEI [28]. It is a low-cost high-benefit method for analyzing software architectures in the light of the key requirements and finding the error factor. It identifies places where the architecture is unsuitable in terms of meeting the requirements at an earlier stage, thus resulting in lower eventual project cost. Added benefits include open communication channels between the architect and the stakeholders.

AREA-TEAM relies on assembling the stakeholders to articulate what the important requirements are, and then exercising the architecture to make sure that those key requirements are satisfied by the software architecture. The result is a high-fidelity architecture analysis coupled with high-quality familiarization with the architecture in

light of the requirements it meets on the part of the stakeholders. It calculates the error factor, i.e., software architect's error percentage as well as customers propagated error percentage separately. Therefore it helps to analyze the performance of architect and judge the customers' maturity. All the stakeholders are saved from loss by adapting a pro-active risk strategy.

The AREA-TEAM draws inspiration and techniques from various areas: the notion of architectural styles; the Architecture Trade-off Analysis Method [18]; and the Software Architecture Analysis Method (SAAM) [17], which was the predecessor to the ATAM and the process of Requirements Engineering [29] and Quantified Design Space [14]. It is intended for analysis of architecture in terms, whether it meets the requirements or not. It also quantifies the error factor. Although this is the AREA-TEAM's focus, but there is a problem in operationalizing this focus. We (and the software engineering community in general) do not understand how to specify the requirements well: the "requirements elicitation", "requirements analysis and negotiation", "requirements specification" changes from system to system, from stakeholder to stakeholder, and from community to community. This propagates errors from one stage to another and finally reflects in the architecture. If the actual requirements are denoted by A and architect error factor and stakeholder error factor are denoted by E_a and E_s respectively, then the equation 4.1 shows the deviation from the desired requirements and deviated requirements are denoted by A'.

$$A' = A + E_a + E_s \tag{4.1}$$

4.1 The Conceptual Model

Proposed method consists of a series of steps for architecture analysis and error factor quantization. First of all it method assembles the identified stakeholders in a meeting. The AREA-TEAM is presented and the core requirements are discussed. The requirements are prioritized by taking votes of the stakeholders and prioritization factor is calculated. Weights are assigned to each requirement in order to calculate the

significance factor of each requirement. This is done with the help of stakeholders by feedback forms. Average requirement significance factor is calculated for each requirement. Then the requirement criticality is computed by finding the mean of Prioritization Factors and Significance Factors for each requirement. The architecture is analyzed by the analysis team and both the architecture accuracy factor and the architecture error factors are calculated. Software development is an agile process. Every time novel requirements are identified. These requirements are given by the stakeholders, generally customers. The error factor is also calculated for these new requirements. This is the stakeholder error factor. The process is further elaborated with the conceptual model of AREA-TEAM in Fig. 4.1.

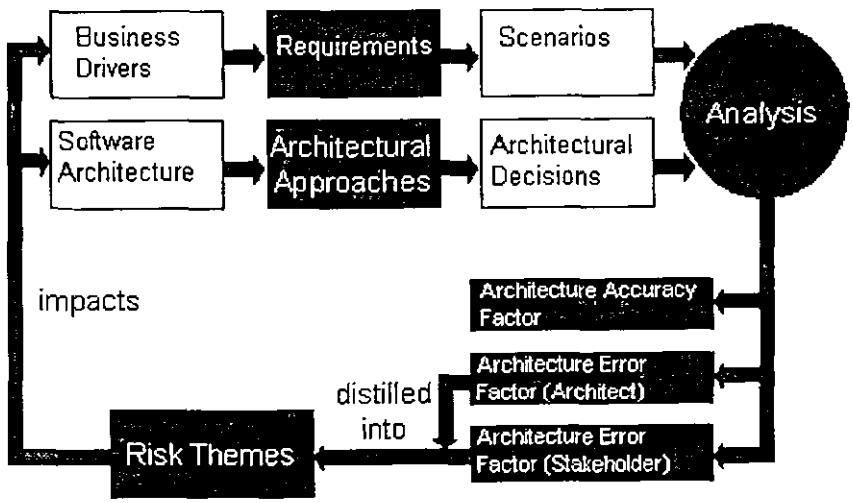


Fig. 4.1 Conceptual Model of AREA-TEAM

4.2 Development of the Method

After problem definition we started research work in order to provide solution of the problem in order to meet the identified objectives (section 1.2.1).

While developing an architecture analysis method we had to consider that the devised method must be acceptable in the Software Engineering community in terms of its techniques and application. Major motivation of the development of the method came from the architecture analysis methods proposed by Software Engineering Institute (SEI)

at Carnegie Mellon University. The methods like SAAM, ATAM and CBAM provide different architecture analysis techniques to analyze the architecture in different aspects. These techniques can be used in combination to obtain early and continuous benefits. Once the software architecture is more fully developed, the ATAM can be used to reveal how well the architecture satisfies particular quality attribute requirements and the risks, sensitivities, and tradeoffs involved in satisfying the requirements. The CBAM guides system engineers and other stakeholders to determine the economic tradeoffs associated with the architectural decisions that result in the system's qualities.

Evaluating an architecture for a large system is a complicated task. First, a large system will have a comparably large architecture that will be difficult to understand in a limited amount of time. Second, according to Nietzsche and the Architecture Business Cycle (ABC), a computer system is intended to support business goals and the evaluation will need to make connections between those goals and the technical decisions [30]. Finally, a large system usually has multiple stakeholders and acquiring their different perspectives in a limited amount of time requires careful management of an evaluation process. We can see from this set of difficulties, managing limited time for an architecture evaluation is a major problem.

While designing the AREA-TEAM, main technique is adapted from the existing architecture analysis methods from SEI mentioned above. AREA-TEAM is designed to evaluate whether the software engineers have properly gathered and understood the requirements during the requirements elicitation phase to meet the business goals for the system as well as for the architecture. It is designed to use those requirements and stakeholder participation to focus the attention of the evaluators on the portion of the architecture that is central to achieve those requirements.

4.2.1 Participants in the AREA-TEAM

The AREA-TEAM requires the participation and mutual cooperation of three groups:

i. **The Evaluation Team:** This group is external to the project whose architecture is being evaluated. It usually consists of three to five people. Each member of the team is assigned a number of specific roles to play during the evaluation. (See Table 4.1 for a description of these roles, along with a set of desirable characteristics for each.) The evaluation team may be a standing unit in which architecture evaluations are regularly performed, or its members may be chosen from a pool of architecturally savvy individuals for the occasion. They may work for the same organization as a development team whose architecture is on the table, or they may be outside consultants. In any case, they need to be recognized as competent, unbiased outsiders with no hidden agendas or axes to grind.

ii. **Project Decision Makers:** These people are empowered to speak for the development project or have the authority to mandate changes to it. They usually include the project manager, and, if there is an identifiable customer who is footing the bill for the development, he or she will be present (or represented) as well. The architect is always included—a cardinal rule of architecture evaluation is that he/she must willingly participate. Finally, the person commissioning the evaluation is usually empowered to speak for the development project; he or she should be included in the group.

iii. **Architecture Stakeholders:** Stakeholders have a vested interest in the architecture performing as advertised. They are the ones whose ability to do their jobs hinges on the architecture promoting the business goals according to the identified requirements. Stakeholders include developers, testers, integrators, maintainers, performance engineers, users, builders of systems interacting with the one under consideration, and others. Their job during an evaluation is to articulate the specific requirements to meet the business goals that the architecture should meet in order for the system to be considered a success. A rule of thumb is that you should expect to enlist the services of twelve to fifteen stakeholders for the evaluation.

Role	Responsibilities	Desirable Characteristics
Team Leader	Sets up the evaluation; coordinates with client, making sure client's needs are met; establishes evaluation contract; forms evaluation team; checks that final report is produced and delivered (although the writing may be delegated)	Well-organized, with managerial skills; good at interacting with client; able to meet deadlines
Evaluation Leader	Runs evaluation; facilitates elicitation of requirements; administers requirements selection/prioritization process; facilitates evaluation of requirements against architecture; facilitates onsite analysis	Comfortable in front of audience; excellent facilitation skills; good understanding of architectural issues; practiced in architecture evaluations; able to tell when protracted discussion is leading to a valuable discovery or when it is pointless and should be re-directed
Requirement Scribe	Writes requirements on flipchart or whiteboard during requirement elicitation; captures agreed-on wording of each requirement, halting discussion until exact wording is captured	Good handwriting; perfectionist about not moving on before an idea (requirement) is captured; can absorb and distill the essence of technical discussions
Proceedings Scribe	Captures proceedings in electronic form on laptop or workstation, raw requirements, issue(s) related to the requirements, and resolution of each requirement when applied to architecture; also generates a printed list of adopted scenarios for handout to all participants	Good, fast typist; well organized for rapid recall of information; good understanding of architectural issues; able to assimilate technical issues quickly; unafraid to interrupt the flow of discussion (at opportune times) to test understanding of an issue so that appropriate information is captured
Process Observer	Keeps notes on how evaluation process could be improved or deviated from; usually keeps silent but may make discreet process-based suggestions to the evaluation leader during the evaluation; after evaluation, reports on how the process went and lessons learned for future improvement; also responsible for reporting experience to architecture evaluation team at large	Thoughtful observer; knowledgeable in the evaluation process; should have previous experience in the architecture evaluation method
Time Keeper	Helps evaluation leader stay on schedule; helps control amount of time devoted to each scenario during the evaluation phase	Willing to interrupt discussion to call time

Table 4.1 AREA-TEAM evaluation team members & their roles

4.2.2 Outputs of the AREA-TEAM

An AREA-TEAM-based evaluation will produce at least the following outputs:

- **A concise presentation of the architecture:** Architecture documentation is often thought to consist of the object model, a list of interfaces and their signatures, or some other voluminous list. But one of the requirements of the AREA-TEAM is

that the architecture be presented in one hour, which leads to an architectural presentation that is both concise and, usually, understandable.

- **Articulation of business goals:** Frequently, the business goals presented in the AREA-TEAM are being seen by some of the development team for the first time.
- **Prioritization of the requirements:** The requirements are ordered in the form of a list according to the priority assigned to each requirement as desired by the stakeholders.
- **Capturing new/changed requirements:** The AREA-TEAM based analysis supports agility. New requirements that were missed due to some reason during requirements elicitation phase are often captured during the analysis process. Change in existing requirements may also be recorded.
- **Mapping of architectural decisions to requirements:** Architectural decisions can be interpreted in terms of the requirements that they support or hinder. For each requirement examined during AREA-TEAM analysis, those architectural decisions that help to achieve it are determined.
- **Quantification of error factor:** The error factor present in the architecture due to problems faced during requirements elicitation phase is calculated. An error factor for the architect (as a measure of performance of the architect) and an error factor for the stakeholder (as measure of maturity of the stakeholder) are included in the output of AREA-TEAM based analysis. Besides that the architecture accuracy factor is also determined.
- **A set of risks and non-risks:** A risk is defined in the AREA-TEAM as an architectural decision that may lead to undesirable consequences in light of stated requirements. Similarly, a non-risk is an architectural decision that, upon analysis, is deemed safe. The identified risks can form the basis for an architectural risk mitigation plan.
- **A set of risk themes:** When the analysis is complete, the evaluation team will examine the full set of discovered risks to look for over-arching themes that identify systemic weaknesses in the architecture or even in the architecture process and team. If left untreated, these risk themes will threaten the project's business goals.

The outputs are used to build a final written report that recaps the method, summarizes the proceedings, captures and prioritizes the requirements and their analysis, and catalogs the findings.

There are secondary outputs as well. Very often, representations of the architecture will have been created expressly for the evaluation and may be superior to whatever existed before. This additional documentation survives the evaluation and can become part of the project's legacy. Also, the requirements discussed and captured by the participants are expressions of the business goals and requirements for the architecture and can be used to guide the architecture's evolution. Finally, the analysis contained in the final report can serve as a statement of rationale for certain architectural decisions made (or not made). The secondary outputs are tangible and enumerable.

Some of the results of an AREA-TEAM based evaluation are intangible. These include a palpable sense of community on the part of the stakeholders, open communication channels between the architect and the stakeholders, and a better overall understanding on the part of all participants of the architecture and its strengths and weaknesses. While these results are hard to measure, they are no less important than the others and often are long-lasting.

4.2.3 Phases of the AREA-TEAM

AREA-TEAM is divided in to two main phases (see table 4.2 to view the phases and their characteristics).

Phase 1 is named as "Rehearsal". During the rehearsal phase the activities are performed in order to prepare the team for the architecture analysis. The evaluation team leadership and the key project decision makers informally meet to work out the details of the exercise. The project representatives brief the evaluators about the project so that the team can be supplemented by people who possess the appropriate expertise. Together, the two groups agree on logistics, such as the time and place of meetings, who will bring the

flipcharts, and who will fulfil the other requirements like refreshment etc. They also agree on a preliminary list of stakeholders (by name, not just role), and they negotiate on when the final report is to be delivered and to whom. They handle formalities such as statement of work or nondisclosure agreements. They work out delivery to the evaluation team of whatever architectural documentation exists and may be useful. Finally, the evaluation team leader explains what information the manager and architect will be expected to show during phase 2, and helps them construct their presentations if necessary. This meeting usually lasts for a day.

Phase 2 "Architecture Requirements Engineering Analysis", is the evaluation phase, where everyone gets down to the business of analysis. Analysis team and the stakeholders are assembled and the main activities of the architecture requirement engineering analysis commence. Nominally this phase takes a day or two.

Phase	Activity	Participants	Typical Duration
1	Rehearsal	Evaluation team leadership and key project decision makers	Typically a day (May vary depending on the project)
2	Architecture Requirements Engineering Analysis	Evaluation team, project decision makers, and stakeholders	Normally a day or two

Table 4.2 AREA-TEAM Phases & their characteristics

4.2.4 Steps of the AREA-TEAM phases

The AREA-TEAM analysis phases (phase 1 and phase 2) consist of thirteen steps. Steps 1 through 5 are carried out in phase 1. In phase 2, with all stakeholders present, those steps are summarized and steps 6 through 13 are carried out. The analysis steps are nominally carried out in sequential order according to the set agenda, but sometimes there must be dynamic modifications to the schedule to accommodate personnel availability or architectural information. Every evaluation is unique, and there may be times when the team returns briefly to an earlier step, jumps forward to a later step, or iterates among steps, as the need dictates.

Step 1

Identify stakeholders: In this case, stakeholders are the customers who will own this software. Software engineers who are expected to use the software architecture, users who will be expected to use the software and the quality assurance inspector. Approximately a dozen stakeholders aimed, but this can vary depending on the size of the user community. Names of the stakeholders who will participate in the analysis process must be finalized in this step.

Step 2

Prepare core requirements: The Software Project Manager, Evaluation Team Leader and the Software Architect identify a set of core requirements, which played the key role in the architectural development. These are identified to develop a consensus between the stakeholders regarding core requirements.

Step 3

Prepare architecture briefing: The Software Architect prepares a briefing explaining the software architecture. A rule of thumb is to aim for two hours' worth of material. Include examples of how the software architecture meets the user requirements. The goal is to present the software architecture in sufficient detail so that a knowledgeable audience member could understand the software architecture.

Step 4

Identify architecture approaches: The AREA-TEAM focuses on analyzing the architecture by understanding its architectural approaches. In this step they are identified by the architect, and captured by the analysis team, but are not analyzed.

Concentrate on identifying architectural approaches and architectural styles because these represent the architecture's ways of addressing the highest priority requirements, the means of ensuring that the critical requirements are met in a predictable way [26]. We look for approaches and styles because not all architects are familiar with the language of architectural styles, and so may not be able to enumerate a set of styles used in the

architecture. But every architect makes architectural decisions, and the set of these we call "approaches." These can certainly be elicited from any conscientious architect. These architectural approaches define the important structures of the system and describe the ways in which this system can grow, respond to changes, withstand attacks, integrate with other systems, and so forth.

Step 5

Prepare materials: Copies of the presentation, core requirements, and feedback forms are produced for distribution to the stakeholders during the Phase 2 meeting. The meeting is scheduled, stakeholders are invited, and steps are taken to assure the presence of a quorum of stakeholders at the meeting.

Start of Phase 2

At this point phase 1 is completed. Next, when the Evaluation team, project decision makers are ready to resume and stakeholders are assembled, phase 2 starts. With this the main activities of the architecture requirement engineering analysis commence.

Step 6

Present AREA-TEAM: Software Project Manager delivers a brief presentation explaining the steps of AREA-TEAM to the participants.

Step 7

Present software architecture and core requirements: The software architect presents architecture overview presentation and backtrack the architecture how it meets the core requirements, which were considered for architecting the system. During this time, a scribe captures each question and any novel requirements mentioned by the stakeholders. The new requirements may or may not be reflected in the present architecture. AREA-TEAM is an agile process that is light and adaptive.

Step 8

Requirements Prioritization Factor: The AREA-TEAM analysis team uses the show-of-hands method in order to get the votes from the stakeholders. The purpose of voting is to identify and prioritize the core requirements as per interest of the stakeholders. These votes are recorded in the requirements prioritization metric. This is to negate the error factor involved in the identification of the prioritized requirements both by the architect and the stakeholders. Then the prioritization factor (PF) is calculated by dividing the number of votes (V) cast by the total number of stakeholders (Ts).

$$PF_i = V_i / Ts \text{ , where } i \text{ is from 1 to } Ts \tag{4.2}$$

Step 9

Requirement Significance Factor: The significance of each requirement is gathered from stakeholders with the help of feedback forms. The significance is constrained between 0 and 100. 0 shows no significance whereas score of 100 shows maximum significance. The significance for each requirement given by each stakeholder is recorded in the requirements significance metric. The average significance factor (ASF) for each requirement is calculated. It is reflected by the percentage of the average of the significance (S) chosen by each stakeholder for each requirement [Equation 4.3].

$$ASF_i = ((\sum_{i=1}^{n} \sum_{j=1}^{Ts} S_{ij}) / Ts) / 100 \tag{4.3}$$

Step 10

Requirement Criticality Analysis: The requirement criticality (RC) is calculated by finding the mean of Prioritization Factors (PF) and Average Significance Factors (ASF) for each requirement [Equation 4.4]. The readings are recorded in the requirements criticality metric.

$$RC_i = (PF_i + ASF_i) / 2 \text{ , where } i \text{ is from 1 to } n \tag{4.4}$$

Step 11

Novel Requirements Prioritization Factor, Average Significance Factor and Criticality Analysis: The novel requirements (if any recorded in step 7) are also processed in the same manner as it was done with the key requirements from step 8 to step 10 [Equations 4.5, 4.6, 4.7].

$$NPF_i = NV_i / Ts \text{ , where } i \text{ is from } 1 \text{ to } n. \tag{4.5}$$

$$NASF_i = ((\sum_{i=1}^{n} \sum_{j=1}^{T_i} NS_{ij}) / Ts) / 100 \tag{4.6}$$

$$NRC_i = (NPF_i + NASF_i) / 2 \text{ , where } i \text{ from } 1 \text{ to } n. \tag{4.7}$$

Step 12

Analyze Architectural Approaches: The AREA-TEAM analysis team examines the highest ranked requirements one at a time. The novel requirements are also analyzed separately. The software architect is asked to explain how the architecture supports each one and there may be a case where the architecture already fulfils some novel requirement. Team members specially the questioners probe for the architectural approaches that the architect used to carry out the requirement. The goal is for the evaluation team to find the architecture approach contribution factor (ACF), that is, be convinced that either the approach is appropriate in meeting the requirement for which it is intended or it is inappropriate. If the architecture approach is appropriate then the score 1 is assigned and if it is inappropriate the score 0 is assigned to the approach.

Step 13

Architecture Accuracy Factor and Architecture Error Factor: The score assigned to each architectural approach formulated during step 12 are put in the architecture accuracy metric. Then the architecture approach contribution (AAC) percentage is calculated [Equation 4.9] for each requirement, by dividing the requirement criticality (RC) for each requirement by the total requirement criticality (TRC) [Equation 4.8]. The data is stored in Architecture Accuracy Metric.

$$TRC = \sum_{i=1}^{n} RC_i \tag{4.8}$$

$$AAC_i = (RC_i / TRC) \times 100, \text{ where } i \text{ from } 1 \text{ to } n. \tag{4.9}$$

Then the architecture accuracy factor (AAF) is calculated.

$$AAF = \sum_{i=1}^{n} AAC_i, \text{ where } ACF_i = 1 \tag{4.10}$$

The architecture accuracy factor (*AAF*) reflects the ratio by which the architecture has successfully addressed the requirements. Then the architecture error factor (*AEF*) is calculated.

$$AEF = \sum_{i=1}^{n} AAC_i, \text{ where } ACF_i = 0 \tag{4.11}$$

The architecture error factor (AEF) reflects the ratio by which the architecture has not addressed the requirements.

For calculating the error factor by the stakeholder, the new requirements recorded in step 7 are also included in the calculations process.

$$NTRC = \sum_{i=1}^{n} NRC_i \tag{4.12}$$

$$NAAC_i = (NRC_i / (TRC + NTRC)) \times 100, \text{ where } i \text{ is from } 1 \text{ to } n. \tag{4.13}$$

Then the new architecture accuracy factor is calculated. The data is stored in Architecture Accuracy Metric with New Requirements.

$$NAAF = \sum_{i=1}^{n} NAAC_i, \text{ where } NACF_i = 1 \tag{4.14}$$

The new architecture accuracy factor reflects the ratio by which the architecture has successfully addressed the requirements including the new requirements. Then the new architecture error factor is calculated.

$$NAEF = \sum_{i=1}^{n} NAAC_i \text{ , where } NACF_i = 0 \quad (4.15)$$

The new architecture error factor reflects the ratio by which the architecture has not addressed the requirements including the new requirements. The stakeholder error factor is calculated with the following equation.

$$SEF = NAEF - AEF \quad (4.16)$$

The adverse effect introduced into architecture accuracy due to the stakeholder is calculated by the following equation.

$$SAF = NAAF - AAF \quad (4.17)$$

This is the last step of the method. After this the analysis is complete and final analysis report may be prepared and presented.

Chapter 5

ISLAMABAD STOCK EXCHANGE (ISE) CASE STUDY

5 Islamabad Stock Exchange (ISE) Case Study

During this research, the Software Engineering Group at IIUI has developed the Architecture Requirements Engineering Error and Accuracy-The Analysis Method (AREA-TEAM). This method permits evaluation of the software architecture in terms of its compliance with the requirement specifications. The devised method "Architecture Requirements Engineering Accuracy and Error-The Analysis Method (AREA-TEAM)", provides solid technical foundations for performing architectural analysis. It is based on the requirements and finding the architecture accuracy factor and architecture error factor. The major contribution of AREA-TEAM is quantification of the error made by the architect as well as the stakeholder, generally the customer. In order to test the method and validate the results, we have selected Date Warehouse project of Islamabad Stock Exchange (discussed in appendix C). The following chapter discusses the evaluation of the reference architecture using AREA-TEAM.

5.1 Start of the Evaluation

At the time of the AREA-TEAM evaluation, the development team for the reference architecture had progressed through the following three phases:

- Understand the problem and scope of the effort
- Document the reference architecture
- Performed the evaluation on the reference architecture using AREA-TEAM

5.2 Phase 1 of the Evaluation

The AREA-TEAM method is applied in two phases. The first phase is named as "Rehearsal". It involves the initial connection of the evaluation team leads with the architects and the beginning of the exploration and analysis. Phase 1 allows the evaluation team to get acquainted with the architect(s), the system purpose, and the architecture and to begin a preliminary analysis of the architecture. Phase 1 also permits

the architects to become familiar with the AREA-TEAM and understand the information they must supply for the complete evaluation to proceed.

An AREA-TEAM evaluation team was selected for this evaluation. The team leads met with reference architecture architects and provided templates for architecture presentation to be given as part of the evaluation exercise. A draft utility tree and seed scenarios were produced. A template was also given to the architecture manager to be used in the presentation of the business drivers.

Phase 1 of the evaluation took place at the Islamabad Stock Exchange Building, Blue Area, Islamabad on April 18, 2005. Five "decision maker" stakeholders were present, which included two members of the International Islamic University (contractor organization) (architect and project manager), two members of the ISE IT Department and the Secretary stock exchange.

5.3 Phase 2 of the Evaluation

All of the AREA-TEAM steps are executed during Phase 2, which typically lasts two days. The phase 2 is called "Architecture Requirement Engineering Analysis. Any preliminary analysis resulting from Phase 1 is presented.

Phase 2 of the evaluation also took place at the Islamabad Stock Exchange Building, Blue Area, Islamabad on April 28, 2005. Ten stakeholders were present, which included Secretary Stock Exchange, 2 members of stock exchange, 2 clients, 1 user, 1 project manager and 1 software architect, 1 IT manager and 1 quality assurance inspector.

5.4 Business Drivers

The system is designed and developed to meet the following objectives.

- ✓ To provide a full fledged and very less error prone online transaction processing system for a stock exchange. It includes real time transaction processing among different agents working under specific members of stock exchange.

- ✓ To enhance the decision support for the stock exchange by developing a data warehouse for bulk storage and efficient retrieval of data. The operational database cannot be used for deep data drilling and decision making, so a data warehouse can be used to store huge volumes of data i.e. up to Terabytes. It also includes the replication, loads and stores of data among distributed databases.

- ✓ To improve the overall business scenario of the stock exchange by retaining the customers. This will help in increasing the market capital and overall business activities. It will lead to gather maximum customers (investors) and to motivate the people to invest in the stock market.

- ✓ To develop software, that provides business intelligent data of a stock exchange on handheld Palm device. Synchronizing the Palm device with the data source for quick and efficient data retrieval.

5.5 Architecture Available for Evaluation

Keeping in view the above mentioned business drivers; software architecture was designed (figure 5.1). Trading OLTP component is present to provide a reliable online transaction processing. Data warehouse components provide the features to provide decision support. Stock market trends can be analyzed. The clients will be facilitated by providing them with useful information related to the condition of the stock market to help them to make decisions to invest. Palm stock exchange explorer and palm suite website components are developed to equip the clients and users with the latest information with comprehensive reports. The architecture must provide a strong basis for the development of a successful software system. Especially for an organization like stock exchange, where the situations vary rapidly and sudden u-turns are also possible, a

smooth operation is not possible without reliable software. Stakeholders need information every moment. The architecture is designed to fulfill the above mentioned business goals. In order to verify our claim that this architecture meets the requirements, we shall analyze this by applying the method AREA-TEAM on it.

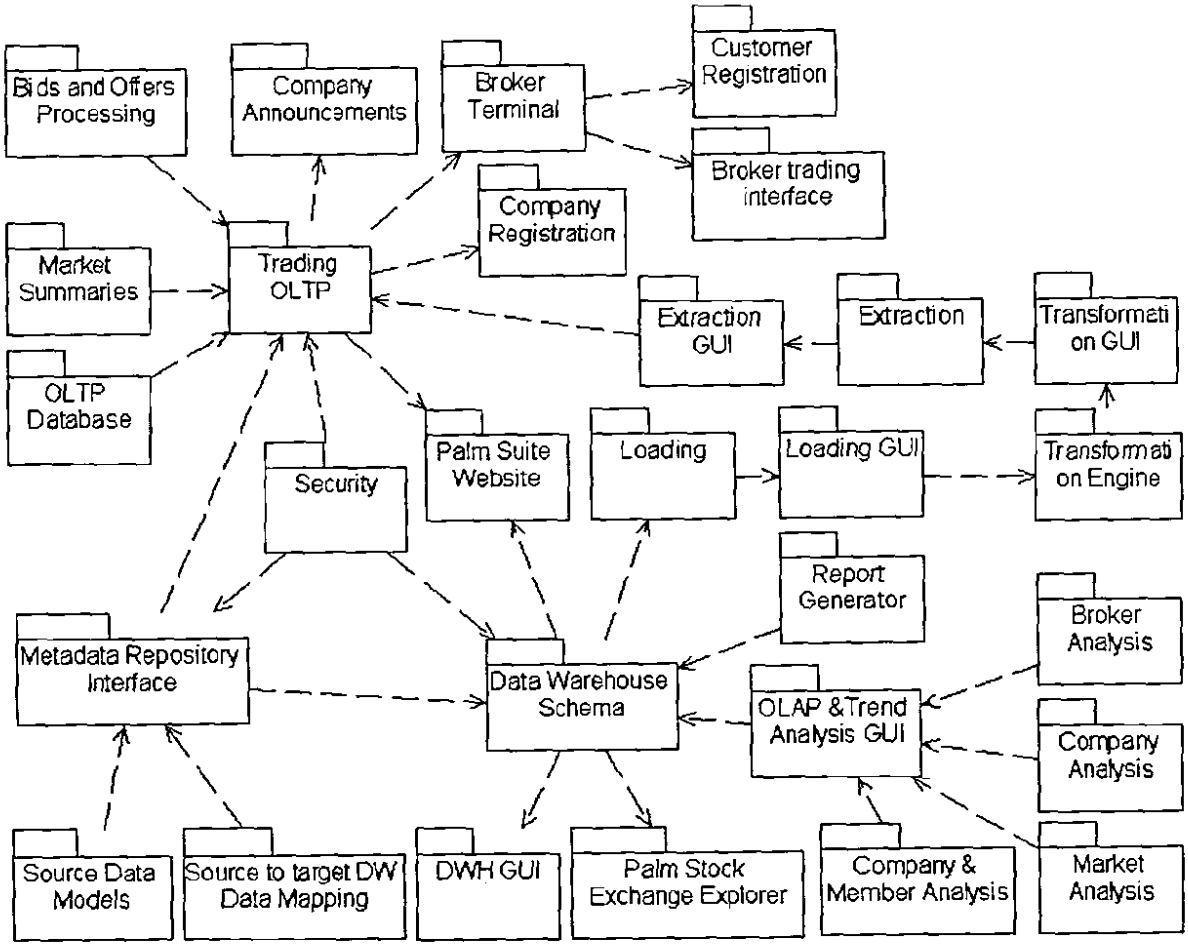


Fig. 5.1 System Software Architecture of the Data Warehouse Solution of Islamabad Stock Exchange

5.6 Step-wise Evaluation of the Architecture

In this phase the AREA-TEAM analysis team prepares itself for the analysis session. The steps-wise solution for phase 1 is given below:

Step 1
Identify Stakeholders:
All the concerned stakeholders of the project were identified which represented the members stock exchange, clients, users, software engineers, IT manager and Secretary Stock Exchange etc.

Step 2
Prepare Core Requirements:
Software Project Manager and the Software Architect identified the core requirements which were used to prepare the architecture and these were the result of the requirements engineering phase.

Step 3
Architecture Briefing:
The Software Architect prepared a briefing that explained all the components and connectors as well as the hierarchy and granularity of the architecture.

Step 4
Architecture Approaches:
A list of architectural approaches was prepared that focused on understanding the architecture and will be analyzed further.

Step 5
Prepare Materials:
Copies of presentations, core-requirements and feed-back forms were produced for distribution to the stakeholders during the next phase. The schedule of the meeting was decided and the stakeholders were invited.

After completion of Step 5, the analysis team entered phase 2 of the AREA-TEAM. In this phase the analysis team analyzed the reference architecture. The steps-wise solution for phase 2 is given below:

Step 6
Present AREA-TEAM:
A brief introduction to AREA-TEAM was given to explain the steps of the process to stakeholders so that the maximum participation of the stakeholders was assured.

Step 7
Present Software Architecture and Core-Requirements:
The core-requirements were distributed among the stakeholders and the software architecture shown in fig. 5.1 was presented in a session of about one-hour. During this session some new requirements were also identified. These are also presented here for evaluation [Table 5.2].

Step 8
Requirements Prioritization Factor:
Requirement Prioritization Factor (PF) was calculated as given in equation 4.2. The votes for each requirement were counted by show-of-hand method. The requirement prioritization metric is shown below [Table 5.1]. The same process is applied on the new requirements captured in step 7 [Table 5.2].

Requirements Prioritization Metric			
	Total Stakeholders	10	
Req ID	Requirement	Votes (V)	PF
REQDWISE1	When and which user tried to access which data	7	0.7
REQDWISE2	Synchronization between Windows and Palm OS	4	0.4
REQDWISE3	Data mapping objects of sources to data warehouse objects	6	0.6
REQDWISE4	Efficient trend analysis mechanism of companies and brokers	10	1
REQDWISE5	Client must be facilitated to access the market status summary and trend analysis through palm device	8	0.8
REQDWISE6	Online Transaction Processing System	10	1
REQDWISE7	Efficient bids and offers processing in OLTP	10	1
REQDWISE8	Generation of daily market and member reports	9	0.9
REQDWISE9	GUI based dynamic back ups and recovery	3	0.3
REQDWISE10	Meta data repository	4	0.4
REQDWISE11	Multipurpose website for Palm software downloading, other software and updates	7	0.7
REQDWISE12	Authorized logging	10	1
REQDWISE13	Member Commission Calculation	7	0.7
REQDWISE14	Companies registration and announcements storage	9	0.9
REQDWISE15	Market summaries of competitive stock exchanges	6	0.6
REQDWISE16	Dynamic calculation of market index on every second	10	1
REQDWISE17	Flushing facility of OLTP data	3	0.3
REQDWISE18	GUI based administration of Data Warehouse	3	0.3

REQDWISE19	3D graph trend analyzers	8	0.8
REQDWISE20	2D and 3D cross tabbed reports	9	0.9
REQDWISE21	GUI for extraction should by there	4	0.4
REQDWISE22	GUI for transformation should be there	4	0.4
REQDWISE23	GUI for loading should be there	4	0.4

Table 5.1 Core Requirements, Votes & Priority Factor

New Requirements Prioritization Metric			
	Total Stakeholders	10	
New Req ID	New Requirement	Votes (V)	NPF
NREQDWISE1	New member registration	10	1
NREQDWISE2	Agents' records storage	6	0.6
NREQDWISE3	Agents' mappings on the transactions	6	0.6
NREQDWISE4	Forecasting of the companies and market index	5	0.5
NREQDWISE5	Loading logs should be maintained	4	0.4
NREQDWISE6	Members loans and balancing	7	0.7

Table 5.2 New Requirements, Votes & Priority Factor

Step 9

Requirement Significance Factor:

The requirement significance factor was then gathered with the help of the feed-back forms. The data from the feed-back forms is recorded in the requirements significance metric [Table 5.3]. Then Architecture Significance Factor (ASF) is calculated according to the equation 4.3. The New Requirements are also processed in the same way [Table 5.4] and nASF is calculated for each new requirement.

Requirements Significance Metric											
Total Stakeholders			10								
Req ID	Stakeholders										ASF
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	
REQDWISE1	70	65	35	55	40	75	65	35	35	65	0.54
REQDWISE2	60	75	45	30	35	30	60	40	30	55	0.46
REQDWISE3	65	65	40	55	65	70	55	45	35	35	0.53
REQDWISE4	65	60	55	60	60	55	60	50	50	60	0.575
REQDWISE5	70	75	65	60	50	60	45	60	55	55	0.595
REQDWISE6	80	80	70	65	60	65	70	75	70	65	0.7
REQDWISE7	75	80	65	70	65	75	70	70	60	60	0.69
REQDWISE8	65	65	75	80	75	70	65	60	55	70	0.68
REQDWISE9	50	60	35	30	45	40	35	55	35	35	0.42
REQDWISE10	70	70	55	45	60	55	45	35	60	50	0.545
REQDWISE11	55	50	45	50	60	65	50	45	65	55	0.54
REQDWISE12	60	70	55	65	60	70	55	60	60	50	0.605
REQDWISE13	70	75	50	60	65	65	70	65	60	65	0.645

REQDWISE14	55	65	55	60	50	50	60	55	35	50	0.535
REQDWISE15	50	55	75	70	65	70	65	70	75	65	0.66
REQDWISE16	80	75	80	70	75	75	75	80	70	75	0.755
REQDWISE17	50	55	35	30	35	45	30	35	35	30	0.38
REQDWISE18	50	65	45	35	50	30	40	30	35	40	0.42
REQDWISE19	65	60	70	75	65	60	65	40	70	45	0.615
REQDWISE20	65	60	70	45	75	60	65	70	70	65	0.645
REQDWISE21	65	65	45	40	35	45	40	35	30	40	0.44
REQDWISE22	65	65	45	40	35	45	40	35	30	40	0.44
REQDWISE23	65	65	45	40	35	45	40	35	30	40	0.44

Table 5.3 Requirements Significance Metric

New Requirements Significance Metric											
Total Stakeholders			10								
New Req ID	Stakeholders										NASF
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	
NREQDWISE1	60	65	65	70	65	60	65	70	65	65	0.65
NREQDWISE2	65	65	60	55	70	65	45	55	60	70	0.61
NREQDWISE3	70	65	55	65	75	65	65	60	55	65	0.64
NREQDWISE4	75	70	65	55	60	70	65	65	55	55	0.635
NREQDWISE5	70	70	50	45	55	45	55	60	40	35	0.525
NREQDWISE6	65	70	70	75	65	70	65	60	55	55	0.65

Table 5.4 New Requirements Significance Metric

Step 10
Requirement Criticality Analysis:
The requirement criticality was calculated by finding the mean of prioritization factors and average significance factors for each requirement [Equation 4.4]. The results are recorded in requirements criticality metric [Table 5.5]. The Total Requirement Criticality (TRC) is then formulated [equation 4.8]. The same process is applied on the novel requirements and the results are shown in new requirements criticality metric [Table 5.6].

Step 11
Novel Requirements Prioritization Factor, Average Significance Factor and Criticality Analysis:
The Novel Requirements Prioritization Factor (NPF), Average Significance Factor (NASF) and the Requirements Criticality (NRC) are calculated in [Table 5.2, 5.4, 5.6] [Equation 4.5, 4.6, 4.7].

Requirements Criticality Metric				
Req ID	Votes (V)	PF	ASF	RC
REQDWISE1	7	0.7	0.54	0.62
REQDWISE2	4	0.4	0.46	0.43
REQDWISE3	6	0.6	0.53	0.565
REQDWISE4	10	1	0.575	0.788
REQDWISE5	8	0.8	0.595	0.698
REQDWISE6	10	1	0.7	0.85
REQDWISE7	10	1	0.69	0.845
REQDWISE8	9	0.9	0.68	0.79
REQDWISE9	3	0.3	0.42	0.36
REQDWISE10	4	0.4	0.545	0.473
REQDWISE11	7	0.7	0.54	0.62
REQDWISE12	10	1	0.605	0.803
REQDWISE13	7	0.7	0.645	0.673
REQDWISE14	9	0.9	0.535	0.718
REQDWISE15	6	0.6	0.66	0.63
REQDWISE16	10	1	0.755	0.878
REQDWISE17	3	0.3	0.38	0.34
REQDWISE18	3	0.3	0.42	0.36
REQDWISE19	8	0.8	0.615	0.708
REQDWISE20	9	0.9	0.645	0.773
REQDWISE21	4	0.4	0.44	0.42
REQDWISE22	4	0.4	0.44	0.42
REQDWISE23	4	0.4	0.44	0.42
			TRC	14.18

Table 5.5 Requirement Criticality Metric

New Requirements Criticality Metric				
New Req ID	Votes (V)	NPF	NASF	NRC
NREQDWISE1	10	1	0.65	0.825
NREQDWISE2	6	0.6	0.61	0.605
NREQDWISE3	6	0.6	0.64	0.62
NREQDWISE4	5	0.5	0.64	0.568
NREQDWISE5	4	0.4	0.53	0.463
NREQDWISE6	7	0.7	0.65	0.675
			NTRC	3.755

Table 5.6 New Requirements Criticality Metric

Step 12

Analyze Architectural Approaches:

Then each requirement was analyzed and the analysis team queried the architect that how he handled the requirements in the software architecture. The analysis is performed in Architecture Approach Analysis Metric. We are showing the analysis for all the key requirements (Table 5.7 to Table 5.35). Some of the examples are discussed below:

The first requirement REQDWISE1 is analyzed. According to [Table 5.7] this requirement is not handled in the architecture and the score 0 is given to the architecture contribution factor (ACF).

Another example is given for requirement REQDWISE4. According to the [Table 5.10] this requirement is handled in the architecture by the architect and score 1 is given to the architecture contribution factor (ACF).

Next example is selected for a new requirement NREQDWISE6. According to the [Table 5.12] this requirement is not handled in the architecture by the architect and the score 0 is given to the architecture contribution factor (ACF).

Last example is provided for a new requirement NREQDWISE5. According to [Table 5.34] the requirement is already handled in the architecture and the score 1 is assigned to the architecture contribution factor (ACF). Here we can see that although the requirement was not provided earlier but the architect handled the requirement.

Architecture Approach Analysis		
Req ID	Requirement	When and which user tried to access which data.
REQDWISE1		
Requirement Criticality		0.62
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.7 Architecture Approach Analysis REQDWISE1

Architecture Approach Analysis		
Req ID	Requirement	Synchronization between Windows and Palm OS
REQDWISE2		
Requirement Criticality		0.43
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.8 Architecture Approach Analysis REQDWISE2

Architecture Approach Analysis		
Req ID	Requirement	Data mapping objects of sources to data warehouse
REQDWISE3		objects
Requirement Criticality		0.565
Architecture Approach		ACF

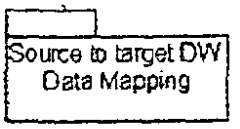
Source to target DW data mapping	1	
Remarks	Source to target DW data mapping module is present	
Architecture Diagram		
		

Table 5.9 Architecture Approach Analysis REQDWISE3

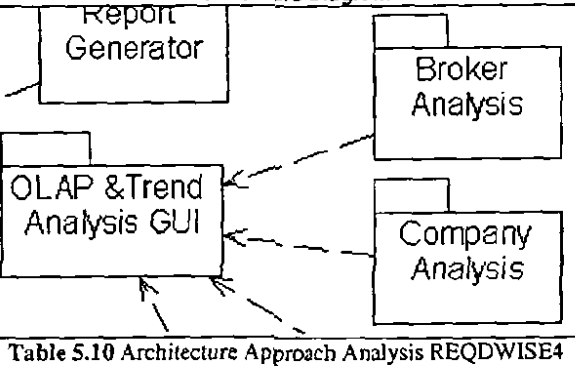
Architecture Approach Analysis		
Req ID	**Requirement**	Efficient trend analysis mechanism of companies
REQDWISE4		and brokers.
Requirement Criticality		0.788
Architecture Approach		**ACF**
Companies Analysis and Broker Analysis Components		1
Remarks		Companies Analysis and Broker Analysis
		Components are present in the architecture
Architecture Diagram		
		

Table 5.10 Architecture Approach Analysis REQDWISE4

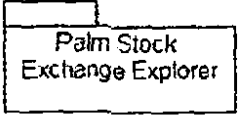
Architecture Approach Analysis		
Req ID	**Requirement**	Client must be facilitated to access the market status summary
REQDWISE5		and trend analysis through palm device
Requirement Criticality		0.698
Architecture Approach		**ACF**
Palm Stock Exchange Explorer Component		1
Remarks		Palm Stock Exchange Explorer Component is present
Architecture Diagram		
		

Table 5.11 Architecture Approach Analysis REQDWISE5

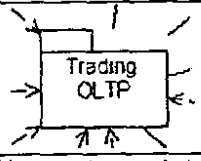
Architecture Approach Analysis			
Req ID	**Requirement**	Online Transaction Processing System	
REQDWISE6			
Requirement Criticality		0.85	
Architecture Approach		ACF	
Trading OLTP Component		1	
Remarks		Trading OLTP Component is present in the architecture	
Architecture Diagram			
			

Table 5.12 Architecture Approach Analysis REQDWISE6

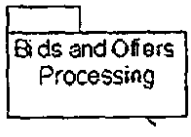
Architecture Approach Analysis			
Req ID	**Requirement**	Efficient bids and offers processing in OLTP	
REQDWISE7			
Requirement Criticality		0.845	
Architecture Approach		ACF	
Bids and Offers Processing Module		1	
Remarks		Bids and Offers Processing Module is present in the architecture	
Architecture Diagram			
			

Table 5.13 Architecture Approach Analysis REQDWISE7

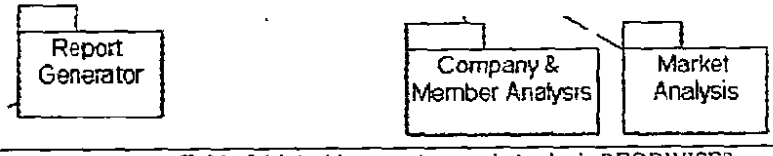
Architecture Approach Analysis			
Req ID	**Requirement**	Generation of daily market and member reports	
REQDWISE8			
Requirement Criticality		0.79	
Architecture Approach		ACF	
Report Generator Module		1	
Remarks		Report Generator Module is present in the architecture	
Architecture Diagram			
			

Table 5.14 Architecture Approach Analysis REQDWISE8

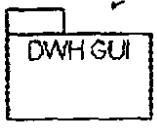
Architecture Approach Analysis		
Req ID	Requirement	GUI based dynamic back ups and recovery
REQDWISE9		
Requirement Criticality		0.36
Architecture Approach		ACF
DWH GUI Module		1
Remarks		DWH GUI Module is present in the architecture
Architecture Diagram		
		

Table 5.15 Architecture Approach Analysis REQDWISE9

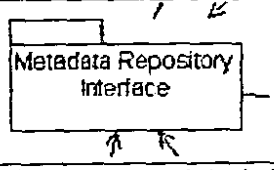
Architecture Approach Analysis		
Req ID	Requirement	Meta data repository
REQDWISE10		
Requirement Criticality		0.473
Architecture Approach		ACF
MetaData Repository and attached Components		1
Remarks		MetaData Repository and attached Components are present in the architecture
Architecture Diagram		
		

Table 5.16 Architecture Approach Analysis REQDWISE10

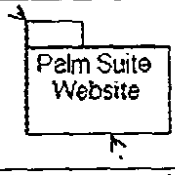
Architecture Approach Analysis		
Req ID	Requirement	Multipurpose website for Palm software downloading, other software and updates
REQDWISE11		
Requirement Criticality		0.62
Architecture Approach		ACF
Palm Suite Website Component		1
Remarks		Palm Suite Website Component is present in the architecture
Architecture Diagram		
		

Table 5.17 Architecture Approach Analysis REQDWISE11

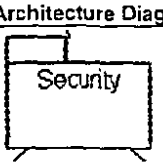
Architecture Approach Analysis		
Req ID	Requirement	Authorized logging
REQDWISE12		
Requirement Criticality		0.803
Architecture Approach		ACF
Security Module		1
Remarks		Security Module is present in the architecture
Architecture Diagram		
		

Table 5.18 Architecture Approach Analysis REQDWISE12

Architecture Approach Analysis		
Req ID	Requirement	Member Commission Calculation
REQDWISE13		
Requirement Criticality		0.673
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.19 Architecture Approach Analysis REQDWISE13

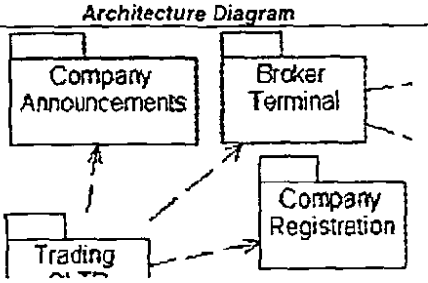
Architecture Approach Analysis		
Req ID	Requirement	Companies registration and announcements storage
REQDWISE14		
Requirement Criticality		0.718
Architecture Approach		ACF
Company Announcements, Company Registration Module		1
Remarks		Company Announcements, Company Registration Module are present in the architecture
Architecture Diagram		
		

Table 5.20 Architecture Approach Analysis REQDWISE14

Architecture Approach Analysis		
Req ID	Requirement	Market summaries of competitive stock exchanges
REQDWISE15		
Requirement Criticality		0.63
Architecture Approach		ACF
Market Summaries Module		1
Remarks		Market Summaries Module is present in the architecture
Architecture Diagram		
		

Table 5.21 Architecture Approach Analysis REQDWISE15

Architecture Approach Analysis		
Req ID	Requirement	Dynamic calculation of market index on every second
REQDWISE16		
Requirement Criticality		0.878
Architecture Approach		ACF
Market Summaries Module		1
Remarks		Market Summaries Module is present in the architecture
Architecture Diagram		
		

Table 5.22 Architecture Approach Analysis REQDWISE16

Architecture Approach Analysis		
Req ID	Requirement	Flushing facility of OLTP data
REQDWISE17		
Requirement Criticality		0.34
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.23 Architecture Approach Analysis REQDWISE17

Architecture Approach Analysis		
Req ID	Requirement	GUI based administration of Data Warehouse
REQDWISE18		
Requirement Criticality		0.36
Architecture Approach		ACF
DWH GUI Module		1

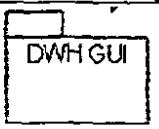
Remarks	DWH GUI Module is present in the architecture
Architecture Diagram	
	

Table 5.24 Architecture Approach Analysis REQDWISE18

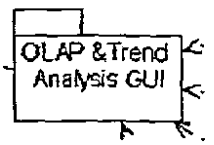
Architecture Approach Analysis			
Req ID	Requirement	3D graph trend analyzers	
REQDWISE19			
Requirement Criticality		0.708	
Architecture Approach		ACF	
OLAP & Trend Analysis GUI		1	
Remarks		OLAP & Trend Analysis GUI is present in the architecture	
Architecture Diagram			
			

Table 5.25 Architecture Approach Analysis REQDWISE19

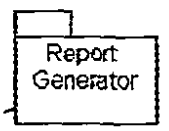
Architecture Approach Analysis			
Req ID	Requirement	2D and 3D cross tabbed reports	
REQDWISE20			
Requirement Criticality		0.773	
Architecture Approach		ACF	
Report Generator Module		1	
Remarks		Report Generator Module is present in the architecture	
Architecture Diagram			
			

Table 5.26 Architecture Approach Analysis REQDWISE20

Architecture Approach Analysis			
Req ID	Requirement	GUI for extraction should by there	
REQDWISE21			
Requirement Criticality		0.42	
Architecture Approach		ACF	
Extraction GUI Module		1	
Remarks		Extraction GUI Module is present in the architecture	
Architecture Diagram			

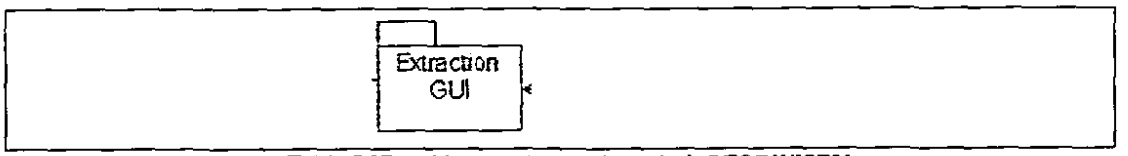


Table 5.27 Architecture Approach Analysis REQDWISE21

Architecture Approach Analysis			
Req ID	Requirement	GUI for transformation should be there	
REQDWISE22			
Requirement Criticality		0.42	
Architecture Approach		ACF	
Transformation GUI Component		1	
Remarks		Transformation GUI Component is present in the architecture	
Architecture Diagram			
			

Table 5.28 Architecture Approach Analysis REQDWISE22

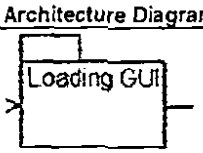
Architecture Approach Analysis			
Req ID	Requirement	GUI for loading should be there	
REQDWISE23			
Requirement Criticality		0.42	
Architecture Approach		ACF	
Loading GUI Module		1	
Remarks		Loading GUI Module is present in the architecture	
Architecture Diagram			
			

Table 5.29 Architecture Approach Analysis REQDWISE23

Architecture Approach Analysis			
Req ID	Requirement	New member registration	
NREQDWISE1			
Requirement Criticality		0.825	
Architecture Approach		ACF	
NIL		0	
Remarks		Not handled in the architecture.	
Architecture Diagram			

Table 5.30 Architecture Approach Analysis NREQDWISE1

Architecture Approach Analysis		
Req ID	Requirement	Agents' records storage
NREQDWISE2		
Requirement Criticality		0.605
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.31 Architecture Approach Analysis NREQDWISE2

Architecture Approach Analysis		
Req ID	Requirement	Agents' mappings on the transactions
NREQDWISE3		
Requirement Criticality		0.62
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.32 Architecture Approach Analysis NREQDWISE3

Architecture Approach Analysis		
Req ID	Requirement	Forecasting of the companies and market index
NREQDWISE4		
Requirement Criticality		0.568
Architecture Approach		ACF
NIL		0
Remarks		Not handled in the architecture.
Architecture Diagram		

Table 5.33 Architecture Approach Analysis NREQDWISE4

Architecture Approach Analysis		
Req ID	Requirement	Loading logs should be maintained
NREQDWISE5		
Requirement Criticality		0.463
Architecture Approach		ACF
Loading GUI Module		1
Remarks		Loading GUI Module is present in the architecture
Architecture Diagram		

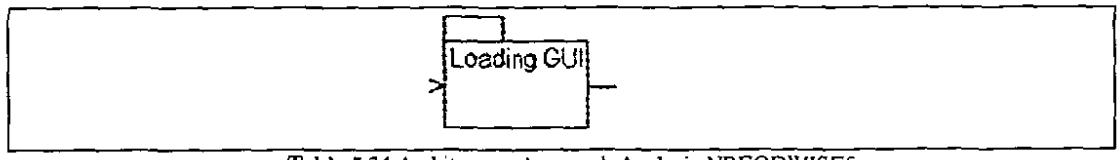


Table 5.34 Architecture Approach Analysis NREQDWISE5

Architecture Approach Analysis			
Req ID	Requirement	Members loans and balancing	
NREQDWISE6			
Requirement Criticality		0.675	
Architecture Approach		ACF	
NIL		0	
Remarks		Not handled in the architecture.	
Architecture Diagram			

Table 5.35 Architecture Approach Analysis NREQDWISE6

Step 13

Architecture Accuracy Factor and Architecture Error Factor:

In the last step for moving towards final results calculations, the architecture approach contribution percentage (AAC%) is calculated for each requirement as per equation 4.9 [Table 5.36]. Then the Architecture Accuracy Factor (AAF) as per equation 4.10. The Architecture Error Factor (AEF) is also calculated here according to equation 4.11.

Architecture Accuracy Factor (AAF) shows the correctness of the architecture in percentage and the Architecture Error Factor (AEF) shows the percentage of error present in the architecture, and this is the error of the architect.

$$E_a \approx AEF$$

To calculate the error propagated due to the stakeholder (normally the customer) we included the new requirements captured in step 7 in our analysis. First of all we calculated the New Total Requirement Contribution (NTRC) for each new requirement captured [Equation 4.12]. Then the Architecture Approach Contribution percentage (NAAC%) is calculated as per equation 4.12 and the results are recorded in Architecture Accuracy Metric with new Requirements [Table 5.37]. The New Architecture Accuracy

Factor (NAAF) and the New Architecture Error Factor (NAEF) is also calculated over here.

The New Architecture Error Factor reflects the ratio by which the architecture has not addressed the requirements including the new requirements. The stakeholder error factor (SEF) is calculated with the equation 4.16. The SEF was 15.31%, which means that the error factor introduced in the architecture due to the stakeholder is 15.31%.

$$SEF = NAEF - AEF$$
$$SEF = 29.87 - 14.56 = 15.31\%$$

$$E_s = SEF$$

The adverse effect introduced into architecture accuracy due to the stakeholder is calculated by the equation 4.17. The resultant SAF is -15.31%, the negative sign shows it is an adverse effect.

$$SAF = NAAF - AAF$$
$$SAF = 70.13 - 85.44 = -15.31\%$$

Also note that both the ratios SAF and SEF are equal, which shows that both the positive and negative sides have the equal effect by the stakeholder.

Architecture Accuracy Metric				
			TRC	14.18
Req ID	Architecture Approach	ACF	RC	AAC%
REQDWISE1	Not handled in the architecture	0	0.62	4.372
REQDWISE2	Not handled in the architecture	0	0.43	3.032
REQDWISE3	Source to target DW data mapping	1	0.565	3.984
REQDWISE4	Company Analysis and Broker Analysis Module	1	0.788	5.554
REQDWISE5	Palm Stock Exchange Explorer Component	1	0.698	4.919
REQDWISE6	Trading OLTP Component	1	0.85	5.994
REQDWISE7	Bids and Offers Processing Module	1	0.845	5.959
REQDWISE8	Report Generator Module	1	0.79	5.571
REQDWISE9	DWH GUI Module	1	0.36	2.539

REQDWISE10	MetaData Repository and attached Components	1	0.473	3.332
REQDWISE11	Palm Suite Website Component	1	0.62	4.372
REQDWISE12	Security Module	1	0.803	5.659
REQDWISE13	Not handled in the architecture	0	0.673	4.743
REQDWISE14	Company Announcements, Company Registration Module	1	0.718	5.06
REQDWISE15	Market Summaries Module	1	0.63	4.443
REQDWISE16	Market Summaries Module	1	0.878	6.188
REQDWISE17	Not handled in the architecture	0	0.34	2.398
REQDWISE18	DWH GUI Module	1	0.36	2.539
REQDWISE19	OLAP & Trend Analysis GUI	1	0.708	4.989
REQDWISE20	Report Generator Module	1	0.773	5.448
REQDWISE21	Extraction GUI Module	1	0.42	2.962
REQDWISE22	Transformation GUI Component	1	0.42	2.962
REQDWISE23	Loading GUI Module	1	0.42	2.962
	AAF	85.44%	AEF	14.56%

Table 5.36 Architecture Accuracy Metric, AAF & AEF

Architecture Accuracy Metric with New Requirements				
NTRC	3.755	TRC+NTRC		17.935
New Req ID	Architecture Approach	NACF	NRC	NAAC%
REQDWISE1	Not handled in the architecture	0	0.62	3.457
REQDWISE2	Not handled in the architecture	0	0.43	2.398
REQDWISE3	Source to target DW data mapping	1	0.57	3.15
REQDWISE4	Company Analysis and Broker Analysis Module	1	0.79	4.391
REQDWISE5	Palm Stock Exchange Explorer Component	1	0.7	3.889
REQDWISE6	Trading OLTP Component	1	0.85	4.739
REQDWISE7	Bids and Offers Processing Module	1	0.85	4.711
REQDWISE8	Report Generator Module	1	0.79	4.405
REQDWISE9	DWH GUI Module	1	0.36	2.007
REQDWISE10	MetaData Repository and attached Components	1	0.47	2.635
REQDWISE11	Palm Suite Website Component	1	0.62	3.457
REQDWISE12	Security Module	1	0.8	4.474
REQDWISE13	Not handled in the architecture	0	0.67	3.75
REQDWISE14	Company Announcements, Company Registration Module	1	0.72	4.001
REQDWISE15	Market Summaries Module	1	0.63	3.513
REQDWISE16	Market Summaries Module	1	0.88	4.893
REQDWISE17	Not handled in the architecture	0	0.34	1.896
REQDWISE18	DWH GUI Module	1	0.36	2.007
REQDWISE19	OLAP & Trend Analysis GUI	1	0.71	3.945
REQDWISE20	Report Generator Module	1	0.77	4.307
REQDWISE21	Extraction GUI Module	1	0.42	2.342
REQDWISE22	Transformation GUI Component	1	0.42	2.342
REQDWISE23	Loading GUI Module	1	0.42	2.342
NREQDWISE1	Not handled in the architecture	0	0.83	4.6
NREQDWISE2	Not handled in the architecture	0	0.61	3.373
NREQDWISE3	Not handled in the architecture	0	0.62	3.457
NREQDWISE4	Not handled in the architecture	0	0.57	3.164
NREQDWISE5	Loading GUI Module	1	0.46	2.579

NREQDWISE6	Not handled in the architecture		0	0.68	3.764
		NAAF	70.13%	NAEF	29.87%

Table 5.37 Architecture Accuracy Metric with New Requirements

Chapter 6

RESULTS & CONCLUSION

6 Results & Conclusion

Adapting the architecture in earlier steps of development is recommended but its evaluation is more important. An architecture analysis method, any architecture analysis method, is garbage-in-garbage out process. The AREA-TEAM is no different. It crucially relies on the active and willing participation of the stakeholders (particularly the architecture team); we advance preparation by the key stakeholders; an understanding of architectural design issues and analytic models; and a clearly articulated set of core requirements and a set of business goals from which they are derived. The purpose in creating a method (rather than, say, just putting some intelligent and experienced people together in a room and having them chat about the architecture or inspect it in an arbitrary way) is to increase the effectiveness and repeatability of the analysis.

6.1 Benefits

Every AREA-TEAM exercise is followed by a survey of stakeholders in attendance. Here is what those responding had to say verbatim:

- Error in the architecture are identified
- The source of error is also identified. For example, in this case study the architect error factor is 14.56% that measure the performance of the architect and the stakeholders' error factor is 15.31% that shows the maturity of the stakeholder.
- Requirements are improved and validated
- New requirements are identified
- Stakeholder knowledge about the system increases
- Software Architecture is revised
- The new requirements are accommodated therefore the method supports agile processes
- Since this review was conducted during the high-level design phase, therefore, the errors are identified at an early stage. So the valuable resources will be saved. "The earliest an error is identified, cheaper it is to fix"

- Important quality requirements will be identified and analyzed
- The method provided the stakeholders with a chance to give a critical look at the system. It validated some architectural decisions and raised questions about others.
- "As an IT manager, it helped me to understand the system."
- The method brought issues and concerns to our attention.
- It will be more useful to use the method earlier. The later use may lead to diminish the effectiveness of the method.

REFRENCES & BIBLIOGRAPHY

References & Bibliography

[1] Bass, Len; Clements, Paul; & Kazman, Rick. ***Software Architecture in Practice, Second Edition***, Boston, MA: Addison-Wesley, 2003 pg: 45.

[2] http://www.sei.cmu.edu/architecture/

[3] http://www.sei.cmu.edu/architecture/ata_eval.html

[4] *Mary Shaw,* ***The coming-of-age of software architecture research,*** International Conference on Software Engineering, Proceedings of the 23rd International Conference on Software Engineering, Toronto, Ontario, Canada , Page: 656, Year of Publication: 2001, ISBN ~ ISSN:0270-5257 , 0-7695-1050-7

[5] *Mary Shaw,* ***Writing good software engineering research papers: minitutorial,*** Proceedings of the 25th International Conference on Software Engineering, Portland, Oregon , 2003, ISBN ~ ISSN:0270-5257 , 0-7695-1877-X.

[6] *Muhammad Amir Aman, Muhammad Sulayman, Khalid Rashid,* ***"Architecture Requirements Engineering Accuracy & Error – The Analysis Method (AREA-TEAM)".*** Published in the first International Conference on Quality of Software Architectures, Dot-Net Object Days, Erfurt Germany, Sep 19-22, 2005, book of proceedings ISBN: 3-9808628-4-4, Pages (461-472)

[7] *Muhammad Amir Aman, Muhammad Sulayman, Khalid Rashid,* ***"Architecture Requirements Engineering Accuracy & Error – The Analysis Method (AREA-TEAM)- Islamabad Stock Exchange Case Study".*** Published in 3rd IASTED International Conference on Software Engineering (SE 2006), February 14-16, 2006, Innsbruck, Austria.

[8] *Thayer, R. H. and M. Dorfman,* ***"Software Requirements Engineering",*** Second Edition, IEEE Computer Society Press, 1997.

[9] *Sommerville, I. and P. Sawyer,* ***"Requirements Engineering",*** Wiley, 1997

[10] *Bass, L., Clements, P. and Kazman, R , What is Software Architecture?, in* ***"Software Architecture in Practice,*** Boston, MA: Addison-Wesley, 2003, pp. 125.

[11] *Dennis, J. B.,* ***"Modularity"*** in Advanced Course in Software Engineering, (F.L.Bauer, ed.), Springer-Verlag, 1973, pp. 128-182.

[12] *Wirth, N.* ***"Program Development by Stepwise Refinement",*** CACM, Vol.14, no. 4, 1971, pp. 221-227.

[13] *Dah. O. E. Dijkstra and C. Hoare,* ***"Structured Programming",*** Academic Press, 1972.

[14] *Asada, T., et al.,* ***"The Quantified Design Space",*** in Software Architecture (Shaw, M. and D. Garlan), Prentice Hall1996, pp. 116-127.

[15] http://www.isixsigma.com/dictionary/Quality_Function_Deployment-305.htm

[16] http://wwwagss.informatik.uni-kl.de/Projekte/GeneSys/Texts/DesignSpaces.html

[17] *Rick Kazman, Len Bass, Gregory Abowd, and Mike Webb,* ***"SAAM: A Method for Analyzing the Properties Software Architectures,"*** Proceedings of the 16th International Conference on Software Engineering, Sorrento, Italy, May 1994, pp. 81-90.

[18] *Klein, M., Clements, P. and Kazman, R.,* ***"ATAM: Method for Architecture Evaluation",*** August 2000, TECHNICAL REPORT, CMU/SEI-2000-TR-004, ESC-TR-2000-004

[19] *Bass, L., Klein, M., Moreno, G.,* ***"Applicability of General Scenarios to the Architecture Tradeoff Analysis Method",*** TECHNICAL REPORT, CMU/SEI-2001-TR014, ESE-TR-2001-014.

[20] *R. Kazman, M. Klein, M. Barbacci, H. Lipson, T. Longstaff, S.J. Carrière,* ***"The Architecture Tradeoff Analysis Method"***, Proceedings of ICECCS, August 1998, Monterey, CA

[21] *Kazman, Rick; Asundi, Jai; & Klein, Mark,* ***Quantifying the Costs and Benefits of Architectural Decisions,*** Proceedings of the 23rd International Conference on Software Engineering (ICSE 23), (Toronto, Canada), May 2001, 297-306

[22] *Kazman, Rick; Asundi, Jai; & Klein, Mark,* **Making Architecture Design Decisions: An Economic Approach** (CMU/SEI-2002-TR-035).

[23] *Bass, Len; Clements, Paul; & Kazman, Rick,* ***Software Architecture in Practice, Second Edition,*** Boston, MA: Addison-Wesley, 2003 pg: 308

[24] *Bashar Nuseibeh, Steve Easterbrook,* ***International Conference on Software Engineering,*** Proceedings of the Conference on The Future of Software Engineering, Limerick, Ireland, Pages: 35 - 46, Year of Publication: 2000, ISBN:1-58113-253-0

[25] *Bass, L., Clements, P. and Kazman, R* ***"Software Architecture in Practice",*** What is Software Architecture?,, Pearson Education 2003, pp. 21

[26] *Buschmann, F.; Meunier, R.; Rohnert, H.; Sommerlad, P.; & Stal, M.* ***"Pattern-Oriented Software Architecture",*** Chichester, UK: Wiley, 1996

[27] *Kazman, R. & Carriere, S.J.* ***"Playing Detective: Reconstructing Software Architecture from Available Evidence",*** Automated Software Engineering, 6, 2 (April 1999): 107-138.

[28] *Clements, P.; Bachmann, F.; Bass, L.; Garlan, D.; Ivers, J.; Little, R.; Nord, R.; & Stafford, J.* ***"Documenting Software Architectures: Views and Beyond",*** Boston, MA: Addison-Wesley, 2003.

[29] *Pressman, R.,* ***"Software Engineering A Practitioner's Approach",*** 5th Edition, Mc Graw-hill, International Edition, pp. 256-266.

[30] *Bass, Len; Clements, Paul; & Kazman, Rick.,* ***Software Architecture in Practice,*** *Second Edition,* Boston, MA: Addison-Wesley, 2003 pg: 276

[31] *STEVENS, W. P, MYERS, G.J., CONSTANTINE, L. L.:* '*Structured design*', IBM Systems Journal, 1974, 13 (2) pp. 115--139

Appendix A Publication # 1

Our first publication from this research is titled "Architecture Requirements Engineering Accuracy & Error – The Analysis Method (AREA-TEAM)". This research paper presents the newly devised software architecture analysis method AREA-TEAM. The paper has been published in the first International Conference on Quality of Software Architectures, Dot-Net Object Days, Erfurt Germany, Sep 19-22, 2005, book of proceedings ISBN: 3-9808628-4-4, Pages (461-472).

The copy of the research paper published is provided in this appendix.

ARCHITECTURE REQUIREMENTS ENGINEERING ACCURACY AND ERROR- THE ANALYSIS METHOD (AREA-TEAM)

Muhammad Amir Aman *amir@iiu.edu.pk*
Muhammad Sulayman *sulayman_67@yahoo.com*
Prof. Dr. Khalid Rashid *drkhalid@iiu.edu.pk*
Department of Computer Science
Faculty of Applied Sciences
International Islamic University, Islamabad

Abstract. Since the software architecture is a key artifact of the software development life-cycle, so architectural analysis is also a key practice involved in the process. Software architectures are usually complex and developed on the quality attributes based on preliminary requirements. Requirements are specified during requirements engineering phase and normally are incomplete and ambiguous and they are propagated in the software architecture. Our devised method "architecture requirements engineering accuracy and error-the analysis method (AREA-TEAM)", provides solid technical foundations for performing architectural analysis. It is based on the requirements and finding the architecture accuracy factor and architecture error factor. The major contribution of AREA-TEAM is quantification of the error made by the architect as well as the stakeholder, generally the customer. The AREA-TEAM is a method that is applied on a leading company's software architecture of a data warehouse solution. AREA-TEAM is refined in practice over the past one year.

1. Introduction

The purpose of this research paper is to describe the approach used behind the architecture requirements engineering accuracy and error-the analysis method (AREA-TEAM) and discusses its steps in practice. Software architecture of a program or computing system is a structure or structures of the system which comprise software elements, the externally visible properties of these elements and the relationships among them [1]. AREA-TEAM is important because it elaborates how well an architecture satisfies important requirements and finds the architecture accuracy factor and architecture error factor both in terms of errors of the architect and error by the stakeholder. Now the question arises, why do we need AREA-TEAM? We all know that architecture is a key ingredient in a successful software engineering process. The software architecture is developed based on the quality attributes identified from the key requirements [2]. The architecture is a key to achieve or failing to meet the requirements [3].

AREA-TEAM is a mean to analyze whether the requirements are achieved in the architecture. Such an analysis is significant because valuable organizational resources are allocated to the project and if at a later stage it is discovered that the architecture was not fulfilling the requirements, the resources may go astray. It also contributes to find the factor of error made by the architect and the customer. Our method uses the standard architecture documentation techniques as prescribed by SEI [15]. Architecture requirements engineering accuracy and error-the analysis method (AREA-TEAM) is a low-cost high-benefit method for analyzing software architectures in the light of the key requirements and finding the error factor. AREA-TEAM is a low-cost software architecture analysis method that produces early identification of places where the architecture is unsuitable in terms of meeting the requirements, thus resulting in lower eventual project cost. Added benefits include open communication channels between the architect and the stakeholders. AREA-TEAM relies on assembling the stakeholders to articulate what the important requirements are, and then exercising the architecture to make sure those key requirements are satisfied by the software architecture. The result is a high-fidelity architecture analysis coupled with high-quality familiarization with the architecture in light of the requirements it meets on the part of the stakeholders. The AREA-TEAM calculates the error factor, i.e., software architect's error percentage as well as customers propagated error percentage separately, and therefore it is also a mean to analyze the performance of architect and judge the customers' maturity. All the stakeholders are saved from loss by adapting a pro-active risk strategy.

The AREA-TEAM draws its inspiration and techniques from various areas: the notion of architectural styles; the Architecture Trade-off Analysis Method [4]; and the Software Architecture Analysis SAAM Method [5], which was the

predecessor to the ATAM and the process of Requirements Engineering [6] and Quantified Design Space [7]. The AREA-TEAM is intended for analysis of architecture in terms, whether it meets the requirements or not, it also quantifies the error factor. Although this is the AREA-TEAM's focus, there is a problem in operationalizing this focus. We (and the software engineering community in general) do not understand how to specify the requirements well: the "requirements elicitation", "requirements analysis and negotiation", "requirements specification" changes from system to system, from stakeholder to stakeholder, and from community to community. This propagates errors from one stage to another and finally reflects in the architecture. If the actual requirements are denoted by A and architect error factor and stakeholder error factor are denoted by E_a and E_s respectively, then the equation 1.1 shows the deviation from the desired requirements and deviated requirements are denoted by A'.

$$A' = A + E_a + E_s \tag{1.1}$$

Our method consists of a series of steps for architecture analysis and error factor quantization. First of all our method assembles the identified stakeholders in a meeting. The AREA-TEAM is presented and the core requirements are discussed. The requirements are prioritized by taking the votes of the stakeholders and prioritization factor is calculated. After that weights are assigned to each requirement in order to calculate the significance of each requirement and significance factor is calculated for each requirement. This is done with the help of the stakeholders by feedback forms.

After this average requirement significance factor is calculated for each requirement. Then the requirement criticality is calculated by finding the mean of Prioritization Factors and Significance Factors for each requirement. Then the architecture is analyzed by the analysis team and both the architecture accuracy factor and the architecture error factors are calculated. Software development is an agile process. Every time novel requirements are identified. These requirements are given by the stakeholders, generally customers. The error factor is also calculated for these new requirements. This is the stakeholder error factor. The process is further elaborated with the conceptual model of AREA-TEAM in figure 1.1.

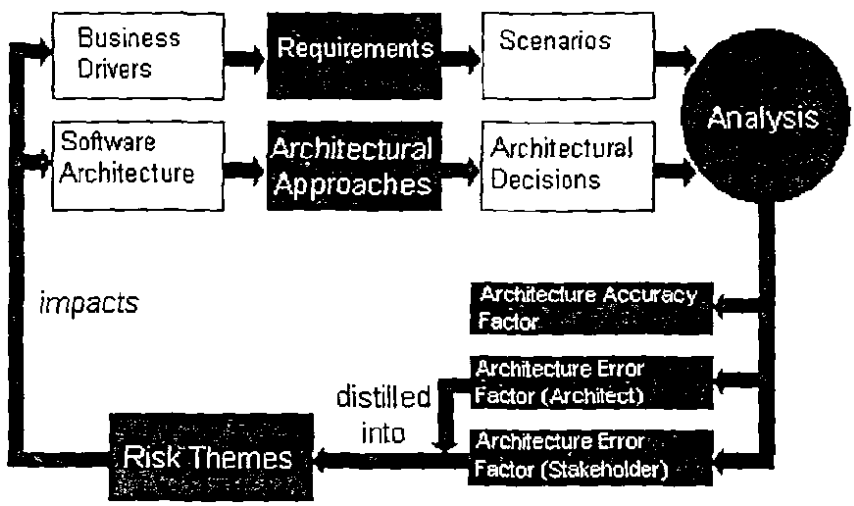


Fig. 1.1 Conceptual Model of AREA-TEAM

2. Literature Survey

The AREA-TEAM focuses on the requirements and error factor calculation during the architecture analysis process. The requirement engineering provides the appropriate mechanism for what customer wants, analyzing need, assessing feasibility, negotiating a reasonable solution, specifying requirement and their validation [8][9]. There is always some percentage of error involved on the part of both the development team and the customers. A plus point in the AREA-TEAM is that it calculates both Ea and Es separately, which helps identifying the competency of the architect and the maturity of the customer. Other methods of architecture analysis with respect to requirements are unable to differentiate between the architect and the customer errors and have no proper measurement mechanism. The initial methods were unable to provide proper mapping from requirements to architecture because many architectural patterns were very

complex and provided no provisions of mapping, for example, call-and-return architecture [10]. For that purpose structured design evolved which stressed *modularity [11], top-down design [12] and structured program [13]. Myers and Stevens* proposed dataflow oriented design translation from the architectural patterns but they were not able to measure neither the performance of dataflow diagrams nor the architectural competency and error responsible authority.

Another method design space quantization [7], a spread sheet model based on realization mechanisms and contribution analysis in the software architecture and the architectural decisions are prioritized. The drawback is that the requirements weights are neglected and no error finding mechanism is present for architecture measurement.

The ATAM [4] is another method which focuses on quality attribute requirements. It evaluates the quality attributes by considering each quality attribute in isolation and identifies sensitivity of quality attributes by various architectural attributes, which may result in evaluating the architecture w.r.t the quality scenarios [14] but there are no proper metrics for architecture correctness percentage and error propagation authority.

In the remainder of this report we will elaborate the steps of the AREA-TEAM, explain its foundations, and concludes with an extended example of applying the AREA-TEAM to a real system.

3. Steps of AREA-TEAM

AREA-TEAM consists of 12 steps divided into two phases:

Phase 1: Rehearsal. First, a meeting between the Software Project Manager and the Software Architect takes place to prepare for the exercise. This meeting usually lasts about a day. In this meeting, the following steps are taken:

Step 1: Identify stakeholders. In this case, stakeholders are the customers who will own the software, software engineers who will be expected to use the software architecture, users who will be expected to use the software and the quality assurance Inspector. We aim for approximately a dozen stakeholders, but this can vary depending on the size of the user community.

Step 2: Prepare core requirements. The Software Project Manager and the Software Architect identify a set of core requirements, which played the key role in the architectural development. These are identified to develop a consensus between the stakeholders regarding the core requirements.

Step 3: Prepare architecture briefing. The Software Architect prepares a briefing explaining the software architecture. A rule of thumb is to aim for two hours' worth of material. Include examples of how the software architecture meets the user requirements. The goal is to present the software architecture in sufficient detail so that a knowledgeable audience member could understand the software architecture.

Step 4: Identify architecture approaches. The AREA-TEAM focuses on analyzing the architecture by understanding its architectural approaches. In this step they are identified by the architect, and captured by the analysis team, but are not analyzed.
We concentrate on identifying architectural approaches and architectural styles (We look for approaches and styles because not all architects are familiar with the language of architectural styles, and so may not be able to enumerate a set of styles used in the architecture. But every architect makes architectural decisions, and the set of these we call "approaches." These can certainly be elicited from any conscientious architect) because these represent the architecture's means of addressing the highest priority requirements, the means of ensuring that the critical requirements are met in a predictable way [2]. These architectural approaches define the important structures of the system and describe the ways in which the system can grow, respond to changes, withstand attacks, integrate with other systems, and so forth.

Step 5: Prepare materials. Copies of the presentation, core requirements, and feedback forms are produced for distribution to the stakeholders during the Phase 2 meeting. The meeting is scheduled, stakeholders are invited, and steps are taken to assure the presence of a quorum of stakeholders at the meeting.

Phase 2: Architecture Requirement Engineering Analysis. Next, the stakeholders are assembled and the main activities of the architecture requirement engineering analysis commence. Nominally this phase takes about a day or two.

Step 6: Present AREA-TEAM. The Software Project Manager delivers a brief presentation explaining the steps of AREA-TEAM to the participants.

Step 7: Present software architecture and core requirements. The software architect presents the architecture overview presentation and backtrack the architecture how it meets the core requirements, which were considered for architecting the system. During this time, a scribe captures each question and any novel requirements mentioned by the stakeholders. The new requirements may or may not be reflected in the present architecture. AREA-TEAM is an agile process that is light and adaptive.

Step 8: Requirements Prioritization Factor. The AREA-TEAM analysis team uses the show-of-hands method in order to get the votes from the stakeholders. The purpose of voting is to identify and prioritize the core requirements as per interest of the stakeholders. These votes are recorded in the requirements prioritization metric [Table 4.1]. This is to negate the error factor involved in the identification of the prioritized requirements both by the architect and the stakeholders. Then the prioritization factor (*PF*) is calculated by dividing the number of votes (*V*) cast by the total number of stakeholders (*Ts*) [Equation 3.1].

$$PF_i = V_i / Ts \text{ , where } i \text{ is from 1 to } Ts \tag{3.1}$$

Step 9: Requirement Significance Factor. The significance of each requirement is gathered from the stakeholders with the help of feedback forms. The significance is constrained between 0 and 100. 0 shows no significance whereas the score of 100 shows the maximum significance. The significance for each requirement given by each stakeholder is recorded in the requirements significance metric [Table 4.3]. Then the average significance factor (*ASF*) for each requirement is calculated. It is reflected by the percentage of the average of the significance (*S*) chosen by each stakeholder for each requirement [Equation 3.2].

$$ASF_i = ((\sum_{i=1}^{n} \sum_{j=1}^{Ts} S_{ij}) / Ts) / 100 \tag{3.2}$$

Step 10: Requirement Criticality Analysis. The requirement criticality (*RC*) is calculated by finding the mean of Prioritization Factors (*PF*) and Average Significance Factors (*ASF*) for each requirement [Equation 3.3]. The readings are recorded in the requirements criticality metric [Table 4.5].

$$RC_i = (PF_i + ASF_i) / 2 \text{ , where } i \text{ is from 1 to } n. \tag{3.3}$$

Step 11: Novel Requirements Prioritization Factor, Average Significance Factor and Criticality Analysis. The novel requirements (if any recorded in step 7) are also processed in the same manner as it was done with the key requirements from step 8 to step 10 [Equations 3.4, 3.5, 3.6].

$$nPF_i = nV_i / Ts \text{ , where } i \text{ is from 1 to } n. \tag{3.4}$$

$$nASF_i = ((\sum_{i=1}^{n} \sum_{j=1}^{Ts} nS_{ij}) / Ts) / 100 \tag{3.5}$$

$$nRC_i = (nPF_i + nASF_i) / 2 \text{ , where } i \text{ from 1 to } n. \tag{3.6}$$

Step 12: Analyze Architectural Approaches. The AREA-TEAM analysis team examines the highest ranked requirements one at a time. The novel requirements are also analyzed separately. The software architect is asked to explain how the architecture supports each one and there may be a case where the architecture already fulfils some novel requirement. Team members specially the questioners probe for the architectural approaches that the architect used to carry out the requirement. The goal is for the evaluation team to find the architecture approach contribution factor (*ACF*), that is, be convinced that either the approach is appropriate in meeting the requirement for which it is intended or it is inappropriate.

If the architecture approach is appropriate then the score 1 is assigned and *if it is inappropriate* the score 0 is assigned to the approach [Table 4.7, 4.8, 4.9, 4.10].

Step 13: Architecture Accuracy Factor and Architecture Error Factor. The score assigned to each architectural approach formulated during step 12 are put in the architecture accuracy metric. Then the architecture approach contribution (*AAC*) percentage is calculated [Equation 3.8] for each requirement, by dividing the requirement criticality (*RC*) for each requirement by the *total requirement* criticality (*TRC*) [Equation 3.7]. The data is stored in Architecture Accuracy Metric [Table 4.11].

$$TRC = \sum_{i=1}^{n} RC_i \tag{3.7}$$

$$AAC_i = (RC_i / TRC) \times 100, \textit{ where i from 1 to n}. \tag{3.8}$$

Then the architecture accuracy factor (*AAF*) is calculated.

$$AAF = \sum_{i=1}^{n} AAC_i \textit{ , where } ACF_i = 1 \tag{3.9}$$

The architecture accuracy factor (*AAF*) reflects the ratio by which the architecture has successfully addressed the requirements. Then the architecture error factor (*AEF*) is calculated.

$$AEF = \sum_{i=1}^{n} AAC_i \textit{ , where } ACF_i = 0 \tag{3.10}$$

The architecture error factor (AEF) *reflects* the ratio by which the architecture has not addressed the requirements.

For calculating the error factor by the stakeholder, the new requirements recorded in step 7 are also included in the calculations process.

$$nTRC = \sum_{i=1}^{n} nRC_i \tag{3.11}$$

$$nAAC_i = (nRC_i / (TRC + nTRC)) \times 100, \textit{ where i is from 1 to n.} \tag{3.12}$$

Then the new architecture accuracy factor is calculated. The data is stored in Architecture Accuracy Metric with New Requirements [Table 4.12].

$$nAAF = \sum_{i=1}^{n} nAAC_i \textit{ , where } nACF_i = 1 \tag{3.13}$$

The new architecture accuracy factor reflects the ratio by which the architecture has successfully addressed the requirements including the new requirements. Then the new architecture error factor is calculated.

$$nAEF = \sum_{i=1}^{n} nAAC_i \textit{ , where } nACF_i = 0 \tag{3.14}$$

The new architecture error factor reflects the ratio by which the architecture has not addressed the requirements including the new requirements. The stakeholder error factor is calculated with the following equation.

$$SEF = nAEF - AEF \tag{3.15}$$

The adverse effect introduced into architecture accuracy due to the stakeholder is calculated by the following equation.

$$SAF = nAAF - AAF \tag{3.16}$$

4. Case Study of AREA-TEAM

This is an example case study based on the data warehouse solution of one of the leading cellular phone providers in Pakistan. The name is not disclosed due to confidentiality as desired by the top management of that company. The problem domain was development of a data warehouse and a trend analysis application for future predictions and analytical processing. Operational databases of the company were unable to process the data due to large volumes, distribution & relational nature. The proposed system software architecture for the solution domain is show n in the figure 4.1.

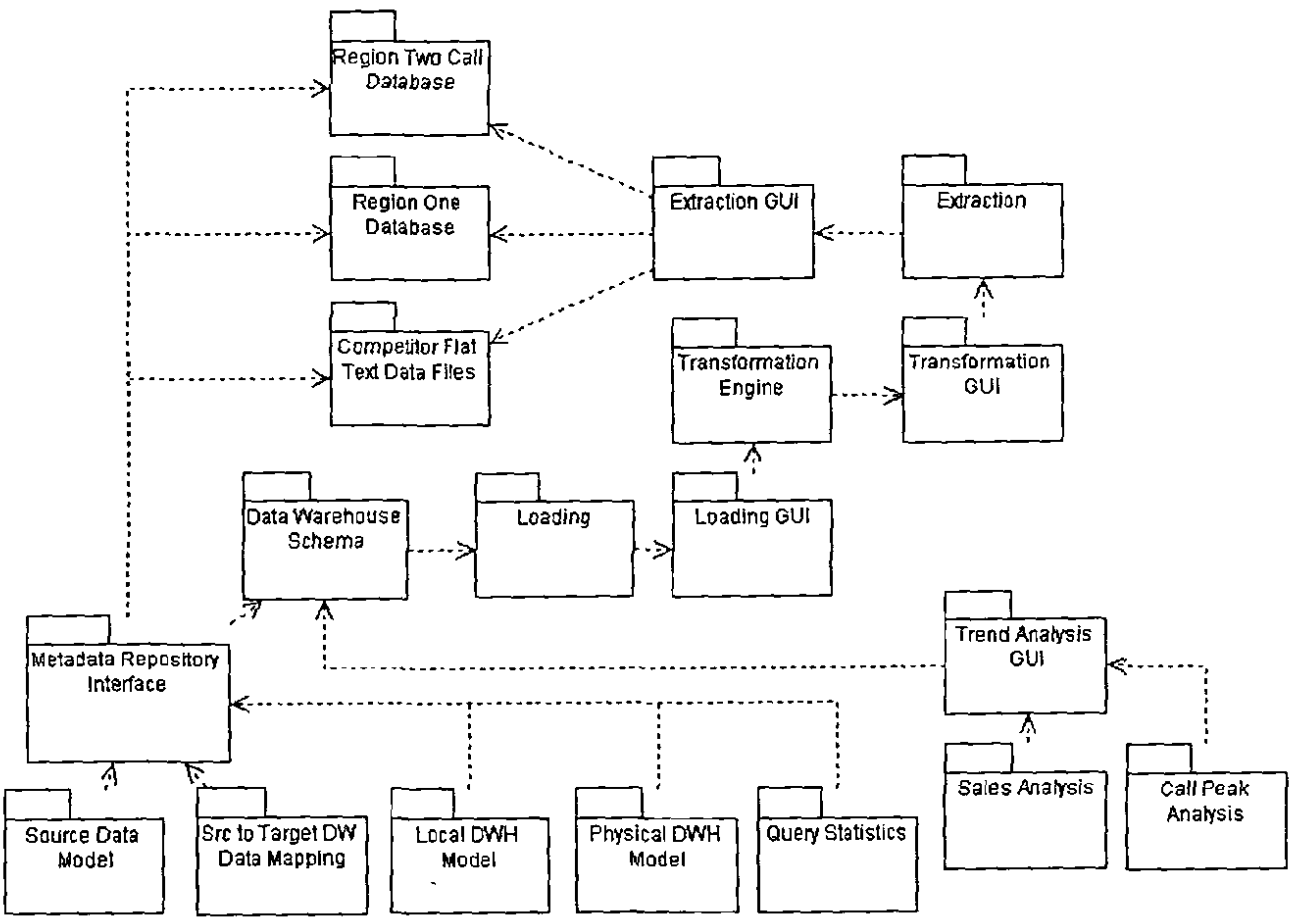


Fig. 4.1 System Software Architecture of the Data Warehouse Solution Domain

There are two phases of developed AREA-TEAM. The sample solution is provided stepwise as mentioned in section 5 of the paper. A portion of the architecture as well as the requirements is used here. The example case study fully elaborates AREA-TEAM. The phase 1 is named as *Preparation*. In this phase the AREA-TEAM analysis team prepares itself for the analysis session. The steps-wise solution for phase 1 is given below:

Step 1: Identify Stakeholders:
All the concerned stakeholders of the project were identified which represented the customers, users, software engineers, quality assurance inspector etc.

Step 2: Prepare Core Requirements:
Software Project Manager and the Software Architect identified the core requirements which were used to prepare the architecture and these were the result of the requirements engineering phase.

Step 3: Architecture Briefing:
The Software Architect prepared a briefing that explained all the components and connectors as well as the hierarchy and granularity of the architecture.

Step 4: Architecture Approaches:
A list of architectural approached was prepared that focused on understanding the architecture and will be analyzed further.

Step 5: Prepare Materials:
Copies of presentations, core-requirements and feed-back forms were produced for distribution to the stakeholders during the next phase. The schedule of the meeting was decided and the stakeholders were invited.

Now we enter the phase 2 of the AREA-TEAM. The phase 2 is called "Architecture Requirement Engineering Analysis.

Step 6: Present AREA-TEAM:
A brief introduction to AREA-TEAM was given to explain the steps of the process to the stakeholders so that the maximum participation of the stakeholders was assured.

Step 7: Present Software Architecture and Core-Requirements:
The core-requirements were distributed among the stakeholders and the software architecture shown in fig. 4.1 was presented in a session of about one-hour. During this session some new requirements were also identified. Three of them are presented here for sample evaluation.

Step 8: Requirements Prioritization Factor:
Requirement Prioritization Factor (*PF*) was calculated as given in equation 3.1. The votes for each requirement were counted by show-of-hand method. The requirement prioritization metric is shown below [Table 4.1]. The same process is applied on the new requirements captured in step 7 [Table 4.2].

Requirements Prioritization Metric			
	Total Stakeholders	15	
Req ID	Requirement	Votes (V)	PF
REQTDW1	There should be efficient data loading mechanism.	4	0.267
REQTDW2	Data from different sources should be transformed into a uniform Data Warehouse schema.	9	0.6
REQTDW3	Logical Design of data warehouse model with all its iterations and revisions.	3	0.2
REQTDW4	Physical Design of data warehouse model with all its iterations and revisions.	3	0.2
REQTDW5	When and which user tried to access which data.	10	0.667
REQTDW6	Loading logs should be maintained.	14	0.933
REQTDW7	Logical and Physical model of the source database with revision facilities.	4	0.267
REQTDW8	Data mapping objects of sources to data warehouse objects.	10	0.667
REQTDW9	Call peak-timing trend analysis reporting should be there.	14	0.933
REQTDW10	GUI for extraction, transformation and loading should by there.	12	0.8

Table 4.1 Core Requirements, Votes & Priority Factor

New Requirements Prioritization Metric			
	Total Stakeholders	15	
New Req ID	New Requirement	Votes (V)	nPF
NREQTDW1	Region and Sub-Region wise trend analysis reporting should be there.	13	0.867
NREQTDW2	Sales trend analysis report should be there.	14	0.933
NREQTDW3	Executive summary of daily sales should be automatically emailed to the top-management.	12	0.8

Table 4.2 New Requirements, Votes & Priority Factor

Step 9: Requirement Significance Factor:
The requirement significance factor was then gathered with the help of the feed-back forms. The data from the feed-back forms is recorded in the requirements significance metric [Table 4.3]. Then Architecture Significance Factor (*ASF*) is calculated according to the equation 3.2. The New Requirements are also processed in the same way [Table 4.4] and nASF is calculated for each new requirement.

Requirements Significance Metric																
Total Stakeholders			15													
Req ID	Stakeholders															ASF
	S1	*S2*	*S3*	*S4*	*S5*	*S6*	*S7*	*S8*	*S9*	*S10*	*S11*	*S12*	*S13*	*S14*	*S15*	
REQTDW1	70	80	66	55	65	84	68	70	90	88	70	50	45	75	48	0.683
REQTDW2	65	76	85	70	55	80	69	77	82	45	75	65	75	67	74	0.707
REQTDW3	30	20	50	70	80	15	85	20	34	28	56	60	20	10	40	0.412
REQTDW4	25	40	45	65	77	10	75	25	30	30	55	60	30	15	40	0.415
REQTDW5	80	75	90	85	44	90	94	100	80	75	20	95	70	50	80	0.752
REQTDW6	30	50	45	60	82	70	35	60	70	42	50	75	25	70	40	0.536
REQTDW7	25	30	50	62	75	5	15	35	35	75	60	50	20	35	45	0.411
REQTDW8	85	80	95	75	50	75	90	82	72	65	90	75	70	62	90	0.771
REQTDW9	80	75	87	95	90	60	90	100	85	95	65	95	85	80	95	0.851
REQTDW10	75	60	80	65	40	75	85	50	90	70	80	50	42	57	75	0.663

Table 4.3 Requirements Significance Metric

New Requirements Significance Metric																
Total Stakeholders			15													
New Req ID	Stakeholders															nASF
	S1	*S2*	*S3*	*S4*	*S5*	*S6*	*S7*	*S8*	*S9*	*S10*	*S11*	*S12*	*S13*	*S14*	*S15*	
NREQTDW1	75	55	70	65	55	75	80	65	80	75	90	80	50	45	68	0.686
NREQTDW2	75	65	70	80	67	70	75	65	78	80	75	65	75	85	74	0.733
NREQTDW3	60	50	55	65	90	20	80	40	55	75	65	70	55	60	65	0.603

Table 4.4 New Requirements Significance Metric

Step 10: Requirement Criticality Analysis:
The requirement criticality was calculated by finding the mean of prioritization factors and average significance factors for each requirement [Equation 3.3]. The results are recorded in requirements criticality metric [Table 4.5]. The Total Requirement Criticality (*TRC*) is then formulated [equation 3.7]. The same process is applied on the novel requirements and the results are shown in new requirements criticality metric [Table 4.6].

Requirements Criticality Metric				
Req ID	Votes (V)	PF	ASF	RC
REQTDW1	4	0.267	0.683	0.475
REQTDW2	9	0.6	0.707	0.653
REQTDW3	3	0.2	0.412	0.306
REQTDW4	3	0.2	0.415	0.307
REQTDW5	10	0.667	0.752	0.709
REQTDW6	14	0.933	0.536	0.735
REQTDW7	4	0.267	0.411	0.339
REQTDW8	10	0.667	0.771	0.719
REQTDW9	14	0.933	0.851	0.892
REQTDW10	12	0.8	0.663	0.731
			TRC	5.867

Table 4.5 Requirement Criticality Metric

New Requirements Criticality Metric				
New Req ID	*Votes (V)*	*nPF*	*nASF*	*nRC*
NREQTDW1	13	0.867	0.686	0.776
NREQTDW2	14	0.933	0.733	0.833
NREQTDW3	12	0.8	0.603	0.702
			nTRC	2.311

Table 4.6 New Requirements Criticality Metric

Step 11: Novel Requirements Prioritization Factor, Average Significance Factor and Criticality Analysis:
The Novel Requirements Prioritization Factor (*nPF*), Average Significance Factor (*nASF*) and the Requirements Criticality (*nRC*) are calculated in [Table 4.2, 4.4, 4.6] [Equation 3.4, 3.5, 3.6].

Step 12: Analyze Architectural Approaches:
Then each requirement was analyzed and the analysis team queried the architect that how he handled the requirements in the software architecture. The analysis is performed in Architecture Approach Analysis Metric. We are only showing one example from each possible category of the requirements. The first requirement REQTDW2 is analyzed. According to the [Table 4.7] this requirement is handled in the architecture and the score 1 is given to the architecture contribution factor (*ACF*).

The second example is given for requirement REQTDW6. According to the [Table 4.8] this requirement is not handled in the architecture by the architect and the score 0 is given to the architecture contribution factor (*ACF*).

The third example is selected for a new requirement NREQTDW1. According to the [Table 4.9] this requirement is not handled in the architecture by the architect and the score *0* is given to the architecture contribution factor (*ACF*).

The fourth example is provided for a new requirement NREQTDW2. According to [Table 4.10] the requirement is handled in the architecture and the score 1 is assigned to the architecture contribution factor (*ACF*). Here we can see that although the requirement was not provided earlier but the architect handled the requirement.

Architecture Approach Analysis		
Req ID	Requirement	Data from different sources should be transformed into a uniform data warehouse schema.
REQTDW2		
Requirement Criticality		0.653
Architecture Approach		ACF
Datawarehouse Schema		1
Remarks		Competitor, Region-One and Region Two data is extracted, transformed and loaded into Data warehouse schema.

Architecture Diagram

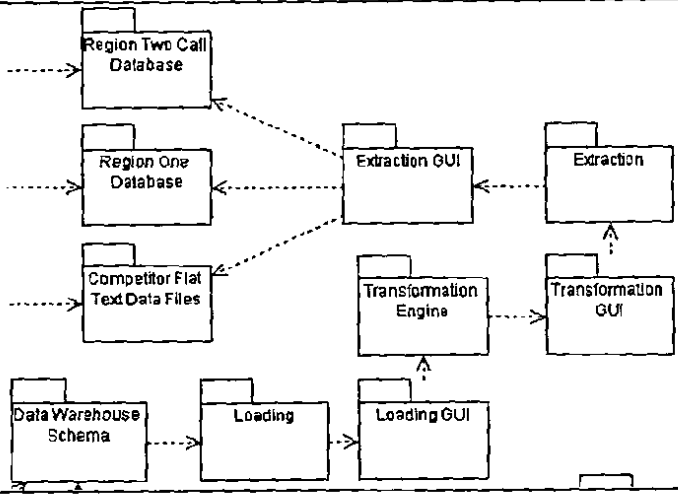


Table 4.7 Architecture Approach Analysis REQTDW2

Architecture Approach Analysis		
Req ID	Requirement	Loading logs should be maintained.
REQTDW6		
Requirement Criticality		0.735
Architecture Approach		ACF
Not Handeled		0
Remarks		No solution is provided by the architect. It should be a part of the Metadata repository but no component is there handling this requirement.

Architecture Diagram

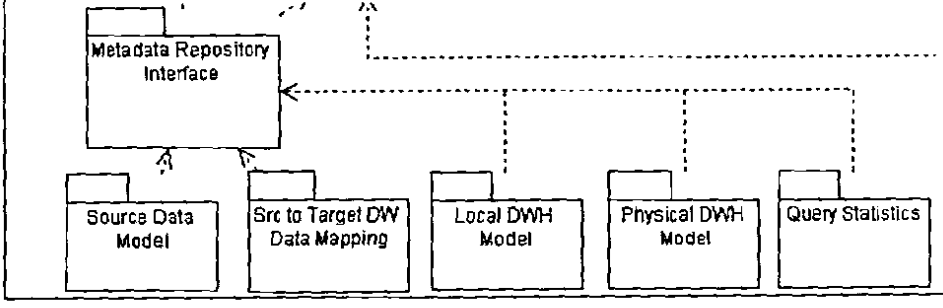


Table 4.8 Architecture Approach Analysis REQTDW6

Architecture Approach Analysis		
Req ID	Requirement	*Region and Sub-Region wise trend analysis reporting*
NREQTDW1		Should be there.
Requirement Criticality		0.776
Architecture Approach		ACF
Not Handled		0
Remarks		No solution is provided by the architect.
		It should be a part of the architecture but no component is there handling this requirement.
Architecture Diagram		

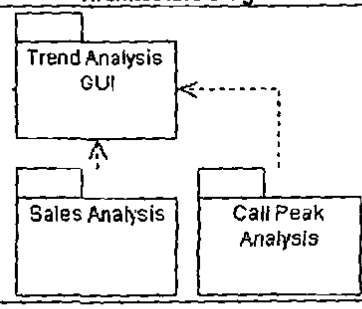


Table 4.9 Architecture Approach Analysis NREQTDW1

Architecture Approach Analysis		
Req ID	Requirement	Sales trend analysis report should be there.
NREQTDW2		
Requirement Criticality		0.833
Architecture Approach		ACF
Sales Analysis		1
Remarks		The architect has already included a sales analysis module.
Architecture Diagram		

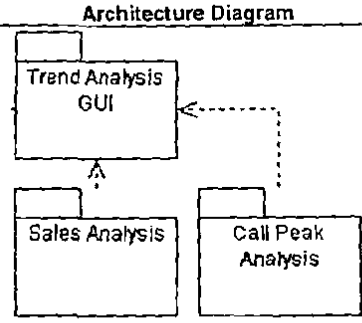


Table 4.10 Architecture Approach Analysis NREQTDW2

Step 13: Architecture Accuracy Factor and Architecture Error Factor:
In the last step for moving towards the final results calculations, the architecture approach contribution percentage (*AAC%*) is calculated for each requirement as per equation 3.7 [Table 4.11]. Then the Architecture Accuracy Factor (*AAF*) as per equation 3.8. The Architecture Error Factor (*AEF*) is also calculated here according to equation 3.9.

Architecture Accuracy Metric				
			TRC	5.867
Req ID	Architecture Approach	ACF	RC	AAC%
REQTDW1	Loading Component	1	0.475	8.091
REQTDW2	Data Warehouse Schema Component	1	0.653	11.14
REQTDW3	Logical DWH Model Component	1	0.306	5.216
REQTDW4	Physical DWH Model Component	1	0.307	5.239
REQTDW5	Query Statistics Component	1	0.709	12.09
REQTDW6	Not Handled	0	0.735	12.52
REQTDW7	Source Data Models Component	1	0.339	5.778
REQTDW8	Source to Target DWH Data Mapping	1	0.719	12.25
REQTDW9	Call-Peak Analysis Component	1	0.892	15.21
REQTDW10	Extraction GUI, Transformation GUI & Loading GUI Components	1	0.731	12.47
	AAF	87.48%	AEF	12.52%

Table 4.11 Architecture Accuracy Metric, AAF & AEF

Architecture Accuracy Factor (*AAF*) shows the correctness of the architecture in percentage and the Architecture Error Factor (*AEF*) shows the percentage of error present in the architecture, and this is the error of the architect.

$$E_a = AEF$$

To calculate the error propagated due to the stakeholder (normally the customer) we included the new requirements captured in step 7 in our analysis. First of all we calculated the New Total Requirement Contribution (nTRC) for each new requirement captured [Equation 3.11]. Then the Architecture Approach Contribution percentage (*nAAC%*) is calculated as per equation 3.12 and the results are recorded in Architecture Accuracy Metric with new Requirements [Table 4.12]. The New Architecture Accuracy Factor (*nAAF*) and the New Architecture Error Factor (*nAEF*) is also calculated over here.

Architecture Accuracy Metric with New Requirements				
nTRC	2.311		TRC+nTRC	8.178
New Req ID	Architecture Approach	nACF	nRC	nAAC%
REQTDW1	Loading Component	1	0.475	5.804
REQTDW2	Data Warehouse Schema Component	1	0.653	7.989
REQTDW3	Logical DWH Model Component	1	0.306	3.742
REQTDW4	Physical DWH Model Component	1	0.307	3.758
REQTDW5	Query Statistics Component	1	0.709	8.674
REQTDW6	Not Handled	0	0.735	8.984
REQTDW7	Source Data Models Component	1	0.339	4.145
REQTDW8	Source to Target DWH Data Mapping	1	0.719	8.788
REQTDW9	Call-Peak Analysis Component	1	0.892	10.91
REQTDW10	Extraction GUI, Transformation GUI & Loading GUI Components	1	0.731	8.943
NREQTDW1	Not Handled	0	0.776	9.493
NREQTDW2	Sales Analysis Component	1	0.833	10.19
NREQTDW3	Not Handled	0	0.702	8.58
	nAAF	72.94%	nAEF	27.06%

Table 4.12 Architecture Accuracy Metric with New Requirements

[3] Kazman, R. & Carriere, S.J. "Playing Detective: Reconstructing Software Architecture from Available Evidence." Automated Software Engineering, 6, 2 (April 1999): 107-138.

[4] Klein, M., Clements, P. and Kazman, R., "ATAM: Method for Architecture Evaluation", August 2000, TECHNICAL REPORT, CMU/SEI-2000-TR-004, ESC-TR-2000-004

[5] Kazman, R.; Abowd, G.; Bass, L.; & Webb, M. "SAAM: A Method for Analyzing the Properties of Software Architectures," 81-90. Proceedings of the 16th International Conference on Software Engineering. Sorrento, Italy, May 1994.

[6] Pressman, R., "Software Engineering A Practitioner's Approach", 5th Edition, Mc Graw-hill, International Edition, pp. 256-266.

[7] Asada, T., et al., "The Quantified Design Space," in Software Architecture (Shaw, M. and D. Garlan), Prentice Hall 1996, pp. 116-127.
[8] Thayer, R. H. and M. Dorfman, "Software Requirements Engineering", Second Edition, IEEE Computer Society Press, 1997.

[9] Sommerville, I. and P. Sawyer, "Requirements Engineering", Wiley, 1997

[10] Bass, L., Clements, P. and Kazman, R , What is Software Architecture?, in "Software Architecture in Practice,", Pearson Education 2003, pp. 125.

[11] Dennis, J. B., "Modularity" in Advanced Course in Software Engineering, (F.L.Bauer, ed.), Springer-Verlag, 1973, pp. 128-182.

[12] Wirth, N. "Program Development by Stepwise Refinement", CACM, Vol.14, no. 4, 1971, pp. 221-227.

[13] Dah. O. E. Dijkstra and C. Hoare, "Structured Programming", Academic Press, 1972.

[14] Bass, L., Klein, M., Moreno, G., "Applicability of General Scenarios to the Architecture Tradeoff Analysis Method", TECHNICAL REPORT, CMU/SEI-2001-TR014, ESE-TR-2001-014.

[15] Clements, P.; Bachmann, F.; Bass, L.; Garlan, D.; Ivers, J.; Little, R.; Nord, R.; & Stafford, J. "Documenting Software Architectures: Views and Beyond". Boston, MA: Addison-Wesley, 2003.

Appendix B

PUBLICATION # 2

Appendix B Publication # 2

Our second publication from this research is titled "Architecture Requirements Engineering Accuracy & Error – The Analysis Method (AREA-TEAM)-Islamabad Stock Exchange Case Study". This research paper has been published in 3rd IASTED International Conference on Software Engineering SE 2006, February 14-16, 2006, Innsbruck, Austria, book of proceedings ISBN: 0-88986-572-8 pages (317-332).

The copy of the research paper published is provided in this appendix.

ARCHITECTURE REQUIREMENTS ENGINEERING ACCURACY AND ERROR- THE ANALYSIS METHOD (AREA-TEAM): ISLAMABAD STOCK EXCHANGE CASE STUDY

Muhammad Amir Aman *amir@iiu.edu.pk*
Muhammad Sulayman *sulayman_67@yahoo.com*
Prof. Dr. Khalid Rashid *drkhalid@iiu.edu.pk*
Department of Computer Science
Faculty of Applied Sciences
International Islamic University, Islamabad

ABSTRACT

Since the software architecture is a key artifact of the software development life-cycle, so architectural analysis is also a key practice involved in the process. Software architectures are usually complex and developed on the quality attributes based on preliminary requirements. Requirements are specified during requirements engineering phase and normally are incomplete and ambiguous and they are propagated in the software architecture. Our devised method "architecture requirements engineering accuracy and error-the analysis (AREA-TEAM)" provides solid technical foundations for performing architectural analysis. It is based on the requirements and finding the architecture accuracy factor and architecture error factor. The major contribution of AREA-TEAM is quantification of the error made by the architect as well as the stakeholder, generally the customer. This research paper describes the application of the Architecture Requirements Engineering Error and Accuracy-The Analysis Method (AREA-TEAM) in the Data Warehouse solution of Islamabad Stock Exchange (DWH-ISE). The system was developed under research collaboration between Islamabad Stock Exchange and International Islamic University, Islamabad.

KEYWORDS

Software Architecture, Architecture Evaluation, Requirements Engineering, Error Factor

1 Introduction

Since software architecture is a major way to measure software quality, due to this we can conclude that good software architecture is essential to the quality of any software-intensive system. For any organization interested in development of a software system, the ability to evaluate software architectures before the anomalies are realized in finished systems can substantially reduce the risk that the delivered systems will not meet their requirements and quality goals.

During the past sometime, the Software Engineering Group at IIUI has developed the Architecture Requirements Engineering Error and Accuracy-The Analysis Method (AREA-TEAM) [1].

This research paper describes an AREA-TEAM evaluation of the software architecture for a Data Warehouse Solution for the Islamabad Stock Exchange (ISE). The system, called the Data Warehouse Solution for Islamabad Stock Exchange (DWH-ISE), is being developed under research collaboration between International Islamic University, Islamabad and Islamabad Stock Exchange.

Following this introduction, Section 2 provides context for the architecture and its evaluation, a description of ISE, and an overview of the DWH-ISE. Section 3 contains an overview of the AREA-TEAM. Section 4 describes how the AREA-TEAM was applied specifically to the DWH-ISE, and Section 5 presents some benefits and conclusions of the AREA-TEAM evaluation.

2 Context for the Architecture Evaluation

2.1 Software Architecture

The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them [2]. The software architecture for a system represents the earliest software design decisions. The right software architecture can pave the way for successful system development, while the wrong architecture results in a system that fails to meet critical requirements and incurs high maintenance costs [3]. According to latest research in the area of software architecture it is found that there are many relevant views of a software architecture. A *view* is a representation of some of the system's elements and the relationships associated with them [2]. Views help us to separate concerns and achieve

intellectual control over an abstract concept. Different views also speak to different stakeholders—those who have a vested interest in the architecture. Which ones are relevant depends on the stakeholders and the system properties that interest them. In addition there are many types of stakeholders associated with the software architecture, for example, the development organization, project manager, software architect, end user, software engineer, data entry operator, and customer organization. Each stakeholder is interested in different system properties and has different goals that are represented by different structural views of the system. These different properties and goals, and their corresponding architectural views are important to engineer, understand, and analyze. They all provide the basis for reasoning about the appropriateness and quality of the architecture [3]. Some experts prescribe using a fixed set of views. Rational's Unified Process (RUP), for example, relies on Kruchten's "4+1 view" approach to software architecture. A current and more healthy trend, however, is to recognize that architects should choose a set of views based on the needed engineering leverage that each view provides and the stakeholder interests that each one serves. This trend is exemplified by the recent American National Standards Institute/Institute of Electrical and Electronics Engineers (ANSI/IEEE) recommended practice for architectural documentation on software-intensive systems [4] and the "views and beyond" approach to architecture documentation from the SEI [5].

2.2 The ISE Organization

Islamabad Stock Exchange (Guarantee Limited) is the youngest stock exchange in Pakistan. It came into existence on October 1989. ISE started trading on 8th October, 1992. At that time there were very few listed companies and members with ISE. But with the passage of time ISE showed extensive improvements in terms of companies, members and business volumes. There are 101 corporate as well as individual members listed with ISE. There are 285 listed companies eligible to be traded by ISE.

2.3 The DWH-ISE

The system is designed and developed to meet the following objectives.

- ✓ To provide a full fledged and very less error prone online transaction processing system for a stock exchange. It includes real time transaction processing among different agents working under specific members of a stock exchange.
- ✓ To enhance the decision support for the stock exchange by developing a data warehouse for bulk storage and efficient retrieval of data. The operational database cannot be used for deep data drilling and decision making, so a data warehouse can be used to store huge volumes of data i.e. up to Tera bytes. It also includes the replication, loads and stores of data among distributed databases.
- ✓ To improve the overall business scenario of the stock exchange by retaining the customers. This will help in increasing the market capital and overall business activities. It will lead to gather maximum customers (investors) and to motivate the people to invest in the stock market.
- ✓ To develop software, that provides business intelligent data of a stock exchange on handheld Palm device. Synchronizing the Palm device with the data source for quick and efficient data retrieval. This also includes synchronization of Windows Operating System and Palm Operating System.

3 The AREA-TEAM

"Architecture Requirements Engineering Accuracy and Error-The Analysis Method (AREA-TEAM)" is a newly devised architecture analysis method [1]. This method permits evaluation of the software architecture in terms of its compliance with the requirement specifications. Our devised method AREA-TEAM provides solid technical foundations for performing architectural analysis. It is based on the requirements and finding the architecture accuracy factor and architecture error factor. The major contribution of AREA-TEAM is quantification of the error made by the architect as well as the stakeholder, generally the customer [1]. In the next section we'll discuss the application of AREA-TEAM on an industrial case study where the method is implemented step-wise.

4 Evaluation of Software Architecture of ISE-DWH

There are two phases of developed AREA-TEAM. The phase 1 is named as "Rehearsal". Phase 1 of the evaluation took place at the Islamabad Stock Exchange Building, Blue Area, Islamabad on April 18, 2005. Five "decision maker" stakeholders were present, which included two members of the International Islamic University (contractor organization) (architect and project manager), two members of the ISE IT Department and the Secretary stock exchange.

In this phase the AREA-TEAM analysis team prepares itself for the analysis session. The steps-wise solution for phase 1 is given below:

Step 1: Identify Stakeholders:
All the concerned stakeholders of the project were identified which represented the members stock exchange, clients, users, software engineers, IT manager and Secretary Stock Exchange etc.

Step 2: Prepare Core Requirements:

Software Project Manager and the Software Architect identified the core requirements which were used to prepare the architecture and these were the result of the requirements engineering phase.

Step 3: Architecture Briefing:
The Software Architect prepared a briefing that explained all the components and connectors as well as the hierarchy and granularity of the architecture.

Step 4: Architecture Approaches:
A list of architectural approaches was prepared that focused on understanding the architecture and will be analyzed further.

Step 5: Prepare Materials:
Copies of presentations, core-requirements and feed-back forms were produced for distribution to the stakeholders during the next phase. The schedule of the meeting was decided and the stakeholders were invited.

Now we enter the phase 2 of the AREA-TEAM. The phase 2 is called "Architecture Requirement Engineering Analysis". Phase 2 of the evaluation also took place at the Islamabad Stock Exchange Building, Blue Area, Islamabad on April 28, 2005. Ten stakeholders were present, which included Secretary Stock Exchange, 2 members of stock exchange, 2 clients, 1 user, 1 project manager and 1 software architect, 1 IT manager and 1 quality assurance inspector.

In this phase the AREA-TEAM analysis team prepares itself for the analysis session. The steps-wise solution for phase 1 is given below:

Step 6: Present AREA-TEAM:
A brief introduction to AREA-TEAM was given to explain the steps of the process to the stakeholders so that the maximum participation of the stakeholders was assured.

Step 7: Present Software Architecture and Core-Requirements:
The core-requirements were distributed among the stakeholders and the software architecture shown in fig. 4.1 was presented in a session of about one-hour. During this session some new requirements were also identified.

Step 8: Requirements Prioritization Factor:
Requirement Prioritization Factor (*PF*) was calculated using equation $[PF_i = V_i / Ts]$, *where i is from 1 to Ts*. The votes for each requirement were counted by show-of-hand method. The same process is applied on the new requirements captured during step 7.

Step 9: Requirement Significance Factor:
The requirement significance factor (S) was then collected with the help of the feed-back forms. The data from the feed-back forms is recorded in the requirements significance metric. Then Architecture Significance Factor (*ASF*) is calculated using equation

$$ASF_j = ((\sum_{i=1}^{n}\sum_{j=1}^{Ts} S_{ij})/Ts)/100.$$

The New Requirements are also processed in the same way and nASF is calculated for each new requirement.

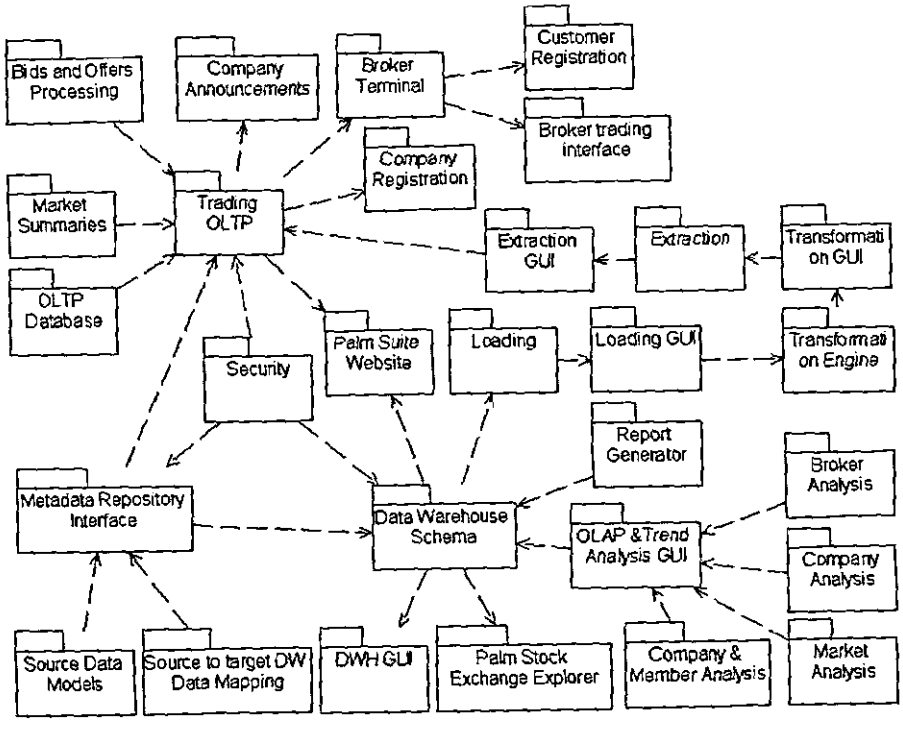


Fig. 4.1 System Software Architecture of the Data Warehouse Solution of Islamabad Stock Exchange

Architecture Approach Analysis		
Req ID	Requirement	Efficient trend analysis mechanism of companies and
REQDWISE4		brokers.
Requirement Criticality		0.788
Architecture Approach		ACF
Datawarehouse Schema		1
Remarks		Companies Analysis and Broker Analysis
		Components are present in the architecture.
Architecture Diagram		

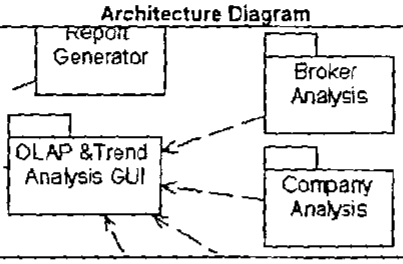


Table 4.3 Architecture Approach Analysis REQDWISE4

To calculate the error propagated due to the stakeholder (normally the customer) we included the new requirements captured in step 7 in our analysis. First of all we calculated the New Total Requirement Contribution (nTRC) for each new requirement captured with the help of equation $nTRC = \sum_{i=1}^{n} nRC_i$.

Then the Architecture Approach Contribution percentage (*nAAC%*) is calculated as per equation $nAAC_i = (nRC_i/(TRC + nTRC)) \times 100$, *where i is from 1 to n* and the results are recorded in Architecture Accuracy Metric with new Requirements. The New Architecture Accuracy Factor (*nAAF*) and the New Architecture Error Factor (*nAEF*) is also calculated over here using equations:

$$nAAF = \sum_{i=1}^{n} nAAC_i \text{, where } nACF_i = 1$$

$$nAEF = \sum_{i=1}^{n} nAAC_i \text{, where } nACF_i = 0$$

The New Architecture Error Factor reflects the ratio by which the architecture has not addressed the requirements including the new requirements. The stakeholder error factor (*SEF*) is calculated as follows:

$$SEF = nAEF - AEF$$

$$SEF = 29.87 - 14.56 = 15.31\%$$

$$E_s = SEF$$

The *SEF* was 15.31%, which means that the error factor introduced in the architecture due to the stakeholder is 15.31%. The adverse effect introduced into architecture accuracy due to the stakeholder is calculated as follows:

$$SAF = nAAF - AAF$$

$$SAF = 70.13 - 85.44 = -15.31\%$$

The resultant *SAF* is -15.31%, the negative sign shows it is an adverse effect. Also note that both the ratios *SAF* and *SEF* are equal, which shows that both the positive and negative sides have the equal effect by the stakeholder.

5 Conclusions

Adapting the architecture in earlier steps of development is recommended but its evaluation is more important. An architecture analysis method, any architecture analysis method, is garbage-in-garbage out process. The AREA-TEAM is no different. It crucially relies on the active and willing participation of the stakeholders (particularly the architecture team); we advance preparation by the key stakeholders; an understanding of architectural design issues and analytic models; and a clearly articulated set of core requirements and a set of business goals from which they are derived. Our purpose in creating a method (rather than, say, just putting some intelligent and experienced people together in a room and having them chat about the architecture or inspect it in an arbitrary way) is to increase the effectiveness and repeatability of the analysis.

5.1 Benefits

Every AREA-TEAM exercise is followed by a survey of stakeholders in attendance. Here is what those responding had to say verbatim:

- Error in the architecture are identified
- The source of error is also identified. For example, in this case study the architect error factor is 14.56%

that measure the performance of the architect and the stakeholders' error factor is 15.31% that shows the maturity of the stakeholder.
- Requirements are improved and validated
- *New requirements are identified*
- Stakeholder knowledge about the system increases
- Software Architecture is revised
- The new requirements are accommodated therefore the method supports agile processes
- Since this review was conducted during the high-level design phase, therefore, the errors are identified at an early stage. So the valuable resources will be saved. "The earliest an error is identified, cheaper it is to fix"
- Important quality requirements will be identified and analyzed
- The method provided the stakeholders with a chance to give a critical look at the system. It validated some architectural decisions and raised questions about others.
- "As an IT manager, it helped me to understand the system."
- The method brought issues and concerns to our attention.
- It will be more useful to use the method earlier. The later use may lead to diminish the effectiveness of the method.

5.2 Summary

Overall, this evaluation succeeded in
- raising awareness of the importance of stakeholders in the architecture process
- establishing a community of vested stakeholders and opening channels of communication among them
- identifying a number of risk themes that can be made the subject of intense mitigation efforts that, even though the system is in development, can be effective in heading off disaster
- raising a number of issues with respect to previously unplanned capabilities for which some stakeholders expressed an acute need
- elevating the role of software architecture in system acquisition

References

[1] Muhammad Amir Aman, Muhammad Sulayman, Khalid Rashid, ***"Architecture Requirements Engineering Accuracy & Error – The Analysis Method (AREA-TEAM)"***. Published in the first International Conference on Quality of Software Architectures in Dot-Net Object Days, Erfurt Germany, Sep 19-22, 2005, book of proceedings ISBN: 3-9808628-4-4, Pages (461-472)

[2] Bass, L.; Clements, P.; & Kazman, R. ***"Software Architecture in Practice"***, 2nd edition. Boston, MA: Addison-Wesley, 2003.

[3] B., Mario; Clements, P.; Lattanze, A, North, L. & W., Wood., ***Using the Architecture Tradeoff Analysis Method (ATAM) to Evaluate the Software Architecture for a Product Line of Avionics Systems: A Case Study"***, Technical Note, CMU/SEI-2003-TN-012

[4] IEEE Computer Society, Software Engineering Standards Committee, Institute of Electrical and Electronics Engineers, IEEE-SA Standards Board, and IEEE Xplore. ***IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*** New York, NY: Institute of Electrical and Electronic Engineers, 2000.

[5] Clements, P.; Bachmann, F.; Bass, L.; Garlan, D.; Ivers, J.; Little, R.; Nord, R.; & Stafford, J. ***"Documenting Software Architectures: Views and Beyond"***. Boston, MA: Addison-Wesley, 2003.

Appendix C

ISE DATA WAREHOUSE PROJECT

Appendix C Data Warehouse Project

The project that we have selected to be used as our case study is the Data Warehouse project of Islamabad Stock Exchange. Following is the detailed discussion about the modules of this project.

Modules of the Data Warehouse Project

Following are the modules of the selected system.

- Bids and Offers processing
- Online Transaction Processing (OLTP)
- Daily Market Reports
- Routine database administrative tasks
- Data warehouse
- Extraction, Transformation and Loading (ETL)
- Data warehouse Administration
- Analytical reports and graphs
- Connectivity/Replication among heterogeneous databases
- Synchronization of aggregated data on Palm Database (PDB)
- Complete application to query Palm Database (PDB)
- Conduit to Synchronize Windows OS and Palm OS
- Forms and reports on handheld Palm Device
- Multipurpose Web Site
- Authorized Logging

Bids and Offers Processing

The main functionality of ISE revolves around bids and offers. If an agent wants to buy the stocks of a specific company on behalf of a member then he puts a bid through the specialized multi user environment. Similarly if an agent wants to sell the stocks of a specific company then he can put the offer on behalf of the member. Once a bid or offer is put by the agent then it can be seen by the other

interested agents. Agents can also view their own put up bids and offers. They are also provided with the facility of viewing the transacted bids and offers. So the hierarchy flows from member to agent, but at this point one should not forget that member always performs the business on the orders put up by his/her clients. As per rules clients cannot directly perform trading without the member's interaction.

Online Transaction Processing System (OLTP)

The developed system provides the facility to perform real time online trading mechanism. As the bids and offers can be viewed as soon as they are entered into the system, the agent can perform the transaction on the most suitable rate with only a couple of clicks. With such a transactional processing system a lot of wasted time can and efforts can be saved. It helps to improve overall trading scenario and can increase the business volume manifolds. All the transactions are efficiently stored in the highly normalized and sophisticated latest database.

Real Time Market Status

As soon as the agent logs into the system, the system shows the latest market status providing him with the latest network index and other useful information. Similarly while putting bids and offers for a specific company the agent is provided with the last transacted rates and volumes of that very company.

Daily Market Reports

Daily market reports can be viewed by the administrator of the OLTP and can also to be provided to any member on request. Daily markets reports provide extended report facility for the daily transactions done. They include the reports showing market summary, companies work files, major gainer and loser companies and volume leaders. It also includes the facility to view previous market summaries and other historic information.

Routine Database Administrative Tasks

There are several important routine database tasks that are to be performed by the database administrator for the purpose of fine tuning and performance. They include the routine updates and routine flushes. Routine updates include the generation of work files and market summary. Routine flushes include the flushing of current day's transactions which are sent to the data warehouse. Administrator will perform these tasks after the market has been closed and the necessary data has been loaded into the data warehouse.

Data Warehouse

The essence of the current system is the data warehouse. Data warehouse is used to store the bulk amounts of categorical historical data of the ISE. It is used perform extended analysis and keep track of all the transactional activity ever performed. Data warehouse leads to a decision support system which will enhance the functionality and will help the investor to minimize the risks. Similarly it is very essential for the members as they can keep track of ever performed buying and selling. This salient feature has been added to overcome the data loss problems of the current system. Data warehouse is fully indexed using Oracle's bitmap indexing that is the latest and best indexing provided so far. Similarly for the efficient retrieval of data tables are fully partitioned.

Extraction, Transformation and Loading (ETL)

Any data warehouse revolves around extraction, transformation and loading. Data is extracted from the source legacy system and is then transformed. Transformation means cleaning the data and making sure that it is free of any bugs. After transformation data is loaded into the data warehouse. This activity will be performed by the administrator of the data warehouse after the market closes. This activity is performed daily to make data warehouse more functional and updated. It also lessens the burden on the operational database.

latest data. Any user who would like to get the updated information will download the application from the website and then will send it to the Palm Pilot with the help of Hot Sync operation. The web site will be updated regularly by the administrator and it will also contain few other salient features of market business.

Authorized Logging

Only the authorized persons will be allowed to use the system. Only agents will be able to correspond with the trading system and administrators will perform rest of the administrative tasks. Special login accounts are created for this purpose. The application on Palm will be provided only to the registered users of the system.