Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:04:2021
DOI 10.17605/OSF.IO/87AGP

DYNAMIC SCENARIOS TRANSFORMATION IN SOFTWARE SYSTEM DESIGN USING ZEND FRAMEWORK

Dr. G MANJULA[1], SOWMYA S R [2], LORATY SHINY[3] and **LAVANYA D[4]**

1,2,4 Associate Professor, Department of ISE, Dayanada Sagar Academy of Technology & Management Bengaluru, Karnataka,
3. Assistant Professor, Department of CSE, Sri Sairam College of Engineering, Bengaluru, Karnataka, India

ABSTRACT

In the real word many software performance prediction methodologies are available such that the design decisions are clearly understandable by the stake holders. To improve the existing software performance Palladio component model methodology is used by little improving algorithm of the problem context which in turn reflected in the system design. The software architectural patterns are the best tool in measuring the quality. There are different types of architectural patterns like distributed systems, interactive systems etc. There are two types of communicating structures known as Model View Controller (MVC) then Presentation Abstraction Controller (PAC). Model View controller is more efficient tool and incorporated with the programming languages like UML, Eclipse, Object oriented patterns languages, java, etc.

1.1 SCOPE OF THIS WORK

Palladio component model is a normal state configuration structure where programming procedure will interact effectively with the quality related issues in the existing system design. The product procedure aimed at calculating quality in the execution, unwavering quality then estimation charge will be utilized in product by using different design programming language like simulation. The simulation is also one of the apparatuses where the system designs execute third party or the outermost layer in the stake holders then in turn calculating measurements. PCM is the process for evaluating the performance prediction by using the architectural patterns and utilized in numerous approaches. Forecast techniques designed for execution then unwavering quality based programming frameworks stand discreet constrained further once now a while utilized as a part of industry Component engineers who deliver segments that are amassed by programming designers and conveyed by framework allocators. The differing data required for the forecast of additional useful properties is in this way spread among these designer parts. PCM can likewise be utilized in light of the distinctive information set, where the behavioural aptitudes of information are information uprightness. The every conduct of information can be put into arrangement outline and follow out. The general procedure of design the MVC for performance is shown in the figure1.There are basically three units interrelated with each other like design inputs, design activities and design outputs. The area of the system is architectural design which in turn related with the interface design and database design. By modifying the architectural design using MVC, the efficiency of the system can be improved. The architecture will be same, when most of the systems domain same. The technology behind the domain may also be same. To fulfil the customer s requirements the application product will in turn bounded by core architecture of structure. The structural design of any classification stays designed by various architectural styles. The architectural patterns are means of reusing the object oriented design methods. This method relates towards programming designs demonstrated

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:04:2021

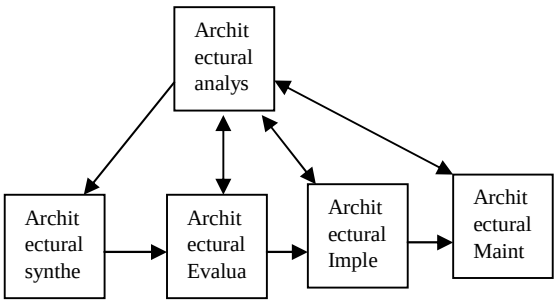
through the Palladio Component Model. It underpins quantifiable execution, dependability, besides charge forecast then correlated with each other, reached out towards extra reckonable excellence standards of programming designs. Through including another segment display in the middle of the every framework remains other compelling in computing also effortlessly reasonable in several professional solicitations.

1.2 PREREQUISITES AND SOFTWARE ARCHITECTURE

1. To tackle an issue an imperative is required so that accomplishing a target of an issue is impressive.
2. The limitation or constraints obligation have the nature of connection by means of framework part ideal to full fill the typical particular. However the additional strictly forced archive.
3. A record requirement reflects the previously revealed phases, then after that can guarantee that the engineering takes accomplished the goal of an issue.

II. REVIEW OF LITERATURE

Jan Bosch and Peter Molin [2], the main objective of well-defined architectural transformation requisite method and not well-defined method are discussed. The well-defined requisite methods are evaluated by iteratively such that the entire not well-defined method requisites are fulfilled successively. The evaluation process is categorized into 3 types, they are Scenario based Simulation based, Mathematical modelling and finally objective reasoning. In this method scenario based and simulation based computation is been used and proved the performance prediction is improved comparatively based on the previous methods. The second mode of evaluation is architectural transformation which categorized into five types, they are architectural styles imposing, architectural patterns imposing, design patterns using, converting not well defined requisite to distributed NFRs which in turn converting into functionality.
H.M. Fahmy, R.C. Holt, and J.R. Cordy [3], the description of how can work around the limitations of relational algebra using Grok. Since Grok is not tuned for the combinatory involved in performing sub graph isomorphism testing, the algorithm is slow even for small graphs.



Review recognizes SA approaches that are completed in administration of refinement, going from an abnormal state SA model to a more point by point one, and those situated to SA display development, remaining at a similar level of reflection. In this manner gives strong standards to assessing SA changes and it calls attention to imperative zones that need additionally inquire about. First architectural analysis, Second architectural synthesis and finally architectural evaluation.

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:04:2021

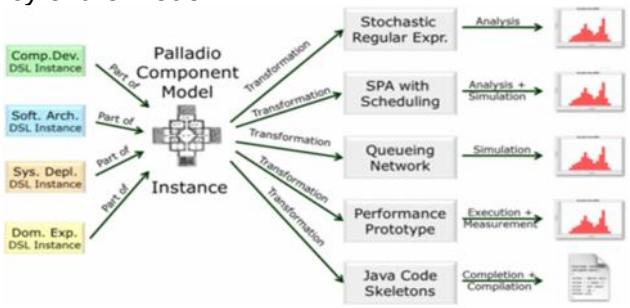
ARCHITECTURAL STYLES

The architectural style, sometime termed as Architectural patterns. The set of protocols which sharpen the application in user requirements is called architectural pattern. It derives the abstract knowledge from all the aspects of a desired system. The structural organization of the system is challenging task of the architectural pattern.

The prime responsibility of the architectural style is as follows.

1. It provides a component; each component specification is prescribed with reference to the family of the organization along with the set bounded around it. Each component is interrelated with other component called connectors.
2. It always improves the requirements solution by partitioning the components specifically and allows the reuse of the design by changing the design decisions. It also provides solutions for frequently occurring problems.

It describes clearly the configuration of the components such as a module transformation, well-defined interprets, reusability of the design and behavioral structure for replace
Algorithm- M-Accurate. The algorithm is to improve the existing system features. The process of inserting and editing the existing components are the major risk. Hover to overcome the existing system drawbacks need to add the CONCRETE model in the class diagram of the source code. The evaluation of performance through the following steps can be incorporated. The process flow of a software architectural design is as follows.
Step1: Defining the architecture with state diagram.
Step2: Identifying basic styles in heterogeneous architecture based on system design features.
Step 3: Mapping the state diagram to Markov model.
Step 4: Integrating Markov models to create an overall Markov model.
Step5: Creating the separate sets for each style and enforcing limitations.
Step 6: Creating the transition probability matrix.
Step 7: Calculating the visit number of each state in Markov model.
Step 8: Evaluating the efficiency of the model.



RESULTS: MULTI-STRATEGIC SOLUTIONS

Case study1

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:04:2021

ZEND FRAME WORK IN MVC PROGRAMMING DESIGN

The methodology applied in two study cases with certain constraints. Investigation of subspace mixture model and various distance measure techniques in class diagram.
The UML-MVC Logical Model
The coherent model explained comprises of stereotyped standard UML charts and is proposed to show the product segments to be created for each of the three layers of the MVC design [8] and additionally the connections among. In particular, the model is organized into the accompanying UML graphs:

1. Model Class Diagram (MCD): an UML class graph demonstrating the classes that take an interest in the Model layer and execute the application business rationale and information persistency.
2. View Class Diagram (VCD): an UML class graph speaking to the customer and server pages in the View layer. These pages have the obligation of displaying information and substance to the client and empowering client connection with the framework. This chart likewise models:
3. Zend framework in MVC programming design

4. A client makes a demand with the framework and the b server handles the demand by executing the bootstrap script index.php

5. The bootstrap scripts make an application case and runs it.

6. The application acquires gritty client ask for data from an application segment named ask.

7. The Application decides the asked for controller and activity with the assistance of an application segment named urlManager.

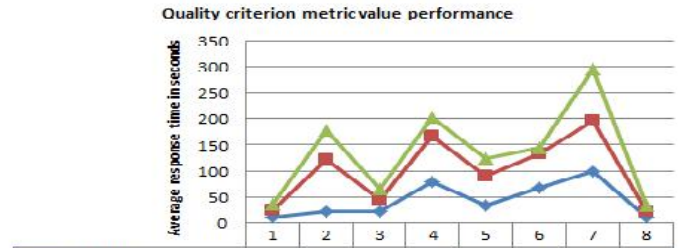
8. The Application makes an occurrence of the asked for controller to further handle the client asks. The controller class. It then makes and executes channels, connected with this activity. The activity is executed on the off chance that it is permitted by the channel.

9. The activity peruses a Post model who's ID is 1 from the database.

10. The activity renders a view named shoe with the Post show.

Software used in the implantation of the algorithm (Markova principle) the majority of the scenario based components automatically will not support for verification and validation. The software engineering process will always compute the new technology related to the execution of the system. The work product of any system can be chosen based on the tool set used for performance prediction. By considering the Palladio component model (PCM) for software prediction, improved the efficiency and cost change of the system as shown in the graph 4.7

Tianjin DaxueXuebao (ZiranKexueyuGongchengJishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online): 0493-2137
E-Publication: Online Open Access
Vol:54 Issue:04:2021

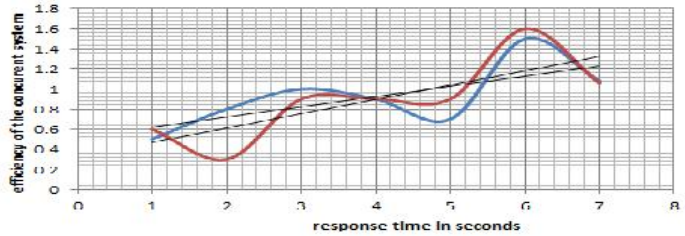


	1	2	3	4	5	6	7	8
reliability	12	56	22	35	34	11	98	11
Average utiliation	12	99	22	80	55	67	98	11
Average respond time	12	22	22	78	34	67	98	11

The software process can be designed based on the above said classifications; hover it is necessary to apply the General Principles of David Hooker [Hoo96] to make the system work efficiently. The David Hooker principles are as follows

1. All the decisions are need such that that user value added in all the aspects.
2. Every design should be as simple as possible, such a way that it should not be so simpler.
3. A strong visualisation is important to the achievement of a software venture of a system.
4. All the time identify, design and implement such that hierarchical growth may be required else will have to recognize in all the aspects.
5. At no time design a system such that it is more specific to a corner.
6. Outline into the future viewpoints with the end goal that reuse-diminishes the cost and expands the estimation of both the reusable part and the framework into which they are fused.
7. Arranging clear, entire thought before activity quite often creates better outcomes are shown in the graph 4.9

Graph 3



SUMMARY AND RECOMMENDATIONS FOR FURTHER STUDY AND PRACTICE

The principal deliberate scenario-based technique is described and assessing programming engineering is maybe the SAAM strategy [2, 3, 4], which will be contemplated in the following section. SAAM strategy was initially recommended to improve the performance prediction tools, reusability, cost and modifiability of programming frameworks in view of data exhibited in building outlines. It has been investigated for the assessment and examination of programming quality credits identified with the alterations, for example, movability, extensibility, integratability, and so on and also practical scope, execution and unwavering quality. Scenario-based techniques for breaking down different programming quality characteristics at design level have additionally been contemplated by Jan Bosch, Perol of Bengtsson for practicality [5], by Perol of Bengtsson, Nico Lassing, Jan Bosch and Hans van Vlient for modifiability [6, 7], and Per Olof Bengtsson and Jan Bosch for scenario-based engineering reengineering [8]. Rick Kazman, Jeromy Carriere and Steven Woods [9]

explored how to inspire scenarios efficiently and introduced a product apparatus to bolster the administration of scenarios in engineering examination.

In a more extensive setting, scenarios are utilized in prerequisites building to evoke programming necessities, see for instance [10], yet for the most part for practical prerequisites. Specifically, utilize cases and scenarios assume a huge part in protest arranged investigation and plan system.

REFERENCES

[1] Karl J. Lieberherr, College of Computer Science, North-eastern University Cullinane Hall, Boston, MA 02115 —Metrics of Graph Abstraction for Component-Based Software Architecture‖ Los Angeles, California USA March 31-April 02 ISBN: 978-0-7695-3507-4
[2] Science of Computer Programming - Special issue on —Applications of graph transformations (GRATRA 2000) — Volume 44 Issue 2, August 2002 Publisher: Elsevier North-Holland, Inc. Amsterdam, The Netherlands, The Netherlands ISSN: 0167-6423 doi:10.1016/S0167-6423(02)00036-9
[3] Guo Wei, XiongZhong-Wei, Xu Ren-Zuo, "Metrics of Graph Abstraction for Component-Based Software Architecture," csie, vol. 7, pp.518-522, 2009 WRI World Congress on Computer Science and Information Engineering, 2009.Articles:Visual Design of Software Architecture and Evolution based on Graph Transformation
[4] Eclipse.org. ATLAS Transformation Language (ATL). http://www.eclipse.org/m2m/atl/.
[5] A. Billig, S. Busse, A. Leicher, and J. G. Süss. —Platform Independent Model Transformation Based on TRIPLE‖. In Middleware‘04: Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware, pages 493–511, 2004. [6] D. Ayed and Y. Berbers. —UML Profile for the Design of Platform-Independent Context-Aware Applications. In MODDM‖ 06: Proceedings of the 1st Workshop on Model Driven Development for Middleware, pages 1–5, 2006.
[7] OMG. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Version 1.0. ttp://www.omg.org/docs/formal/08-04-03.pdf, 2008.
[8] S. Link, T. Schuster, P. Hoyer, and S. Abeck. —Focusing Graphical User Interfaces in Model-Driven Software Development‖. In ACHI‘08: Proceedings of the 1st International Conference on Advances in Computer-Human Interaction, pages 3–8, 2008.
[9] D. Habich, S. Richly, and W. Lehner. IgnoMDA —Exploiting Cross-layer Optimization for Complex Database Applications‖. In VLDB‘06: Proceedings of the 32nd International Conference on Very Large Data Bases, pages 1251–1254, 2006.
[10] David Garlan and Mary Shaw,‖ An Introduction to Software Architecture, In Advances in Software Engineering and Knowledge Engineering‖, Volume 1, World Scientific Publishing Company, 1993.
[11] C. He, F. He, K. He and W. Tu. —Constructing Platform Independent Models of Web Application‖. In SOSE‘05: Proceedings of the 2005 IEEE International Workshop on Service-Oriented System Engineering, pages 85–92, 2005
[12] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. —Pattern-Oriented Software Architecture: A System of Patterns‖. John Wiley & Sons, Inc., 1996.
[13] M. López-Sanz, C. Acuña, C. Cuesta, and E. Marcos. —UML Profile for the Platform Independent Modeling of Service-Oriented Architectures. Software Architecture‖, pages 304–307, 2007.
[14] eclipse.org. Model to Model (M2M) Project. http://www.eclipse.org/m2m/.
[15] T. Fink, M. Koch, and K. Pauls. —An MDA approach to Access Control Specifications Using MOF and UML Profiles. Electronic Notes in Theoretical Computer Science‖, 142:161–179, 2006. [16] J. Bezivin, S. Hammoudi, D. Lopes, and F. Jouault. —Applying MDA

approach for Web service platform. In EDOC‘04‖ Proceedings of the 8th IEEE International Enterprise Distributed Object computing Conference, pages 58–70, 2004. Dynamic scenario transformation in software system design Page 46

[17] International Journal of Computer Science & Information Technology (IJCSIT) Vol 3, No 4, August 2011 DOI: 10.5121/ijcsit.2011.3405 57 —-BASED ATL TRANSFORMATION TO GENERATE MVC 2 WEB MODELS‖.

[18] Liliana DOBRICĂ1, Anca Daniela, —mda-based atl transformation to generate mvc 2 web models‖ in AUTOMATIC TRANSFORMATION OF SOFTWARE ARCHITECTURE MODELS. IONIŢ Radu PIETRARU3, Adriana OLTEANU4 U.P.B. Sci. Bull., Series C, Vol. 73, 2011 ISSN 1454-234x [19] Daniel Toll and Morgan Ericsson, —Evolution and evaluation of the Model view controller architecture in games‖, by in IEEE/ACM 2015 4th international workshop on games and Software Engineering.

[20] Daniel Lucredio, Renata P m Fortes, Eduardo S Almedia and SilvoL.Meira, —Designing domain architectures for Model-driven engineering‖, in 2010 Fourth Brazilian symposium on software components, Architectures and reuse.

[21] Ramon Hugo de Souza, Paulo arion Flores, —Architecture recovering model for distributed databases: A reliability, Availability and Serviceability approach‖ in IEEE symposium on Computers and communication in 2016.

[22] Zaki Brahmi, Jamel Feki, SlimaneHammoudi —Towards semi-Automatic transformation Process in MDA Model driven software architecture evolution information capture — 2015 second international conference on Information science and control engineering.