EFFECTIVE INVOLVEMENT OF VERIFICATION AND VALIDATION TO IMPROVE SOFTWARE QUALITY

Radha Krishna Rambola,SVKM'S NMIMS University, India (radhakrishna.rambola@nmims.edu)
Nonita Sharma, National Institute of Technology, Jalandhar, India (nonita@nitj.ac.in)
Alok Deepak

ABSTRACT

Quality planning includes planning for the processes to build and test software to create a right product. It may include software certification "validation" as well as finding conformance to the requirements defined by the processes and standards called "verification" Quality is an outcome of planned activities carried out in various phases of SDLC. This paper discusses about the overview of different methods and levels in software verification and validation. Only proper verification without the proper validation, and vice versa cannot ensure the good software product. Rather, it needs the combination of theboth.

I. INTRODUCTION

Verification and validation activities are the two branches of the software testing. They are complementary to each other and not substitutes of each other (Deshmukh O. et al.,2013). Verification and validation are completely dependent on each other. They are not mutually exclusive. Auditing, which works on sampling basis, verification and validation ensure that processes are followed correctly, the process definition is appropriate and process capability is
ensured by optimizing processes through feedback loop.

Verification

Verification (Deshmukh O. et al., 2013). is a disciplined approach to evaluate whether a software product fulfils the requirements or conditions imposed on them by the standards or processes. It is done to ensure that the processes and procedures defined by the customers and/or organization for development and testing are followed correctly.
Verification is also called as static technique as it does not involve the execution of code , program or work product.

Advantages:
a. Verification can confirm that the work product has followed the product correctly as defined by the organization or customer.
b. It can find defects in terms of deviation from standards.
c. Itcanreducethecostoffindingandfixingthedefectsaseachworkproductisreviewedandcorrectedfaster.

Disadvantages:
a. It cannot show whether the developed software is correct or not.
b. Actual working software may not be assessed by verification as it does not cover any kind of execution of a work product.

Prerequisites for verification:
a. Training required by people for conducting verification.
b. Standards, guidelines, and tools to be used during verification.
c. Verification Do and check process definition.

Validation

Validation (Rambola,2018) is a disciplined approach to evaluate whether a software product fulfils its specific intended use. It meant to validate the requirements as defined in requirement specification, ensure that the application as developed matches with the requirements defined, and is fit for the intended use. Validation steps may involve test bed preparation, test scenario definitions, test case definitions, test data definitions, test case execution, defect identification, retesting and regression testing
Validation is also called as "dynamic testing" as the application is executed during validation, with the intention to find defects.

Advantages:
a. Black box testing approach is used for the system, integration and acceptance testing.
b. Validation is the only way to show that the software developed by the following all processes of development is actually functioning as desired and is usable.

Disadvantages:
a. No amount of testing can improve that software does not have defects.
b. Forunit testing and module testing, stubs and drivers are needed to be designed and used during testing.

Pre-requisites for validation
a. Training required by people for conducting validation.
b. Standards, guidelines, and tools to be used during validation.
c. Validation Do and check process definition.

II. RELATEDWORKS

(Sethi A. et al., 17) has researched on various testing types and techniques. The testing techniques broadly classified into Static and Dynamic testing which is also referred as Verification and Validation respectively. In Static Testing the code is not executed so we doesn't require very highly skilled persons. It starts with the initial phase of SDLC. It includes Inspection, Walkthrough, Technical Reviews and Informal Reviews. In Dynamic Testing the behaviour of the software is analyzed. In this the code is executed and requires highly trained persons. Its types are Black Box Testing and White Box Testing. (H.Hedoo andKhandelwal ,2017) has researched on various aspects of dynamic testing and its techniques. In this it mainly focuses on Validation techniques.

Dynamic Testing can be broadly classified into Black Box Testing and White Box Testing.

In Black Box Testing various techniques are-
• Equivalence Class Partitioning
• Boundary Value Analysis
• Decision Tables

In White Box Testing Various Techniques are-
• Static White Box Testing
• Structural White Box Testing

R has researched on structural software testing coverage approaches. In this paper it focuses on Static and Dynamic Testing and its various types. In this it talks about reviews, walkthroughs or inspection which are part of the Static testing whereas white box testing and black box testing are part of dynamic testing. Static Testing involves verification process and Dynamic Testing involves validation process to improve the quality of the software.

III. METHODS OFVERIFICATION

There are many methods available for verification of software work products.
a) Self-review: Self review may not be considered as an official way in software development as it is expected to be done by the author/creator of an artifact by default. It is a primary expectation that no one must say that his work is done before doing Self review and being satisfied that there are no problems in the output. Self-review is excellent in defect prevention through self-learning. Defects found in self-review can help in self-education and self-improvement.
b) Peer review: peer review is conducted frequently in SDLC at various stages of development. Example of peer review could be code review done by the peer or fellow developer, test case review done by fellow tester and so on. It may happen that the person collecting requirements from a customer may ask his peer to go through the requirement document to find out if anything is missing or wrongly interpreted in the requirement statement. It is also called "desk review".
c) Superior review: superior review is also considered as peer review, as a superior may be considered as peer of the author with some more experience and visibility. Superior reviews are conducted to avoid some limitation of peer review related to approach, visibility, etc. A superior such as team leader or module leader may have to review the artifacts made by his subordinates to find its sustainability with reference to the expected outcome.
d) Walkthrough: walkthrough is more formal than peer review but less formal than inspection. It is termed as "semi-formal review" as only related people are involved in walkthrough. In a typical walkthrough, some members of the project team are involved in examining an artifact under review. Author of an artifact presents it, and the entire team discusses various aspects of the artifact.
e) Inspection: inspection is the very formal way of reviewing the work product. Agenda of an application is decided in advance. People attending the inspection are given inputs and background documents beforehand. The author of the work product, and a recorder makes notes of comments given by the inspectors. The actions identified during inspection are tracked till closure. The people present in inspection are experts in the domain under consideration. Inspectors may be external to the organization.
f) Audit: audit is defined as an independent assessment of the process or product under progress. Audit may be categorized into quality assurance activity or quality control activity depending upon its nature, breadth and depth. 100% auditing may be considered as quality control activity while sampling may be considered as quality assurance activity.

IV. LEVELS OFVALIDATION:

Unit testing: Unit testing is a type of testing which involves testing of each component of a software. Unit which means a small part of the software which is testable. Unit testing uses black box methodology. Black box testing concentrates on requirement of the system. In unit testing, units are tested for the design in addition to requirements as low-level design defines the unit.
• Individual components and are tested to ensure the proper functioning of the system.
• In unit testing, individual files may not be testable or executable without the drivers and stubs which are required while performing the test.
• Gray box testing is also considered as "unit testing technique" because it examines the code in full detail along with its functionality.
• Unit test cases may be derived from use cases/design components used at the lowest level ofdesign

Integrationtesting:Integration testing involves integrating the units that are test edintheunittestingprocess.By integrating all the small units to form a module, after that all modules are combined together to form a complete system. Integration testing uses stubs and drivers for testing. Stub is a dummy piece of a code. Driver is a piece of a code emulating a calling function.
There are various approaches in integration testing:
Approach 1: Bottom-up testing
Bottom up testing focuses on testing the lowermost unit and the module. The testing is carried out by using driver and stubs. And after that it goes on integrating the units and modules upwards till it reaches to the top most component. This approach is difficult to implement.

Approach 2: Top-down testing
In top-down testing approach, the top level of the software is tested first and then it goes downward it reaches to the lowestunitofthesystem.Alltop-levelcomponentscalledbytestedcomponentsarecombinedonebyoneandtested in process. Top down approach only requires stubs for the implementation.
Approach 3: Sandwich testing
Sandwich testing is carried out in two parts: top-down approach and bottom-up approach either simultaneously or one after another. This type of testing combines the pro of both the top-down as well as bottom- up
In Bottom-up testing, testing starts from the middle layer and then goes upward to the top layer. Bottom up approach starts at subsystem level and goes upwards.
In Top down testing also the testing starts from middle layer but goes downward. Top-down approach starts at subsystem level and then goes downwards.

System testing: System testing represents the final that is done on the system before being delivered to the customer. It is done on integrated subsystems that make up the entire system, or the final system getting delivered to the customer. System testing mainly validates that the entire system meets its functional/non-functional requirements as defined by the customer in software requirement specification.
System testing goes through the following stages:
• Functional testing: Functional testing is the testing process in which the software is tested to keep up with all the requirements.This testing ensures that all the functionality that is mentioned in the functional requirements.
• User interface testing: Once the functionalities are set correct, the next step is to set the user interface correct. User interface testing may involve colors, navigation, spellings andfonts.

Acceptance testing: Acceptance testing is generally considered as the final stage of testing of a software, before it is accepted formally by a customer for operational purpose. Acceptance testing is generally done by the customer or by somebody on the behalf of the customer. Scope of acceptance testing must be covered in contract or statement of work so that both the parties of contract are aware of acceptance criteria.
Acceptance testing validates the following:
• Whetheruserneeds,asdefinedinsystemrequirementspecifications,areachievedbythesystemornot.
• Whether system performance meets the expectation of customer as defined/documented in system requirement statement.
Itisaformal testing conducted, generally at the end of the software development,tocheck if the application satisfies its acceptance criteria or not.

V. CONCLUSION

Verification and Validation is an essential part of SDLC. Verification also known as Static testing starts at the requirements phase of the SDLC where the gathered requirements are checked properly and thoroughly whereas in Validation also known as Dynamic Testing starts at the designing phase and continues until the software is deployed. It involves testing of the code which helps is reducing the number of errors and bugs in the software. V & V together helps in improving the quality of the software by first verifying the requirements provided at the starting phase of SDLC and then checking the code written and any issues related to the designing of the software. The more Verification and Validation testing are done on the product the better it gets and better customer satisfaction isachieved.

REFERENCES:

Gaur J., Goyal A., Choudhury T. and Sabitha S. (2016). A walk through of software testing techniques, 2016 International Conference System Modeling & Advancement in Research Trends (SMART), Moradabad.

Deshmukh O., Kaushik M. (2013), A Overview of Software Verification & Validation and Selection Process, International Journal of Computer Trends and Technology, 4(2).
Jain M. and Gopalani D. (2016). Aspect Oriented Programming and Types of Software Testing, 2016 Second International Conference on Computational Intelligence & Communication Technology (CICT),

Ghaziabad, 2016

Sethi A. (2017), A review paper on levels, types & techniques in software testing, International Journal of Advanced Research in Computer Science, 8(7).

HHedaoo A. , Khandelwal A. (2017), Study of Dynamic Testing Techniques, International Journal of Advanced Research in Computer Science and Software Engineering, 7(4).

Rambola R.K. , Mandarjatkar *(2018). An Effective Synchonization of ERP in Textile Industries,* In Proc.(IEEE Xplore) IEEE International Conference on Electronic, Communication and Aerospace technology(ICECA-2018) India during 29& 30 March 2018.

Jatkar M. , Rambola R.K. (2018). *How Information technology Helps Textile Manufacturing Unit in Different Steps?,* In Proc.(Springer)International Conference on Information Technology and Digital Applications(ICITDA-2018) India during 6& 7 April, 2018.

Lodha K. N Rambola R.K.(2018), *New Approach of Software Development Life Cycle to improve Software Quality Management,* In Proc.(Springer) International Conference on Information Technology and Digital Applications(ICITDA-2018) India during 6& 7 April, 2018.

Mehra S. , Anand S. , Rambola R. K. (2018). Synchronization of software Quality Management Through Testing, In Proc.(Springer) International Conference on Information Technology and Digital Applications(ICITDA-2018) India during 6& 7 April, 2018.

Verma U. , Rambola R. K. , Meshram P.(2018). *A Model Based Testing Approach for Generation of Test Case from Problem statement using Activity diagram,* In International Conference on Innovative Research in Science, Management and Technology (ICIRSMT-2018) State Government Bilaspur university, Bilaspur, India during 4-5, August, 2018.

Rambola R. K. , Verma U. , Meshram P. (2018), *Design and Development of a Technique for Hidden Data in Watermark Image Using Steganography: An Exploration,* In International Conference on Innovative Research in Science, Management and Technology (ICIRSMT-2018) State Government Bilaspur university, Bilaspur, India during 4-5, August, 2018.