AN EFFECTIVE VERIFICATION AND VALIDATION STRATEGY FOR SAFETY-CRITICAL EMBEDDED SYSTEMS

Manju Nanda[1], Jayanthi J²

[1]Scientist, National Aerospace Laboratories, Bangalore, India
manjun@nal.res.in
[2] Scientist, National Aerospace Laboratories, Bangalore, India
jayanthi@nal.res.in

ABSTRACT

This paper presents the best practices to carry out the verification and validation (V&V) for a safety-critical embedded system, part of a larger system-of-systems. The paper talks about the effectiveness of this strategy from performance and time schedule requirement of a project. The best practices employed for the V &Vis a modification of the conventional V&V approach. The proposed approach is iterative which introduces new testing methodologies apart from the conventional testing methodologies, an effective way of implementing the phases of the V&V and also analyzing the V&V results. The new testing methodologies include the random and non-real time testing apart from the static and dynamic tests. The process phases are logically carried out in parallel and credit of the results of the different phases are taken to ensure that the embedded system that goes for the field testing is bug free. The paper also demonstrates the iterative qualities of the process where the iterations successively find faults in the embedded system and executing the process within a stipulated time frame, thus maintaining the required reliability of the system. This approach is implemented in the most critical applications —-aerospace application where safety of the system cannot be compromised. The approach used a fixed number of iterations which is set to4in this application, with each iteration adding to the reliability and safety of the embedded system. Data collected and results observed are compared with a conventional approach for the same application and it is demonstrated that the strategy proposed reduces the time taken by 50% as compared to a conventional process that attains the same reliability as required in the stipulated time.

KEYWORDS

Verification and Validation process, safety, critical systems, embedded systems, reliability

1. INTRODUCTION

This paper presents an effective and reliable V&V approach to be carried out as part of the embedded system development. The modified approach effectively implements the Verification and Validation process ensuring the safety, reliability of the system within the project schedules. A similar work to improve the V&V process is described in the paper by NASA [1].

DOI : 10.5121/ijsea.2013.4209

Verification & Validation of safety critical subsystems found inside larger systems is a great challenge, not least of all as the Verification and Validation process needs to be carried out within a stipulated time without compromising on the final safety of the overall system. The traditional V&V approach in case of safety critical systems is to test the embedded system with the test cases till such time as all the faults are presumed to be detected and all the tests generated for the system pass. The assumption for execution of these tests is that the test setup is correct and the test cases for the system testing are also correct. It is observed that the time taken to execute these tests often take longer than the required time, many a times effecting the project schedules [2].

The conventional V&V approach for an embedded system talks about the sequential execution of the low level testing, software-software integration testing, hardware-software integration testing and finally the system testing. This approach is a proven process to capture bugs, defects in the system during the embedded system development lifecycle. New tools enable this process for better productivity and reliability as the errors with the human in the loop is reduced. Model based development is a step towards in that direction. It is the most appropriate approach in speeding up the process activity and reducing the human errors in critical applications. Model based approach automates the Verification and Validation process resulting in reduced time to carry out this process. This approach is slowly gaining maturity. The tools for V&V of the model-based approach are available but are not qualified. This becomes a bottleneck in case the embedded system is used for a safety critical application in specific the aerospace application. The qualified tool that exists for carrying out the V&V is a very expensive tool affecting the budge of a project and also the entire software cannot be verified especially the input-output processing. The input-output processing reads the various inputs from the external interfaces and validates it for its health and reliability before they are used for critical computations and the output processing provides reliable signals to the external interfaces. The V&V of this functionality of a system is to be carried out with the conventional approach.

This paper proposes a modified conventional approach, which uses the conventional approach as the framework, more parallel activities are carried and some of effective testing methodologies are introduced. The low-level and the software-software integration is carried out sequentially but the hardware-software integration is carried out in parallel with the software testing. This helps in executing the V&V task in the project stipulated time. There may be scenarios where it is not possible to carry out hardware-software integration due to the limitation of the test set up or tools incapability or not possible to test during this phase then if it’s tested at software level the credit is taken from that. New testing methodologies are introduced apart from the existing ones to detect the defects earlier in the phase of the system development. This modified approach is proposed as best practices to be carried out for a V&V process to achieve the reliability and the safety of the system in the project stipulated time.

The iterative V&V process includes tests to be carried out at software level and at the hardware-software integration level in parallel . This approach helped in executing and capturing defects, capturing the requirement either at the software or at the hardware-software integration level. The process in each iteration includes document review, traceability establishment, test execution and report generation. Iterative approach is a good practice proposed for the Verification and Validation process and building up the system functionality-wise [3].
This modified approach resulted in establishing best practices which can be employed for the conventional V & V process and also reducing the V&V in process time to 50% less time than the conventional iterative approach where the numbers of iterations are not defined because some of

the test methods described in the best practices are not followed. The paper discusses in detail the strategy to carry out the V & V process, comparison with the conventional approach and the results of the two approaches. This approach is carried out for the embedded system development for use on a larger system, a 14-seater civilian aircraft. The embedded system is a critical system as it monitors critical parameters in the aircraft during its flight so the reliability requirement for this system is very high.

Section II defines the terminology used in this paper. Section III gives an overview of the literature survey. Section IV describes the iterative Verification & Validation process carried out, and Section VI describes the implementation. SectionVII illustrates the effectiveness of the V& V process and Section VIII summarizes the conclusion.

2. TERMINOLOGY

Definitions of some of the terminologies used in this paper are defined in this section and the definitions are pertaining to this paper.

Verification and Validation process: In any process driven system or software development the verification process verifies if the system or software is being developed as per the requirement and the validation process confirms the system or software is developed as per the requirement. The verification process is carried out during the different developmental phases of the system or the software. The validation process is done at the end of the system or software development. 50-70% of the developmental resources are consumed by this process. [4]. This process consists of reviews, analysis, checklist generation and testing.

Safety critical system: A system whose functionality affects the safety of the environment, human life, or the nation’s interests is a safety critical system.

Embedded system: A system comprising of hardware, software and additional mechanical parts designed to perform specific function. [5]

Testing: Testing is a part of the Verification and Validation process where the system or software is checked by executing specific test cases. These test cases are developed based on the system or software requirement. The results of these test cases are pass or fail based on the system or software functional requirement. Testing needs a test tool in case of software testing and a test set-up in case of system testing.

DO-178B: DO-178B is the Software Considerations in Air-borne Systems and Equipment Certification. FAA′s Advisory Circular AC20-115B established DO-178B as the accepted means of certifying all new aviation software. It provides the guidelines for developing airborne software. Based on the increasing order of the software criticality the software is categorized as Level E,D,C,B and A.

Build: The software released by the design team for the V&V process is called a Build. Each Build has a specific software functionality to be tested. The numbers of builds to be released are to be decided by the V&V team.

3. BACKGROUND

A lot of work is being carried out in this field as the reliability of any system is affected by the Verification and the Validation process. The V&V process consumes 50-70% of the system development cycle. Various techniques are being researched on for reducing this time. Various papers talk about the different Verification and Validation process like the model based, conventional and model and conventional based Verification and Validation and the effective software testing strategies [6] [7] [8] .

The current work is different from the literature surveyed as this paper discusses about the effective testing strategy for a safety critical systems in the stipulated time of frame with high reliability. The paper [2] talks about the similar work carried out by this paper but the concentration is more on the reliability rather than the testing methodologies to reduce the Verification and Validation time maintaining the required reliability. Other papers talk about the Verification and Validation approaches for specific applications, reliability computations from the testing results of the software. A brief overview of the literature survey is given below.

Tal et.al [2] discusses the importance of testing and reliability in safety-Critical applications is described , in particular defense applications. Two new testing methodologies, Single Risk Sequential Testing (SRST) and the Probability Ratio Sequential Testing (PRST) are described and compared for the consumer risk. The optimal testing policies are described to maintain a high reliability and this is demonstrated with a new reliability model for safety-critical systems. Statistical data are provided to demonstrate it. The paper also describes the methods for improving the calculated reliability or shortening the testing duration.

Bertolino [9] deals with software testing, software reliability and the effectiveness of the testing. ‘Testability’ is the word which is being used in this paper to analyse the effectiveness of testing. The paper describes a model of program execution and testing, the parameters, events and the probabilities of interest for computing the testability of the system. The description about the relationship between the testability, reliability, software design and the test set-up are given. Formulae to compute the failures even after a successful testing is given and the Bayesian inference is used to find the reliability after the testing is done with the testability approach.

Miller et.al [10] proposes a formula for estimating the probability of failure when the testing process reveals no errors in the system. The formula incorporates random testing results, information about the input distribution and prior probability of failure of the software. The formulae are based on discrete sample space statistical model of software and include Bayesian prior assumptions. The method described assumes that the input distributions are generally not equally likely and that the input distribution assumed during testing may not be the same as that during use. The formulae proposed will help the professionals to quantify the techniques used during the software development and improve the software quality.

Antonia [11] presents the statistical approach for estimating software reliability. The paper talks about the Bayesian Inference mathematical used for statistical testing. Analysis of the expected failure after the testing phase is done and an approach to achieve a reliability target is given. The paper also gives a detailed reliability approaches with various prior distributions and how the prior distributions help in checking the correctness of the program. These approaches are tested using examples.

Ammann et.al [12] assesses the software reliability by running a large number of test cases and observing the test results. Estimation of the system reliability is achieved by life-testing or the statistical usage testing. The paper talks about the effects of imperfect error detection and the means to compute realistic reliability bounds.

The Handbook released by the U.S Department of Transportation [3] talks about the software life cycle, software testing, safety in software, numerical reliability and tools for software reliability. The chapter 4 of this handbook talks about the testing approach and methodology for safety critical applications. This chapter concludes by suggesting that in case of safety critical applications the deployment should be done in succession of phases and in each deployment the number of deployments is a multiple of the deployments in the previous phase.

Ayaz [13] deals with the importance of Verification and Validation process and the future research directions in this field. The paper speaks about the current Verification and Validation standards like the Testing Maturity Model, Test Process Improvement Model, Test Management Approach, Metrics based Verification and Validation Maturity Model, IEEE standard for Software Unit Testing to name a few. The future research areas in the Verification and Validation techniques are the provision of efficient defect detection and prevention, minimization of test and development costs and schedules, scalability and production of quality products.

Bouali [14] describes Verification methods for model based system development in safety critical systems. Formal methods are used to verify these systems. This paper talks about the formal verification tools from SCADE to carry out the verification. The model based system development is the emerging technology and the tools to verify them should be qualified as the systems are used for safety critical applications. The entire system development and formal verification as per IEC61508 software process model is explained with a performance table to show the effectiveness of the process.

Mats [4] highlights the issues which are still not well understood, namely safety, demonstrating the safety of a system, increasing reliance on the models and automated tools used for development and testing and the importance of the data for a system safety. The system safety aspects to be performed as an integral part of the iterative system development. The verification and the Validation process parallels this development process and verifies that the implemented system satisfies the safety requirement. The demonstration of the safety can be done as per the safety standards. Model based system development is an approach to reduce the verification and Validation cycle process helping in dramatic reduction in cost savings. The model verification and validation is the approach for model based development. With safety the tools used for modeling should be qualified else it may lead to safety issues. The author has elaborated this with examples. Techniques to assure validation of the data is required in the safety critical systems as an incorrect data can lead to catastrophe.

Jeppu et.al [15] discuss the novel non-real time testing of the control law algorithm for the Indian Light Combat Aircraft which resulted in the fault free software performance during its maiden flight. This paper demonstrates the effectiveness of the non-real time testing of the software developed on Ada code and discusses the procedure to generate the non-real time test scenarios.

4. A BRIEF OVERVIEW OF THE TWO APPROACHES OF THE VERIFICATION AND VALIDATION PROCESS

The paper presents an efficient implementation of the V&V process for a safety critical application. The efficient implementation of the process helps in meeting the project schedule requirement. Every project whether it is safety or non-safety related is time bound and if it is not executed on time there are economic implications. The strategy proposed for carrying out the Verification and Validation process is motivated by the safety critical system development for an indigenous 14-seater aircraft. The system is a part of the Avionics system developed as per the aircraft standards of DO-178B Level A. This system is used to inform the pilot about the critical aircraft warnings. The system requirements are well defined and documented. As per the Safety critical process and the aircraft standards the Failure Hazard Analysis and the Failure Mode and Effect Analysis is also done for the system. These are required to assess the safety requirement of the system. Mats also discusses this [4], and gives an overview of the importance of V&V, noting that 50–70% of the resources are consumed by the Verification and Validation phase and good techniques in this approach can save cost at the same time the time to market of the product. The conventional V&V process flow will carry out the testing of the system until the testing converges to no failure of the test cases [2]. This approach also has a iterative process but the number of iterations depend till the test cases do not converge based on the system application and the safety implication of the test cases which have not converged. This is an ideal approach for a safety critical system V&V process. The prolonged V&V process effects the cost and the time of the project. The documentation involved in each of the iteration also becomes voluminous. The documentation in each iteration involves generation of the artifacts like the test procedure, traceability matrix, test cases, test reports, test observations and the test limitations. The test methodologies used in such approach include —-static and dynamic test procedure. Figure 1 gives the V&V process flow for an conventional process

In the above figure the system is divided into various functionalities as it is being developed. The design team releases the V&V builds. Each build has specific functionalities to be tested. The test cases to test the system functionality are developed and once the build is received the test cases are executed and the results observed. If any test fails, the design team fixes the bug and releases the same build with a different version. This is a continues process till the time no test cases fail. This process is followed with other incremental builds and continued till all the builds are tested. Once all the builds are tested, the system tests are carried out. System test is an end to end testing of the embedded system. If there is any failure during this test, the system design is analyzed and many a times there is a change in the system design resulting in carrying out the critical V&V process activities. In this approach each build goes iteration till the build release is not cleared from the safety aspect. In case the project has strict time schedule then this approach may affect the schedule, money and the resources.

The test cases developments for each of the builds are based on static, dynamic and safety based scenarios.

Static based testing: A conventional test case generation method where the different signal values (Analog, Discrete, Digital and ARINC) are generated at fixed intervals of time based on the signal processing frequency of the embedded system. The test cases generate the signal static values in the normal and the robustness range. Dynamic tests based testing: Conventional test case generation method where the different signal values (Analog, Discrete, Digital and ARINC) are generated in real time using signals like the sinusoidal waveforms, ramp waveforms or random signals based on the rate at which the system processes the data.

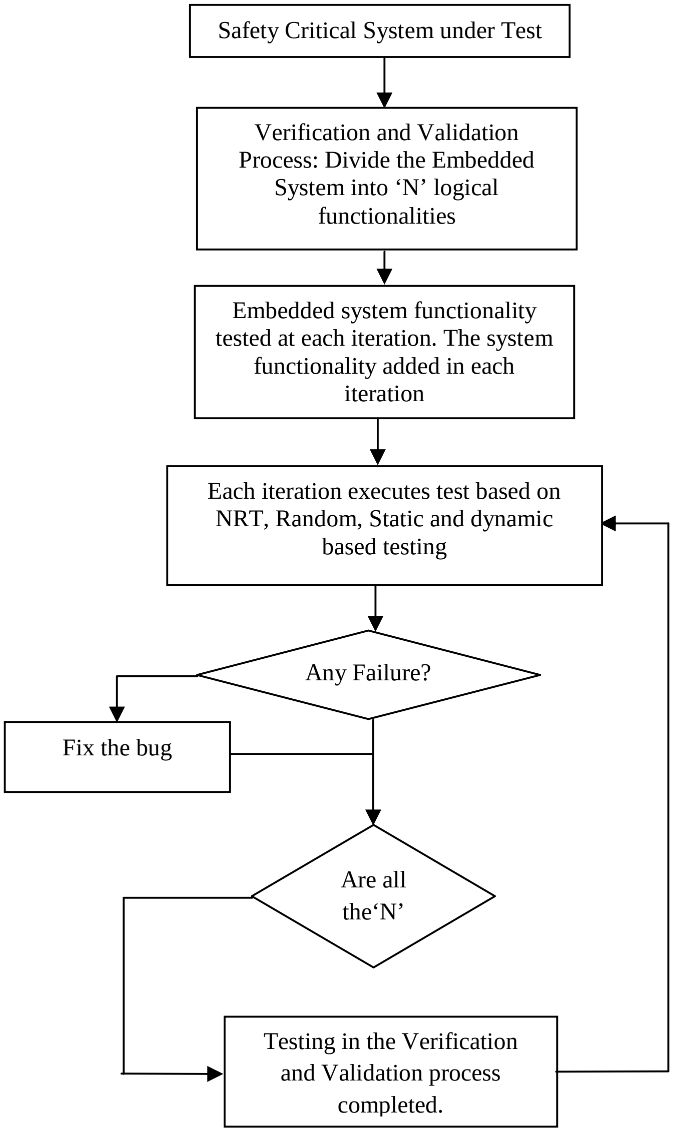
Safety Critical System under Test
Verification and Validation Process: Divide the Embedded System into 'N' logical functionalities
Embedded system functionality tested at each iteration. The system functionality added in each iteration
Each iteration executes test based on NRT, Random, Static and dynamic based testing
Any Failure?
Fix the bug
Are all the'N'
Testing in the Verification and Validation process completed.

Figure 1 Conventional Verification and Validation process flow

The signal frequency is based on the rate at which the system processes the data and the signal amplitude is based on the operating range of the signal.

Safety tests based testing: The hazard analysis, Failure mode effect analysis of the embedded system helps in generating the safety related issues. The software responses to the various safety issues of the system are captured. The test cases are designed to make sure that under any unforeseen circumstances the embedded systems does not behave unpredictable and also does not harm the external interfaces, environment and the humans .

The current project requirement is to carry out the V&V process effectively capturing all the bugs early in the process and to complete it within the schedule time. The approach taken is to fixed the numbers of V&V process iterations and also minimize the number of iterations in each of the builds. This is possible by developing the test cases in the most optimized way to capture maximum bugs in minimum number of test cases. This is shown in the Figure 2.

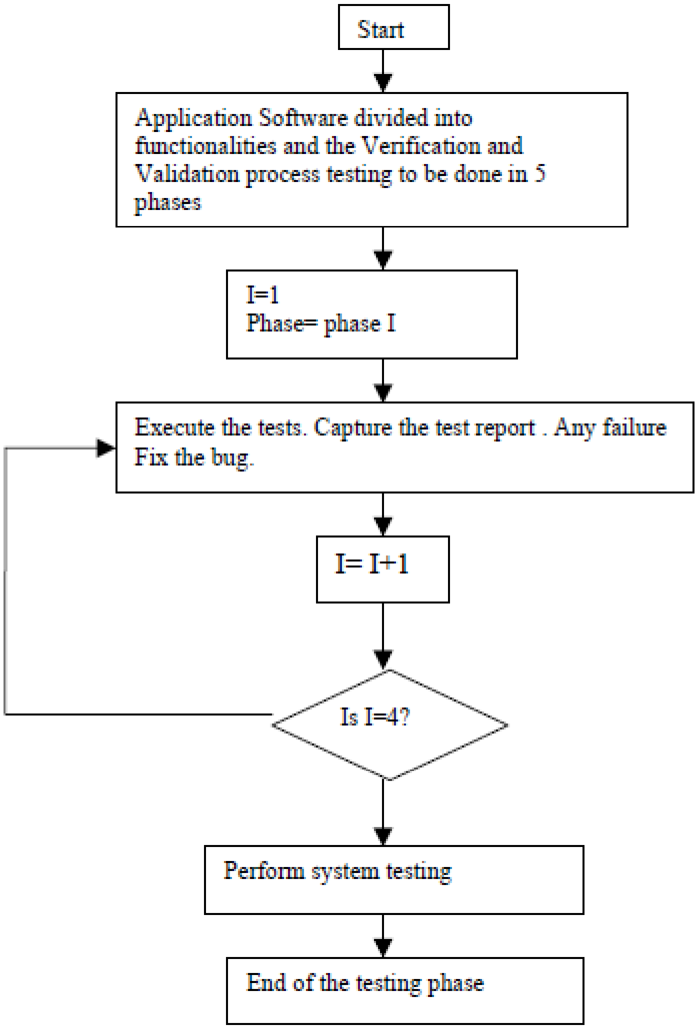
Start
Application Software divided into functionalities and the Verification and Validation process testing to be done in 5 phases
I=1
Phase= phase I
Execute the tests. Capture the test report . Any failure Fix the bug.
I= I+1
Is I=4?
Perform system testing
End of the testing phase

Figure 2 Effective Verification and Validation process flow

The process flow is similar to the one in the conventional approach but the major difference is that the build does not undergo iterations until there is a requirement change due to the test case failure. In the above figure the system is divided various functionalities as it is being build up. Unlike the conventional process each build had more than one functionality to be tested. Each build provided by the designer had functionality which is logically grouped. The numbers of V&V process iterations are also limited to four. This number is limited to four based on the complexity and the various functionality tested in each of the build. For each build the test cases are developed priori. The test cases are executed once the build is received from the design group. Any failure during the build test is analyzed for its impact. If the bug leading to the failure is minor i.e. some code logic implementation change then this bug is fixed and is tested as part of the next build as the incremental builds are built upon the previous build. But in case the test failure is due to a bug which needs a requirement change then the bug is fixed and the same build is run again as the requirement change may lead to other test case changes. The possibility of this is remote as the requirement is reviewed thoroughly by group of system experts. Once the predefined build iterations are over the system tests are carried out. Similar to the conventional approach any failure during this stage is analyzed. Based on the impact analysis sometimes the entire process is redone affecting the project schedules. The probability of such a case is very rare. The number of iterations for project is fixed to four. The test methodologies used for carrying out the V&V process introduces new methodologies which help in detecting critical failures. The random test and the non-real time testing are two such methodologies added to the conventional approach. Apart from the new methodologies added, the approach uses optimized test cases which cover all the requirements of the system. These optimized test cases are generated by doing a detailed study of the system, its safety and functionality and helps in testing the embedded system in the most efficient and complete way. The techniques to optimize the test cases are itself a subject of study.

The test cases developed during the V&V process for each of the incremental build are based on Static, Dynamic, Safety, Random and Non-Real-time based scenarios. Static, dynamic and Safety based test cases are explained in the previous paragraph.

Random tests based testing: In this testing the input domain of the oracle consists of generating random signals being fed to the embedded system. In this test the signal values required by the critical system are generated randomly. This is done by using well defined random functions with different seed values. The range of the random signals is as per the different signal ranges interfacing with the critical system when it interfaces with the various other systems.

In the random tests based testing the critical sections of the software are tested to find the non-obvious bugs. This method is the fastest way to find bugs by generating signals randomly and record the software behavior. Most of the hidden and nonobvious bugs can be detected by this method easily.

Non-Real Time based testing: Safety critical application of the software which are modeled can be tested by this method. The critical section in the unit under test and the other sections interacting with this section are simulated using the normal range test case and robustness range test case to capture the response of the software critical sections. The interacting functions are dummy and the signals generated or the data provided by these functions are simulate. The simulation of the signal or data can either be static or dynamic. In case of static signal or data simulation the data is generated by giving a fixed value of data or signal at any instance of time. In case of dynamics signal generation signals like sine wave, ramp or other random signals are generated at a known frequency and amplitude. In static as well as dynamic tests the signals or data are generated as per the processing frequency of the critical system under test. An example of this testing is the control law applications for aerospace applications. The Non-Real-time test results help in executing the same tests on the test set up faster as the critical function logical functionality has already been done and passed. The test set up for a non-real time testing is shown in Figure 3

Test set up is an oracle with (human or computer) agent that decides whether the program behaved properly on a given test [4]. The statement of an oracle is given below. Oracle → Input domain X Output domain X. *Sequence of values → Approve, Rejected*

The testing for an embedded system is done at the software level as well as the hardware level with the software embedded in the hardware. The software testing is done on the native hardware with appropriate test tools making the software testing automated or it can be tested manually. The software testing tests the software functionality, capturing the software coverage using the test cases based on the requirement. The tests done at the software level using the tools are static tests. In this project dynamic testing at the software level is introduced. The dynamic test scenarios are developed based on the feedback by the system designers. Test cases are developed based on the requirements, signal type, signal range.

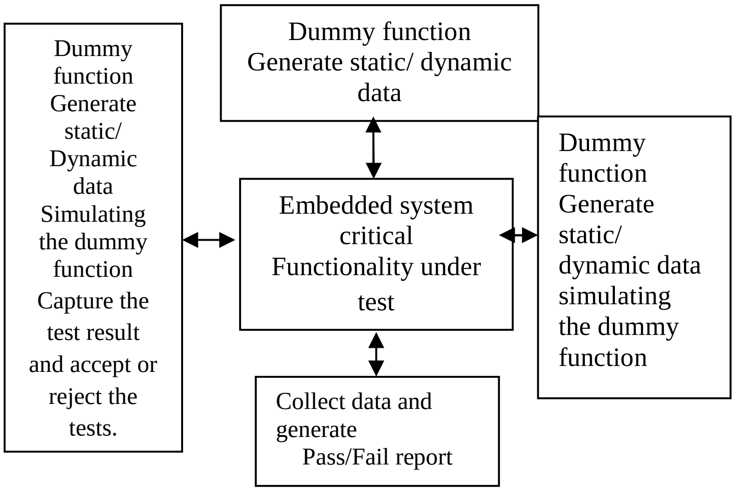
Dummy function Generate static/ Dynamic data Simulating the dummy function Capture the test result and accept or reject the tests.
Dummy function Generate static/ dynamic data
Embedded system critical Functionality under test
Dummy function Generate static/ dynamic data simulating the dummy function
Collect data and generate Pass/Fail report

Figure 3 Non-Real Time Test Set Up

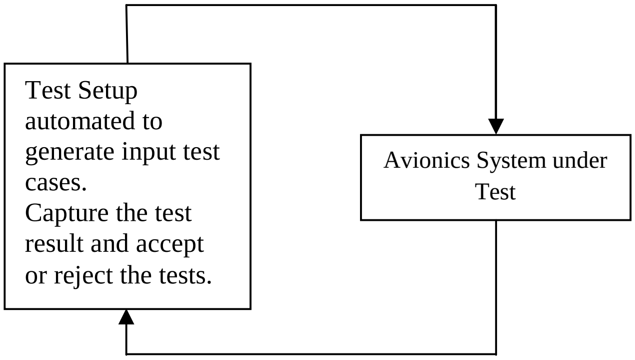
Test Setup automated to generate input test cases. Capture the test result and accept or reject the tests.
Avionics System under Test

Figure 4 Generic Oracle

The test cases also cover the normal range of the signals as per the system requirements and the robustness range where the performance of the software is checked under the out of range signal condition. This is done to make sure that the software architecture is designed for the software's performance under robust conditions. With every build, the functionality is defined and the build consists of the modules which perform these functionality. Further the software testing is done at two levels—module level and the module integration level. The software and the system testing continues in parallel with each build. This approach also helps in accelerating the process and complementing the test scenarios that are not possible in software level, the credit is taken from the hardware-software integration. Vice-versa is also true. In the conventional approach the software and embedded testing reports are used separately for the iteration to be carried out in each build. In case of the effective approach any failure in the embedded testing is complemented by the test passing at the software level. This is done many a times because sometimes it is not possible to test some requirement at the system level due to the limitation of the test set-up. In case the functionality requirement test gives correct result at the software level and the same functionality requirement fails at the system level then this requirement can be cleared provided the failure is due to the test set up limitation. 'Coverage' is an important criteria in the Verification and Validation process. All system requirements have to be covered at any level of testing—software module level, integration level or at the system level. Sometimes if it is not possible to do at either of the levels then such requirements are cleared through analysis. This is possible if the Verification and Validation team is equipped with experienced people to understand and analyze the requirement, develop test cases, execute them, observe the results and generate the report. The Verification and Validation report generated is one of the artifacts for the certification of the critical system. These reports give the confidence of the correctness of the system being deployed in the safety critical systems. AV&V process needs test tools, test setups and a very good set of test cases. In case of safety critical system development the tools used need to be qualified. A qualified tool is a tool in which the outputs are reliable and predictable. There is a rigorous process to qualify tools for safety critical system development and V&V process. Many commercially available tools are qualified these days. The test set up also needs to be qualified. A well qualified tool, test set up and a set of optimized test cases captures most of the bugs in the system development making the system safe and reliable.

5. BEST PRACTICES FOR THE VERIFICATION AND VALIDATION PROCESS

The best practices employed to make the V&V effective and the project a success is discussed in this section. The activities to be carried out are the thorough document review, impact analysis of the iterative build, strategy and documentation for the V&V activities in the current build, enhanced and improved testing methodologies, review of the testing procedure and the detailed report generation. Each of the best practices will be described in the following paragraphs.

Document Review: Document review is one of the earliest activity to be carried out once the build is released by the design team for the V&V activities. The document review provides an in depth information regarding the requirement, design and the code implementation. Based on the review , the test procedures are developed.

Impact Analysis: In case of the iterative builds, the new build is reviewed and the changes in comparison to the previous build are analyzed for their impact from the changes to the test procedure strategy.

Test Strategy and Documentation: The impact analysis on the build helps in developing or updating or modifying the test strategy for the test procedure. The impact analysis, strategy are documented for reference during the build activity.

V&V activities for the iterative build: The document also documents the various activities for the iterative build. All the activities will not be applicable for a particular build based on the functionality implemented by the build.

Enhanced and improved methodologies: The testing methodologies include software testing that include static and dynamic test. Hardware Software Integration testing includes static, dynamic and random testing. Based on the functionality the non-real time based testing can be carried out for Hardware Software Integration testing.

Review of the Testing Procedure: This is to make sure that the test procedure is correct and compete before the execution of the tests.

Execution and Report Generation: This activity is the last activity to be carried out and the results obtained are analyzed and based on the pass fail criteria the test reports are generated with the information of the requirements captured, limitation and the observations during the tests.
These activities are carried out for the V&V process activity in a sequential way ensuring the effectiveness of the process and capturing the bug early in the process.

6. Implementing the Effective Verification and Validation Process

The previous section gives an overview of the two V&V approaches. This section talks about the implementation of the V&V approach proposed to show its effectiveness. As mentioned earlier the approach is developed for a safety critical avionics application.

There is also a different school of thought for the effective approach for Verification and Validation process in which the testing for the system requirements is carried out at the system level. The testing of the software is not carried out separately on the native platform. Instead the software is tested as part of the system. This approach reduces the time required to carry out the Verification and Validation process further. This approach can be used if the critical system is reused for different applications with changes in the software and the same team carries the Verification and Validation process. In case of a new project with new test scenarios it is advisable and necessary that the entire process of software testing, system testing to cover the requirements and gain confidence about the system. Many may contest this thought but with experience we have realized this.

The tools used for the software testing were qualified and commercially available. The test set up for the embedded system testing is developed in-house as it is specific to the application. The qualification of the setup is carried out by an independent team. This team is a part of the quality team. Qualification is done by testing the features provided by the test set up. The test set-up test procedure is developed by the V&V team as it is aware of the requirements to be provided by the test set up which will help them in carrying out the tests. These test procedures are documented in the Acceptance Test Procedure document. The test procedure underwent a lot reviews before it was accepted.

In the test set up for the application the various signals the critical system uses are simulated through the data acquisition cards. These cards are programmed to generated signals required for testing the system under test. The build released by the designers provide test points in the system to debug and trace the signal and data flow from the various functionalities in the critical system from the input to the output. The test points, system outputs are received in the receiver computer. The receiver computer has a in-house built software which captures these results and generates a Pass, Fail report. The test set up used for the project is shown in figure 5

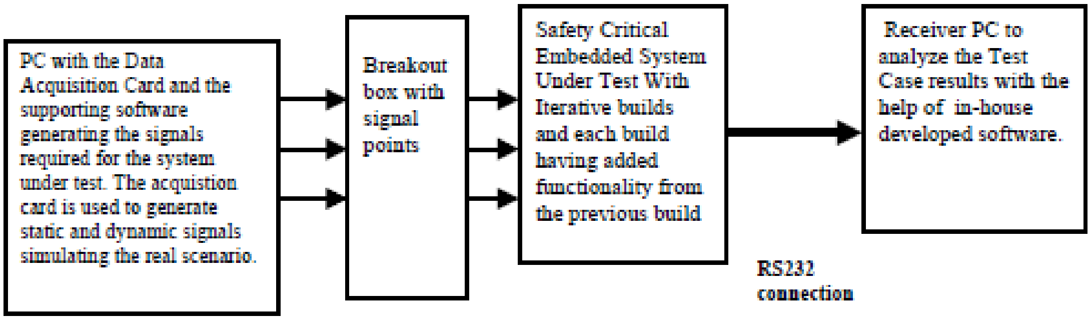
PC with the Data Acquisition Card and the supporting software generating the signals required for the system under test. The acquistion card is used to generate static and dynamic signals simulating the real scenario.
Breakout box with signal points
Safety Critical Embedded System Under Test With Iterative builds and each build having added functionality from the previous build
Receiver PC to analyze the Test Case results with the help of in-house developed software.
RS232 connection

Figure 5 Test Set-Up Used For The Application

Qualification of the test set up and the test tools help in building up the confidence of the V&V team, certification team, quality team and the design team about the results generated by it.

The critical system which is being developed and tested consists of 22 functionalities in all. The 22 functionalities are logically divided into four builds. Each of the build is given by the design team and is V&V iteration for the V&V team. The building up of the functionality in each build by the design team is in concurrence with the V&V team. The buildup of the functionalities by the design team should be such that the regression testing of some of the test cases of the previous build can be executed. If this provision is not provided then any build fixes cannot be tested in the next build resulting in the iterations of the same build till all the fixes are done before proceeding to the next build. Description of each of the V&V iteration with the functionality tested in each iteration is given below. Each of the V&V iteration executes the software tests using the software test tool, system test using the test set up developed for the app Application software 22 functionalities

Iteration 1 → Input, output interfaces, input, output processing and System initialization
Iteration2 → Iteration1 +warnings +Built In Tests
Iteration3 → Control Law +Monitors +Iteration2
Iteration4 → System Time Response +Failure Management +Data Logging +Iteration3
Iteration5 → System testing
The number of functions in each of the iteration is given below.
Iteration1 → Comprises of 9 functions of the system
Iteration2 → 2 functions+Iteration1
Iteration3 →2 functions +Iteration2
Iteration4 → 9 functions +Iteration3
Iteration5 →System testing

This iterative approach used was effective as in each of the iteration the regression testing of the previous iteration was possible. The functionalities to be tested were defined and the test cases were developed as per the tests to be carried out in each of the builds.

The total functionality of the critical system adds up to 22,000 software lines of code. This is implemented in 132 modules. The implementation in each module is dictated as per the coding standard and coding style. The number of modules tested in each of the iteration is shown below.

Iteration 1 → 9 functions to be tested at system level and 40 modules to be tested at the software level
Iteration 2 → 2 functions+Iteration1 at system level and 40 modules to be tested at software level + regression testing of previous iteration
Iteration 3 → 2 functions+Iteration2 at system level and 45 modules to be tested at software level +regression testing of previous iteration
Iteration 4 → 9 functions+Iteration3 at the system level and 5 modules at software level
Iteration5 → System testing

The static, dynamic and random based test cases are executed in each of the V&V iteration mentioned above. The Non-Real time testing is executed in the Iteration 3 because this iteration tests the critical functionality of the application and it is very logical intensive. The logical correctness of this functionality is to be checked thoroughly. The Safety based test cases are part of each of the iteration and extensively checked in the Iteration 4 as the total application is available in iteration 4.

There are various tasks to be carried out by the V&V team before and after the tests are carried out. Test cases were developed after a complete understanding of the system, tools used and the test set up. The various references for generating these test cases were the system requirements, implementation of the requirement in the design, failure hazard analysis report of the system, Failure Mode and Effect Analysis report of the system, the various signals coming to the system, their range, frequency and tolerance level of the various signals. Test set up was qualified and the software was developed in-housed to automate the test scenario. This software also underwent qualification. Qualification reports were generated after successfully completing these qualifications. Analyzing the test result and generating the test report with test pass or fail information is carried out after the execution of the test cases at the software and system level.

Table 1 Time Taken and the Test Cases in each iteration

Iteration	Test Cases	Time taken	Loops
First Iteration	3507	25	2
Second Iteration	1698	30	1
Third Iteration	2639	50	1
Fourth Iteration	454	10	1
Fifth Iteration	1900	45	1
Iteration	**Test Cases**	**Time taken**	**Loops**
First Iteration	1061	10	1
Second Iteration	1688	45	4
Third Iteration	1000	35	2

Fourth Iteration	200	15	1
Fifth Iteration	1600	40	3
Sixth Iteration	2577	40	1
Seventh Iteration	1000	35	1
Eight Iteration	64	10	3
Ninth Iteration	270	30	1
Tenth Iteration	175	40	2
Eleventh Iteration	1900	50	1

The test cases developed for carrying out the software testing in each iteration is briefed below.
Iteration1 → 1200 test cases for 40 modules
Iteration2 → 1300 test cases for 40 modules
Iteration3 → 1600 test cases for 45 modules
Iteration4 → 125 test cases for5 modules
The test cases developed for carrying out the system testing in each iteration is briefed below.
Iteration1 → 2307 test cases
Iteration2 → 398 test cases
Iteration3 → 1039 test cases
Iteration4 → 449 test cases
Iteration5 → 1900 test cases

The table below shows the time taken by each of the iteration and the test cases executed at each iteration. The test cases are inclusive of the test cases for software and system testing.

The entire V&V process took six months to complete the complete the V&V process from the task of test set up qualification, test cases generation, test cases execution at the software level and at the system level, analyzing the results, generating the test reports and the decision to go ahead with the next iteration or loop in the same iteration till the bugs are fixed. It was observed that the fist iteration was looped twice as there was the initialization of the hardware which did not functional properly. Once this bug was fixed all the other iteration bugs were fixed in the subsequent iteration.

7. COMPUTING THE EFFECTIVENESS OF THE STRATEGY

The effectiveness of the approach can be realized only by comparing the data with the different existing approach on the same application. We used the conventional approach for the same aerospace application. The conventional approach to carry out the V&V is already explained. In this section we talk about the implementation of the application using this approach and compute the parameters like the number of iterations, number of test cases and the time taken for executing these iterations. The process also generates the same artifacts, efforts to qualify the tools, test set up and the same team strength and experience is used for this approach. This hypothetical approach is taken to compare the metrics generated by both the approaches and then quantify the effectiveness of either of the approaches.

The conventional approach is described in the earlier section and the data collected by implementing the V&V process is analyzed. The number of iterations the conventional V&V approach takes for this application is 11 and the functionality covered in each of the iteration is shown below.

Application software 22 functionalities

Iteration1 → System Initialization
Iteration2 → Input, output interfaces +Iteration1
Iteration3 → Input, Output processing +Iteration2
Iteration4 → Built In Tests +Iteration3
Iteration5 → Warnings +Iteration4
Iteration6 → Control Laws +Iteration5
Iteration7 → Monitors +Iteration6
Iteration8 → System Modes +Iteration7
Iteration 9 → System Time Response, Data Logging +Iteration8
Iteration 10 → Failure Management +Iteration9
Iteration 11 → System testing

The conventional approach takes each functional of the application and tests those particular functions till all the bugs are cleared. The next iteration is not executed till the time the current iteration bugs are not fixed resulting in more time needed to execute the iterations. The table below shows the time taken by each of the iteration, number of test cases executed and the looping done at each iteration for the conventional V&V approach.

Some of the major strategy which makes the ‘effective’ approach for V&V effective are discussed. Iterations 2,3,5,8 and 10 loop for more than one time. Since the system is being developed for a safety critical avionics application the looping does not happen in every iteration. This ensures that the process followed for the requirement establishment, test case generation, qualification of the tools and the test set up is good as that of the ‘effective’ approach. The second iteration loops for two times because there are system related bugs which need to be corrected before the next iteration is executed. The same bug is found in the first iteration of the ’Effective’ approach and the analysis showed that the bug needs to be fixed in the current iteration as the subsequent iteration will not perform correctly. The bug was related to the resource clash happening at the hardware communication channels. The looping of other iterations are removed in the ’Effective’ approach by fixing the bug and performing the regression testing in the subsequent iteration. The build released by the design team is such that as we build up the functionality of the system, the earlier functionality can be tested by the V&V team.

The other difference in the proposed approach is of postponing some of the requirement testing at the later stage without effecting the schedule. Many a times the requirement cannot be tested because the entire application is not available, so such requirements are test during the end to end system testing. But at the software test level the partial functionality can be tested for its correctness. This approach helps in preceding further rather than failing the requirement and trying to fix the bug.

Table 2 TimeTakenandtheTestCasesineachiterationfortheConventionalV&Vprocess

Iteration	**Test Cases**	**Time taken**	**Loops**

First Iteration	1061	10	1
Second Iteration	1688	45	4
Third Iteration	1000	35	2
Fourth Iteration	200	15	1
Fifth Iteration	1600	40	3
Sixth Iteration	2577	40	1
Seventh Iteration	1000	35	1
Eight Iteration	64	10	3
Ninth Iteration	270	30	1
Tenth Iteration	175	40	2
Eleventh Iteration	1900	50	1

Table 3 Functionality comparison between the Conventional and the Effective Verification and Validation process

Functionality	**Conv**	**Eff**
Iterations	11	5
Number of Test Cases	12000	11000
Time taken to complete	351 days	180 days

The critical sections of the applications are logical intensive and to check the correctness of these logics the Non-Real-time testing approach is used. These logics are demonstrated by the design team to prove the correct implementation of the critical sections. The test cases and their results are appended with the test cases and the results developed by the V&V team. This will prevent the repetition of the test cases execution. Many of these tests are done at the model levels and using the simulation the results are generated. This approach helps in reducing the time in executing the critical sections by repeating the same tests. The report generated appends the results of the tests carried out at the simulation level and at the system level.

Optimization of the test cases help in generating the optimized test cases covering the entire requirements. Lot of simulations and detail study of the system is required before the optimized set of test cases can be used. This approach reduces the time to execute the test cases and also the numbers of test cases are reduced.

The above mentioned approaches are used in the proposed V&V strategy and the results obtained are discussed in the earlier section. The results obtained using the conventional approach are T compared with respect to the number of iterations, number of test cases and the time taken to complete the V&V process. Note ’Conv’ is used instead of Conventional V&V process and ’Eff’ is used instead of Effective V&V process.

The effectiveness of the two approaches can be seen in the bar chart shown in Figure6

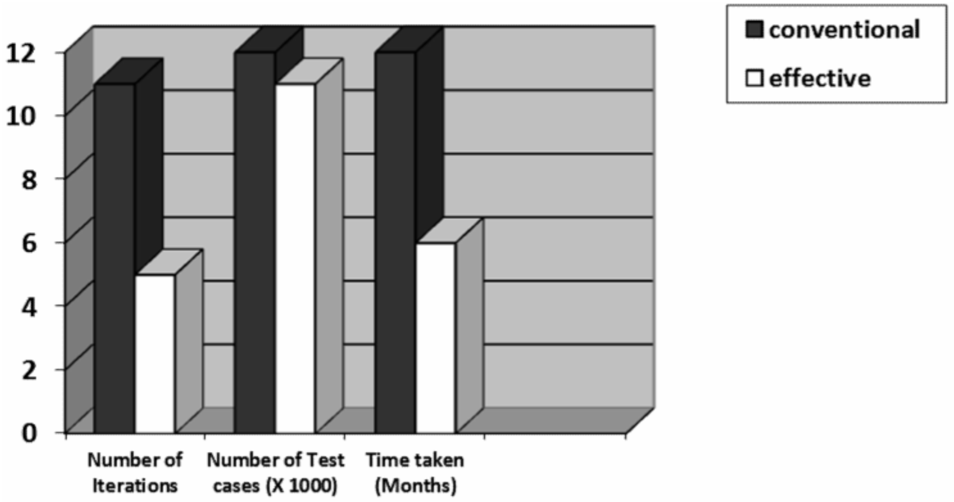

Figure 6 Comparison between the Conventional and Effective Verification and Validation Approach

Figure6 shows the time reduction in carrying out the effective approach strategy maintaining the prescribed reliability of the critical system. The same reliability is obtained with the conventional approach but the time taken is more.

The ’Effectiveness’ of the strategy is justified not only by the data obtained during the V&V process but the system is successfully installed in the aircraft after getting a nod from the certification agency. The success of the V&V strategy for V&V is quantified by the fact that the system is performing as per the requirement during the flight tests. The post-flight data analysis shows that there is no nuisance warning or failure of the system since it is installed in the aircraft.

8. CONCLUSION

This paper presents an implementation efficient strategy to carry out the V&V process. The paper proposes an ’Effective’ strategy to execute the process within the project schedule without effecting the safety and reliability of the system. The strategy proposes a fixed iteration V&V process. In each of the iteration the functionality of the embedded system is added and test by means of test methodologies like static, dynamic, random, non-real time and safety related testing. These testing methodologies are capable of capturing system bugs to improve the testing process. Limiting the number of iterations reduces the time taken by the V&V process for a project. The documentation for each of the iteration also reduces and the time taken to produce the artifacts also reduces.

This approach has a lot of scope in applications where the completion of the project on schedule effects economy and success of the project. The strategy proposed can be implemented for any safety critical embedded system applications.

The strategy proposed and its effectiveness is shown by implementing this approach for an aerospace application. The Verification and Validation data is collected within the required time frame and attaining the same prescribed reliability. The same application is hypothetically tested using the regular approach and the data collected with the time taken to test the safety critical embedded system. The time taken to complete the Verification and Validation of the aerospace application reduces the time by 50% with the strategy proposed maintaining the same probability of success. Based on the experience of carrying out the V&V for safety critical embedded system, a guideline to decide the number of V&V iterations is given. This will help in analyzing the number of iterations to be taken for the system based on this complexity, time schedule of the project and the maximum allowable failures based on the criticality of the application.

REFERENCES

[1] J. Callahan and G. Sabolish, “A process improvement model for software verification and validation,” Proceedingsof19thAnnualSoftwareEngineeringWorkshop, NASA Goddard Space Flight Center, Dec. 1994.

[2] O. Tal, C. McCollin, and T. Bendell, “Reliability demonstration for safety critical systems,” IEEE Transactions on Reliability, vol. 50, no. 2, Jun. 2001.

[3] Software Reliability: A Federal Highway Administration Preliminary Handbook. Federal Highway Administration, Sep. 2004.

[4] M. P. Heimdahl, “Safety and software intensive systems: Challenges old and new,” International Conference on Software Engineering, pp. 137–152, 2007.

[5] M. Barr, “Programming embedded systems.”

[6] Basili,V.R., Selby, and R.W, “Comparing the effectiveness of software testing strategies,” IEEE Transactions on Software Engineering, vol. 13, no. 12, pp. 1278–1296, Dec. 1987.

[7] G.S. Tallant, J. M. Buffington, W. A. Storm, P. O. Stanfill, and B. H. Krogh, “Validation & verification for emerging avionic systems,” chess.eecs.berkeley.edu/hcssas/papers/Storm-HCSS-avionics-positon-paper.pdf.

[8] N. Juristo,A.M, Moreno, ands. Vegas, “Reviewing25 years of testing technique experiments,” Empirical Software Engineering, vol. 1, no. 1-2, pp. 7–44, Mar. 2004.

[9] A. Bertolino and L. Stringi,“On the use of testability measures for dependability measures,” IEEE Transactions on Software Engineering, vol. 18, no. 1, Jan. 1996.

[10] K.W. Miller,L.J. Morell,R.E. Noonan, S.K.Park,D.M. Nicol, B.W. Murrill, and J. M. Voas, “Estimating the probability of failures when testing reveals no failures,” IEEE Transactions on Software Engineering, vol. 18, no. 1, Jan. 1992.

[11] A. Bertolino and L. Stringi,“Predicting software reliability from testing taking into account other knowledge about a program,” Quality Week, May 1996.

[12] P.E. Ammann, S. S. Brilliant, and J. C. Knight, “The effect of imperfect error detection on reliability assessment via life testing,” IEEE Transactions on Software Engineering, vol. 20, no. 2, Feb. 1994.

[13] A.Farooq and R. R. Dumke, “Research directions in verification & validation process improvement,” ACMSIGSOFT Software Engineering Notes, vol. 32, no. 4, Jul. 2007.

[14] A. Bouali and B. Dion, “Formal verification for model-based development,” SAE International, no. 05AE-235, 2004.

[15] Y.V. Jeppu, K. Karunakar, and P. S. Subramanyam, “Testing safety critical ada code using non real time testing,” Empirical Software Engineering,vol.1,no.1-2,pp.7–44,Mar.2004.

AUTHORS

Ms. Manju Nanda: Scientist at CSIR-NAL, Bangalore

Her research interests include design, development and qualification of safety critical embedded systems , formal methods for safety critical systems, software engineering , software engineering and systems engineering life cycle processes

Ms. J Jayanthi: Scientist at CSIR-NAL, Bangalore

Her research interests include design, development and qualification of safety critical embedded systems , software defined radio for aerospace application , software engineering , software engineering and systems engineering life cycle processes