IJCSC International Journal of Computer Science and Communication Vol. 3, No. 1, January-June 2012, pp. 105-107

NEED FOR SPECIAL PLANNING OF VERIFICATION AND VALIDATION BY EXTERNAL ORGANIZATIONS IN SOFTWARE PROJECTS

K. Jagan Mohan[1] and T. Venkat Rao[2]

Varaha Lakshmi Narasimha Swamy Engineering College, Narava, Visakhapatnam-27
E-mail: kammili_jaganmohan@rediffmail.com

Venkat Educational Academy, Ramavarappadu, Vijayawada, *E-mail: venkateducation@yahoo.com*

ABSTRACT

When we look into the goals of verification and validation activities as per the quality assurance of the software products are concerned, they were meant for assessing and improving the quality of the work products generated during development and modification of software. Quality attributes may include correctness, completeness, consistency, reliability, usability, efficiency, conformance to standards and overall cost effectiveness. In some high quality software products it is realized that the independent organization should be given preference to provide verification & validation for the work-products. As per the work-products are concerned, the process of verification should ensure that various work products are complete and consistent with respect to the other work-products and customer needs. For this reason, it is necessary that an external organization should be employed to verify that the design specifications are complete and consistent with respect to the system definition and software product specifications and that the source code is complete and consistent with respect to the design specifications and the requirements.

Keywords: Work-product, system definition, software product specification, acceptance test plan, stress tests, performance test, structure test, software quality assurance plan.

1. INTRODUCTION

Generally, a software development life cycle consists of Requirement-Analysis, Software Design, Implementation, Testing and Maintenance. The process of verification & validation took place during development and modification of software, which ensures the software quality assurance. There are two types of verification. They are - Life cycle verification and Formal verification.

The process of life-cycle verification should ensure that the degree to which the work-products of a given phase of software development cycle fulfil the specifications established during prior phases. Formal verification should ensure that the source code should conform to its specifications through a rigorous mathematical demonstration. The process of Validation should ensure that the software produced at the end of the software-development should comply with the requirements. The real meaning of the terms verification and validation in terms of software development and maintenance is given by-

Verification - "Are we building the product right?"

Validation - "Are we building the right product?"

It is always the best method to minimize the number of errors in a program by catching and removing the errors during analysis and design, which help us to minimize a maximum number of errors during source code testing.

2. PRESENT-WORK

Verification and Validation process involve the assessment of work-products to determine conformance to their specifications. These specifications include the requirements specifications, design documentation, implementation language standards, project standards, organizational standards, and user expectations, meta-specifications for the formats and notations used in various product specifications. The requirements must be examined for conformance to user needs, environmental constraints, and notational standards. The design documentation must be verified with respect to the requirements and notational conventions. The source code must be examined for conformance to the requirements, the design documentation, the user expectations and various implementation and documentation standards. In addition to this, the supporting documents like user's manual, test plan, principles of operation etc must be examined for correctness, completeness, consistency, and adherence to standards.

The errors may occur when any software product is incomplete, inconsistent, or incorrect. There are 3 kinds of errors to be considered. They are: requirement errors, design errors, and implementation errors. Requirement errors are caused by incorrect statement of user needs, failure in specifying functional and performance requirements completely, inconsistencies among

requirements, and infeasible requirements. The Design errors are caused by failure to translate the requirements into correct and complete solution structures, inconsistencies within the design specifications, inconsistencies between design specifications and requirements. The implementation errors are the errors made in translating design specifications into source-code. Implementation errors can occur in data declarations, in data referencing, in control flow logic, in computational expressions, in sub-program interfaces, and in input/ output operations.

We realized that it is very much needed to see that the quality of the work products generated during analysis and design can be assed and improved by using systematic quality assurance procedures, by inspections, and by automated checking for consistency and completeness. Assessing and improving the quality of source code can be done by implementing the quality assurance procedures, inspections, static analysis, symbolic execution, unit testing, integration testing.

The main purpose of a software quality assurance group is to provide assurance that the procedures, tools, and techniques used during product development and modification are adequate to provide the desired level of confidence in the work-products. The software quality assurance personnel are organizationally distinct from the software development group. This adds a degree of impartiality to quality assurance activities, and allows quality assurance personnel to become specialists in their discipline. The quality assurance personnel may function either as an advisory committee in the same organization or they may actively develop standards, tools, and techniques, and examines all work-products for conformance to specifications.

It is very much necessary that the software quality assurance group is responsible for software verification plan, Acceptance test plan, source code test plan, and quality assurance plan.During analysis and design, a software verification plan and an acceptance test plan are prepared. The verification plan describes the methods to be used in verifying that the requirements are satisfied by the design documents and that the source code is consistent with the requirements specification and design documentation. The acceptance test plan includes test cases, expected outcomes, and capabilities demonstrated by each test case. Following completion of the verification plan and the acceptance plan, a software verification review is held to evaluate the adequacy of the plans. During product evolution, in-process audits are conducted to verify consistency and completeness of the work-products. The items to be audited for consistency include interface specifications for hardware, software and people. Intensive audits can be done on internal design versus functional specifications; source code versus design documentation; and functional requirements versus test descriptions. Prior to product delivery, a functional audit and a physical audit are performed. The functional audit reconfirms that all requirements have been met. The physical audit verifies that the source code and all associated documents are complete, internally consistent, consistent with one another and ready for delivery. A software verification summary is prepared to describe the results of all reviews, audits, and tests conducted by quality assurance personnel throughout the development cycle. The quality assurance group will work with the development group to derive the source code test plan which specifies the objectives of testing, the test completion criteria, and the system integration plan, methods to be used on particular modules, and particular test inputs and expected outcomes. A source code test must satisfy function tests, performance tests, stress tests, and structure tests.

The function test cases specify typical operating conditions, typical input values, and typical expected results. They also test the behavior in, around and just beyond the functional boundaries. Performance tests are designed to verify response time under varying loads, percent of execution time spent in various segments of the program, primary and secondary memory utilization, and traffic rates on data channels and communication links. Stress tests are designed to overload a system in various ways. Ex: Attempting to sign on more than the maximum number of allowed terminals, processing more than the allowed number of identifiers or static levels, or disconnecting a communication link. Structure tests are concerned with examining the internal processing logic of a software system. The objective of structure test is to traverse a specified number of paths through each routine in the system to establish thoroughness of testing.

Validation typically involves planning and execution of test cases. On projects using independent verification and validation, test cases are developed and are executed by the independent organization. Independent software verification and validation by an external organization results in producing the high quality software products. However, the cost of performing the independent verification and validation may be as much as 25 percent of the development cost. The benefits must be judged in terms of high cost.

3. CONCLUSION

It is now realized that the independent verification and validation by an external organization leads to the production of high quality software. Hence, the need for Special Planning of Verification and Validation by External Organizations in Software Projects is necessary. This technical paper explained the verification and validation techniques and will be helpful for the software engineers particularly software quality assurance personnel to go ahead with the special planning for verification and validation techniques apart from the software development group.

REFERENCES

[1] Software Engineering concepts - Richard Fairley.

[2] Software Engineering - Pressman.

[3] Software Engineering (7th Edition) - Ian Sommerville.

[4] Specification of Software Systems - V.S. Alagar, K. Periyasamy.

[5] Software Engineering: Theory and Practice (4th Edition) - Shari Lawrence Pfleeger and Joanne M. Atlee.

[6] Object-Oriented Software Engineering: Practical Software Development using UML and Java - Timothy Lethbridge, Robert Laganiere.

[7] Model-Based Software Performance Analysis - Cortellessa, Vittorio, Di Marco, Antinisca, Inverardi, Paola.

[8] Practical Formal Software Engineering - Bruce Mills.

[9] Software Modeling and Design - Hassan Gomaa.

[10] Software Management, 7th Edition -Donald J. Reifer, Barry Boehm.