Software Engineering Process Model: A cost and quality enhancive technique.

Mohammad Asaad Subih
Department of CS & IT
University of Lahore, Gujrat
Gujrat, Pakistan.
asad.subih32@gmail.com

Maria Latif
Department of CS &IT
University of Lahore, Gujrat
Gujrat, Pakistan.
maria.latif@cs.uol.edu.pk

Jamshaid Amjad
Department of CS &IT
University of Lahore, Gujrat
Gujrat, Pakistan.
jamshaid11601@gmail.com

Abstract—The Software industry has flourished and changed our lives in a way like never before. The software is considered as paramount for our life. The exceeding entanglement in this context is the Quality and Cost-effectiveness of the software products. Previously many approaches have been used in order to enhance the quality of the software products (i.e. validation) affecting the overall cost of the software product. Our Research work can be divided in two steps. First step is to review all top-notch Software Engineering Process Models in practice i.e. water fall model, spiral model etc. and Second and most cardinal part of the research work is to propose a new software engineering process model which will help in making software products more efficient both in terms of both cost and quality. Our research will ensure that the proposed model follows all the basics practices of Software Engineering process.

Index Terms— Quality, Validations, Models, Increment, Pipe lining, Z-notations, Feasibility Test, Software Engineering Process.

1 INTRODUCTION

THE Software Industry has reciprocated our life like never before. The software is in use everywhere from our pocket i.e. cell phones to the up high in the sky i.e. satellites. The Software has impacted our lives in terms of communication, management and other elements of life. In the year 2012, the Software industry has participated in 3.2% (452 $ Billion + 101$ Billion) of the total GDP in that year [11]. "Software is eating the world" [8]. The cost of poor quality in the year 2018 in the USA is $2.84 trillion [13], which is far too much. The situation is alarming and demanding necessary corrective actions in order to counter this dilemma. Many models and techniques have been developed and used but there is always a gap for betterment.

Software quality is elucidated as "The degree to which system components or process meets the specified Requirements" [9]. Software quality assurance is a planned systematic approach to evaluate the quality and standards of the software product [14]. It requires a lot of cost and effort to review, reviving back the whole product and to do the major changes after the product has finished its manufacturing chain, as reviews take a lot of time. Time and cost are directly proportional to each other, greater the time spent higher will be the cost and vice versa. Our Research will review all the top-rated software engineering models and will propose a a new model which will help us in achieving the quality software products while being cost efficient as well.

2 A REVIEW OF THE EXSISTING SOFTWARE ENGINEERING

Process is defined as the series of action taken to achieve some particular goals. In context to the software engineering the models are defined in terms of requirement analysis phase of each model [1], but our research work will take in account all the differentiating factors of all the Software Engineering Process Models in practice specially their working scenarios. The Detailed Analysis of top-notch software engineering process model is given below in detail

2.1 A. Water Fall Model

Water fall Model is commonly used for the development of the small-scale projects or the projects having small level of complexity. It is the most commonly used and understood model. The water fall model is a static model and it approaches the model in linear and sequential manner [4]. It is an underprivileged option for the development of the software because in this model Requirements are not understood by the development team and other than that the verification is specifically is done only in particular phase, which means that if the software is not according to the specified requirements i.e. not fulfilling the quality factors the reviews should take place from the scratch, reviewing the whole project.

More Over Water Fall Model is rigid in nature and does not support any kind of flexibility for the development, causing increase in expenses in terms of verifications and reviews and delay in delivering the project. The water fall model is shown below

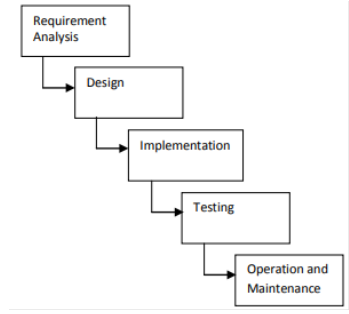

Fig.1 showing the Water fall model {15].

The Major deportment related to the water fall model is that the model cannot be adopted for any change and if any change is incorporated the project leads to delay and confusions [1].

2.2 V-Shaped Model

V Shaped Model takes Water Fall Model a step forward. V-Shaped Model is a sequential model very similar to the water fall model implying the series of actions in a sequence.

The V-Shaped Model can be said as the "next generation water fall model" or "water fall model which supports testing". There is an Old Saying "Prevention is better than cure", In terms of software it can narrated as "It's better to Verify then to Validate". The V-Shaped model takes it in account as it involves the testing from the very start of the process. The initialization stage for Water Fall Model and V-Shaped Models are same but V-Shaped is better because it involves the Testing in very beginning of the process.

In V-Shaped Model the testing team has to perform a series of actions like planning the tests, designing the test cases so that actual testing can take place. The testing will take place in parallel of each activity. In V-shaped Model the flow of the process changes from "Left" to "Right". "Left" ones are the "Do-Procedures" i.e. the activities which are needed to be performed by the development team and on the "Right" are the "Check-Procedures" i.e. the activities which are needed to be performed by the testing team.

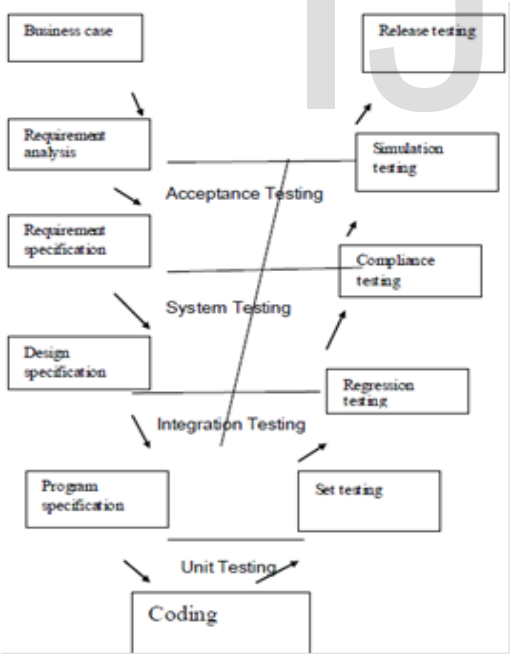

Fig.2 V shaped Model and its internal Processes [7].

The V-shaped has solved the problem related to the testing to some extent in the water fall model but still it is rigid which creates problem in terms of flexibility of the model.

2.3 Spiral Model

The Spiral Model was developed by Kotonya and Summerville. The Spiral Model uses evolutionary approaches as it recognizes a great series of uncertainty and it allows the developers to attain additional information and handle it while working on the project. The Working structure of the Spiral Model is clockwise direction [4]. The activities performed during the spiral model are

- Analysis
- Design
- Implementation
- Testing

The Spiral Model is further divided into multiple frame work activities known as "Task Regions". The task regions usually range from 3-6. In 1987 Berry Boehm integrated and corrected the primitive Spiral Model with some techniques from Prototyping and some from the classical Software Engineering Process Models I.e. Water Fall Model. The Spiral Model scrutinizes the extent of the possible risk and eliminating the risks increasing the cost of the project [5]. The Spiral Model is very good in Risk analysis. The end product i.e. Software is developed usually early in the life cycle. Spiral Model is commonly used in large projects having high scale of uncertainty i.e. risk involved in them. The Biggest problem prevailing in the Spiral Model is that it is not cost effective at all [3] [4].

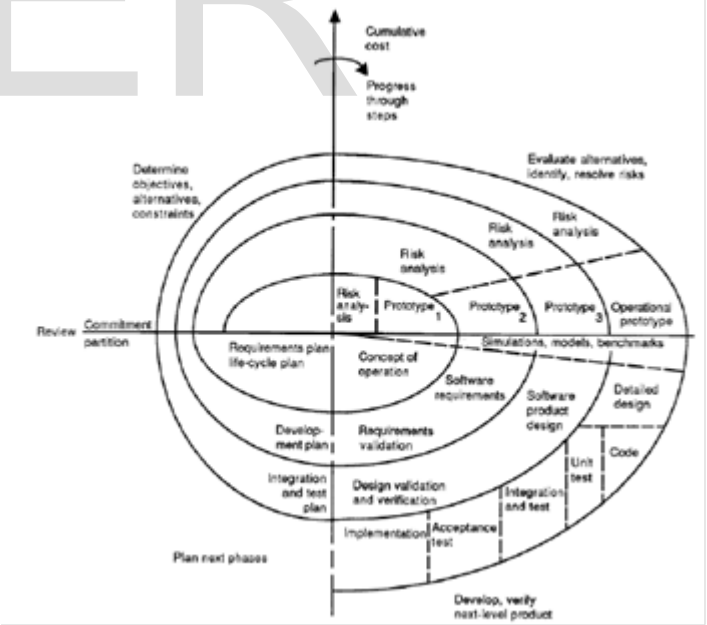

Fig. 3 showing Spiral Model proposed by Boehm[3].

2.4 Incremental Model

Incremental Model is an Increment based model. In this model the requirements are taken as an increment and developed. This process continues until the end product is ready. The incremental model is used where the requirements are clear and easily understood. The major concern in this model is that all the requirements are not understood and al-

ways increasing until the time of completion [17].

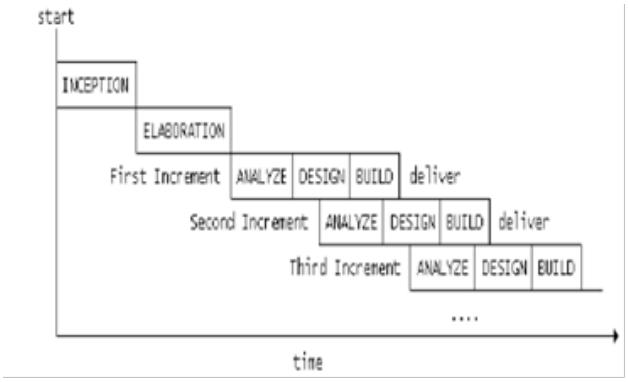

Fig.5 Showing the Incremental Model [4].

Other than that, at a certain time when requirements become mush complex and large then it becomes rigid as well.

2.5 RAD Model

RAD stands for "Rapid Action Development". RAD is an incremental model. It is a coordinated development technique in which peripherals are developed in lateral. If the requirements are completely understood by the development team, RAD enables the development team to develop a fully working prototype in no time [1]. RAD is used where requirements are Static and are Complete. The Major deportment related to the water fall model is that the model cannot be adopted for any change and if any change is incorporated the project leads to delay and confusions [1].

The RAD can be narrated as a sheer force approach to some extent. The ultimate aim of this approach is to develop the deliverable in terms of the project as early as possible. The RAD Model is shown below. The RAD joins together some concepts from the PROTOYPING and some techniques from the basic Software Engineering Model I.e. Water fall model.

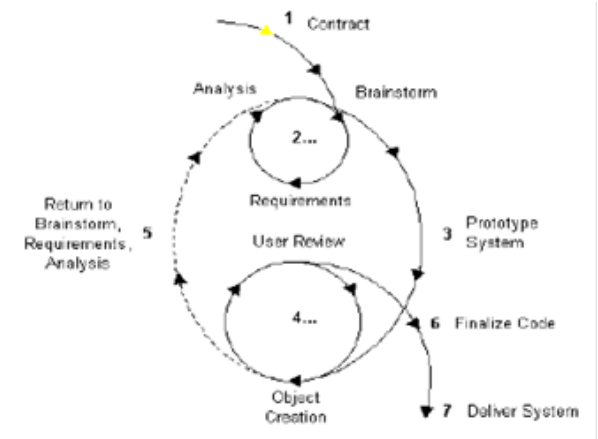

Fig. 4 Showing the RAD and its Processes.

3 THE PROPOSED MODEL

In previous section we have taken in account all the major and widely used software engineering process models and the issues related to them. The one of the serious issue pertaining in the existing model is that models are rigid and are not flexible enough to accept any change and in case they are flexible they are not cost effective.

The structure of the proposed model is shown below

The Proposed model starts from the "Requirement Gathering" phase. In this phase, Requirements are gathered from the customers and stakeholders. The Second Phase is "Analyze Requirements and Perform feasibility test". In this Phase it is checked the Requirements are realistic ones or not and in the second stage it is checked that requirement is "fit for the job" or not. The third and most important Phase is the "Setup System Development Standards and Constraints". This Phase has a vital role in the project development as in this stage the decisions regarding what is to be done and how it is to be done is taken. The process involves the development team, management, and stakeholders involved in it. In this phase system constraints and decisions regarding the system architecture, development platform and development tools are taken into account. After that in "formalizing the Requirements" the re-

quirements are formalized using formal methods specifically Z-notations in order to remove all ambiguousness from the requirements.

Customer Verification is considered as the nucleus of the proposed model, as it involves the verification and completion of requirements from the customer end. In previous practices, it was observed that most of the times the cause of poor quality was the ever-increasing requirements from the customer end. In order to counter this Problem, this phase is very important as it re ensures that either customer or stakeholder has given the complete detail of requirements and if not, the customer can give the new details of required specifications at this stage and all of the above-mentioned activities are performed. The main activity involved in this phase is that customer certify and ensure by signing on the agreement that further no more change or new requirement can penetrate until the deployment of the Product.

After that "Marking the Increment and Passing" takes place in this phase each requirement is considered as an increment and that increment is passed on to the development team. Each Requirement (i.e. functional one) is Passed on to a specifically assigned development team which will work on that particular requirement. One important thing to accord at this stage is that each increment is going to be assigned to a specific development team which means that multiple teams will work in parallel and each requirement will be tested after the development which means that error or bug can be detected in premature stage in context to the whole project and necessary corrective action can be taken. After that the increments i.e. Functional units are integrated into "Integration Unit", which means that all the increments join up to form a complete product and at the last and final the product is deployed for the user and stakeholders for use.

Our proposed model will make sure that the product is thoroughly tested from requirement till the development phase and the development takes place at a rapid pace because the concept of "Pipelining" is used and deployed in the model. Testing indulged with the concept of pipelining will ensure that the developed product is true quality and cost-effective one.

4 Conclusions

Software Industry has flourished and made progress with passing time and it is still flourishing in the way like never before. The situation seems to be very charming but it gives rise to new challenges. One of the major challenges in this context is about the quality and cost effectivity of the software product. The Research work has taken in account and reviewed all of the top-notch software engineering model exploiting their shortcomings and deficiencies. This research will help the masses in deciding which model to choose in certain circumstances. The Research Work Has Proposed a new Software Engineering Process model which ensures the quality and cost effectivity of the software product.

5 Referances

[1] Er.Neha Budhija ,Er. Satinder Pal Ahuja, "Study of Software Process Model Selection", International Journal of Advanced Research in Computer Science, Volume 2, No. 6, Nov-Dec 2011.

[2] Mona Batra, Dr. Archana Bhatnagar ,"A Comparative Study of Requirements Engineering Process Model", International Journal of Advanced Research in Computer Science, Volume 8, No. 3, March – April 2017.

[3] Mr. Preyash Dholakia, Mr. Dishek Mankad, "The Comparative Research on Various Software Development Process Model", International Journal of Scientific and Research Publications, Volume 3, Issue 3, March 2013.

[4] Ashwini Mujumdar, Gayatri Masiwal and P. M. Chawan," Analysis of various Software Process Models", International Journal of Engineering Research and Applications, Vol. 2, Issue 3, May-Jun 2012, pp.2015-2021.

[5] John Wiley and Sons, "Process Models in Software Engineering", Walt Scacchi, Institute for Software Research, University of California, Irvine February 2001.

[6] Silvia T. ACUÑA and Xavier FERRÉ, "Software Process Modelling", Universidad Nacional de Santiago del Estero.

[7] Ravi Shanker Yadav, "Improvement in the V-Model" , International Journal of Scientific & Engineering Research, Volume 3, Issue 2, Februaryy-2012.

[8] Marc Marc Andreeson, August 20, 2011, Wall Street Journal.

[9] Online, IEEE.

[10] Sacha Martin, Aybüke Aurum, Ross Jeffery and Barbara Paech, "Requirements Engineering Process Models in Practice", 3Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.

[11] Robert J. Shapiro of Sonecon, "The U.S. Software Industry: An Engine for Economic Growth and Employment", SOFTWARE & INFORMATION INDUSTRY ASSOCIATION (SIIA).

[12] J. Breckling, Ed., The Analysis of Directional Time Series: Applications to Wind Speed and Direction, ser. Lecture Notes in Statistics.

[13] Herb Krasner, "The Cost of Poor-Quality Software in the USA 2018 Report", Consortium for IT Software Quality.

[14] C. Senthil Murugan and S. Prakasam, "A Literal Review of

Software Quality Assurance", International Journal of Computer Applications, Volume 78 – No.8, September 2013.

[15] Adetokunbo A.A. Adenowo, Basirat A. Adenowo, "Software Engineering Methodologies: A Review of the Waterfall Model and Object-Oriented Approach", International Journal of Scientific & Engineering Research, Volume 4, Issue 7, July-2013.

[16] P Beynon-Davies, C Carne andD Tudhope, "Rapid application development (RAD): an empirical review", European Journal of Information Systems.

[17] Dominique Mirandolle, Inge van de Weerd and Sjaak Brinkkemper, "Incremental Method Engineering for Process Improvement - A Case Study" , Chapter in IFIP Advances in Information and Communication Technology · January 1970.