Advanced Model of Software Testing Management

Debashis Ghatak
IT Engineer (SI)
CMC Ltd.
(A TATA Enterprise)
SDF Building (3rd Floor) Module No.219-222
Sector-V, Block GP, Salt Lake City
Kolkata-700091

ABSTRACT

Software Testing is the most important phase of the Software Development Life Cycle. Software testing is essential to ensure software quality. Software quality very much depends on “Software Testing”. And “Software Testing” is also very much dependent on “Validation”. To improve the software quality is a great challenge in front of us. This article has discussed an improved model on the basis of “Verification & Validation”. There are several model exists on the basis of “Verification & Validation” like V-Model, Waterfall Model, Spiral Model etc. This improved model makes the software testing pass through the each stage of software development cycle. This new discovers much reduces the software bugs & errors specially in respect of the “Software Development Company” in respect to the present one.

General Terms

“D-LADDER MODEL”, Architecture of “D-LADDER MODEL”.

Keywords

Verification, Validation, Feasibility Specification, Feasibility Testing.

1. INTRODUCTION

Basic component of any Software Products stands on the two phase “Verification” & “Validation”. In the “Verification” where the process of evaluating a system or component which is given at the time of development phase that measure whether the said product satisfy the requirement which is imposed at the start of this phase. In the “Validation” where it measures the correctness of the product which is developed by developer according to the feasible requirement of the customer. In the “Software Testing” which is the most valuable part to deliver a good product to the customer. “Software Testing” is the process of excising software which measures the correctness between actual feasible requirement & corresponding development by the developer. Also find out bugs, error between them.

“Verification” is providing that a product meets the requirements according to the specification of the customer. “Validation” checks that how much systems fulfill the customer requirement. If both “Verification” & “Validation” meet simultaneously then it can easily optimize the problem and also get a consolidated optimized data.

IT Products consists of large number of and various combinations of critical data of the organization dependent. During the time of development of these such kind of long live software require detail level analysis, feasibility of the requirement, technical specification, design and coding. To get the proper accuracy according to the feasible requirement it require some detail level testing like unit testing, module testing, integration testing, regression testing, load testing, stress testing, Security Testing, System Testing, Security Testing, feasibility Testing, Acceptance Testing.

2. TYPES OF MODEL IN SOFTWARE TESTING MANAGEMENT

Relation between “Verification and Validation” is a Parent & Child. So if verification stage will not cover all steps during the time of software development then validation will not work according to the same.
There are several models based on “Verification and Validation” which belongs to the Software field like “Waterfall Model”, “V-Model”, etc.

3. DETAIL DISCUSSION AND FIND OUT ADVANTAGES AND DISADVANTAGES OF EXISTING MODEL

Detail Discussion about Two Models such as “Waterfall Model” & “V-Model”. First discuss about “Waterfall Model”.

Waterfall Model

The Water fall Model is the mother of all models in the Software Engineering. It follows sequential approach time of software development period. Waterfall model follows the “Top Down” approach. It consists of combination of “Verification and Validation” process. Waterfall model steps forward through

1) **Customer Requirement**
2) **Design**
   a) **Architecture Design**
   b) **Logical Design**
   c) **Functional Design**
3) **Coding**
4) **Testing**
5) **Maintenance**

In the “Waterfall” model first steps is to gather the customer requirement of the software.
To understand the nature of the requirement “Analyst” will evaluates the requirement and to take required steps accordingly.

Software Design is a bridge between "Requirement of the Software" and "Representation of the software". Software design is the multi steps process. Software designer will design the software architecture of the requirement. Software designer generates the "Data Structure" of logical design according to the software architecture. Software designer also define the "Interface Tools" or Functional Design to develop the said software.

Software testing measures the actual requirement and corresponding output comes from the developed software which is developed by the software coder and find out the bugs, errors from the developed software.

After implementation of the software at the customer end some modification is always needed due to some new requirements added in the software or some errors have occurred. This software maintenance comes in the existing software not in the new software.

Advantages:

1. Waterfall model is a sequential model so it is very easy to understand.
2. It is very useful in the very small project.

Disadvantages:

1. Main disadvantage in the "Waterfall" model is that in the "Testing" section is not clearly understandable that what types of **"Software Testing"** have done?

2. As "Waterfall" model follows "Top down" approaches then if any "Bug" will found during the time of "Testing" then it is very difficult to optimize the said error spontaneously.

3. In the "Waterfall" model "Testing" only have done on the basis of "Coding". But in other two modules "Analysis" and "Design" have not any provision for testing or measure the correctness of those two modules.

Figure of Waterfall Model is as follows:

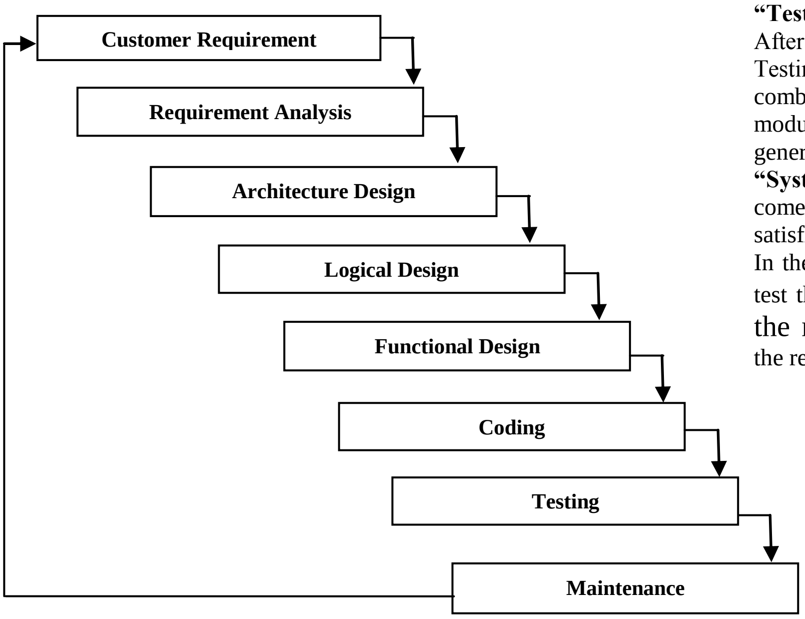

Fig. 1 Waterfall model

Second discuss about **V-Model**

V- Model

"V-Model" comes from the "Water Fall" model. It is also modified version of "Water Fall" model. In the "V-Model" corresponding "Testing Phase" is divided into several sub phase. "V-Model" into two modules such as

Module 1) Analysis, Design and Execution

Module 2) Testing and Implementation

The "Module-1" belongs to the "Verification" phase and "Module-2" belongs to the "Validation" phase.

"Module-1" of **"V-Model"** has several sub modules such as

a) **Requirement Analysis**
b) **Architecture Design**
c) **Logical Design**
d) **Functional Design**
e) **Coding**

"Module-2" of **"V-Model"** has several sub modules such as

a) **Unit Testing**
b) **Module Testing**
c) **Integration Testing**
d) **System Testing**
e) **User Acceptance Testing**

"Unit Testing" is a part of **"White Box"** testing. **"Unit Testing"** is basically **"Coder's"** end testing. After completion of coding and before send it for testing "Coder" must test their coding and try to find out some error within the boundary of the specific module and resolve the same.

"Module Testing" is a part of **"Black Box"** testing. According to the requirement tester will first generate a **"Test Case"** for corresponding module. On the basis of **"Test Case"** tester will prepare **"Test Bed"** for each module and try to find out bugs and error on the basis of the **"Test Case"** & **"Test Bed"**.

After completion of the several "Module" of "Module Testing,", in the "Integration Testing" said module will combined as like as **"Big Bang"**. The entire consolidated module will tested as a whole. Usually a set of errors will generate during the time of **"Integration testing"**.

"System Testing" will perform that whatever the output has come after completion of **"Integration Testing"** that is satisfied according to the customer requirement or not.

In the **"User Acceptance Testing (UAT)"** the customer will test the final product and verify the output according to the requirement. If the output will satisfy according to the requirement then customer will accept the product

Advantages

1. It is easy to Understand and easy to use.
2. Presently this model is used in the complex software designing

Disadvantages

1. **"Coding"** is the heart of this model and testing phase has done on the basis of the output of the **"Coding"**. But in the real time scenario design is the heart of every development. There is no such provision is there to find out errors in the design phase.

2. Major testing facility like **"Regression Testing"**, **"Security Testing"** is not available in the **"V-Model"**.

Figure of V-Model Model is as follows:

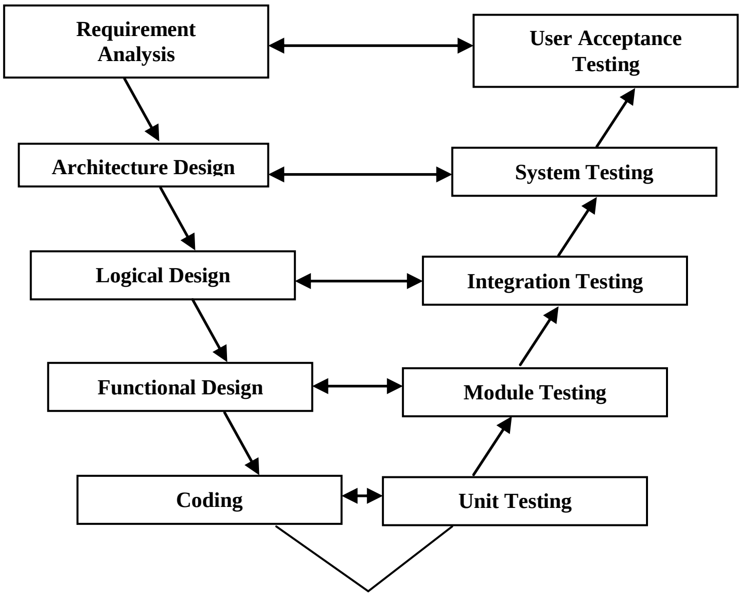

Fig. 2 V-Model

3.1 Introduction About New Model

This new model is generated on the basis of **"Verification" & "Validation"**, the new model divided into two phases such as **"Top Down"** & **"Bottom Up"**. Each phase have multi sub phase but each sub phase is related and connected with other sub phase. **"Top Down"** phase is related with **"Development"** related purpose and **"Bottom-up"** phase is related with **"Testing"** related purpose.
In the **"Top Down"** phase have the following sub-phases such as

a) **Requirement Analysis**

b) **Feasibility Specification**

c) **Technical Specification**

d) **Review of Technical Specification**

e) **Design Specification**

f) **Review of Design Specification**

g) **Coding**

h) **Review of coding**

The above steps are related with **"Development"** phase but here some sub phases have introduced which is related with **"Software Testing"**. In generally the well-known that **"Software Testing"** starts after completion of **"Coding"**. But if start testing from the very beginning of the development phase then at the end it will get an almost bug or error less software.
In the **"Bottom-up"** phase have the following sub phases such as

a) **Unit Testing**

b) **Module Testing**

c) **Integration Testing**

d) **Regression Testing**

e) **Load, Stress and security Testing**

f) **System Testing**

g) **Feasibility Testing**

h) **User Acceptance Testing(UAT)**

A new sub phase as well as a new testing technique have introduced in this article that is **"Feasibility Testing"**. Now question can arise **what is the purpose of "Feasibility Testing"?** In the **"Load, stress & Security"** testing if some major functionality will change according to the actual requirement then it will not identify in the **"System Testing"** phase. According to the existing definition of **"System Testing"** will work only on the output of **"Integration Testing"** for the said reason some times when product moves in to the **"User Acceptance Testing (UAT)"** customer rejects the entire development which had delivered to their end because that delivered product does not fulfill the actual requirement.
In the "Feasibility Testing" will test all the major functionality and corresponding required output with **"Feasibility Specification" (Sub phase of "Top Down" Phase)** which is changed or developed by the developer during the time of **"Load, Stress and Security"** testing at that time if any major mistake will come due to that changes, then it will found and rectified also fulfill the criteria of **"Verification and Validation"**. Hence it will improve the quality of the product and reduce the percentage of rejection from **"User acceptance Testing (UAT)"**.

In the proposed diagram there are two sections one is **"Top Down"** and another is **"Bottom Up"**. **"Top Down"** section starts with **"Requirement Analysis"** where first customer requirement will analysis. Whether the said requirement is feasible or not and how many percentage of the **"Requirement"** will feasible it can analysis in the **"Feasibility Specification"** phase. According to the feasible requirement it will prepare **"Technical Specification"**. **"Technical Specification"** is a part of **"Design"**. **"Technical**

Specification" team will decide what type of interface is required to build the software. Some of the experts will review whatever the **"Technical Specification"** has selected for this purpose that is correct or not. **"Design"** is the heart of **"Software Development"**. It has a two part first one is to design **"Architecture"** of the software and second one is to design **"Logical Structure"** of the said **"Architecture"**. Some of the experts will review that whatever the **"Design"** has prepared for this software that is correct or not. In the **"Coding"** phase coder will implement the **"Logical Structure"** of the **"Software Architecture"** into machine readable format. Some of the Technical expert will review the **"Coding"** which has developed by developer.

Before send the said coding for testing purpose the developer must check the said **"Coding"** in the **"Unit Testing"** phase. There after tester will test each separate module whatever the developed by developer in the **"Module Testing"** section. After completion of testing of each **"Module"**, tester will integrate all modules and re-tested it in the **"Integrated Testing"** section. In the **"Regression Testing"** tester must check the previous version's development & output with present version's development & output and try to find out bugs and errors. There after tester must test the **"Load, Stress & Security" testing** of the software which is developed by the developer in order to check the systems compliance to various security threats. And finally tester must check according to the **"Feasible Requirement"** whatever developer has developed that satisfies the requirement or not. If not then it will further send to the **"Coding"** phase and if satisfy then the said product will send for **"User Acceptance Testing"**.

Figure of Proposed Advanced Model is as follows:

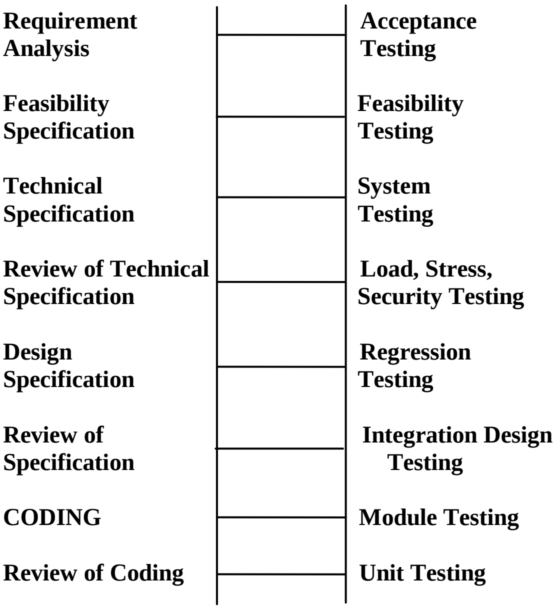

Fig 3: D-LADDER MODEL
***(© All Rights Reserved)**

4. ARCHITECTURE OF THE NEW (D-LADDER) MODEL

In the below architectural diagram (Fig 4) shows that how **"Validation & Verification"** is worked simultaneously. In every time of software development there are not necessary that first complete **"Verification"** then **"Validation"** will complete. If **"Verification & Validation"** will combine every time then it will get better output.

According to the architectural diagram first steps is **"Requirement Analysis"**. According to the customer requirement first steps is to evaluate the requirement of customer thoroughly. For e.g suppose an Insurance company want to develop their own "Core Insurance" software.

"Requirement Analysis" team will first split-up the requirement of the customer along with time frame and delivered it to the **"Feasibility Specification Team"**, that whatever they split up the requirement within the schedule time frame is that possible for complete development of the software within schedule time frame or not. Also clarify the detail information against each module.

"Feasibility Specification" team will first evaluate the feasibility of the analysis which is analyzed by the "Requirement Analysis Team" according to the requirement of the customer. If the analysis is not feasible or further analyses is required from the customer in concern with the product then send it to the **"Requirement Analyses Team"**. For e.g suppose a Insurance company want to deposit their premium directly from their software to any specific Bank Account through "Net Banking" during the time of premium collection. This type of requirement is a good requirement but it is not feasible to develop within in a fixed schedule time frame. Because "Banking" software is the different software and supposes to develop by different company and it is not a feasible requirement to complete total development within a fixed time frame according to requirement of the customer.

The entire software development process stands on the **"Verification & Validation"**. If **"Verification"** has done thoroughly and simultaneously if **"Validation"** doesn't have done **"Validation"**, then entire software will developed but it may not run properly in the client location. Hence total work will goes in to the water. Here it can assume that **"Requirement Analysis"** is the **"Verification"** stage and also assume that **"Feasibility Specification"** is the **"Validation"** Stage. So if combine **"Verification & Validation"** simultaneously means **"Requirement Analysis"** & **"Feasibility Specification"** then a complete requirement will get before start of any development until it will create lots of problem during the time **"User Acceptance Testing"**.

Same thing is also applicable for **"Technical Specification"**, **"Review of Technical Specification"**, **"Design Specification"** & **"Review of Design Specification"**. Especially in **"Design"** phase because **"Design"** is a combination of **multi-features** element like **"Architecture Design"**, **"Logical Design"** etc. If the **"Design"** will not review then it is not understandable the said design is correct or not for development. Sometimes it happens that after completion of entire development it has discovered that entire design was wrong. It is the vital stage of review **"Design Specification"** before send it to the **"Coding"** stage.

Coding is one of the major stages of software development. According to the **"Design"** coder generates the code and shape the software. But major problem is that if the code will not review the said item it may create a several problems in the next stage. And this **“Software Testing”** comes only because of **“Coding”**. And **“Coding”** and **“Software testing”** is co-related with each other as **“Verification & Validation”**.

After completion of coding **“Review of Coding”** is mandatory because it optimize the complexity of the code, tune the code and finally it will help to improve the performance of the product.

A new testing phase have introduced in this architecture that is **“Feasibility Testing”**. Question can arise why **“Feasibility Testing”** is required? According to the original definition **“System Testing”** covers two areas one is **“Functional Requirement”** and another is **“Business Requirement”**. In the **“Functional Requirement”** it has covered only the consolidated output of the **“Integration Testing”**. But it has not covered the outcome of **“Regression Testing”** also **“Load, Stress, Security Testing”**. These steps is also a major steps of testing so whatever changes will come during the time of **“Regression Testing”** & **“Load, Stress, Security Testing”** according to the definition it will not reflect in the **“System Testing”**. For the said reason **“Business Requirement”** testing will not applicable in this stage according to the above architecture.

According to this architecture **“Regression Testing”** will cover to take consolidated output of **“Integration Testing”**. After completion of **“Regression Testing”** one major thing still pending that is **“Load, Stress & Security”** testing. According to the above architecture **“System Testing”** will test only **“Requirement Based”** testing which comes according to the test case of actual requirement. And **“Feasibility Testing”** will test or compare the outcome has come from **“System Testing”** that is matched with actual **“Feasibility Specification”** or not. After tested the software thoroughly if the present output is matched according to the **“Feasibility Specification”** then send it to the **“User Acceptance Testing”**. So many times the entire software has returned from **“User Acceptance Testing”** reason behind is that before send it to the **“User Acceptance Testing (UAT)”** the said software has not checked actual requirement with present output.

4.1 Differences With Exting Models

A) The major difference between existing **“V-Model”** and new model **(D-LADDER)** is that in the **“V-Model”** first **“Verification”** have done then **“Validation”** has done. But in the new model **(D-LADDER) “Verification & Validation”** have done simultaneously in each phase.

B) In the **“V-Model”** testing phase have described elaborately but not systematically. Some major testing techniques have not taken in the **“V-Model”**. But in the new model **(D-LADDER)** have designed according to the demand of present **“Software Industry”**. All major testing technique has introduced and that technique is required every time during the period of **“Software Development”** for e.g **“Regression Testing” “Load, Stress, Security Testing”.**

C) A new technique has introduced in the new model i.e **“Feasibility Specification”** and **“Feasibility Testing”**. Such kind of technique is not available in the **“V-Model”** and other existing model.

D) Review of every phase is the **KEY** of this new model. In the **“V-Model”** and other existing model there is no scope of review in any phase. So, more bugs & errors will be optimized to use this new model **(D-LADDER)** than existing model and also get a maximum bug less or error less software than existing one.

E) This new model **(D-LADDER)** is applicable for both small (easy or complex) and Large (easy or complex) software development. But **“V-Model”** logically supports small projects.

F) This new model **(D-LADDER)** model will reduce the dependency on **“User Acceptance Testing”**.

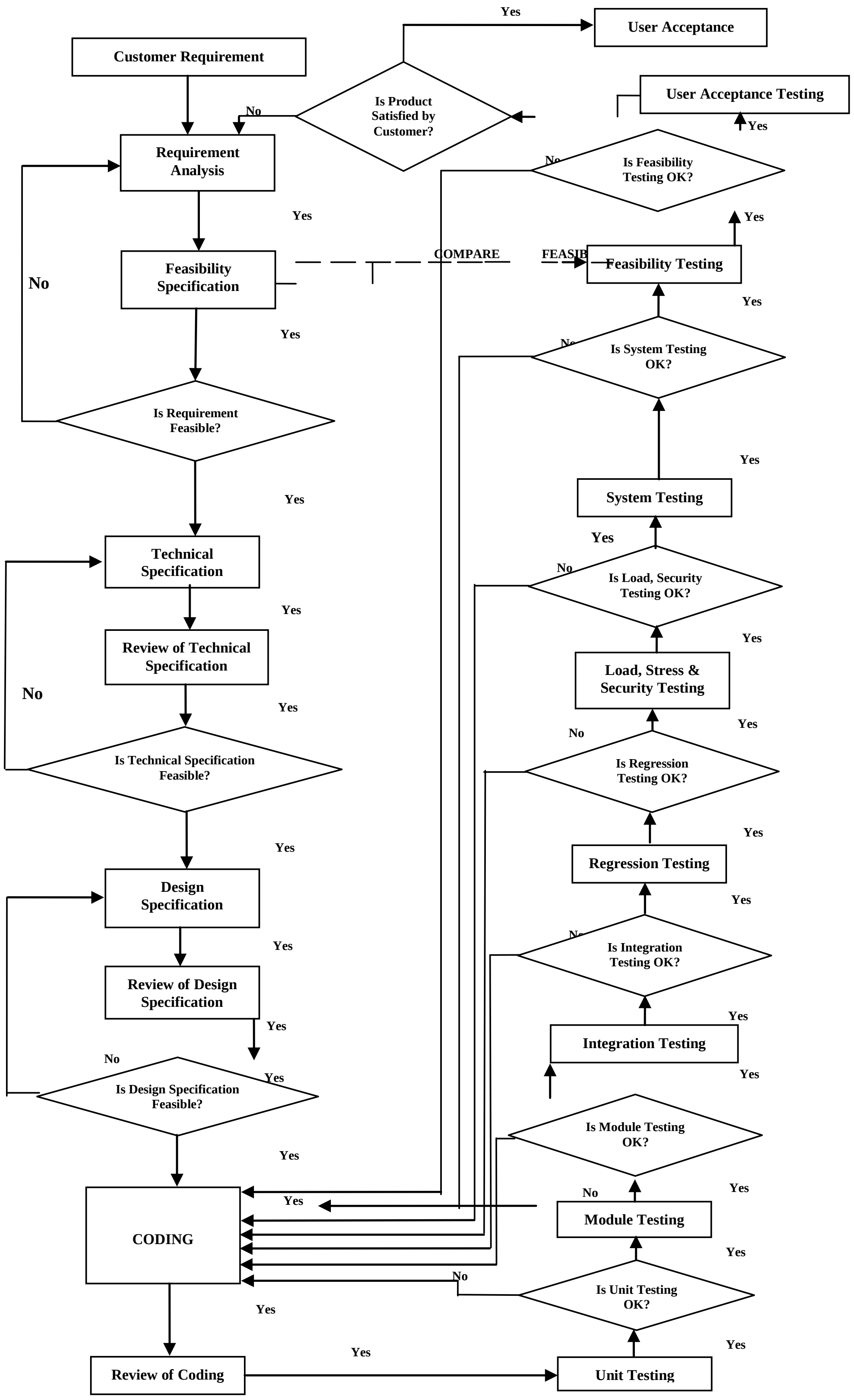

Fig 4: Architecture Design Of D-Ladder Model

5. CONCLUSION

In this research and main things observed that in the present existing models specially “V-Model” is applicable for “Product” based software. Generally in the “Product” based software modification does not depends on the customer it depends on the source. So in the product based software “V-Model” is very much applicable. But in the “Project” based software specially in the maintenance of the project where modification depends on the customer in that case “V-Model” is not suitable specially in today’s complex requirement of the software.

This new model will provide a guideline to the software industry where technology keeps on changing every day and according to the requirement of the customer software it is going to complex. Thus this new model is very efficient for software testing management as well as a new testing technique have introduced in this model which will enhance the “Software Testing” as well as it will more reduces the bugs and errors from the software.

6. REFERENCES

[1] IEEE. 1993. IEEE Standard for Software Maintenance. IEEE Std 1219-1993. Institute of Electrical and Electronics Engineers, inc.,New York, NY.

[2] IEEE Standard for Software Verification and Validation Plans (Reaff.1992). IEEE Std 1012-1986.

[3] IEEE Standard Glossary of Software Engineering Terminology. IEEE Std 610.12-1990.

[4] JJ Kuhl, "Project Lifecycle Models: How They Differ and When to Use Them",2002 .

[5] Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[6] Fundamental of Software Engineering: Rajib Mall.

[7] Boehm, B. and Port, D., Educating software engineering students to manage risk, Proc. Int. Conf on Software Engineering, 2001.

[8] Ludewig, J., Software Engineering in the year 2000 minus and plus ten, in R. Wilhelm (ed.): Informatics:10 years back, 10 years ahead, Springer- Verlag,Berlin, Heidelberg, 2001, pp. 102 - 111.

[9] JJ Kuhl, "Project Lifecycle Models: How They Differ and When to Use Them",2002

[10] Rlewallen, "Software Development Life Cycle Models", 2005.