AEROSPACE
Assuring Space Mission Success

Welcome

Honoring the Legacy. Assuring the Mission.
50 YEARS

NPOESS Flight and Ground Software Verification: Insights, Issues and Lessons

Daniel R. Vanderwarker
The Aerospace Corporation

Introduction

Background and Perspective

- Author is a software/systems engineer with over 25 years experience in all phases of ground and flight systems software acquisition
- Joined the Aerospace Corporation in 1987
- Has supported NPOESS ground and sensor development programs since 2001
- Extensive experience in software verification and validation activities in support of NPOESS
- Has supported numerous NPOESS program milestones in the areas of software development and software/systems requirements verification
- Has represented the NPOESS Integrated Program Office (IPO) as both technical reviewer and as test witness as part of government-contractor teams

Background and Perspective (cont)

- Author has been immersed in software verification activities in support of the NPOESS Program for over 4 years
  - Role of test witness/observer as representative of the NPOESS Integrated Program Office (IPO) team
  - Participation in requirements and software verification activities for NPOESS ground and flight software
    - Preliminary Configuration Audits (PCAs) for NPOESS ground and flight software
    - Functional Configuration Audits (FCAs) for NPOESS ground and flight software
    - Formal Qualification Testing (FQT), Factory Acceptance Testing (FAT), and Segment Acceptance Testing (SAT) activities for NPOESS ground software
    - Requirements "Sell-off" verification effort for NPOESS flight (sensor performance) software requirements
    - Critical Design Audits (CDAs) for NPOESS ground software
- View this presentation as an opportunity to "step back" and share some of the insights, issues and lessons that I have derived from my involvement in these NPOESS software verification activities

Purpose of Presentation

- Summarize and describe selected software verification efforts performed in support of the National Polar-orbiting Operational Environmental Satellite System (NPOESS) program
- Identify a set of insights, issues and lessons derived from my involvement in these verification efforts as a member of the NPOESS Integrated Program Office (IPO) team
- Where feasible, frame the above insights, issues and lessons within the context of flight and ground software verification
  - Suggest some common themes
- Discuss possible implications based on these experiences

NPOESS Overview

- The National Polar-orbiting Operational Environmental Satellite System (NPOESS) is a system of polar orbiting weather satellites and ground equipment used for collecting global multi-spectral radiometry and other specialized meteorological, oceanographic, and solar-geophysical data

- NPOESS disseminates the above data to field users deployed worldwide, the environmental remote sensing community, and to the NPOESS system's four operational Centrals
  - National Environmental Satellite Data and Information Service/National Center for Environmental Prediction (NESDIS/NCEP) located at Suitland, MD
  - Air Force Weather Agency (AFWA) located at Offutt Air Force Base (AFB), Omaha NE
  - Navy Oceanographic Office (NAVOCEANO) located at NASA Stennis Space Center, Bay St. Louis, MS
  - Fleet Numerical Meteorology and Oceanography Center (FNMOC) located at Monterey, CA

- NPOESS system is composed of the following five segments
  - Space Segment (SS)
  - Command, Control and Communications Segment (C3S)
  - Integrated Data Processing Segment (IDPS)
  - Launch Support Segment (LSS)
  - Field Terminal Segment (FTS)

Focus of NPOESS Verification Efforts

- Flight Software: Ozone Mapping and Profiler Suite (OMPS) Instrument
  - Preliminary Configuration Audit (PCA)
  - Functional Configuration Audit (FCA)
  - Requirements Sell-Off Activity (Pre-Ship Review Acceptance)
    - Compliance and verification examinations conducted for a large number of OMPS sensor performance requirements
    - This activity helped to drive OMPS to pre-ship review acceptance by the NPOESS IPO and the Government Independent Review Team

- Ground Software: Integrated Data Processing Segment (IDPS)
  - Formal Qualification Testing (FQT)
  - Factory Acceptance Testing (FAT)
  - Segment Acceptance Testing (SAT)
  - Post-SAT Preliminary Configuration Audit and Delta PCA

OMPS Characterization

- The Ozone Mapping and Profiler Suite (OMPS) system provides NPOESS users with data products describing the vertical, horizontal and temporal distribution of ozone in the Earth's atmosphere
  - These data products are known as Environmental Data Records (EDRs)
  - OMPS EDRs are derived from the space-borne ultraviolet and visible observations of two sensors
    - Nadir viewing sensor
    - Limb viewing sensor
  - These two sensors, together with the interface and control electronics, comprise the OMPS sensor suite
  - Calibrated and uncalibrated sensor data are also provided to NPOESS users in the form of Raw Data Records (RDRs) and Sensor Data Records (SDRs)
- OMPS is comprised of three subsystems
  - Sensor suite
  - EDR and SDR generating algorithms
  - Ground support and test equipment necessary to verify the suite performance

IDPS Characterization

- The Integrated Data Processing Segment (IDPS) is one of the five segments comprising the NPOESS system
- IDPS consists of ground hardware and software that
  - Ingests Stored Mission Data (SMD) provided by the Command, Control and Communications Segment (C3S)
  - Converts the SMD into Raw Data Records (RDRs)
  - Processes the RDRs to create Sensor Data Records (SDRs), Temperature Data Records (TDRs) and Environmental Data Records (EDRs)
  - Provides these data records to the four weather processing Centrals on a time critical basis
- IDPS also provides support for calibrating algorithms, validating EDRs against externally generated truth data, and monitoring satellite sensor performance
- The Data Processing Element (DPE) is the sole component of the IDPS, and resides at the weather-processing Centrals
  - DPE provides the common core functionality for both the IDPS and the Field Terminal Segment (FTS)

Flight Software Verification Focus

Flight Software Verification: Insights Gained

- A requirements database audit is valuable for verifying the completeness and accuracy of flight software requirements as part of a coordinated sell-off strategy
  - Helps ascertain and verity that requirements content, format (structure) and traceability objectives have been satisfied
  - Helps ensure that each software requirement has been assigned an acceptable verification method
- It is risky to assume that requirements waivers will definitely be accepted by the program for which the requirements sell-off process is being conducted
  - If a waiver is found to be unacceptable, it can be very difficult to satisfy the original requirement at a later stage

Flight Software Verification: Insights Gained (continued)

- A requirements sell-off review process can benefit from explicitly defining the “things to look for” (e.g., criteria) during the review and evaluation of requirements for approval and closure
  - Wording of [each] requirement for completeness and accuracy
  - Verification method identifier
  - Reference material (review artifacts) identified/provided to assist evaluation of each requirement, along with location pointers to assist artifact review
  - Dialogue, rationale and justification presented by each reviewer concerning closure decisions
  - Previously documented closure recommendations concerning approval or non-approval of each requirement
- Defining and utilizing the above criteria can serve to facilitate team coordination and collaborative information sharing during the review and sell-off effort

Flight Software Verification: Insights Gained (continued)

- Implementing a rigorous, comprehensive and well-defined process for incorporating known problem change requests (PCRs) and lessons learned from previous ("heritage") software builds into software development for "follow-on" (derived) builds can increase confidence in the quality of derived build, and can be expected to
  - Accelerate the software integration process, if implemented carefully
  - Facilitate the early identification of software and integration risks
- All new releases of flight software (FSW) and ground software element (GSE) software changes should require a Delta Test Readiness Review (TRR) / Formal Qualification Test (FQT) to verify
  - That changes made since the previous release of the software were successfully tested
  - That these changes do not impact the functionality of the software

Flight Software Verification: Issues and Risks Identified

- Incomplete or inadequate specification of reference “pointers” for locating and examining the documentation (review artifacts) needed for evaluating the requirements as part of the flight software approval (“sell-off”) process
  - For example, when specific sections or paragraphs of review artifacts were not completely or precisely specified, this impeded the requirements sell-off process
- Decentralized (“fragmented”) approach among technical reviewers; absence of face to face technical working meetings for the review and closure of flight software requirements
  - Email correspondence and weekly teleconferences (decentralized, or “fragmented” review approach) not entirely sufficient
  - Face to face working group meetings (centralized review approach) were necessary to complete the FSW sell-off process
- Sell-off requirements that reviewers determined to be confusing due to ambiguity, completeness or syntax (e.g., how they were structured / worded)
- Tendency to address and close the “easy” software-related requirements and to defer (or ignore) the more difficult or challenging verification issues

Flight Software Verification: Adjustments and Risk Mitigation

- Implementation of weekly teleconference meetings to review sell-off closure status, discuss issues, and resolve inconsistencies or areas of confusion/conflict
- Mandating and implementing technical interchange meetings (face to face working meetings) to expedite the review and approval for sell-off of a large number of software (and related) requirements
- Insistence that specific documentation “reference pointers” be provided to the flight software sell-off review team to allow a faster, more thorough and efficient review process for the requirements to be evaluated for approval and closure decisions
  - Reviewers were requested to provide and share information on which documentation artifacts (and where within the artifacts) that they based their decisions on closing any given requirement (page number, paragraph, appendix, etc.)
  - This encouraged a more streamlined, faster and efficient review process
  - Insisted on and provided more efficient cross-referencing between requirements and their corresponding documentation and review artifacts

Flight Software Verification: Lessons Learned

- Care must be taken in the structuring and wording of requirements that must eventually be reviewed and approved as part of a requirements sell-off activity
  - Each requirement must be worded so as to be unambiguous, as unique as possible (e.g., non-redundant with other requirements) and explicitly verifiable
    - Verification method language must be clear and unambiguous
- Good documentation is essential for the requirements sell-off process
  - Contractor/subcontractor must document their design well, so that it will be easier to trace all designs, even when the original designer has left the company
- Contract should be written such that the contractor/subcontractor is accountable for the risk of their design and responsible for developing the risk mitigation plan (rather than the customer)
- Requirements needing waivers should be identified early and evaluated as soon as they are available
  - Such requirements should be promptly communicated

Flight Software Verification: Lessons Learned (continued)

- A requirements verification team should be involved early in the sell-off process to better facilitate their understanding of system/software design, so that the verification can be thoroughly and efficiently performed
  - Early review of acceptance test plans and test data for the requirements
  - Otherwise, requirements that have been closed may have to be later reopened for further investigation when additional clarifications are made available
    - Can adversely impact schedule
- When reviewing and evaluating a large number of requirements for the purpose of approval and sell-off, group and face to face technical meetings are essential activities
  - Technical interchange meetings (TIMs) can be very effective as part of the requirements sell-off process
  - These TIMS help to facilitate and expedite the review and approval for closure (e.g., sell-off) of a large number of requirements
  - Contribute toward an efficient, focused and collaborative working process
  - Easier to review each requirement and compare it against relevant documentation for evaluation and closure (or to determine that further review is necessary)

OMPS Flight Software Verification: "What Was Positive"

- Value that was realized through a combination of individual review, information sharing, and team collaboration during the review and evaluation of a large number of sensor performance requirements
- Fact that such a large number of requirements could be reviewed for "selloff" in a relatively short time frame
- Spirit of cooperation and camaraderie which contributed to the success of this review effort
- Increased insight into sensor performance and interface considerations specific to OMPS program
- Compilation of a large amount of review information into a shared database repository
  - Overall, documentation package to be reviewed was determined to have been complete

OMPS Flight Software Verification: "What Needed Improving"

- Reference pointers for the requirements to be reviewed were at first incomplete and inadequate
  - For example, reviewers often had to ask "exactly where within this particular specification is this requirement (that I have been assigned or volunteered to review) located?"
  - "Pointer" information was initially unsatisfactory
- Compilation of a large amount of review information into a shared database repository
  - This was found to be a challenge until better reference ("pointer") information was provided
- Review determined that some requirements had been structured/worded as to appear (in some cases)
  - Confusing and ambiguous
  - Redundant with other, similar or related requirements

Ground Software Verification Focus

Ground Software Verification: Insights Gained

- Test procedure dry runs can significantly contribute to the smooth and efficient conduct of the actual Run for Record (RFR) event
  - During software verification activities, an Integrated Program Office (IPO) test witness team must “tread a fine line” between the informality of a dry run and the formality of a customer-observed Run for Record (RFR) event

- As previously noted, implementing a rigorous, comprehensive and well-defined process for incorporating known problem change requests (PCRs) and lessons learned from previous (“heritage”) software builds into software development for “follow-on” (derived) builds can increase confidence in the quality of derived build, and can be expected to
  - Accelerate the software integration process, if implemented carefully
  - Facilitate the early identification of software and integration risks

- All new releases of ground software element (GSE) software changes should require a Delta Test Readiness Review (TRR) / Formal Qualification Test (FQT) to verify
  - That changes made since the previous release of the software were successfully tested
  - That these changes do not impact the functionality of the software

Ground Software Verification: Issues and Risks Identified

- Lack of sufficient detail provided with respect to test input data and expected outcomes for verifying satisfaction of requirements
- Numerous and pervasive test procedure “red lines” observed during Dry Run qualification activities
- Numerous and pervasive test procedure “red lines” observed during the execution of certain Run for Record (RFR) qualification tests
- In certain instances, incomplete tester familiarity with and understanding of the test procedures being run, to the extent that underlying design and implementation of the software did not always appear known or fully understood
  - Instances of testers having to consult with the developers (during a test event) to more fully understand system design or what functions were being addressed as part of the verification event
- Lack of depth in certain test cases and test procedures
- Insufficient time allowed for the review of test cases and test procedures prior to test execution

Ground Software Verification: Issues and Risks Identified (continued)

- Test procedures in need of substantial modification during dry run verification activities, either because they were incomplete or out of date
  - Lack of updates since previous software build
  - Failure to reconfigure system for new software build
  - New requirements imposed since previous software build
- During software qualification dry runs, not all expected results appeared to be well written
  - Situations where expected results looked like test procedure steps
  - Situations where test procedure steps looked like expected results
  - Situations where expected results were loosely worded or confusing
- In certain situations, contractor not having followed all established processes and procedures inherent in software peer reviews

Ground Software Verification: Issues and Risks Identified (continued)

- Within the context of a Post Segment Acceptance Testing (SAT) Functional Configuration Audit (FCA), the following observations were determined to be risks
  - "As Run" test procedures that did not support verification of certain requirements
  - Procedural steps referenced in the test procedures that did not correspond to steps actually appearing in the As Run version of the test
    - Sometimes difficult to locate evidence that a requirement successfully satisfied its criteria required for verification
  - Certain As Run test procedures that were devoid of Tester initials and Quality Assurance (QA) stamps
  - Certain As Run test procedures containing procedural sequences that were neither "checked off" by the Test Conductor nor stamped by QA (with no explanation provided)
  - Requirements appearing in the Requirements Verification Configuration Matrix (RVCM) for which no Test Report was identified
  - Multiple versions of a given, As Run test procedure, making it sometimes confusing to efficiently locate requirements that were addressed and verified as part of the test (versioning control issue)

Ground Software Verification: Adjustments and Risk Mitigation

- Workaround procedures were developed and implemented to address and resolve problems encountered during Dry Run activities for the verification of ground software
- Recommended that prior to commencement of Dry Runs, all procedures necessary for setting up the test environment be thoroughly documented, including
  - Noting which steps are applicable to the Build software testing (e.g., Qualification and Segment Integration) and which steps can also be used for FAT and SAT
  - Encouraging testers in the Test Team to review all (or as many as possible) of the test procedures as a quality check for the early detection of problems
- Recommendation that test team aggressively look for opportunities for using scripting and “self-scoring” techniques to streamline / expedite the test process
- Encouragement of “cut and paste” techniques for test procedures in situations where scripting either not possible nor appropriate, recognizing that this is an effective technique that can reduce data entry errors

Ground Software Verification: Lessons Learned

- Test procedure dry runs can significantly contribute to the smooth and efficient conduct of the actual verification event (e.g., Run for Record)
- When planning software development as a series of “builds” or “baselines” it is important to carefully maintain distinctions from one build or baseline to the next
  - Important from a configuration management perspective
  - Important as software changes are “merged” from one build or baseline into the next, so as to preclude unplanned or unintended changes or regression effects
- A sound approach for the verification of ground (and flight) software includes the use of unambiguous pass/fail criteria
- When carefully implemented, scripting of test procedures can be effectively used to eliminate manual steps, thereby expediting the verification process while improving repeatability

Ground Software Verification: Lessons Learned (continued)

- Implementing a “freeze date” for specifying requirements is useful in terms of avoiding requirements “creep” as the software or program evolves toward the verification phase
- Prior to a formal software qualification test, the system should be “wiped clean,” brought up from scratch, audited by QA, and should subsequently be locked down for the duration of the test
- Test procedure redlines from dry run verification events must be thoroughly and carefully incorporated into the test procedures for future Dry Run or Run for Record (RFR) events
- Verification method(s) for each requirement as defined in an Interface Control Document (ICD) must be consistent with the corresponding method(s) identified for these requirements in the Verification Cross Reference Matrix (VCRM)
- Software peer reviews are important as the verification team prepares for FQT, Factory Acceptance Testing (FAT), or Segment Acceptance Testing (SAT)

IDPS Ground Software Verification "What Was Positive"

- Sense of camaraderie demonstrated by contractor responsible for conducting the IDPS verification program
  - Demonstrated ability to work cooperatively toward resolving software verification issues during both the Dry Run and RFR activities
  - Long hours were often required and put in by testers
- Contractor's demonstrated ability to develop and implement "workaround" solutions for issues encountered during the test process
- Contractor adhered to and followed through on their established verification processes (known as "Work Instructions")
- Demonstrated ability of IPO team to engage collaboratively and share information
- Successful verification of a large number of requirements for a large and significant ground system effort
  - And evaluation of the various software artifacts related to IDPS software verification

IDPS Ground Software Verification "What Needed Improving"

- Initial readiness of test procedures with respect to software verification activities
  - Especially specific to the Dry Run procedures
  - However, certain of the test procedures required substantial modification ("redlining") during RFR as well
- Ensuring that the proper (e.g., current) software configuration was loaded onto the system prior to commencement of certain of the tests
  - Sometimes encountered a lack of updating with respect to prior software builds
- Test process could sometimes have benefitted from more review time as devoted to the review, evaluation and improvement of test procedures prior to test execution
- More review time should have been provided with respect to the requirements to be verified in advance of the actual test activities

Wrap-up and Conclusions

NPOESS Software Verification Experiences: Some Common Themes

- Precision is important when defining and structuring software requirements and verification criteria
- Value of software technical reviews and audits
- Importance of Dry Run activities as applied to software verification
- Importance of team collaboration and information sharing
- Value of direct (face to face) technical interchanges as part of a coordinated requirements sell-off and software verification strategy
- Importance of configuration management and version control as part of software verification activities
- Early and extensive system/software involvement by the verification team is helpful in terms of software verification readiness and execution
- Verification artifacts must be reviewed and evaluated strategically and systematically
  - Reference pointer and "what to look for" criteria (e.g., guidance) important

Summary and Implications

- Author has summarized a set of insights, issues and lessons derived from his involvement in these verification efforts as a member of the NPOESS Integrated Program Office (IPO) team
- The above experiences, insights, issues and lessons suggest and bear out the importance of
  - Defining and implementing a disciplined software test planning process
  - Addressing program considerations, collaboration, and coordinated information sharing as essential elements in the requirements and software verification process
  - Early and thorough test planning
- The above has hopefully provided a somewhat detailed glimpse into the
  - Importance of detailed review and analysis as part of software verification
  - Resource-intensive nature inherent in the process of software verification

AEROSPACE
Assuring Space Mission Success

Thank you

Honoring the Legacy. Assuring the Mission.
50 YEARS