Evaluation Techniques applied for Mission-Critical Software of Satellite Launch Vehicles of ISRO

Dr. B. Valsa
Deputy Director, VSSC (SR)
Thiruvananthapuram

20th February 2017
SQA Organisation in VSSC/ISRO
Flight Software Development Process
Independent Verification & Validation
Reviews & Quality Audits
Best Practices in SQA of ISRO
Process Improvement
Quality in Launch Vehicles

Quality is never an accident, it is always the result of high intention, sincere effort, intelligent direction and skillful execution. It represents wise choice of many alternatives”

“Quality is not an act, it is a habit”

Launch Vehicle Technology
- Complex
- State of the art
- High degrees of accuracy and precision
- Very unforgiving - a small deviation or fault in even one of the thousands of elements/subsystems/software can cause a catastrophic mission failure
Quality System in Launch Vehicles

Strong Quality System, catering to

Mission  Mechanical Systems
Software  Chemicals & Propellants
Avionics  Composites
Based on PSLV-D1 failure (due to a software error) National level Failure Analysis Committee (FAC) recommended an independent third party SQA for on-board software

Formed as an independent Division (SQAD) in 1994 under System Reliability Entity. Renamed as Quality Division Flight Software (QDFS) in 2004

Main Objectives
- Quality Assurance and Independent V& V
- Configuration Management of Software & Data
- Independent Assessment and Certification for Flight
- Software Process Standardization across ISRO Centres

Scope further extended to Mission Design, Ground (Simulation & Check-out) & Telemetry System Software
Software is considered as a sub-system like Avionics, Mechanical etc.

Followed all qualification & clearance procedure similar to other sub-systems
### Software Categories of ISRO Launch Vehicles

#### Flight Software
- Software residing in the on-board computers of Launch vehicles & Spacecraft
  - e.g.: NGC Software, Cryo Control Electronic Software
- Mission- Critical

#### Ground Software
- Software residing in Checkout systems and simulation test facilities of Launch vehicles / Spacecraft
  - e.g: Test Station Console Software, Automatic Launch Sequence Software
- Mission- Critical / Critical

#### Mission Design Software
- Software for mission design/analysis activities of Launch Vehicles / Spacecraft
  - e.g: 6D Simulation Software (ICOMETS)
- Critical

#### FPGA Design Software
- FPGA design using HDL for realization of digital circuit
- Mission- Critical / Critical
Launch Vehicle software development takes place in an environment where:

- Mission Requirements are evolving
- Functional Requirements revised frequently
- Software Changes from mission to mission
- Mission constraints may be changed from time to time
- Mission design parameters often fine tuned to meet the requirements
- Software to be finalized sufficiently early to enable completion of Integrated System Level Test and Evaluation
The **Onboard software** used in Launch vehicle is highly complex

- Has to be highly reliable for efficient performance & error free operation
- More and more of the critical aspects of a Launch vehicle’s design is being implemented in software
- Should provide accurate information on Vehicle states, execute appropriate Sequencing and stage separation events, Guide and Control the Vehicle to inject the satellite into the precise orbit
- Should recover from faults or provide a degraded service in the event of failures
- Success or failure of Satellite Launch Vehicle depends to a great extent on **Quality & Reliability** of On-board Software
Fault Tolerant Features

- No software fault tolerance. Only system level fault tolerance designed to tolerate of 2 non-identical hardware failures.

- Hardware fault tolerance features - Cross strapping at input & output and switchover to a redundant chain:
  - Processor self check
  - Task incompletion
  - Handling of exception conditions and switching over after ‘n’ cycles
  - Sensor failures detection and isolation & reconfiguring
  - Processor synchronization & communication failure and reconfiguring
  - Built-in salvage options in case of severe performance deviations, to redirect the vehicle to the nearest better target to avoid catastrophic mission failure.
  - Robust Module Design to handle unforeseen nature of input conditions.
**Basic On-board Hardware & Software Configuration**

**Hardware**
- Multiple computers in hot redundant mode
- Capable of reconfiguring under failure detection of any Computers

**Basic Software Configuration**
- Real Time Executive (REX) Software: Custom made OS
- Application Software
  - Data Acquisition Software
  - Navigation
  - Guidance
  - Digital Auto Pilot
  - Sequencing
  - Pneumo Hydraulic Algorithm
  - Cryo Stage Control Electronic Software
- Interface Software (REX-Application)
- GAINS & NIM Software

**Language**: ALFA & Assembly
**Target Platform**: Vikram 1601 & i960
**Size**: ~60KLOC
**Init Data**: ~40KBytes
On-board Software Functions

**Navigation**
- Generate attitude quaternion, body rates, acceleration, position and velocity components, FDI of sensors.

**Guidance**
- Generate steering commands, follow DOL-WB at OLG, guidance based events, switching over to OLG steering as mission salvage option.

**DAP**
- Computes control commands to ensure smooth lift off and stabilization of vehicle in pitch, roll and yaw.

**Sequencing**
- Execute vehicle sequencing functions from lift off and detect RTD based on trigger conditions, execute hardware/software commands.

**REX**
- Maintain time synchronization, task scheduling, telemetry, error handling and redundancy management.

Fig.1: Navigation, Guidance, and Control Functions
- Sensing Vehicle acceleration, rates
- Computes Vehicle present attitude, position, velocity with respect to Inertial reference frame.
- Computes desired attitude command to follow the desired trajectory for achieving the target.
- Computes engine deflection commands to steer the vehicle from initial attitude to desired attitude.
- Direct the thrust to generate control force required to steer the vehicle
On-Board Software Progression across ISRO Launch Vehicles

- SW criticality increased manifold from critical to mission-critical & safety-critical
- More design/fault tolerant features in SW
- Complexity increased & more challenges in QA

- Assembly code, CLG (VG guidance), simple Sequencing, minimum Error Handling

- HLL, NGC functions, Double stream Sequencing, Mission Salvage, EGC/MRC Algo, LOX/LH2 pressure regulation, RTDs, Sensor FDI, System re-configuration, scheduling

- HLL, Complex DAP/Sequencing, FE & EE Guidance, EGC/MRC Algo, FDI for EGC-LVDT, LOX/LH2 pressure regulation, Improved Sensor FDI, Dynamic system re-configuration

- RLV-TD

- No application SW. Analog Control, Stored OLG

- Micro, minor & major task scheduling, Navigation, Re-entry phase Control & guidance, Safe-mode guidance, APL Guidance, Sequencing, Improved Sensor FDI, System re-configuration
Software Standardisation in ISRO

- Software Processes across ISRO Centres standardised based on ISRO Software Process Document (ISPD), a tailored version of IEEE12207:2015 - Provides common framework for ISRO software lifecycle processes
- Entire software in ISRO classified into 11 categories based on its Applications.
- Process Model for each category of software well defined along with its expected outcomes, qualification process and review mechanism
- Standardisation of document templates in line with IEEE standard
- Adherence to the Standard ensured by SQA group of respective Centres – process violations raised as non-conformances.

<table>
<thead>
<tr>
<th>Software Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-board</td>
</tr>
<tr>
<td>Checkout &amp; Simulation</td>
</tr>
<tr>
<td>Launch Operations and Test Facilities</td>
</tr>
<tr>
<td>Mission design</td>
</tr>
<tr>
<td>Image /data processing</td>
</tr>
<tr>
<td>Scientific</td>
</tr>
<tr>
<td>Information services</td>
</tr>
<tr>
<td>Spacecraft operations</td>
</tr>
<tr>
<td>System software</td>
</tr>
<tr>
<td>Mobile applications</td>
</tr>
<tr>
<td>FPGA Design</td>
</tr>
</tbody>
</table>
Realisation of Zero defect Flight Software

- Achieved through well defined SW Development Process & Stringent QA and IV&V Activities
- Entire V&V Cycle to be repeated even for incremental changes in flight proven software

Effective V&V

Fault detection & removal
- Code inspection
- Module testing
- Validation tests
- Performance Analysis
- Software Audits

Fault prevention
- Formal reviews
- Design guidelines
- Coding Guidelines
- Adherence to S/w std (IEEE 12207)

Fault tolerance
- System level fault tolerance
- Self tests

Zero defect Software

Right QA & Development Process

By Design

No Software anomaly in any of the Launch Vehicle missions since PSLV-D1
On-board Software Life Cycle Model

- Iterative waterfall model – Based on ISRO Software Standard (ISPD)
- Mission Design as additional phase
- Standardised across ISRO Centres
On-board Software Development Process Model
Major SQA Practices of On-board Software

- Standardization – generation of Plans/Guidelines / Checklists
- Reviews
- Quality Audits
- Root Cause Analysis, Corrective & Preventive Action
- QA Assessment & Clearance
- Continual Improvements based on Lessons Learned

- Verification of Life cycle documents
- Independent V&V
- Simulations
- Performance analysis

- Initialisation data verification & clearance
- RAM Dump Verification

- Configuration Management of Software & Data
Development Phase V&V

Development Cycle

Algorithm Formulation

Software Req

Software Design

Code

Development Cycle

Validation

Peer Reviews

QA Audit

Review MFRT

Peer Reviews

QA Audit

Review DRT-NGC

Peer Reviews

QA Audit

Review DRT-NGC

QA Audit

Review SSRB-NGC

Validation

Peer Reviews

QA Audit

Review MFRT

Peer Reviews

QA Audit

Review DRT-NGC

Peer Reviews

QA Audit

Review DRT-NGC

QA Audit

Review SSRB-NGC

Qualification Cycle

CC Release

IV&V

Simulations

Flight Clearance
Development Phase V&V

- Review of Algorithm, SW Requirements & Design by DRTs
- Detailed Review & Verification of Life Cycle Documents (Formulation, requirements & Design) by Peer Review Teams
- Designer Level Test of Integrated Software (SIP) and Independent Test Results analysis by QA Teams
- Formal review of designer level test results with QA participation
- Audits of Designer level Test cases & results by QA to ensure adequacy and completeness
- Independent Document Assessment by QA
Verification of Requirements & Design

- Verification of Requirements & Design by Peer Review Teams

- Major aspects verified
  - Completeness, consistency & correctness
  - Correctness of Interfaces & Error Handling Requirements
  - Traceability w.r.t parent documents
  - Impact Analysis of Changes
  - Adherence to Guidelines & Documentation Templates

- Proven designs of similar requirements within Centre considered as part of review


- Documents updated based on peer review recommendations & released as new revision/updates - Document Assessment by QA for CC release
Independent Verification & Validation

Software Quality Assurance

SQA Planning

Configuration Management

Static Analysis
Code Inspection
Init. Data Verification
Module Test
SFIT
Simulation

Reviews
Audits
QA Assessment
Performance Assessment
Process Improvements

Quality Assurance
Code Inspection

- Carried out independently by QA - In Team Mode

- Key aspects verified
  - Logical correctness & type inconsistencies
  - Design Traceability
  - Interfaces & Error handling
  - Identification of unreachable or dead codes, unused codes
  - Violation of coding & optimizing guidelines

- Independent inspection & Usage of procedure / checklist
- Verification of software-software interfaces and software-hardware interfaces
- System level impact analysis
- Inspection based on failure modes.
- High level reference documents
Module Level Test

- Independent Testing carried out by QA Team as per Test plan
- Features tested
  - Correctness of functional / error handling requirements, interface
  - Behavior to valid and invalid inputs
  - Protection against overflows
  - Data representation and precision on host platform
  - Performance at boundary
- 100% coverage for requirements, statement & basis path

- Test Cases Review by independent Team other than QA
- Emphasis on requirements based Testing & Structural coverage
- Testing at Boundary Condition
- Modified Condition Decision Coverage (MCDC)
- Test oracle development from functional requirements.
• Objective: To validate the Error Handling features and Hardware specific features of the Software.

• Software instrumented to trigger the faults based on fault handling requirements.

• Test Configuration: Open loop Test with single Target hardware. Redundant Target & all RTs are simulated in RTLinux platform.

• Process Improvement based on Task Incompletion anomaly in one of the PSLV missions (PSLV-C17.)

Test Case Details

<table>
<thead>
<tr>
<th>Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Arithmetic error set by processor</td>
</tr>
<tr>
<td>S</td>
<td>Arithmetic Errors set by Software Library Functions</td>
</tr>
<tr>
<td>M</td>
<td>Memory Error</td>
</tr>
<tr>
<td>L</td>
<td>Self check Error</td>
</tr>
<tr>
<td>E</td>
<td>ACE Read back Error</td>
</tr>
<tr>
<td>T</td>
<td>Task Incompletion</td>
</tr>
<tr>
<td>C</td>
<td>Communication errors</td>
</tr>
<tr>
<td>O</td>
<td>Other Category</td>
</tr>
<tr>
<td></td>
<td>Sequencing Table integrity error</td>
</tr>
<tr>
<td></td>
<td>EEPROM checksum test failure</td>
</tr>
<tr>
<td></td>
<td>LMP set before T0</td>
</tr>
<tr>
<td></td>
<td>LMP not detected after T0 for 1500s</td>
</tr>
<tr>
<td></td>
<td>Integrity failure of Composite Error Flag Mask</td>
</tr>
<tr>
<td></td>
<td>RGTM failure</td>
</tr>
<tr>
<td></td>
<td>Minor and Major task stack growth</td>
</tr>
</tbody>
</table>

Process Flow - SFIT

Instrument Source code to trigger the Fault

Program the OBC with instrumented code

Run the Onboard Computer with required inputs

Analyse the Results

Code to test SECDED

If (clock_tk) = 100) then
TestVar := 16#5555#
Write_Memory(SECDED_ENDIS_PORT,16#AA55#)
TestVar := #5554#
Write_Memory(SECDED_ENDIS_PORT,16#55AA#)
Temp:= TestVar;
end if;
Initialisation Data Verification

- Part of Flight software. Can be tuned independently without affecting software - Possible to finalize & qualify SW much earlier than Data

- Comprises of
  - Design numbers, Open Loop Trajectory
  - Software related flags, counters, limits
  - Sequencing Commands & RTD window-In / window-out timings
  - HW dependent parameters – scale factors, offset, coefficients
  - Address map

- Verified independently by QA against reference - Vehicle Data, High level reference, Init. Data Reference Manual, T&E values

- Correctness w.r.t SW implementation
- Derived parameters & HW related parameters (gain, offset etc) independently computed
- Address map verified against Linker Table independently generated by QA
- Sequencing commands and relay numbers verified against HW User manual
Verification Of FPGA Codes

- **Functional Verification**
  - Exhaustive RTL Functional Simulations to ensure VHDL/Verilog Design as per requirements
  - Test bench evaluated using code coverage matrices. Augmented with 100% statement, branch, condition & FSM coverage. More than 95% Toggle coverage

- **Assertion based verification**

- **Independent Code Walkthrough** - Ensure adherence to design/coding guidelines of ISRO for ASIC/FPGA

- **QA Analysis** - Guideline Violations & Static Timing Analysis

- **Tools used**: Questa SIM (Simulator)
SIMULATIONS

- Closed loop validation at various system levels of nearness to flight environment - Onboard packages as in Flight Configuration
  - Test cases
    - Performance dispersions within ±3 sigma
    - Very large disturbance at staging instants
    - Unmodelled disturbances
    - Guidance & DAP saturation limits
  - Each test case contains different combination of flight conditions – Motor performance, Aero perturbations, CG offset, thrust misalignment

- Major Simulations
  - Integrated Processor Test (IPT)
  - Onboard In Loop Simulation (OILS)
  - Hardware in Loop Simulation (HLS)
  - Actuator in Loop Simulation (ALS)

- Carried out independently by Simulation Agency
- QA Participation as Test Bed T&E, Test case Review & Simulation Results Analysis
An open loop simulation test to

- Validate System Level Error Handling & Fault Tolerant scheme
- Qualify NGC Redundancy Management Scheme
- Evaluate Software performance in the Integrated environment

Input Profile derived from Mission simulation software (ICOMETS).

Reference outputs tapped from ICOMETS
Objective

- Validate SW under normal & extreme flight conditions including failures
- Estimate mission performance under extreme flight environments
- Validate switch over logic, control command generation & selection of proper thrusters
- Finalize the Flight software residing in the respective hardware
Hardware in Loop Simulation (HLS)

Objective

- Evaluate mission performance in closed loop with sensors
- Study effects of sensor errors in the mission
- Evaluate mission performance under sensor failure condition.
**Objective**

- Evaluate mission performance in closed loop with CPP under different flight environment
- Evaluate actuator performance in closed loop environments
- Validation of Actuator models
# Quality Audit in Flight Software

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Audits</th>
<th>Objective</th>
<th>Tool Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Product Quality Audit</td>
<td>SW development in adherence to SQA/V&amp;V Plan</td>
<td>Template</td>
</tr>
<tr>
<td>2.</td>
<td>Compliance Audit</td>
<td>Compliance of Review Recommendations / Actions</td>
<td>Template</td>
</tr>
<tr>
<td>3.</td>
<td>Traceability Audit</td>
<td>End-to-End Traceability.</td>
<td>Template</td>
</tr>
<tr>
<td>4.</td>
<td>Audit of designer level tests</td>
<td>Test case adequacy</td>
<td>Procedure</td>
</tr>
<tr>
<td>5.</td>
<td>Document Assessment</td>
<td>Document compliance w.r.t review recommendations</td>
<td>Guideline</td>
</tr>
<tr>
<td>6.</td>
<td>Audit of SW Integration &amp; program process</td>
<td>Correctness of SW integration &amp; Programming process</td>
<td>Procedure Checklist</td>
</tr>
<tr>
<td>7.</td>
<td>Configuration Audit</td>
<td>SW &amp; data compliance to CM Plan</td>
<td>Template</td>
</tr>
</tbody>
</table>

Audits carried out independently by QA Agency
Product Quality Audit (PQA)

- End-to-End Audit covering the entire Life Cycle Phases
- Reference for Audit: SQA/V&V Plan, Guidelines, Quality Records, Review Minutes
- Audit carried out by IV&V Team. Audit findings presented to CMB as part of Flight Initialisation Data Clearance

Ensures

- Adherence of SW development to SQA/V&V Plan
- Review & CC release status of Documents
- Init. Data Verification & Clearance Status
- Follow up of modifications & its clearance
End to End Traceability Audit

- Ensures traceability from Requirements to Design to Code to Test Cases
- Ensures all requirements are tested and validated.
- Provides additional confidence in the Flight Software Validation.

Lessons Learned from Task Incompletion anomaly in PSLV-C17 Mission.

- Inducted in V&V Cycle from PSLV-C18 & GSLV-D5 onwards
- Methodology: Manual
Online QC in Flight Software

- On-line QC implemented at various checkpoints in Flight software realization
  - SW Integration & programming
  - Init. Data generation & Integration
  - Designer level Test & Simulation Tests
  - TTC / ICU Package Programming

- On-line QC as per QC guidelines

QC Guidelines – Software Integration & Programming

- Ensure CC release of code / data and its clearance.
- Usage of correct version of software by verifying against VDN.
- Ensure integration as per approved Integration Plan.
- System used for software integration and programming shall be standalone and virus protected.
- Minimal manual interventions during software integration and programming.
- Usage of CC released fuse files for programming.
- Ensure EEPROM/PROM dump verifications, monitor mode checks and functional checks before package delivery.
- After PROMing read back the checksum & version numbers.
- Ensure the validity of the calibration of the programmer used for PROMing.
Anomaly observed in software after CC release & cleared for flight (during system level test, global checks) treated as failures and reviewed in FAB

Objective: Root cause analysis and corrective & preventive action

Independent Analysis by QA (in addition to SDAs analysis)

Process non-conformances also addressed
Critical Design Review (CDR) of Flight Software

- CDR conducted for new software – generally applicable to new mission software

- Objective: Review of software design / design constraints, fault tolerant/ error handling features, interface details and its qualification criteria

- Conducted by FRR-NGC (No separate CDR committee) before software finalization.

- Presentations by SDAs and QA in prescribed CDR format
Review Mechanism of VSSC / ISRO

- DRT/FRR-STR
- DRT/FRR-SPS
- DRT/FRR-LPS
- DRT/FRR-SAS
- DRT/FRR-NGC
- DRT/FRR-CPP
- DRT/FRR-MISSION

FRR-LAUNCH VEHICLE
FRR-SATELLITE
FRR-LAUNCH FACILITIES
FRR-TRACKING FACILITIES

MISSION READINESS REVIEW

LAUNCH AUTHORISATION BOARD
Issues specific to flight software

- Latest version software / data may not always be used
- Different initialization data may be prepared for a particular flight. On the launch date, the right data set to be loaded
- Data may depend on factors like the geographical location of the test conducted, calibration coefficients of the hardware elements
- Parameters based on launch azimuth measurements, drift coefficients of inertial sensors, OL Trajectory etc are updated in the last minute

- CM of Software & Data independently carried out by QA - As per CM Plan
- Configuration Audit conducted to track the versions
- Minimum Tool Support
- Internal CC maintained at SDA level

CM very critical for Flight SW
Configuration Management

Best Practices

- A well defined version numbering scheme for tracking the data sets
- A unique file naming conventions for identifying the data sets
- A verification procedure with checklist to ensure latest data only enters these files
- A few minutes prior to launch, automated verification to ensure the correct data has been loaded to the onboard computers by the checkout computers

CM Ensured

- Right Software with right set of data set used in different simulations and for final flight successfully.
- All change control procedures & configuration item identification schemes documented
- Good Tracking of changes
- SCM scheme implemented helped in attaining the required reliable performance of onboard software and data for Launch Vehicles.
Various processes of mission design identified and following QA activities being carried out for each process:

- Verification of design/simulation inputs, drawings, models & outputs
- Assessment of requirement & formulation documents
- Audit of design reviews actions/recommendations
- Review of documents and configuration control of mission Design SW.

Product Quality Assessment Report, consolidating all QA activities, released before every mission.

Based on the observation during QA activities, wherever necessary, recommendations were made to improve the process.

Quality Assurance Plan for Mission Design
<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Deliverables</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Plans, Procedures &amp; Guidelines</td>
</tr>
<tr>
<td>2</td>
<td>Inspection Reports</td>
</tr>
<tr>
<td>3</td>
<td>Test Case Design &amp; Test Results Document</td>
</tr>
<tr>
<td>4</td>
<td>Anomaly / Non-conformance Reports</td>
</tr>
<tr>
<td>5</td>
<td>Failure Analysis Reports</td>
</tr>
<tr>
<td>6</td>
<td>QA Audit Reports</td>
</tr>
<tr>
<td>7</td>
<td>QA Assessment Reports</td>
</tr>
<tr>
<td>8</td>
<td>Document Assessment report</td>
</tr>
<tr>
<td>9</td>
<td>Simulation Analysis Reports</td>
</tr>
<tr>
<td>10</td>
<td>Initialisation Data Verification Reports</td>
</tr>
<tr>
<td>11</td>
<td>QA Clearance Certificate &amp; QA Alerts</td>
</tr>
<tr>
<td>12</td>
<td>Impact Analysis</td>
</tr>
<tr>
<td>13</td>
<td>Configuration Audit Report</td>
</tr>
</tbody>
</table>
Continual Process Improvement

- DRT recommendations compiled & followed up by QA
- End to end signal flow from control command generation to actuators prepared for reference by all concerned for verifying the interfaces
- Sign check committee monitors the signals as per the checklist prepared from EID details of packages.
- Simulation agencies connecting the different hardware packages as per the EID
- T&E of test bed as per the checklist
Continual Process Improvement …

- Designer level tests strengthened & results reviewed
- Independent testing by QA augmented to carry out hardware dependent features testing (SFIT)
- End-to-End Traceability Audit to ensure all requirements are validated
- On-line QC at major checkpoints in flight SW realization
- Revised the EEPROM/PROM Programming Checklist to include Package clearance status & programming directory
Method of Achieving Zero Defect Software

- Adopt Best Software Engineering practices
- Strict Adherence to Software Development Process
- Implementation of Changes in a controlled manner
- Effective Configuration Control & Better Traceability
- Independent QA, Reviews & Exhaustive simulations
- Corrective & Preventive actions
- Analysis of Recurring deviations
- Issues & Lessons learned and its implementation
- Extensive training in Software development
CONCLUSION

- By defining a set of best practices, we have more chances of successful completion of Missions.
- Continual improvement should be the driving force.
- The best practices cited are evolved over years, and is not the end.

RESULT – 38 Successive Successful PSLV Missions
Thank You