Volume 1, No. 2, July-August 2010

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Variations in V Model for Software Development

Mr. Ganesh B. Regulwar*
Lecturer, Info-Tech Dept
Jawaharlal Darda Institute of Engg. & Tech.,
Yavatmal ,Maharashtra, INDIA
ganeshregulwar@gmail.com

Mr. P. M. Jawandhiya
HOD & Asst Prof CSE Dept
J.D.I.E.T., Yavatmal
Maharashtra, INDIA
pmjawandhiya@rediffmail.com

Mr. V.S. Gulhane
Asst Prof in CMPS & IT
Sipna's CET, Amravati,
Maharashtra, INDIA
v_gulhane@rediffmail.com

Mr. R. M. Tugnayat
Asst.Prof. & HODIT,
J.D.I.E.T.,Yavatmal
Maharashtra, INDIA
rmtugnayat@gmail.com

Dr. P .R. Deshmukh
Professor& HOD CMPS &IT
Sipna's CET, Amravati
Maharashtra, INDIA

Abstract: V Model Represents one-to-one relationship between the documents on the left hand side and the test activities on the right. This is not always correct. System testing not only depends on Function requirements but also depends on technical design, architecture also. Couple of testing activities is not explained in V model. This is a major exception and the V-Model does not support the broader view of testing as a continuously major activity throughout the Software development lifecycle. Paul Herzlich introduced the W-Model approach in 1993. The W-Model attempts to address shortcomings in the V-Model. Rather than focus on specific dynamic test stages, as the V-Model does, the W-Model focuses on the development products themselves. In its most generic form, the W-Model presents a standard development lifecycle with every development stage mirrored by a test activity. On the left hand side, typically, the deliverables of a development activity (for example, write requirements) is accompanied by a test activity test the requirements and so on. If your organization has a different set of development stages, then the W-Model is easily adjusted to your situation. The important thing is this: the W-Model of testing focuses specifically on the product risks of concern at the point where testing can be most effective. The main contribution in this paper is Sawtooth model and Sharktooth model. Sawtooth model is another variation of the V model. The Sawtooth model is actually an extension of the V-model. The only difference between the Sawtooth and the V-model is that prototypes are created and shown to the client for validation. Sharktooth model is more detailed view of the sawtooth model. In contrast to the sawtooth model the sharktooth model puts the manager into account. By presenting the manager a certain abstraction it also introduces new activities.

Keywords*:* Quality software through V Model; Overcome W Model on V Model; Sawtooth & Shartooth model

I. INTRODUCTION

The verification and validation model commonly known as "V Model". It is considered to be an extension of the Waterfall model. This is because just like the waterfall model, it's a well structured method in which the different phases progress in a sequential or linear way. That means each phase begins only after the completion of the previous phase.An important aspect of this model is that testing activities like planning, test designing happens well before coding. The advantage is that it saves ample amount of time and since the testing team is involved early on, they develop a very good understanding of the project at the very beginning. The V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing. The horizontal and vertical axes represents time or project completeness (left-to-right) and level of abstraction (coarsest-grain abstraction uppermost), respectively.Early test preparation finds faults in baselines and is an effective way of detecting faults early. This approach is fine in principle and the early test preparation approach is always effective. However, there are two problems with the V-Model as normally presented. Thus, Paul Herzlich introduced the W-Model approach in 1993. The W-Model attempts to address shortcomings in the V-Model. Rather than focus on specific dynamic test stages, as the V-Model does, the W-Model focuses on the development products themselves. Essentially, every development activity that produces a work product is shadowed by a test activity. The purpose of the test activity specifically is to determine whether the objectives of a development activity have been met and the deliverable meets its requirements.

II. STRUCTURE OF V-MODEL

A. Requirements analysis (SRS)

In the Requirements analysis phase, the requirements of the proposed system are collected by analyzing the needs of the user(s). This phase is concerned about establishing what the ideal system has to perform. However it does not deter-

mine how the software will be designed or built. Usually, the users are interviewed and a document called the user requirements document is generated.

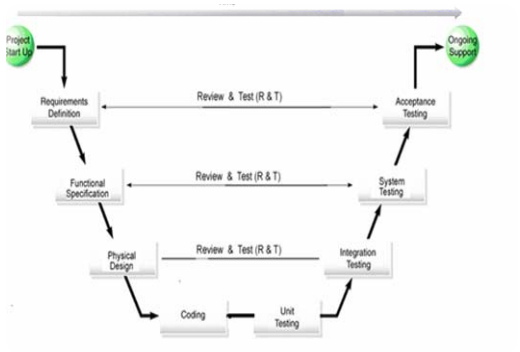

Figure 1: V Model

The user requirements document will typically describe the system's functional, physical, interface, performance, data, security requirements etc as expected by the user. It is one which the business analysts use to communicate their understanding of the system back to the users. The users carefully review this document as this document would serve as the guideline for the system designers in the system design phase. The user acceptance tests are designed in this phase.

B. System Design/ Functional Specifications

Systems design is the phase where system engineers analyze and understand the business of the proposed system by studying the user requirements document. They figure out possibilities and techniques by which the user requirements can be implemented. If any of the requirements are not feasible, the user is informed of the issue. A resolution is found and the user requirement document is edited accordingly.

The software specification document which serves as a blueprint for the development phase is generated. This document contains the general system organization, menu structures, data structures etc. It may also hold example business scenarios, sample windows, reports for the better understanding. Other technical documentation like entity diagrams, data dictionary will also be produced in this phase. The documents for system testing are prepared in this phase.

C. Architecture Design/ Physical Design

The phase of the design of computer architecture and software architecture can also be referred to as high-level design. The baseline in selecting the architecture is that it should realize all which typically consists of the list of modules, brief functionality of each module, their interface relationships, dependencies, database tables, architecture diagrams, technology details etc. The integration testing design is carried out in the particular phase.

D. Module Design / Coding

The module design phase can also be referred to as low-level design. The designed system is broken up into smaller units or modules and each of them is explained so that the programmer can start coding directly. The low level design document or program specifications will contain a detailed functional logic of the module, in pseudo code:

- database tables, with all elements, including their type and size
- all interface details with complete API references
- error message listings
- Complete input and outputs for a module.

The unit test design is developed in this stage.

E. Unit Testing

In the V-model of software development, unit testing implies the first stage of dynamic testing process. According to software development expert Barry Boehm, a fault discovered and corrected in the unit testing phase is more than a hundred times cheaper than if it is done after delivery to the customer.

It involves analysis of the written code with the intention of eliminating errors. It also verifies that the codes are efficient and adheres to the adopted coding standards. Testing is usually white box. It is done using the Unit test design prepared during the module design phase or while coding. This may be carried out by software developers.

F. Integration Testing

In integration testing the separate modules will be tested together to expose faults in the interfaces and in the interaction between integrated components. Testing is usually black box as the code is not directly checked for errors.

G. System Testing

System testing will compare the system specifications against the actual system. After the integration test is completed, the next test level is the system test. System testing checks if the integrated product meets the specified requirements. Why is this still necessary after the component and integration tests? The reasons for this are as follows:

1. In the lower test levels, the testing was done against technical specifications, i.e., from the technical perspective of the software producer. The system test, though, looks at the system from the perspective of the customer and the future user. The testers validate whether the requirements are completely and appropriately met.
2. Example - The customer (who has ordered and paid for the system) and the user (who uses the system) can be different groups of people or organizations with their own specific interests and requirements of the system.
3. Many functions and system characteristics result from the interaction of all system components; consequently, they are only visible on the level of the entire system and can only be observed and tested there.

H. User Acceptance Testing

Acceptance testing is the phase of testing used to determine whether a system satisfies the requirements specified in the requirements analysis phase. The acceptance test design is derived from the requirements document. The main purpose of acceptance testing is to verify the system or changes according to the original needs. The acceptance test phase is the phase used by the customer to determine whether to accept the system or not. The following description is unacceptable in and overview article Acceptance testing:

- To determine whether a system satisfies its acceptance criteria or not.
- To enable the customer to determine whether to accept the system or not.

Procedures for conducting the acceptance testing:

1. Define the acceptance criteria:
   a. Functionality requirements and Performance requirements.
   b. Interface quality requirements.
   c. Overall software quality requirements.

2. Develop an acceptance plan:
   a. Project description.
   b. User responsibilities.
   c. Acceptance description.
   d. Execute the acceptance test plan.
   e. To develop

III. ADVANTAGES OF V MODEL

1. Proactive defect tracking i.e defects are found at early stages even may be in the development phase before application is tested.
2. It avoids the downward flow of the defect
3. It reduces the cost for fixing the defect since defects will be found in early stages
4. The users of The V-Model participate in the development and maintenance of The V-Model. A change control board publicly maintains the V-Model. The change control board meets once a year and processes all received change requests on The V-Model.
5. At each project start, the V-Model can be tailored into a specific project V-Model, this being possible because the V-Model is organization and project independent.

IV. DISADVANTAGES OF V MODEL

1. The biggest disadvantage of V-model is that it's very rigid and the least flexible. That is if any changes happen mid way, not only the requirements documents but also the test documentation need to be updated.
2. It needs lot of resources and money, so it can be implemented by only some big companies.
3. The organization and execution of operation, maintenance, repair and disposal of the systems are not covered by the V-Model. However, planning and preparation of a concept for these tasks are regulated in the V-Model.
4. It is not proposed for short term projects as it requires reviews at each stage.

V. MODEL TO W- MODEL

The V-model promotes the idea that the dynamic test stages (on the right hand side of the model) use the documentation identified on the left hand side as baselines for testing. The V-Model further promotes the notion of early test preparation.

Early test preparation finds faults in baselines and is an effective way of detecting faults early. This approach is fine in principle and the early test preparation approach is always effective. However, there are two problems with the V-Model as normally presented.

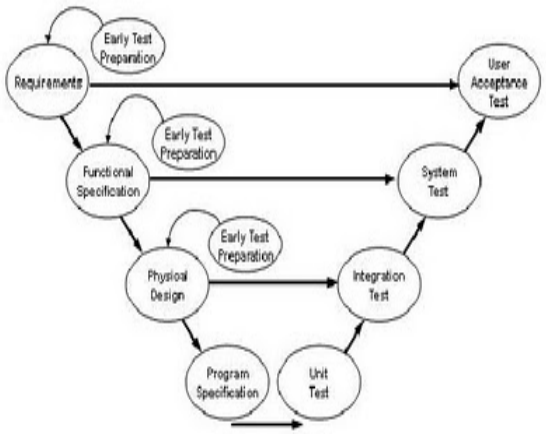

Figure 2: The V-Model with early test preparation.

There is rarely a perfect, one-to-one relationship between the documents on the left hand side and the test activities on the right. For example, functional specifications don't usually provide enough information for a system test. System tests must often take account of some aspects of the business requirements as well as physical design issues for example. System testing usually draws on several sources of requirements information to be thoroughly planned. V-Model has little to say about static testing at all. The V-Model treats testing as a back-door activity on the right hand side of the model. There is no mention of the potentially greater value and effectiveness of static tests such as reviews, inspections, static code analysis and so on. This is a major omission and the V-Model does not support the broader view of testing as a constantly prominent activity throughout the development lifecycle. Thus, Paul Herzlich introduced the W-Model approach in 1993. The W-Model attempts to address shortcomings in the V-Model. Rather than focus on specific dynamic test stages, as the V-Model does, the W-Model focuses on the development products themselves. Essentially, every development activity that produces a work product is shadowed by a test activity. The purpose of the test activity specifically is to determine whether the objectives of a development activity have been met and the deliverable meets its requirements. In its most generic form, the W-Model presents a standard development lifecycle with every development stage mirrored by a test activity. On the left hand side, typically, the deliverables of a development activity (for example, write requirements) is accompanied by a test activity test the requirements and so on. If your organization has a different set of development stages, then the W-Model is easily adjusted to your situation. The important thing is this: the W-Model of testing focuses specifically on the product risks of concern at the point where testing can be most effective.

VI. STRUCTURE OF W- MODEL

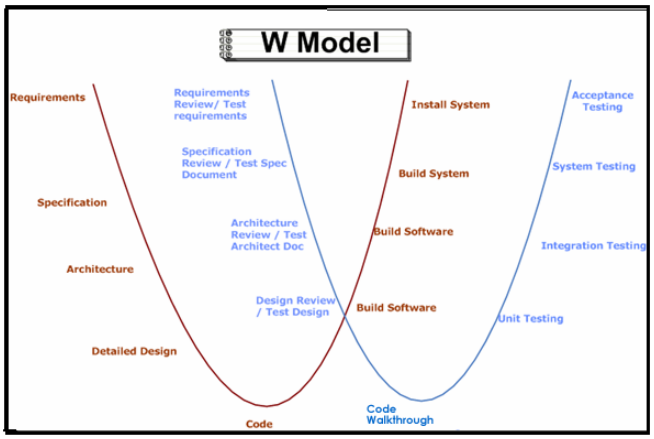

Figure 3: W Model

We already discuss that V-model is the basis of structured testing. However there are few problems with V Model. V Model Represents one-to-one relationship between the documents on the left hand side and the test activities on the right. This is not always correct. System testing not only depends on Function requirements but also depends on technical design, architecture also. Couples of testing activities are not explained in V model. This is a major exception and the V-Model does not support the broader view of testing as a continuously major activity throughout the Software development lifecycle. The W model is more accurately described as a variation of the V model. In W Model, those testing activities are covered which are skipped in V Model.

Like the V model, the W model emphasizes the correlation between development phases (requirements analysis, system design, and program code) and testing phases (acceptance testing, system testing, and unit testing). The W model includes a prototyping step, which is conducted immediately following the architectural design of the overall system, and an implementation phase that is realized as a series of individual builds as opposed to a staggered incremental model, which is considered to be more risky. For each build, construction follows a 4-step process:

1. Digitalization of materials and design of art work.
2. coding of the program,
3. unit testing of models and integration with previous builds, and
4. Validation of the product with the customer.

The 'W' model illustrates that the Testing starts from day one of the of the project initiation. If you see the below picture, 1st "V" shows all the phases of SDLC and 2nd "V" validates the each phase. In 1st "V", every activity is shadowed by a test activity. The purpose of the test activity specifically is to determine whether the objectives of that activity have been met and the deliverable meets its requirements. W-Model presents a standard development lifecycle with every development stage mirrored by a test activity. On the left hand side, typically, the deliverables of a development activity (for example, write requirements) is accompanied by a test activity test the requirements and so on.

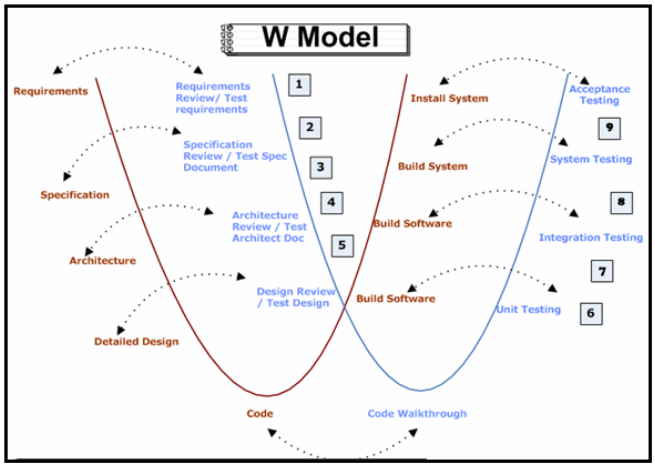

Figure 4: W Model where Each phase is verified/validated. Dotted arrow shoes that every phase in brown is validated/tested through every phase in sky blue.

Now, in the above figure,

- Point 1 refers to - Build Test Plan & Test Strategy.
- Point 2 refers to - Scenario Identification.
- Point 3, 4 refers to – Test case preparation from Specification document and design documents
- Point 5 refers to – review of test cases and update as per the review comments.
- So if you see, the above 5 points covers static testing.
- Point 6 refers to – Various testing methodologies (i.e. Unit/integration testing, path testing, equivalence partition, boundary value, specification based testing, security testing, usability testing, performance testing).
- After this, there are regression test cycles and then User acceptance testing.

VII. ADVANTAGES OF W MODEL

1. **I**n the W-model the importance of the tests and the ordering of the individual activities for testing are clear. Parallel to the development process, in a tighter sense, a further process - the test process - is carried out. This is not first started after the development is complete.
2. The strict division between constructive tasks on the left-hand side and the more destructive tasks on the right-hand side that exists in the V-model is done away with. In the W-model it is clear that such a division between tasks is not sensible and that a closer co-operation between development and testing activities must exist. From the project outset onwards the testers and the developers are entrusted with tasks and are seen as an equal-rights partnership. During the test phase, the developer is responsible for the removal of defects and the correction of the implementation. The early collaboration and the tight co-operation between the two groups can often in practice avoid conflict meetings
3. The W-model becomes closer to practice when the test expenditure is given 40% and more. The model clearly emphasizes the fact that testing is more than just construction, execution and evaluation of test cases.

VIII. DISADVANTAGES OF W MODEL

1. Models simplify the real facts. In practice there are more relations between the different parts of a development process. However, there is a need for a simple model if all people involved in a project are to accept it. This is also a reason why the simple V-model so frequently used in practice.
2. The models of software development presented do not clarify the expenditure needed for resources that need to be assigned to the individual activities. Also in the W-model it appears that the different activities have an equal requirement for resources (time, personnel, etc.) In practice this is certainly not the case. In each project the most important aspects may vary and so therefore the resource allocation is unlikely to be equal across activities. For highly critical applications the test activities certainly have higher weighting or at least equal weighting with other activities.

IX. SAWTOOTH MODEL

The W model helps to overcome these limitations by advocating a prototyping step following architectural design and the gradual delivery of the system via incremental builds. To emphasize the increased interaction with the client/user, the depiction of the W model includes a dashed line with life cycle phases above the line indicating heavy client involvement and phases below the line denoting little or no client interaction.

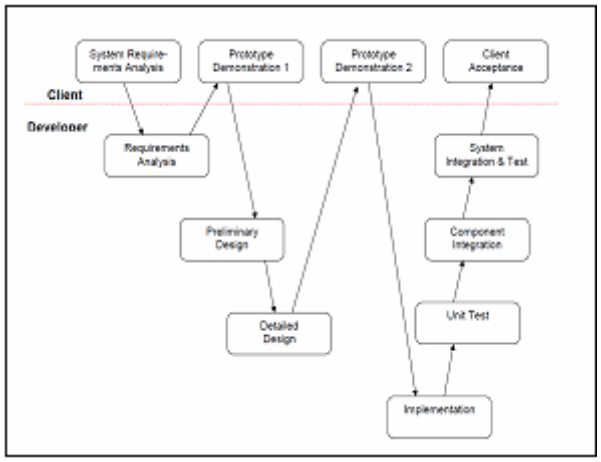

Figure 5: Sawtooth Model

This part of the W software life cycle model is similar to the sawtooth model, another variation of the V model. The Sawtooth model is actually an extension of the V-model. The only difference between the Sawtooth and the V-model is, that prototypes are created and shown to the validations in between the critical phases. The client is involved which is supposed to guarantee that the the client for validation. The prototypes are present right after the analysis phase and in between the design and the implementation phase. By making sure that the client is getting an insight into the progress, checkpoints should ensure that development is going in the right direction. The major benefit comes from project will become a success.

The huge problem is the time consumption with presenting the prototypes to the client. Depending on the complexity of the project the time might be considerable.

X. SHARKTOOTH MODEL

This is another, more detailed view of the sawtooth model. In contrast to the sawtooth model the sharktooth model puts the manager into account. By presenting the manager a certain abstraction it also introduces new activities.

As it's pretty similar to the sawtooth model it has all of the advantages and disadvantages of the sawtooth model.

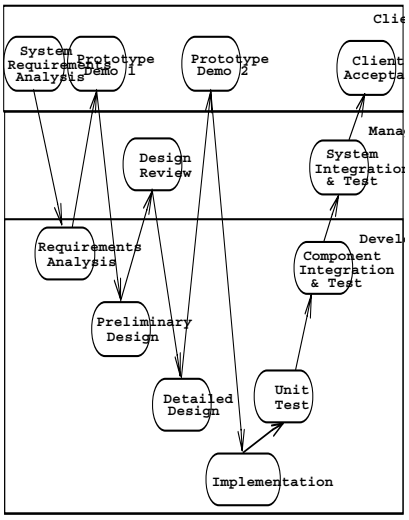

Figure 6: Shark tooth model with two prototype demonstrations

XI. CONCLUSION

As mentioned before: there is neither a worst nor best model. And there is also no generally good or bad approach to managing software development. Each model has its own advantages and disadvantages. And each model is meant to deal with (sometimes only slightly) different aspects than the other. Which of these models should be applied to a certain problem depends on the number of persons involved, the complexity of the software to be developed and the goals and even more aspects. The above introduction might already help evaluating the correct choices for each case. Most of the time the model is chosen indirectly by the way the software is supposed to be delivered and developed. If you really have the chance to chose a model for a future project, it might be best to not only think about the software lifecycle at hand but also about the tools, frameworks, and knowledge in need to support this kind of development.

The major contribution of this research is the design of a specialized software process model i.e W model, Sawtooth Model , Shark-tooth Model.

XII. REFERENCES

[1] sqa.fyicenter.com/.../Software-Development- models/ software_ Development_ Models_ W_ Model.html
[2] www.ics.uci.edu/~wscacchi/Papers/.../Process-Models-SE-Encyc.pdf
[3] http://www.computer.org/portal/web/csdl/proceedings/
[4] http://gerrardconsulting.com/?q=node/531
[5] http://www.augustana.ab.ca/~mohrj/courses/2000.winter/csc220/lecture_notes/lifecycle.html
[6] www.scribd.com/.../Advantages-and-Disadvantages-of-v-Model
[7] wiki.answers.com/Q/V_ model_ disadvantages.
[8] en.wikipedia.org/wiki/V-Model

[9] The W Life Cycle Model and Associated Methodology for Corporate Web Site Development - By Linda B. Sherrell , Department of Computer Science, University of West Florida , Lei-da Chen, College of Business Administration Creighton University. lchen@creighton.edu

[10] Software Life-cycle ,Peter Wittmann, peter@wittmannclan.com, www.wittmannclan.com

[11] Software Life Cycles, Prof. Bernd Brügge, Ph.D ,Technische Universität München, Institute für Informatik, Lehrstuhl für Angewandte Softwaretechnik , http:// www. bruegge. in.tum.de

[12] The Software Development Life Cycle (SDLC), For Small to Medium Database Applications.

[13] The W-MODEL – Strengthening the Bond Between Development and Test Andreas Spillner, University of Applied Sciences Bremen, Germany.