*I.J. Information Technology and Computer Science,* **2012, 12, 31-39**
Published Online November 2012 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2012.12.03

# Improvement in Quality of Software Architecture via Enhanced-Pattern Driven Architecture (EPDA)

**Muhammad Fahad Khan, Kanwal Yousaf, Anam Mustaqeem, Muazaam Maqsood**

University of Engineering and Technology, Taxila, Pakistan
E-mail: fahad.khan@uettaxila.edu.pk, kanwal_400@yahoo.com, anam_846@yahoo.com, muazzammaqsood@yahoo.com

***Abstract*—** No doubt software plays an important role in improvement of our lives. Great demand of software makes software architecture more complex. Flaws in any software have direct impact on diverse fields of life (such as business, science, engineering etc). The main reason of any software failure is due to poor software architecture or quality attributes. This paper focuses on factors that affect the quality of software architectures and highlighted the major reason of the defects through questionnaire and survey. In the light of this survey a technique is proposed to improve the quality of any software architecture. The proposed architecture is Enhanced-Pattern driven architecture (EPDA). This architecture focuses on the improvement of design phase in any architecture. This will also help in resolving lots of problems which arise due to usage of different traditional architectural styles.

***Index Terms*—** Architecture, Architecture Pattern, Software Architecture, Quality Attributes

## I. Introduction

In the past decade, the software plays an important role in improving our life style. As the clear relation between different fields of life and software no one can deny the importance of software. For the effectiveness of any software depends upon its architecture. Great demand of any software also increases the complexity in the software architecture. The design of any software is very crucial. So the architecture of software should focus on different diverse factors to resolve them. To understand the architecture of any software, firstly develop an understanding about simple architecture. Architecture is basically related to its outlook, services and firmness between its different components. The software architecture of any system is structure/ structures consist of different modules, relation between these modules and its properties. Each module can be helpful in accomplishing set of quality attributes [1].

Major fault in any software is due to the early selection of architecture pattern. The impact of these selections on quality attributes is not fully understood, until it becomes difficult to change the architecture style. Any software architecture may compose of set of patterns and may have multiple quality attributes. Thus the interaction among quality attributes and patterns introduces the complexity in the system. As mentioned in Fig. 1, any software architecture can follow one or more patterns (depending upon its nature) and each pattern have a negative or positive impact on certain quality attributes.

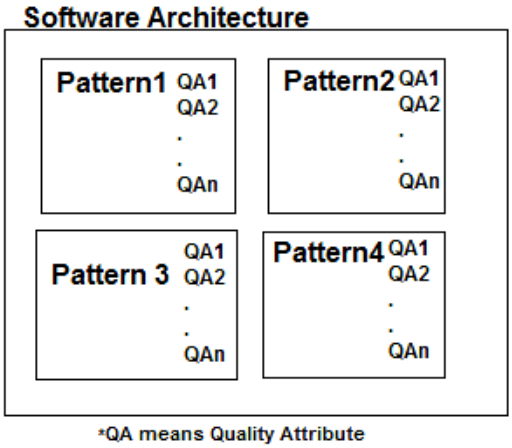


Fig. 1: General Software Architecture

Poor interaction among patterns lead to the failure in achieving its quality attribute requirements. The failure consequences can be very severe. The quality attributes are related to the attributes that system must fulfill. Some major quality attributes are mentioned below:

1) Modifiability (Maintainability, Configurability)
2) Usability (User Error Avoidance and Handling)
3) Portability
4) Performance (storage and execution efficiency )
5) Interoperability
6) Security (Authentication, Integrity)
7) Dependability (Availability, Reliability) etc

While some software architecture related patterns are:

 

1) Model View Controller
2) Layered Pattern
3) Shared Repository
4) Pipes and Filters
5) Client-Server
6) Broken
7) Black-Board etc

The major purpose of any quality oriented software architecture is to address three basic problems shown below:

1) Good Quality Software architecture design
2) Defining, Implementing and Evaluating architecture's quality
3) Management of Architecture's Quality

So to improve the quality of any software architecture, it's better to minimize the contradictions between quality attributes caused due to different patterns. A survey is conducted for this purpose in order to analyze the problems in already existing architecture and by using this knowledge a new architecture is proposed named as Enhanced-Pattern Driven Architecture (EPDA).

The rest of this paper is organized as follows: Section 2 provides a brief review of different quality attributes and different architecture patterns. Section 3 explains about the survey which we had carried out from the people with diverse fields. Section 4 presents the proposed methodology and. Conclusion and future work are given in the final section.

## II. Literature Survey

Lundberg, Bosch, Häggander and Bengtsson [1] explain that quality attributes of larger systems depends upon its architecture. They focused on performance and modifiability by comparing five different industrial applications (i.e. billing gateway, fraud control, generic database, haemo-dialysis and measurement system). They proposed eight design guidelines defined to gain advantage over quality attributes of any architecture.

Figueiredo, Reis, and Rodrigues [2] proposed a framework that allows virtual community to search for software architecture's information. Their software architecture is based on the Toeska Architecture Ontology. This architecture provides a light-weight mechanism that can easily be adopted by the virtual community.

Marten, Kozioleky, Beckerz and Reussner [3] provided an approach to improve the architecture of any software automatically. They concerned with performance, reliability and cost related quality attributes. Based on the initial model of a system, new candidates are generated automatically and evaluated for different quality criteria. They focused on the trade-off decisions between multiple qualities criteria.

Silva, Terra, and Valente [4] proposed improvement of architectural constraints based on maintainability and evolvability. They have three-staged goals to achieve quality oriented architecture. In first stage, analyze the importance of architecture in any system. In next stage, take control of the different of architectural constraints for removing the negative impact of quality attributes in any software system. In last stage, use simple language (i.e. Dependency Constraint Language (DCL)) to express these architectural constraints that have the clear impact on maintainability and evolvability of real-world applications.

F. Bachmann, L. Bass, M. Klein and C. Shelton [5] proposed that all the quality attributes are treated in a same way. Normal practice is to separate and hide the most identifiable large modules of responsibility. They used a reasonable framework for modifiability for supporting separation and encapsulation of these modules. These are explicitly included in different quality attributes.

Steffen Thiel [6] explains that the quality attributes play a major role in any architecture of application. He proposed few techniques related to QUDRAD (Quality Driven Architecture Development). He explains about the scenario of quality attributes related to the source of stimulus (actor) to request for accessing the system's artifact. Then the response of the system is being measured in order to check the quality of software system.

## III. Research Methodology

A research methodology is carried out in order to understand the importance of quality driven architecture and the shortcomings with the traditional architectural styles. This could be done through the proper approach adopted to conduct a survey. A questionnaire-based approach is used to perform this survey. The overall survey process includes 6 basic stages as shown in Fig. 2 [7] [8].

Data is gathered from different sources in order to make an effective questionnaire. These questionnaires broadly cover the major portion of software architecture and quality attributes. The survey team include group of people such as from educational background, professionals and technical people. The result is collected and mentioned in the form of pie-chart to represent the response of different groups of people. The survey includes various questions as shown below along with responses.

 

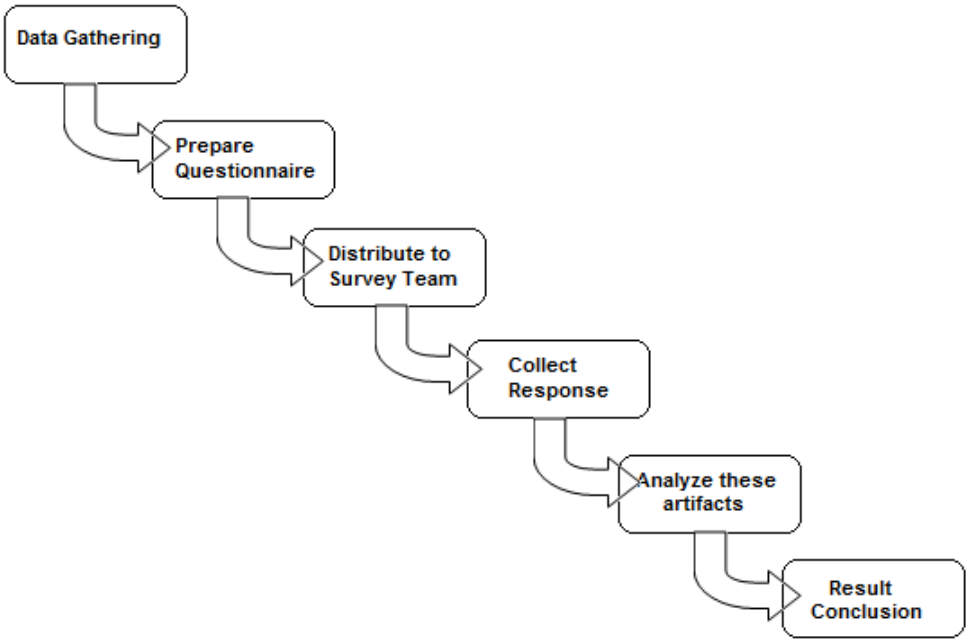


Fig. 2: 6-Stages Research Methodology

### 3.1 Observations and Analysis

1) Are the traditional architectures help in solving the emerging problems of any application?

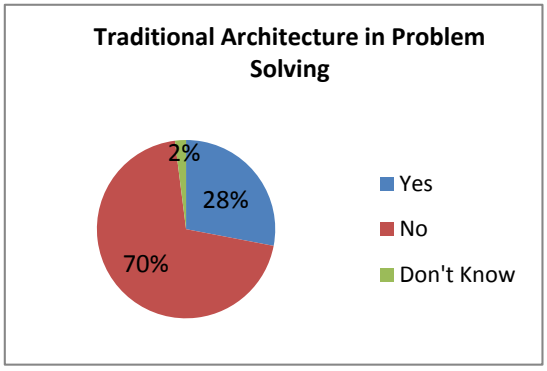


Fig. 3: Traditional Architecture's Importance in Problem Solving

2) Can architecture of any system/ application follows multiple architecture patterns at a same time?

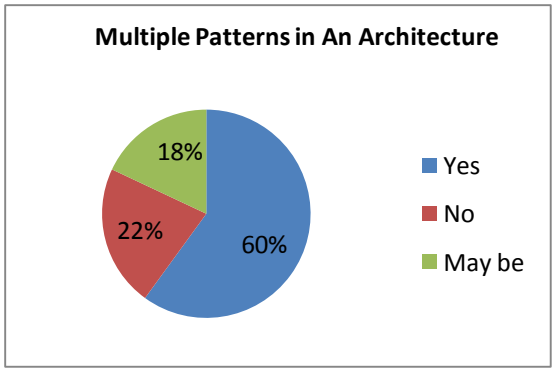


Fig. 4 Multiple Architecture Patterns in any Application

3) Is it beneficial to use different patterns in a same application's architecture?

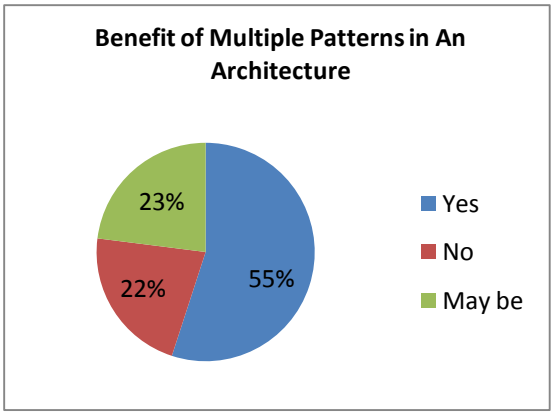


Fig. 5: Benefit of Using Multiple Patterns in Architecture

4) Do these patterns should follow the rule of "High-Cohesion and low-Coupling"?

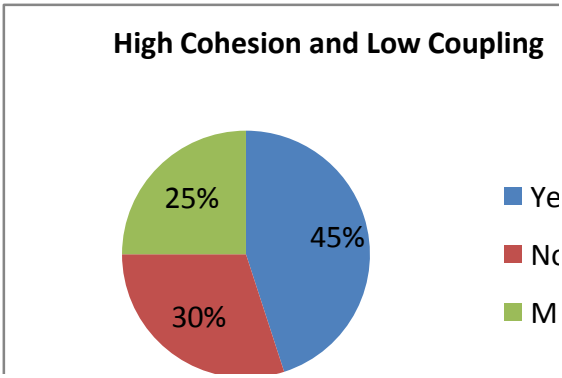


Fig. 6: High Cohesion and Low Coupling between Patterns

5) What are the levels of patterns use in any software architecture?

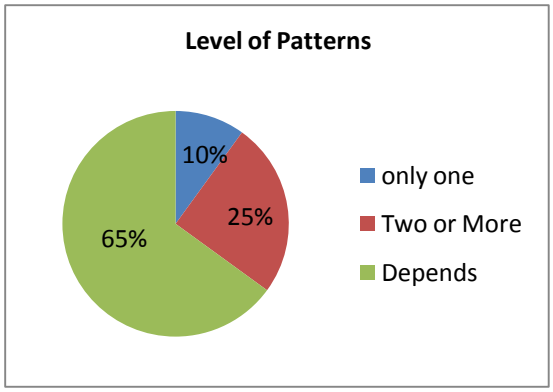


Fig. 7: Levels of Patterns in any Application

6) Do the quality attributes of any software architecture have an impact by using different patterns?

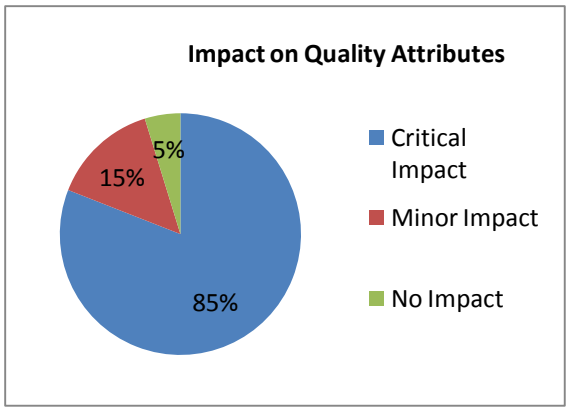


Fig. 8: Impact of Quality Attributes by Using Multiple Patterns

7) What are the most common patterns used in any software architecture?

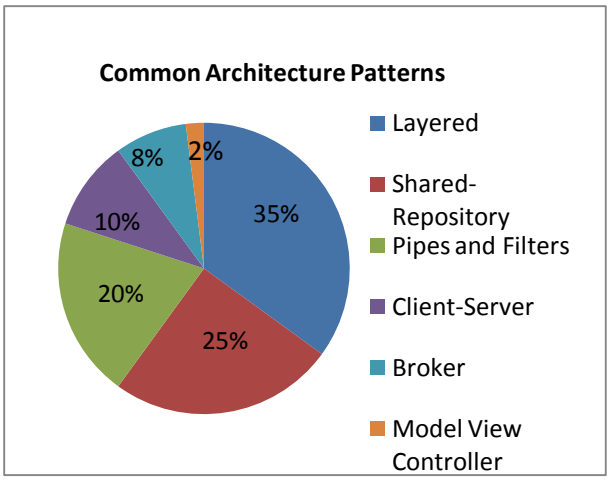


Fig. 9: Common Architecture Patterns in Different Applications

8) For different application domains (such as web-applications, embedded systems, games, and scientific applications) can use same pattern?

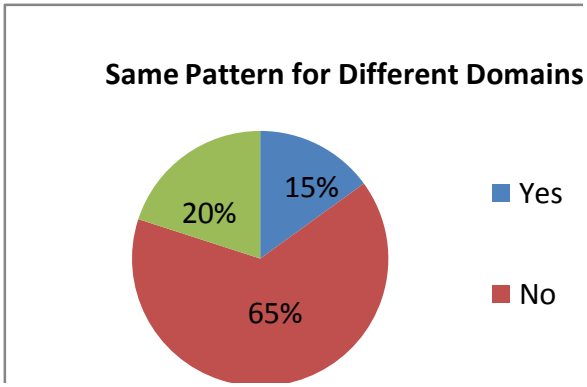


Fig. 10: Same Patterns in Different Applications

9) Is it possible that single attribute can be handled by more than one architectural pattern?

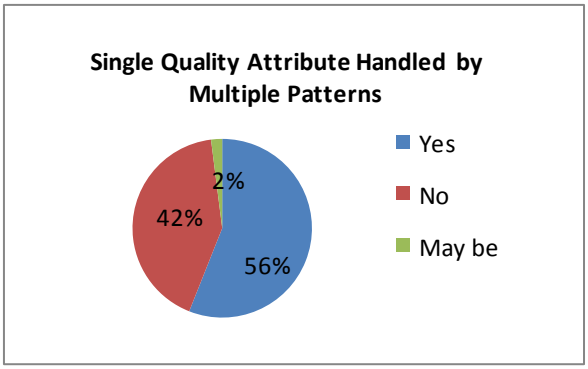


Fig. 11: Single Quality Attribute Handled by Multiple Patterns

10) Do you prefer to use prototyping for the creation of any software architecture?

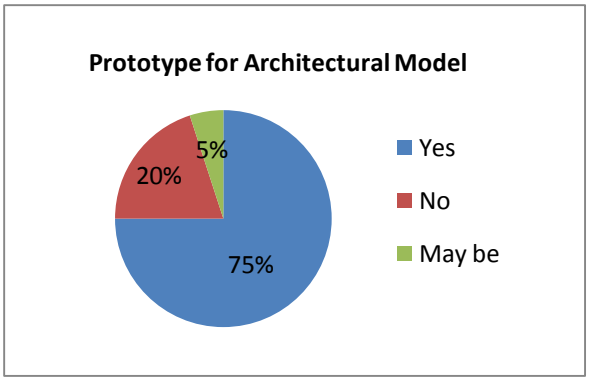


Fig. 12: Prototypes for any Architectural Model

11) Is it valid that "by checking the impact of any pattern in overall architecture of system helps in identifying the quality-oriented architecture"?

 

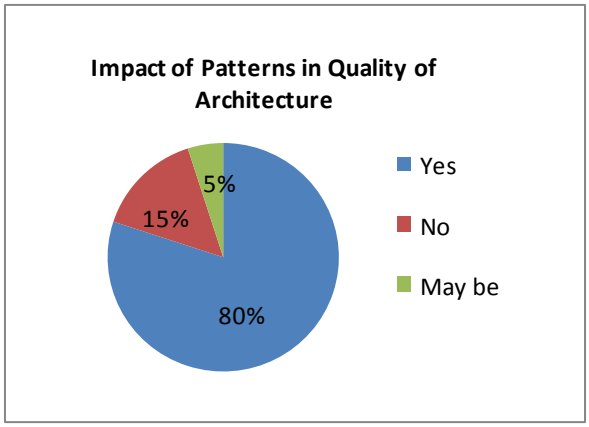


Fig. 13: Impact of Patterns in Quality of Architecture

### 3.2 Conclusion of Survey

We concluded from the survey that traditional architectures are no more helpful in solving problems of daily routine life. So there is a need of architecture to overcome this issue. Majority of the people said that multiple patterns can be used in any system architecture depending upon its nature and complexity. These patterns are very helpful in achieving different quality attributes.

The rule "High Cohesion and Low Coupling" should also follow to the patterns. But in some cases patterns can become dependent on others (e.g. in online airline reservation system, web application having layered pattern depends upon the database of repository pattern). Depending upon the architecture different levels of patterns can be used. There is a diverse feedback regarding the usage of different patterns in any architecture. Prototyping can be helpful in achieving the right architecture while keeping in mind that same architecture pattern cannot be used in different application domains. But single quality attribute can be handled by multiple patterns (e.g. security is handled by layers, MVC).

## IV. Enhanced-Pattern Driven Architecture (EPDA)

Different architectures are used by multiple applications. Currently existing architectures have few short comings. Because the patterns used by these architectures don't have large caliber to run on diverse applications. Pattern-driven architecture (PDA) was used to overcome the problems of traditional architecture [9]. But this architecture has few limitations. The key limitation of PDA is the amount of source information of patterns that the architect has. The lesser knowledge an architect has about patterns, the less PDA can be used effectively. It has difficulty in scalability, which can be used for the evaluation of larger system. So for this purpose an enhanced-pattern driven architecture (EPDA) is proposed in order to avoid such situations. The main architecture of EPDA is shown in Fig. 14.

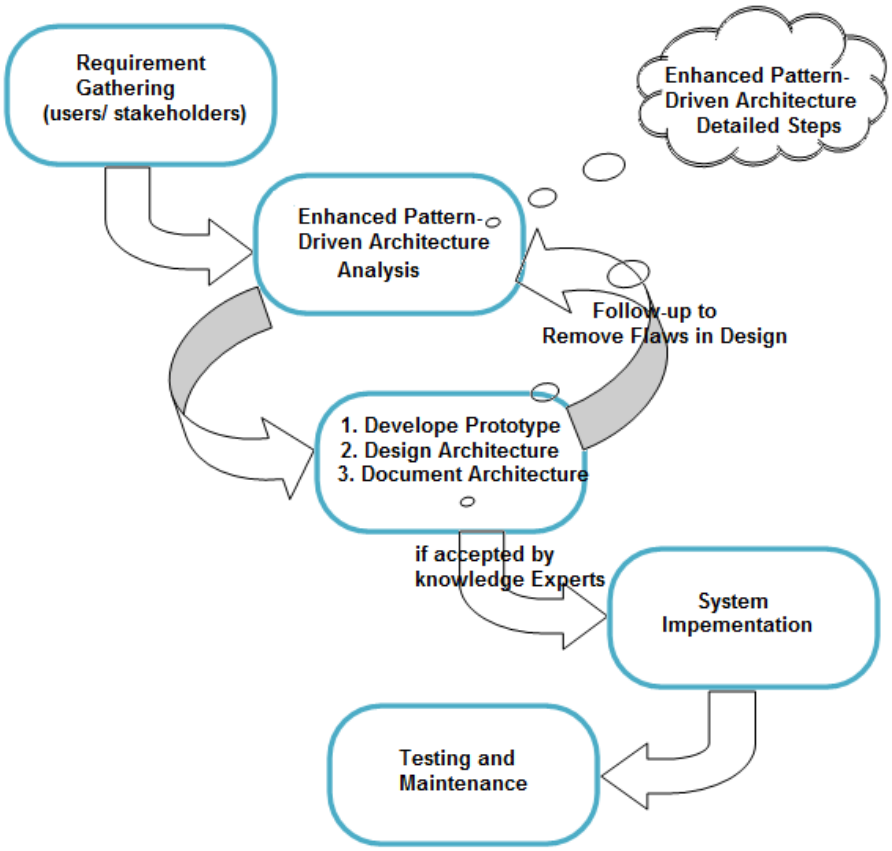


Fig. 14: Model of Enhanced Pattern Driven Architecture

 

This architecture is based on the concept of SDLC (Software Development Life Cycle). But the analysis and design phases are different and involve nested procedures. As any architecture may compose of one or more patterns so the EPDA resolves the problem of scalability by carrying not only single pattern oriented system as well as multiple-pattern based applications. EPDA firstly check the pattern/patterns used for any application domain. Then identify the level of patterns. If a single pattern (pattern level=1) is followed by application then single pattern driven model is used. If more than one pattern (pattern level>=2) is followed by application then multiple patterns driven model is used as mentioned in Fig. 15.

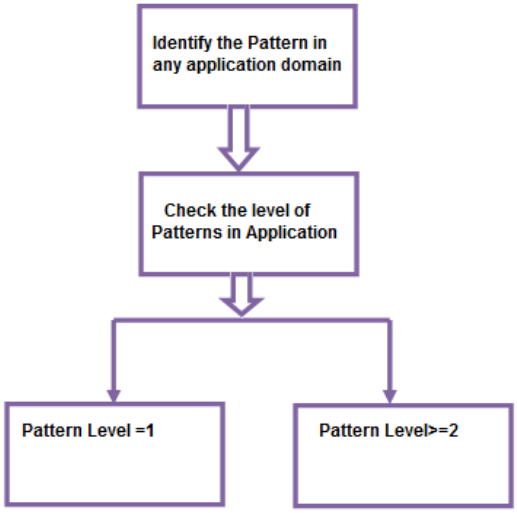


Fig. 15: Pattern Identification in EPDA analysis phase

### 4.1 Single-Pattern Driven Architecture

The single-pattern oriented architecture follows 6 simple steps as mention in Fig. 16:

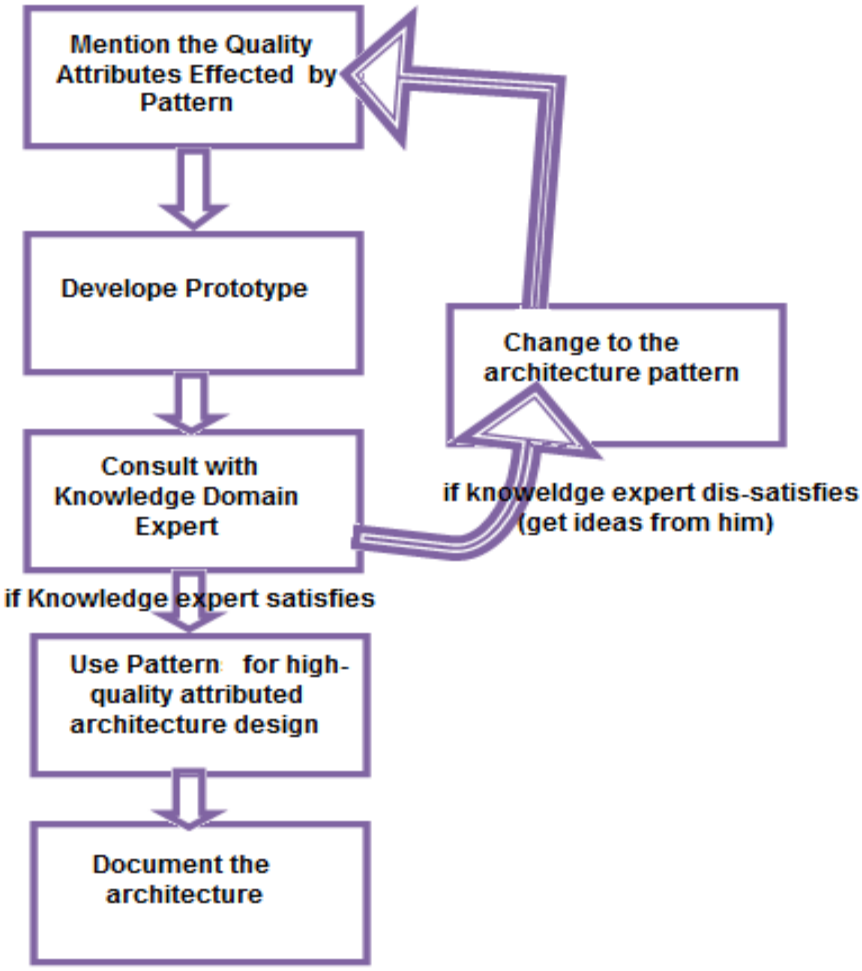


Fig. 16 Single Pattern-Driven Architecture

 

1) Mention or highlight all the quality attributed that effects pattern. These quality attributes may have positive or negative impact on the pattern. Make the selection of these quality attributes that have more positive impact.
2) An architect develops prototype according to his/her own knowledge.
3) Then this prototype is verified by the domain knowledge expert.
4) If knowledge expert is not satisfied, then he/she should mention the reasons of dissatisfaction. Also get help from an expert in development of quality attributed architecture.
5) So this pattern can be used as a high-quality attributed architecture design
6) Document the architecture for later purpose usage.

### 4.2 Multiple-Patterns Driven Architecture

The multiple-pattern oriented architecture follows 9 steps as mention in fig 17:

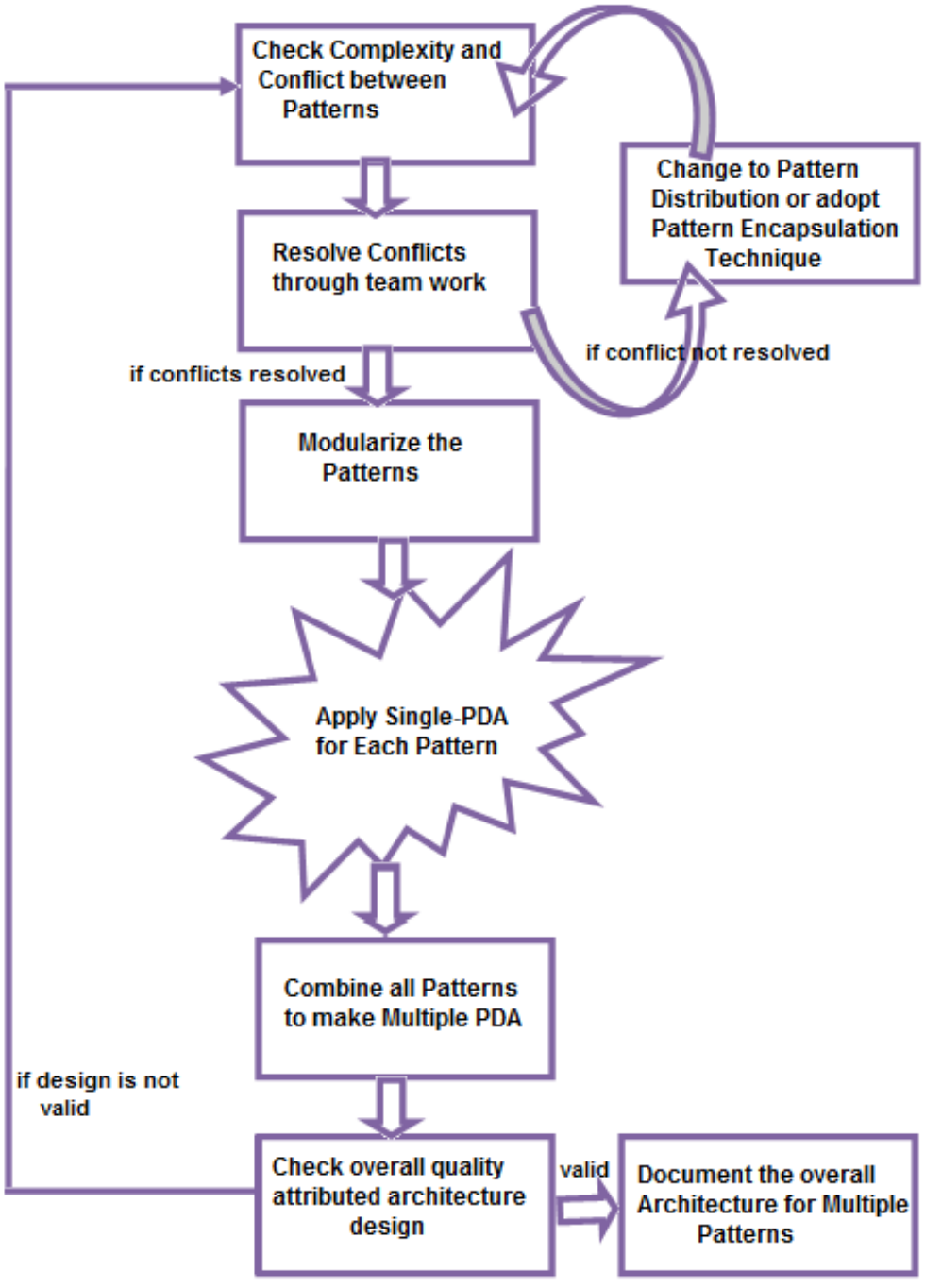


Fig. 17: Multiple-Patterns Driven Architecture Model

The steps are:

1) As multiple patterns are involved in architecture. So check for the conflicts and complexity constraint between multiple patterns.
2) Resolve these conflicts through advance knowledge on patterns and software architecture.
3) If the conflicts are still unresolved then bring changes to patterns i.e. encapsulates some information from one pattern to other
4) Modularize the patterns
5) Then follow same 6 steps of single-PDA for each individual pattern
6) Combine all the patterns by using simplest rule as a baseline. The rule is "High-Cohesion and Low Coupling". Each pattern should have one distinct purpose and it should be independent of other patterns. Low coupling will help in easy modularization and combination.
7) Check overall architecture's quality and rate it. This rating is done by knowledge experts and other technical people.
8) If the architecture is above the certain range in quality attribute then document the architecture for combine patterns
9) Otherwise start from the scratch.

These architectures resolve the conflicts caused in Pattern-Driven Architecture. The major conflicts were lack of knowledge by architects and scalability. In this proposed methodology, architects can get the knowledge from the domain experts by consulting them. Scalability is resolved by using the architecture for multiple patterns model.

## V. Conclusion

Traditional architectures become less effective with the passage of time. The alternative was proposed and based on pattern-driven architecture. As architecture may comprise of more than one pattern but due to few short-comings in pattern-driven architecture (PDA) such as scalability and knowledge oriented issues, an enhanced PDA is proposed. The proposed architecture basically divides into two basic parts single PDA and Multiple PDA. These architectures are related to developing a prototype of architecture and then consult with knowledge experts to determine knowledge related issues and make architecture more effective. This proposed architecture also resolves the issue of scalability by providing a framework for multiple pattern related architecture.

## References

[1] Lars Lundberg, Jan Bosch, Daniel Häggander and Per-Olof Bengtsson "Quality Attributes in Software Architecture Design".

[2] Adriana Maria Figueiredo, Julio C. dos Reis, Marcos A. Rodrigues," Improving Access to Software Architecture Knowledge An Ontology-based Search Approach", International Journal Multimedia and Image Processing (IJMIP), Volume 2, Issues 1/2, March/June 2012

[3] Anne Marten, Heiko Koziolek, Steffen Becker, Ralf Reussner, "Automatically Improve Software Architecture Models for Performance, Reliability, and Cost Using Evolutionary Algorithms", WOSP/SIPEW 2010, San Francisco Bay Area, USA Copyright 200X ACM

[4] Leonardo Humberto Guimaraes Silva, Ricardo Terra, Marco Tulio Valente, "A Case Study on Improving Maintainability and Evolvability using Architectural Constraints"

[5] F. Bachmann, L. Bass, M. Klein and C. Shelton, "Designing software architectures to achieve quality attribute requirements", IEE Proc.-Software, Vol. 152, No. 4, August 2005

[6] Steffen Thiel, "A Framework to Improve the Architecture Quality of Software-Intensive Systems", 28th October 2005.

[7] Syeda Uzma Gardazi, Haroon Khan, Syeda Faiza Gardazi, and Arshad Ali Shahid, "Motivation in Software Architecture and Software Project Management", 2009 International Conference on Emerging Technologies

[8] Liliana Dobrica, "Exploring Approaches of Integration Software Architecture Modeling With Quality Analysis Models", 2011 Ninth Working IEEE/IFIP Conference on Software Architecture

[9] Neil Harrison, "Improving Quality Attributes of Software Systems Through Software Architecture Patterns", Copyright © 2011, Neil B. Harrison, ISBN: 978-90-367-4893-3

**Engr. Muhammad Fahad Khan:** is a PhD Scholar in the Department of Computer Engineering at University of Engineering and Technology Taxila, Pakistan. He has received his Master degree in Computer engineering from university of Engineering and Technology Taxila, Pakistan in February, 2010. He graduated from University of Engineering and Technology Taxila in Software Engineering in September 2007. His areas of interest are Video Summarization, Computer vision, Software Designing, Design Patterns, Software Requirements Analysis and Mobile application development.

**Kanwal Yousaf:** MSc Scholar in Department of Software Engineering at University of Engineering and Technology, Taxila. She completed her Bachelor's degree in Software Engineering from UET, Taxila in 2010. She is currently working on Web 2.0 based E-learning by using social media. Her area of interest is Software Quality Assurance, Internet application development, Digital Image Processing and Wireless networks.

**Anam Mustaqeem**, is MSc Scholar in Department of Software Engineering at University of Engineering and Technology Taxila. She completed her Bachelor's degree in Software Engineering from University of Engineering and Technology Taxila in 2010. She has done extraordinary work in her four years degree of engineering. She is currently working on multimedia transmission over vehicular adhoc networks (VANETs). Her areas of interest are digital image processing, Medical Imaging, Software Quality Assurance, Wireless Networks and Adhoc Networks. She is trying hard to bring innovation in above mentioned fields.

**Engr. Muazzam Maqsood** is MS Scholar in Department of Software Engineering at University of Engineering & Technology Taxila. He holds a Bachelor's degree in Software Engineering from University of Engineering & Technology Taxila and has performed exceptional on his 4 year degree program. His core areas of interest are software design and architecture, digital image processing, Software quality assurance, Software Project management and Video Summarization. Muazzam has been striving to bring innovations in the said fields through his research.

**How to cite this paper:** Muhammad Fahad Khan, Kanwal Yousaf, Anam Mustaqeem, Muazaam Maqsood,"Improvement in Quality of Software Architecture via Enhanced-Pattern Driven Architecture (EPDA)", International Journal of Information Technology and Computer Science(IJITCS), vol.4, no.12, pp.31-39, 2012. DOI: 10.5815/ijitcs.2012.12.03

 