Indian Journal of Science and Technology, *Vol 9(30), DOI: 10.17485/ijst/2016/v9i30/96653, August 2016*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Novel Creative Innovative Patterns for Architecture Analysis (CIPA)

S. Angeline Julia[1]* and Paul Rodrigues[2]

[1]Computer Science Department, SRM University, SRM Nagar, Potheri, Kattankulathur, Kancheepuram District, Near Potheri Railway Station, Chennai - 603203, Tamil Nadu, India; sa_julie_2k3@yahoo.co.in
[2]Computer Science Department, King Khalid University, Abha, Saudi Arabia; drpaulprof@gmail.com

Abstract

Objectives: To improve the quality attributes of software architecture using new patterns. **Methods/Statistical Analysis:** Software architecture analysis methods are developed to reduce risk and to improve software quality. Patterns which have impact on the quality of software systems are used for the development of software. In this paper, novel Creative Innovative patterns are used. **Findings:** When Creative Innovative patterns are applied in the case study of hospital management system, the quality attributes like bug fixing cost, scalability, availability, maintainability, usability and reliability are improved. **Application/Improvement:** This paper uses Hospital management system which is enriched by Creative Innovative patterns and explains how these patterns help to improve the quality attributes of the architecture of a Hospital management system.

Keywords: ATAM, Creative Innovative Patterns Pattern, Hospital Management System, Software Architecture Analysis Methods

I. Introduction

Perhaps the most complex activity during application development is the transformation of a requirement specification into application architecture. Architectural design is a typical multiple objective design activity where the software engineer has to balance the various requirements during architectural design. Although there are methods for analyzing specific quality attributes, these analyses have typically been done in isolation[1-3]. A Software Architecture system is a composition of components such as Global control structure of the system, Synchronization and data access, Assignment of functionality to design elements, Composition of design elements, physical distribution, scaling and performance, dimensions of evolution, and selection of design alternatives.

The scientific and industrial communities have recognized that Software Architecture (SA)[4,5] sets the boundaries for the software qualities and it is not possible to measure the quality attributes of the final system based on the SA design[6-8]. This would imply that the detailed design and implementation, represents a strict projection of the architecture. Future work is needed to develop systematic ways of bridging the quality requirements of software systems with their architecture[9,10].

Software quality is the degree to which software possesses a desired combination of attributes (e.g., reliability, interoperability)[11,12]. Quality of architecture can be improved by using appropriate patterns. Patterns are the solutions for a recurring problem[13-15]. Patterns are applied to increase software quality by improving its quality attributes[16,17].

Hospital management system is taken as a case study for analyzing the quality attribute using patterns. Hospital management system (HMS) converts hospital activates into automated ones by making entries and providing reports as output. This HMS has to undergo improvement in its quality attributes. If patterns are tailored with the HMS, then the quality attributes of the HMS tremendously improves[18,19].

Section 2 shows the various methods of architecture analysis available. Section 3 explains in detail the ATAM

*Author for correspondence

method. Section 4 explains about the proposed patterns and its usage in hospital management system. Section 5 explains the procedure for applying the Creative Innovative patterns. Section 6 provides the analysis and discussion. Section 7 concluded the paper.

2. Methods Available to Analyze Software Architecture

The purpose of the software architecture evaluation method is to analyze the SA to identify potential risks and verify that the quality requirements have been addressed in the design[3]. In addition, it helps to know the important properties of software, even before it is developed and to predict the downstream effects of the software[20]. Therefore, it has been decided to study such methods in order to cover as many particular points of view of objective reflections as possible, to be derived from the general goal. SA is considered the first product in an architecture based development process[21]. There are eight methods available to analyze software architecture such as SAAM (Software Architecture Analysis Method), SAAMCS (Software Architecture Analysis Method Founded on complex Scenario), ESAAMI (Extending SAAM by Integration in the Domain), SAAMER (Software Architecture Analysis Method for Evolution and Reusability), ATAM (Architecture Trade-Off Analysis Method), SBAR (Scenario-Based Architecture Reengineering), ALPSM (Architecture Level Prediction of Software Maintenance), and SAEM (Software Architecture Evaluation Model). All the above methods are used to evaluate a software architecture for its functional requirements. These techniques required give very low consideration for non functional requirements like availability, scalability, maintainability etc. In this research set of patterns called Creative Innovative patterns are used for architectural analysis. These patterns focus on improving quality attributes of the architecture.

3. ATAM

From literature survey it is found that most widely accepted industrial and researched architectural Analysis Method is ATAM[22]. Following paragraph explains ATAM process and its lacunae.

The Architectural Trade-off Analysis Method (ATAM) was initially developed as a software architecture design method to support design trade-offs[4,5]. After some years it was progressed as a model for software architecture analysis. The targeted goal of ATAM is to give a disciplined reasoning while analyzing the capacity of software architecture. It also maintains trade-offs between competing attributes. The major inputs of ATAM are business goals, software specification and software description. The outputs of ATAM are list of scenarios, sensitivity points, trade-off points, risks, software architecture approaches and so on. ATAM is the best software architecture analysis technique ever used. ATAM is easy to implement but in complex. Even though ATAM is a best method it is also having the drawbacks.

3.1 Drawbacks of ATAM

- ATAM did not consider business view in its analysis.
- The quality attributes such as extensibility, maintainability, scalability, and reusability are not fulfilled in large systems.
- ATAM did not fulfill stakeholder participation.

The drawbacks identified in the ATAM method like maintainability, extensibility, scalability and reusability are not fulfilled to the entire satisfaction of stakeholders'. Proposed frameworks using CIPA introduced in the following section bridges the above drawbacks.

4. Proposed System

To overcome the disadvantages of ATAM, new patterns are introduced here. In this research eight patterns called Creative Innovative Patterns for Architecture Analysis (CIPA) are used for Architectural analysis purpose. These patterns have emerged from our historical analysis of product development and research trends. Research indicates that the more successful product or research innovation fits into at least one of these patterns[23]. Indeed, it is found that the patterns can help predict the emergence of new products before the appearance of signals indicating the market demand. In this research complete Hospital Management System product has been used to evaluate the proposed method. Following sections depict CIPA. The diagrammatic view of Creative Innovative Pattern for Architecture Analysis is shown Figure 1.

4.1 Characteristic Dependent Pattern

This pattern produces its results based on the characteristics of its environment[24].

Figure 1. Diagrammatic view of Creative Innovative Patterns for Architectural Analysis.

A hospital management system should support the characteristic dependent pattern based billing system so that the billing section maintains different billing methods which are adaptable to the different types of insurance of patients and hospitals have.

This pattern can be explained by some examples:

- Mobile companies first introduced mobiles in common colors for all people. But later Samsung company introduced panels in different colors focusing on the tastes of the young generation. Now Apple has introduced different colored mobiles.
- In DOS OS, the calculator application has the behavior of placing major mathematical calculations in a single window. But in Windows 7 OS the built in calculator is designed to satisfy the four types of users such as, Standard, Scientific, Programmer and Statistics. For these four types of users, the Microsoft organization has designed four different types of calculators to satisfy them perfectly.

4.2 Task Association Pattern

Here an existing product is modified by attaching another task oriented component but the existing component and the new component share the same body. So a single product embedded by double task components performs double duty.

In a hospital management system, normally the reception desk contributes in patient registration only. In accordance with the task association pattern, if the reception desk has the provision of giving medicines for contagious diseases, (with the assistance of nurses) then the reception desk performs dual activity by providing the best support to patients.

This pattern can be explained by some other examples as follows:

- Normally the split AC is used for air condition purpose only. But now Samsung company has introduced a mosquito repellent unit which is attached to the AC as a requirement of the bedroom. Thus it performs double duty.
- Earlier, Windows OS was not able to compress data by itself. So software like WinZip or WinRar was downloaded to compress data. But now Windows XP software has inbuilt compression facility.

4.3 Split up Pattern

A complex product is decomposed into small subsystems so that all the small subsystems can produce better result than the parent system.

Normally alert handling does not have the provision of priority in the hospital management system. Patient alert is not an emergency alert but theater alert is an emergency one. If the alerts are divided using split up pattern as Normal alert and Emergency alert, then the performance of alert handling becomes better. The Normal alerts are produced to remind patients whereas the Emergency alerts are produced for the theater, the doctors and anesthetics.

This pattern can be explained by some examples as follows:

- In early days the most commonly used air conditioner was Window-AC. But now split AC is being used. The usage of split AC has resulted in the reduction of sound, the size of the hole on the wall and building damage.
- Microsoft Visual studio version 6.0 package seems like a single unit at the time of installation. But after installation it gets split into Microsoft Visual Basic 6.0, Microsoft Visual C++ 6.0, Microsoft Visual InterDev 6.0 and Microsoft Visual Studio 6.0 tools.

4.4 Add-on Pattern

To an existing product, one or more components are added with slight modification to ensure better results.

In the hospital management system, using the add-on pattern, heavily crowded hospitals such as Apollo and KIMS can open multiple reception computer desks to efficiently handle the patient crowd. The software development for the hospital must be supported for multi system data entry. This will reduce the impatience of patients during registration.

This pattern can be explained by some examples:

- In the Sony double deck audio player, two audio cassette player units are kept in a single audio set. One is to play and the other is to record the audio.
- Normally Windows supports a single desktop only but now it can support multi desktops and so photo editors' are benefitted much.

4.5 Reduction Pattern

Here the non desirable components are removed from the whole system to gain some quality attributes.

In the hospital management system, instead of paying bills at different counters inpatients, through the reduction pattern can pay all the bills together at one counter. This behavior type of reduction pattern is not applicable for outpatients.

This pattern can be explained by examples like:

- The Sony Xperia Mini pro model has done away with the keyboard feature and now looks more stylish and compact.
- In the earlier version of Windows media player, the equalizer component is displayed but is not used much. In the recent versions, there is a button with options to use the equalizer component.

4.6 Limited Lifetime Pattern (LLP)

With the passage of time, softwares get updated and even undergo change. So the use of LLP is beneficial as it reduces cost expenses and opens space for change[23]. This pattern can be explained by the following example:

The three year old data of the patients are shifted periodically to another database in order to make the current retrieval process easy and quick. This process of archival increases the availability of the system and quick search response.

4.7 Agent Pattern

A complex system fulfills its sub component works with the help of inbuilt actives or mobile actives. Thus the main system is not compelled to develop or produce all its components by itself.

In hospitals, if a patient comes with a strange disease which is difficult for the doctor to come to conclusion the auto diagnosis software gives a solution to him. But owning this software is expensive. Using agent pattern, instead of buying a software doctor can use the software when needed by paying for that time of usage only. The architecture should be designed in such a way that the agent softwares should be able to adapt in that system. Thus the agent softwares can give a solution for the problem and removed after use.

4.8 Progressive Pattern

Doing a step by step work is normally called as progressive. In hospitals, the patient data can be searched using advanced search process by giving step by step inputs. Another example is in emergency case admission only the name filed is made mandatory in order to create unique ID. So the rest of the details can be done in a later date.

5. Procedure for Applying Creative Innovative Patterns

Input: The software architecture which is to be refined.

A set of new patterns a = {Active pattern, progressive, Limited Lifetime, Characteristic Dependent, Task association, Split up, Add-on, Reduction}.

Output: Improves the quality attributes and thereby the overall performance and efficiency of the specific architecture under consideration.

Step 1: Find the business goal, stakeholder's requirements, technical and managerial constraints.

Step 2:Collect and prioritize the requirements and analyze the architectural approach.

Step 3:Categorize the quality attributes appropriate to the architecture.

Step 4: Apply CIPA patterns.

Step 5: Create architecture based on CIPA patterns.

Step 6:Compute the final result and get feedback from the customer.

Step 7: Repeat the steps 2 to 5 if needed.

When CIPA is applied, the typical issues that are under examination are,

- Whether all the stakeholders considered?
- Check whether all the requirements are identified?
- Find whether the architectural solution being provided, appear rational?

- Check whether the documents are well managed?

The issues asked here can be examined by using the above steps and the results are discussed below.

6. Analysis and Discussion

6.1 Case Study

In order to evaluate the merits and demerits of the proposed approach (use of CIPA for software architecture) hospital management system, a large scale software project is considered and detailed analysis is carried out. A host of relevant parameters (analysis time, reliability, bug fixing cost, cost, and maintainability) and their variations were compared using both ATAM approach and CIPA. The result of the study is presented below.

Analysis Time

The analysis time is reduced for the development of the product. There are a reduced number of components by the application of the reduction pattern and the task association pattern. The cost of the product can be calculated with the formula,

Cost of the product = Human resource required for the product × Time required for analysis × The cost per unit time.

Figure 2 shows the analysis in time.

Reliability

Reliability of a system is defined as the mean time to failure. Reliability of the system was higher in this pattern than in ATAM. This is measured using the formula,

$$\text{Reliability} = \frac{\text{Duration for which the software worked without failure}}{\text{Total duration}} \times 100$$

Figure 3 shows analysis of reliability.

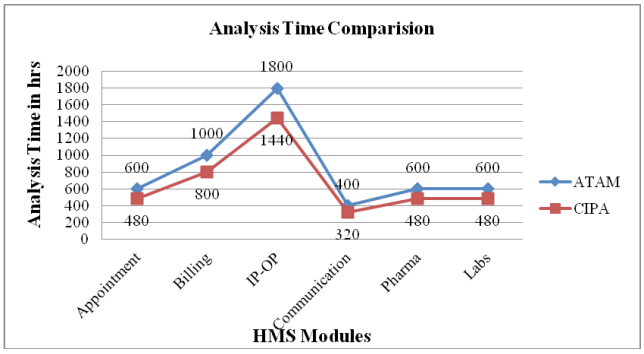

Figure 2. Analysis of Time.

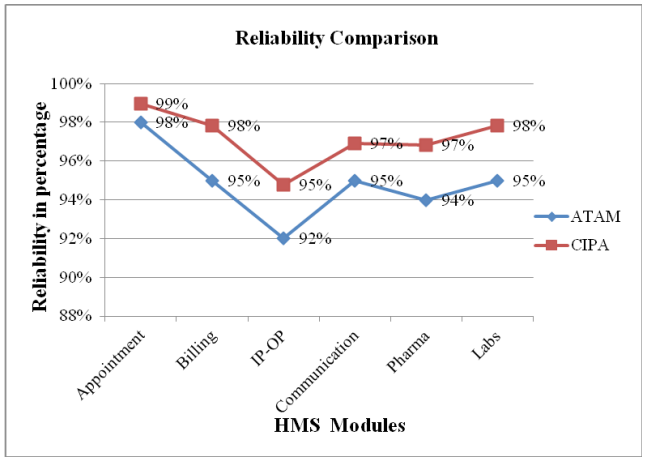

Figure 3. Analysis of Reliability.

Cost

The development cost also reduced significantly by the implementation of the reduction pattern. Figure 4 shows the analysis of development cost.

Cost of the product = Human resource required for the product × Time required for analysis × The cost per unit time.

Maintainability

Maintainability is defined as the percentage of total cost that is required to maintain the product. The maintainability is measured as,

$$\text{Maintainability} = \frac{\text{Number of application scenarios to which thesystem correctly adapted}}{\text{Total number of application that contains either false or require modifications}} \times 100$$

Figure 5 shows the analysis of maintainability.

Analysis Level Bugs

The result shows that there was a reduced number of bugs in patterns than in ATAM due to the refinement done by the application of the reduction pattern and task association pattern. Figure 6 shows the bug fixing cost in $.

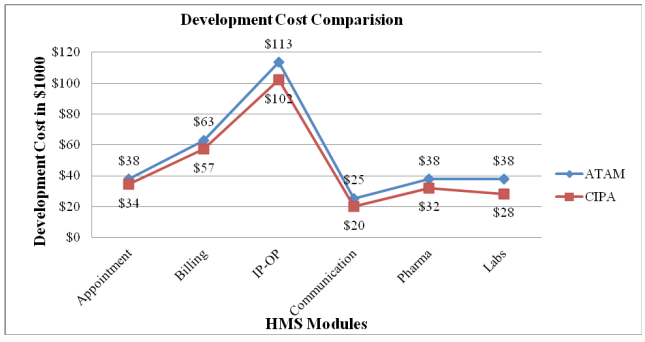

Figure 4. Analysis of Development Cost.

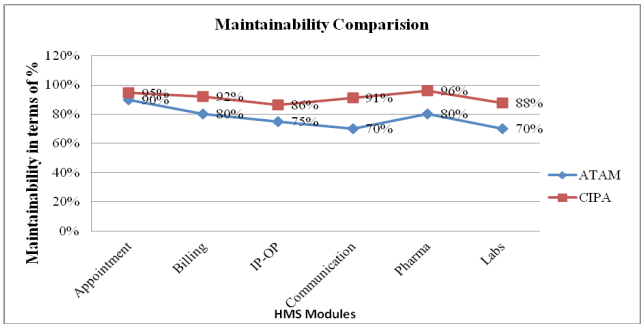

Figure 5. Analysis of Maintainability.

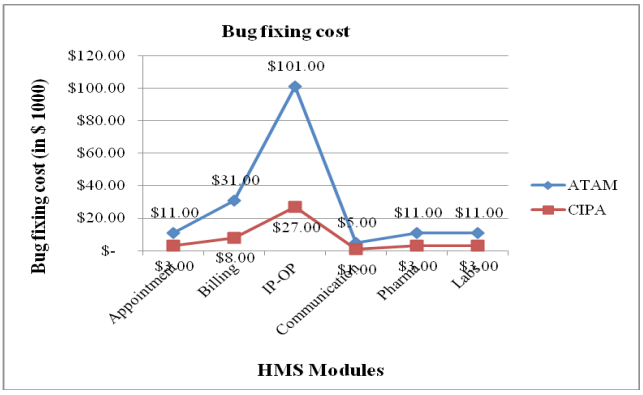

Figure 6. Bug fixing cost (in $ 1000).

7. Conclusion

The Creative Innovative pattern for architecture analysis method improves the quality attributes of the hospital management system more than the existing method. To improve the quality of the system, eight patterns are used. These patterns improve the software quality attributes usability, availability, scalability and maintainability. This also reduces analysis level bugs, cost and analysis time. The CIPA method is best suited for designing large and complex systems with improved quality attributes. In future, the proposed system can be updated with more powerful design patterns which can be used to hike the quality attributes.

8. References

1. Kazman R, Abowd G, Bass L, Clements P. Scenario-Based Analysis of Software Architecture. IEEE Software. November 1996; 13(6):47-55.
2. Klein M, Ralya T, Pollak B, Obenza R, Harbour MG. Kluwer Academic: A Practitioner's Handbook for Real-Time Analysis. 1993.
3. Len B, Clements P, Kazman R. Boston, Addison-Wesley: Software Architecture in Practice. Second Edition. 2003.
4. Chen Y, Li X, Yi L, Liu D, Liu T, Yang H. Beijing: A Ten Year Survey of Software Architecture. IEEE. 2010; p. 729-33.
5. Patidar A, Suman U. New Delhi: A Survey of Software Architecture Evaluation Methods. IEEE. 2015; p. 967-97.
6. Smith C, Williams L. Software Performance Engineering: A Case Study Including Performance Comparison with Design Alternatives. IEEE Transactions on Software Engineering. 1993; 19(7):720-41.
7. Bachmann F, Bass L, Klein M, Shelton C. Designing Software Architectures to achieve quality attribute requirements. IEEE Proceedings. 2005; 152(4):153-65.
8. Kruchten P, Obbink H, Stafford J. The Past, Present and Future of Software Architecture. IEEE Software. 2006 March/April; 23(2):22-30.
9. Dobrica L, Niemela E. A Survey on Software Architecture Analysis Methods. IEEE Transactions on Software Engineering. 2002; 28(7):638-53.
10. Hamidreza H, Homayun M, Hossein N. Selection of Appropriate Software Architecture Style with Square Spline in Style Based Systems. Indian Journal of Science and Technology. 2014 January; 7(6):804-08.
11. Bannerman PL. Vancouver. Canada: IEEE Publications: Software architecture: Organizational perspectives. LMSA'09. 2009; p. 37-42.
12. Sima E, Fereidoon S. A New Executable Model for Software Architecture Based on Petri Net. Indian Journal of Science and Technology. 2009 Sep; 2(9):15-25.
13. Buschmann F, Meunier R, Rohnrert H, Sommerland P, Stal M. Wiley Publications: Pattern-Oriented Software Architecture – A system of Patterns. 2014; p. 1-5.
14. Ali M, Mahmoud O, Elish E. Suwon: A Comparative Literature Survey of Design Patterns Impact on Software Quality. IEEE. 2013; p. 1-7.
15. Khwaja S, Alshayeb M. Niigata: A Framework for Evaluating Software Design Pattern Specification Languages. IEEE. 2013; p. 41-45.
16. Harsha SV, Outi S, Kai K, Kari S. Using Constrained Satisfaction and Optimization for Pattern based Software Design. NSW: 23rd ASEC, IEEE, Milsons Point. 2014; p. 29-37.
17. Joel C, Adams A. Patternlets: A Teaching Tool for Introducing Students to Parallel Design Patterns IPDPSW. IEEE. 2015; p. 752-59.
18. Hu D, Xu W, Shen H, LM. Study on Information System of Health Care services Management in Hospital. IEEE. 2005; 2:1498-501.
19. Muzamal RL, Athar A, Saqib NA. Seoul: Intelligent Agent based system for monitoring and control of hospital management system. IEEE. 2015; p. 1-3.

20. Shaw M, Garlan D. Prentice-Hall: Software Architecture - Perspectives on an Emerging Discipline. 2008.
21. Clements P, Bachmann F, Bass L, Garlan D, Ivers J, Little R, Nord R and Stafford. Addison Wesley: Documenting Software Architectures: Views and Beyond. 1st Edition. 2002.
22. Kazman R, Klein M, Clements P. Pittsburgh: Software Engineering Institute: ATAM: Method for Architecture Evaluation. 2000.
23. Goldenberg J, Louzoun Y, Solomon S, Mazursky D. Finding your innovation sweet spot. Harvard Business Review. 2003; 81(3):1-11.
24. Ram S, Rodrigues P. Innovative Patterns for Finding Enhanced Solutions to Your architecture. IJCSNS. 2008 Jul; 8(7):1-29.
25. Julia A, Rodrigues P. Quality Hike Patterns for Performance Elevation of Architecture Trade-off Analysis Method. IJETCA. 2015; 5(2):1-11.