Sumerianz Journal of Scientific Research, 2018, Vol. 1, No. 1, pp. 22-27
ISSN(e): 2617-6955, ISSN(p): 2617-765X
Website: https://www.sumerianz.com

Original Article Open Access

The Development of Life Cycle Technique for Software Verification and Validation

Mohammed Nazeh Abdulwahid
Centre of Postgraduate Studies, LUCT, Malaysia

Abstract

In the century of the Internet revolution and technology innovation, that enter in every part of our life , from simple microwave applications used at home to surgeries and critical applications in hospital, Hence came the need to make an assured quality software product and systems . The quality of the software product is depends on the quality of the each phase of the development process used to create it. Verification and Validation techniques used across full phases of Software Development Life Cycle (SDLC) to ensure that the phase satisfy its requirement, error free, and correctly completed. Verification and Validation Techniques are set of testing and analysis activities by different methods. This paper described first the definitions of verification and validation concepts, followed by verification and validation role in each phase of Software Development Life Cycle SDLC), and finally the Methods of software verification and validation.

Keywords: Software verification and validation; Software development life cycle (SDLC); Methods of software verification and validation.

1. Introduction

The traditional development life cycle usually confines testing nearly at the end phases or before the operation phase, to determine if the proposed design meets the initial set of business goals. Testing may be repeated, specifically to check for errors, bugs and interoperability. This testing will be performed until the end user finds it acceptable, [1], which often , countless cases of software errors are found , that have proved costly, even dangerous and sometimes embarrassing [2]. Most of the projects start out this way end up with either a total failure or a tremendously expensive self-fulfilling prophecy and going to have a whole lot of de bugging to do [3]. The best protection against such failure is to verify and validate the correctness of each development phase and improve the overall quality of the software before it is used. And Verification and Validation (V&V) techniques are the prominent methodologies used to accomplish this.

Software verification and validation are software quality assurance activities that aim to ensure that the software system is developed according to a development process and meets the customer's needs. In other words, verification is about "are we building the product right", and validation is about "are we building the right product" [3]. In the following sections, we first provide definitions of verification and validation concepts, followed by verification and validation in the software lifecycle and finally formal verification and software testing techniques.

2. Software Verification and Validation Definition

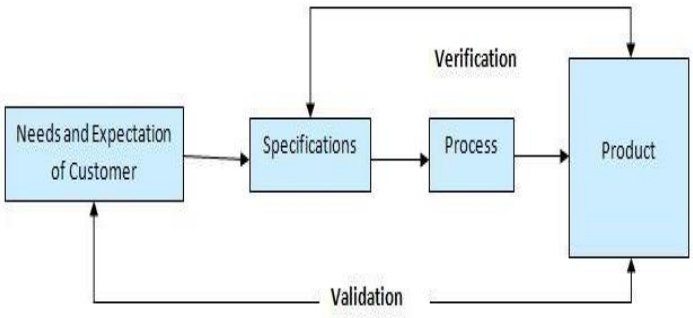
The IEEE Standard Glossary of Software Engineering Terminology' defines "verification" and "validation" as follows:

- Verification. The process of determining whether or not the products of a given phase of the software development cycle fulfill the requirements established during the previous phase.
- Validation. The process of evaluating software at the end of the software development process to ensure compliance with software requirements. [3]

From Wikipedia definition, Verification and validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it fulfills its intended purpose. These are critical components of a quality management system such as ISO 9000. [4] IEEE Std 1012TM-2012 is a process standard that defines the V&V processes in terms of specific activities and related tasks. The standard also defines the contents of the V&V plan (VVP), including example formats as the following brief:

Verification and validation (V&V) is a technical discipline of systems engineering. The purpose of V&V is to help the development organization build quality into the system during the life cycle. It performed at the level of the system, software element, or hardware element, or on any combination of these. Throughout this standard, V&V processes provide an objective assessment of products and processes throughout the life cycle. This assessment demonstrates whether the requirements are correct, complete, accurate, consistent, and testable [5]

Figure-1. Software Verification and validation



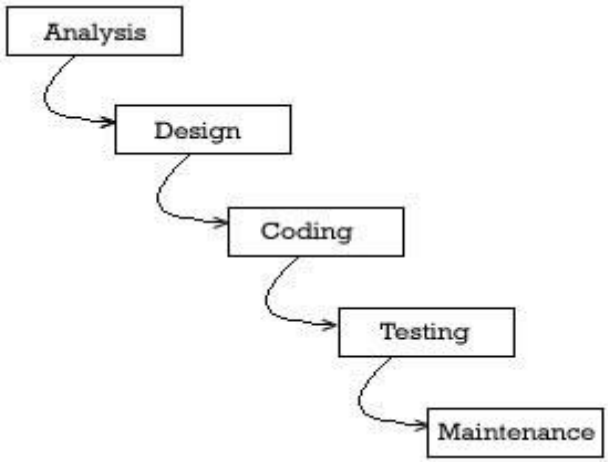
The V&V processes determine whether the development products of a given activity conform to the requirements of that activity, and whether the product satisfies its intended use and user needs. The determination includes the assessment, analysis, evaluation, review, inspection, and testing of products and processes. V&V is performed in parallel with all life cycle stages, not at their conclusion as the traditional software life cycle.

V&V is an extension of program management and systems engineering that employ a rigorous methodology to identify objective data and conclusions to provide feedback about quality, performance, and schedule to the supplier. This feedback consists of anomaly resolutions, performance improvements, and quality improvements not only for expected operating conditions but also across the full spectrum of the system and its interfaces. Early feedback results allow the development organization to modify the products in a timely fashion and thereby reduce overall project and schedule impacts. Without a proactive approach, the anomalies and associated system changes are typically delayed to later in the program schedule, resulting in greater program costs and schedule delays [6].

3. Verification and Validation in Software Development Lifecycle

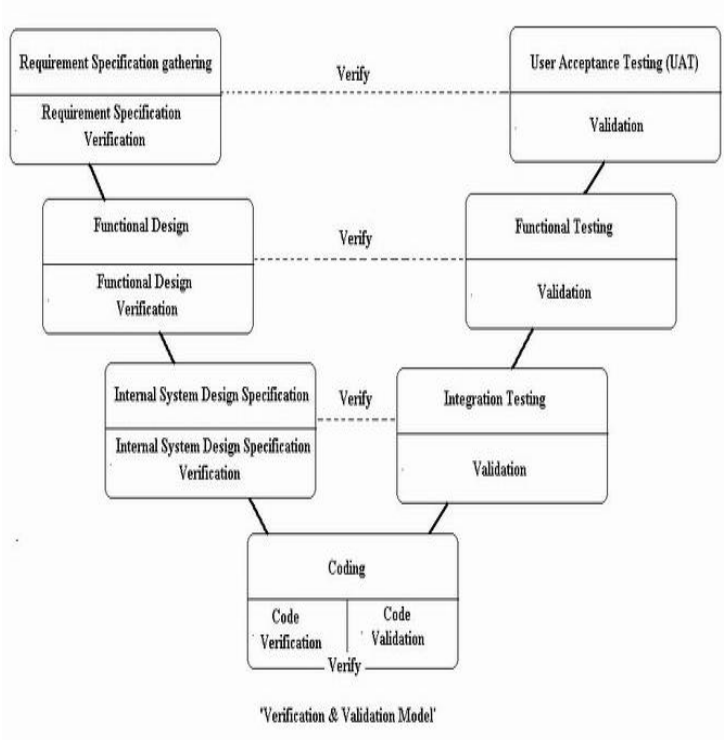
Testing is the only verification technique used to determine the adequacy of the software in the traditional development life cycle, and it usually confines at phases immediately prior to operation and maintenance as shown in figure 2. That result in an error is found at letter phase of development and require higher cost of its correction[7]. That's way verification should not be isolated to a single phase in the development process but should be incorporated into each phase of development life cycle to reduce the cost and increase the quality.

Figure-2. Software Development Lifecycle



Barry Boehm [BoEH77] has stated that one of the most prevalent and costly mistakes made in software projects today is deferring the activity of detecting and correcting software problems until late in the project, [8] as shown in fig.2. The primary reason for early investment in verification activity is to catch potentially expensive errors early before the cost of their correction escalates. And the incorporated of verification and validation into each phase is not enough to success of performing verification throughout the development cycle , [9] but also require existence of a clearly defined and stated product at each development stage (e.g., a requirement specification at the requirements stage) [10] . The more formal and precise the statement of the development product, the more amenable it is to the analysis required to support verification. Many of the new software development methodologies encourage a visible, analyzable product in the early development stages [9].

Figure- 3. Verification and validation Model



'Verification & Validation Model'

This section presents the role of verification and validation in the software lifecycle. That is, what is checked in each of the lifecycle phases and who performs the checking. Moreover, what are the techniques used to perform the checking.

3.1. Verification and Validation for the Requirements Phase

3.1.1. Requirement Analysis Phase

Verification and validation in the requirements phase aims at detecting errors in the requirements specification and the analysis models to ensure that the requirements are stated in clear and unambiguous terms, the verification and validation activity starts when the user group outlines the requirements for its software product, Then requirements are spelt out to both the design team as well as the V&V team. See fig.3 [11].

3.1.2. Specification Requirements Phase

Verification and validation in this phase aims at inspecting the SRS document to ensure that the specifications are traceable to the user's requirements documents. Software development group draws a Software Requirement Specification (SRS) document Based on the user group's requirements [12]. And the SRS document can take many forms, it can be in simple informal style to very formal specifications.

3.2. Verification and Validation for the Design Phase

Verification and validation in the design phase are aimed at assessing the correctness, consistency, and adequacy of the design with respect to the requirements and analysis models. Verification and validation activities in the design phase starts when the The design team develops a software design based on the SRS document, the verification and validation team ensures that the design is traceable to the SRS, the documentation of the design follows established standards and test cases have been formulated for testing the software product[13]. The design is documented using Structured Engineering methods, such as, Data Flow Diagrams, Structure Chart Diagrams, Data Structure Diagrams, Flow Charts, or Object Oriented methods, such as, Use Case Diagrams, Class Diagrams, Sequence Diagrams and Deployment Diagrams. The Verification and validation team in the design phase use review, inspection, walkthrough, formal verification, and prototyping techniques.

3.3. Verification and Validation for the Coding Phase

Verification and validation for the coding phase that occurs during the construction stage of development aimed to ensure that the source code complies with the organization’s coding standards, implements the required functionality, satisfies performance, real time and security requirements, and properly handles exceptional situations. The design team codes the software Based on the Software Design Document (SOD), then the verification and validation team inspects the code to ensures that the code is traceable to the SOD, standard practices are followed in implementing and documenting the code, and tests are conducted to validate the code with requirements[13].

Desk checking, code review, inspection, and walkthrough are commonly referred to as verification [14] and static validation methods while testing is commonly referred to as the dynamic validation method[8]. All these are commonly used in the coding phase.

3.3.1. Testing Phase

During the system testing phase, the software system is integrated with other systems and tested against the software system requirements [13], the verification and validation team then validates the integration of software developed with the system under development to ensure that the system satisfies the functional and non-functional requirements. In addition, the system satisfy the constraints stated in the requirements specification. And finally the end product of system testing is a system that is ready for deployment and acceptance test in the customer's target environment.

3.4. Verification and Validation for Operation and Maintenance Phase

Over 50 percent of the life-cycle costs of a software system are maintenance [15], As the system is used, it often requires modification either to correct errors or to augment its original capabilities. During the system maintenance phase, the software system after each modification must be retested. The V & V activity is termed *regression testing*. *The team* inspects those portions of the system affected by the modifications to ensure that changes at a given level will not affect other part of the system and then updating documentation at all levels below that change [8].

For example, a change at the design level requires design reverification, and unit retesting and subsystem and system retesting at the construction level. During regression testing, test cases generated during system development are reused or used after appropriate modifications. Since the materials prepared during development will be reused during regression testing, the quality of the test documentation will affect the cost of regression testing. If test data cases have been cataloged and preserved, duplication of effort will be minimized.

4. Methods of Software Verification and Validation

The reasons for the great success of verification and validation techniques is their uniform applicability at requirements, design, and coding phases. These techniques can be used without massive capital expenditure. However, to be most effective, they require a serious commitment and a disciplined application. Careful planning, clearly stated objectives, precisely defined techniques, good management, organized record keeping, and strong commitment are critical to successful validation [16].

4.1. Methods of Verification

4.1.1. Review

A software review process is an overview of the whole process to detect and records defects, the review process includes guidelines and specification which help the reviewer verify the quality and accuracy of assets according to their specific area of expertise.

4.1.2. Walkthrough

Walkthrough process are the most time consuming and most formal of the informal method involves larger team along with the author reviewing a work product. Walkthrough can be effectively used for communication between team. For Ex. Software development team reviewing a product before the final product is sent for approval by the customer.

4.1.3. Inspection

Inspection is a software verification method that is used to check how matches the conceptual model to the executable model but with no given solution and that gives the organization its own action plan to solve the system problem.

4.1.4. Audit

A software audit review is used to establish how well a model matches the guidelines that are set in place. Software audit occur both during, as well as at the end of the software development life cycle to ensure that all requirements have been fulfilled.

4.2. Methods of Validation

4.2.1. Unit Testing

Unit testing is a level of software testing where individual units or components of software are tested and comparing it with requirements to make sure that it performs as designed, and then make ready for integration. It is usually performed by the developer by using white boxing method.

4.2.2. Integrated Testing

It is the systematic combination and testing of software components as a group to insure consistency of component interfaces and to expose faults in interaction between integrated units. It is usually performed by the developer as well as an independent test team.

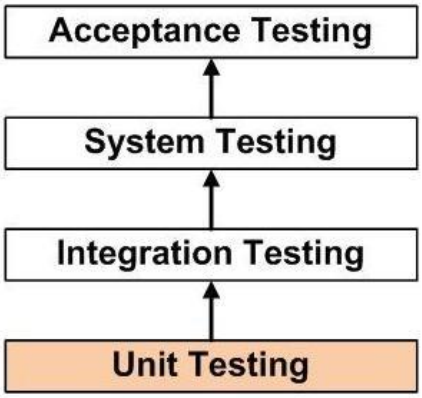
4.2.3. System Testing

It is an integrated software system where complete and integrated software is tested and compared with software system requirements. It includes functional and structural testing. It is usually performed by the independent test team by using black box testing method.

4.2.4. User Acceptance Testing

This software Testing method is an integrated hardware and software system. It is performed by the user with support of independent test team. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery. It is performed after system testing and before making the system available to the user.

Figure-4. Methods of validation



5. Conclusion

Through literature survey of many literature sources as the methods in this paper, we know that verification and validation are very important to be involved in system development life cycle SDLC, as they play essential role for the quality insurance and successful of software system. verification and validation through disciplined manual techniques, such as walkthroughs, reviews, and inspections, and with systematic combination and testing techniques, such as unit test, Integrated test, system teat, and user acceptance test of software components that applied to all stages in the life cycle, reduces the risk associated with any software project and helps in detection and correction of errors and mistakes, which are unknowingly done during the development process.

Errors Discover within the first stages of development will reduce the cost of correcting these errors, where if they remain undiscovered until later stage or until the development products the correction cost will escalates significantly, that way verification and validation should be part of each system development life cycle SDLC.

The quality of the software product is largely control by the quality of the process used to create and maintain it. Where the software process is the set of activities, methods and practices that guide people in production of the software, Software verification and validation performs tasks such as ,requirement analysis, requirement tracing, structure review, design review, code inspections and validation testing, That ensures a high quality product for mission critical applications software.

References

[1] Architects, I., 2012. "The seven phases of the system-development life cycle." Available: https://www.innovativearchitects.com/KnowledgeCenter/basic-IT-systems/system-development-life-cycle.aspx

[2] Hill, M., 1997. "Rogers pressman, Software engineering- A practitioner s approach." *Software Engineering-A practitioner S Approach,*

[3] Boehm, B. W., 1984. "Verifying and validating software requirements and design specifications." Los Alamitos 1.1, B. W. Ieee Software.

[4] Wikipedia, 2017. "Verification and validation." Available: https://en.wikipedia.org/wiki/Verification_and_validation

[5] Std, I., 2004. "Ieee standard for system and software verification and validation."

[6] Std, I., 2004b. "IEEE standard for system and software verification and validation."

[7] Willmer, H., 1985. "Literatur. Systematische software-qualitätssicherung anhand von qualitäts-und produktmodellen."

[8] Adrion, W. R., Branstad, M. A., and Cherniavsky, J. C., 1982a. "Validation, verification, and testing of computer software." *ACM Computing Surveys (CSUR),* vol. 14, pp. 159-92.

[9] Richards, A. M. A. B. A. J. C. C. W., 1982. "Validation, verification, and testing of computer software." *Computing Surveys,* vol. 14, p: 10

[10] Rakitin, S. R., 2001. *Software verification and validation for practitioners and managers*. Artech House, Inc.

[11] Sujatha, P., Sankar, G. V., Rao, A. S., and Satyanarayana, T., 2001a. "The role of software verification and validation in software development process." *IETE Technical Review,* vol. 18, p 7.

[12] Sujatha, P., Sankar, G. V., Rao, A. S., and Satyanarayana, T., 2001. "The role of software verification and validation in software development process." *IETE Technical Review,* vol. 18, pp. 23-26.

[13] Sujatha, P., Sankar, G. V., Rao, A. S., and Satyanarayana, T., 2001b. "The role of software verification and validation in software development process." *IETE Technical Review,* vol. 18, pp. 23-26.

[14] Fagan, M., 2002. *Design and code inspections to reduce errors in program development*. Springer: Software pioneers.

[15] Zelkowitz, M. V., 1978. "Perspectives in software engineering." *ACM Computing Surveys (CSUR),* vol. 10, pp. 197-216.

[16] Adrion, W. R., Branstad, M. A., and Cherniavsky, J. C., 1982b. "Validation, verification, and testing of computer software." *ACM Computing Surveys (CSUR),* vol. 14, pp. 159-192.