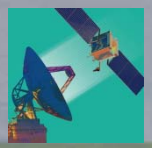
An Increase in Software Testing Robustness: Enhancing the Software Development Standard for Space Systems

Karen Owens and Suellen Eslinger
Software Engineering Subdivision



15th Ground System Architectures Workshop (GSAW)

Computers and Software Division
The Aerospace Corporation

Outline

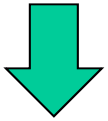
- Software mission assurance
- Evolving software testing requirements
- Lessons learned
- Levels of software testing
- Proposed enhancements for software testing
  - Consistency across test levels
  - Parallel subsections
  - Definitions
  - Operations-like test environment
  - Enumerated requirements
  - Reduce redundant requirements

Software Mission Assurance

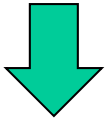
- **Software mission assurance requires two essential components:**
  1. **Building quality in throughout the entire development life cycle**
     - **Using techniques focused on finding and removing defects within each development activity**
     - **Example techniques: peer reviews, product evaluations, joint technical reviews, and software quality audits**
  2. **Conducting a robust software test program**
     - **Focused on finding defects that escaped the quality gates for earlier software development activities**
- **This paper focuses on the second component**
  - **Requirements proposed for the next revision of the Software Development Standard for Space Systems (SDSSS) targeted toward ensuring that a robust software test program is implemented**

Evolving Software Test Requirements

- **MIL-STD-498, "Software Development and Documentation" updated in**



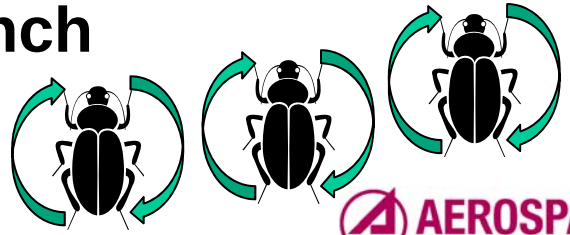
- **R. J. Adams, S. Eslinger, P. Hantos, K. L. Owens, L. T. Stephenson, J. M. Tagami, and R. Weiskopf. "Software Development Standard for Space Systems." TOR-2004(3909)-3537 Revision B, The Aerospace Corporation, 11 March 2005, SMC-S-012**



- **Proposed enhancements for Revision C**

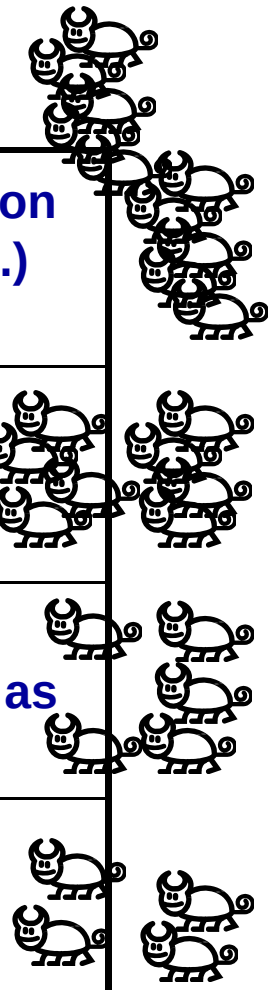
Lessons Learned

- **Requiring standards on contracts improved testing rigor**
- **However, testing still needs <u>more</u> improvement**
  - **Testing starts late due to many factors (e.g., trying to finish sooner without mission assurance measures)**
  - **Quality of software entering various levels of testing is often lower than expected**
  - **Contractor and Acquirer discuss how much testing is enough – late in the testing**
  - **Later phase testing discovers defects that could have been found in peer reviews or unit testing**
  - **Test and fix cycle keeps going, thus delaying system integration, and in some cases, launch**



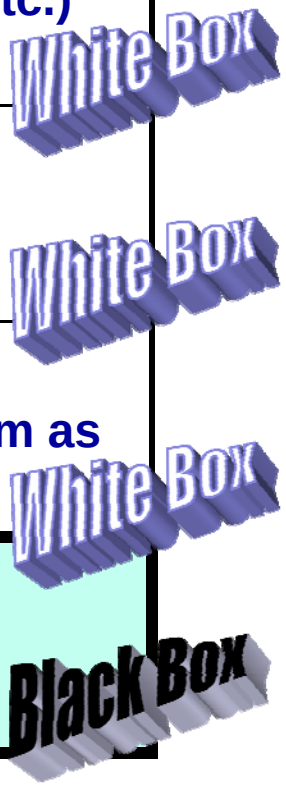
Four Required Levels of Software Testing

Software Unit Testing	**Testing to verify that the implementation of the software unit (code, scripts, etc.) performs as designed**
Software Unit Integration and Testing	**Testing to verify that the integrated software units perform as designed**
Software-Hardware Integration and Testing	**Testing to verify that the integrated hardware and software items perform as designed**
Software Item Qualification Testing	**Testing to verify that the software satisfies its requirements**



Four Required Levels of Software Testing

Software Unit Testing	Testing to verify that the implementation of the software unit (code, scripts, etc.) performs as designed
Software Unit Integration and Testing	Testing to verify that the integrated software units perform as designed
Software-Hardware Integration and Testing	Testing to verify that the integrated hardware and software items perform as designed
Software Item Qualification Testing	Testing to verify that the software satisfies its requirements



Proposed Enhancements

- Divide testing requirements at each level into consistent subsections to reduce redundant requirements
  1. Independence in testing (only for qualification testing)
  2. Testing on the target computer system
  3. Preparing for testing
  4. Dry run of testing (only for qualification testing)
  5. Performing testing
  6. Analyzing and recording test results and analysis
  7. Regression testing
  8. Revision and retesting
- Define several terms
- Add more operations-like test environment requirements
- Split multipart “shall”s into separate statements or lettered items

Note: Test planning and traceability are not discussed in this briefing

Problem: Inadequate number of types of test cases and number of test cases

- Test coverage was neither complete nor representative of the range of operational situations
- Contractor and Acquirer discuss whether the set of test cases was complete – LATE in the testing cycle
- Solutions
  - Involve testers to review requirements during requirements development
  - Define “representative set”, “equivalence class”, and “reusable software”, and other terms
  - Determine test case completeness during development and test planning
  - Review the representative sets and tests in the test plans and descriptions

Definitions - 1

Representative set.

- **A set of values that is representative of the distribution of values or conditions**
- **The size of the set (i.e., the quantity of the values) within each equivalence class is dependent on the software item in question and the required level of confidence for successful execution**
- **The size of the set is defined in:**
  1) **the acquirer-approved software development plan, and**
  2) **the software unit, integration, and qualification test plans based on the number of equivalence classes**
- **As a default, the data sample should be of a size such that a 90% confidence* of successful execution can be established for each equivalence class given that no failures are observed during the testing**

*** Reference: National Institute of Standards and Technology (NIST), "Engineering Statistics Handbook", Chapter 1 and Chapter 6, Section 2**

Definitions - 2

- **Equivalence class. An input set ("class") in which all elements cause the same ("equivalent") execution path, regardless of which element from the class is chosen.**

- **Reusable software. Software developed for one use but having other uses, or developed specifically to be usable on multiple projects or in multiple roles on one project. Each use may include all or part of the software product and may involve its modification. Examples of reusable software include, but are not limited to:**
  - **pre-existing developer software**
  - **software in reuse libraries**
  - **Government Off-The-Shelf (GOTS) software**
  - **acquirer-furnished software**
  - **open source software (OSS) and**
  - **Commercial Off-The-Shelf (COTS) software**

Consistency Across Test Levels

Using representative sets of nominal and off-nominal conditions the test cases shall cover, as a minimum, correct execution of all:	**Unit Testing**	**Unit Integ and Testing**	**Software-Hardware Integ and Testing**	**Software Item Qual Testing**
Algorithms	**X**	**NEW**	**NEW**	
Software requirements allocated to the [software unit(s) (or portion thereof), hardware unit(s), software item] under test	**NEW**	**X**	**X**	**X**
Statements and branches	**X**			
End-to-end functional capabilities through the [software units, software items, hardware items] under test		**X**	**X**	**X**
Interfaces **among** the software [and hardware units or items] under test	**X**	**X**	**X**	**X**
Software interfaces **external** to the [unit or software item] under test	**NEW**	**NEW**	**NEW**	**X**
[Integrated] error and exception handling across the [software units, hardware units, software item] under test	**Error & exception handling within unit**	**Integrated**	**Integrated**	**Integrated**
Fault detection, isolation, and recovery handling (e.g., fault tolerance, fail over, data capture and reporting)	**NEW**	**X**	**X**	**X**

Note: **Bold text** in first column indicates changes or additions.

Consistency Across Test Levels (Cont.)

Using representative sets of nominal and off-nominal conditions the test cases shall cover, as a minimum, correct execution of all:	**Unit Testing**	**Unit Integ and Testing**	**Software-Hardware Integ and Testing**	**Software Item Qual Testing**
Start-up, termination, and restart (when applicable)	**X**	**X**	**X**	**NEW**
Performance testing, including timing and accuracy requirements		**X**	**X**	**NEW**
Resource utilization measurement (e.g., Central Processing Unit (CPU), memory, storage, bandwidth)		**X**	**X**	**X**
Stress testing, including worst-case scenarios **(e.g., extreme loads, frequency of inputs and events, large number of users, simulated failed hardware, missing interfaces)**		**MOD**	**MOD**	**MOD**
Software specialty engineering requirements (e.g., supportability, testability, dependability, reliability, maintainability, availability, safety, security, and human system integration, as applicable), including, in particular, verification of software reliability requirements		**NEW**	**NEW**	**X**
Endurance testing using normal and heavy operational workloads		**NEW**	**NEW**	**NEW**

Note: **Bold text** in first column indicates changes or additions.

More Operations-like Test Environment

Requirements by Level	Unit Integ and Testing	Software-Hardware Integ and Testing	Software Item Qual Testing
Perform testing using the target computer system	X[1]	X	X
Have the target computer system be as close as possible to the operational target hardware	X	X	X
Configure the target computer system to be as close as possible to the operational configuration	X	X	X
Conduct all testing under conditions as close as possible to those that the software will encounter in the operational environment (e.g., operational data constants, operational input and output data rates, operational scenarios)			X
Use actual interfaces wherever possible	NEW	NEW	X
If using actual interfaces is not possible for software item qualification testing, then use simulations of the interfaces	NEW	NEW	MOD validated high-fidelity simulations

[1] Unit I & T may begin in the development environment, but it generally transitions to the target computer system in the software I & T environment as larger sets of software and the hardware become available.

More Operations-like Test Environment (Cont.)

Requirements by Level	Unit Integ and Testing	Software-Hardware Integ and Testing	Software Item Qual Testing
Perform testing with the entire software item under test (including newly developed software, COTS software, and all other modified and unmodified reusable software), installed in the target computer system			**NEW**
Perform testing with the target computer system in the operational software configuration, including all other software executing on that system in addition to the software item under test (e.g., operating system, COTS software, and other software items)			**NEW**
The target computer system and configuration used for testing is subject to approval by the acquirer			**NEW**
The operational environment for testing (e.g., operational data constants, operational input and output data rates, operational scenarios) is subject to approval by the acquirer			**NEW**

Note: **Bold text** in first column indicates changes or additions.

Enhancements for Software Unit Testing Reusable Software

- The following reusable software within the unit shall be tested as part of unit testing:
  a) all modified reusable software
  b) all reusable software where the track record indicates potential problems (even if the reusable software has not been modified)
  c) reusable software which has record of being inadequately unit tested and
  d) all critical reusable software (even if the reusable software has not been modified).
- **When source code for reusable software is not available, then that reusable software is not required to be unit tested**

Conclusion

- **A robust software testing program is an essential component of software mission assurance**
- **The proposed updates to Software Development Standard for Space Systems", TOR-2004(3909)-3537, Revision B contain additional software testing requirements to ensure that a robust software testing program is implemented**
  - **Over and above the original requirements in MIL-STD-498 and**
  - **Over and above those in Revision B**
- **These additional software testing requirements are designed to improve software mission assurance, based on**
  - **Experience from multiple space programs**
  - **Documented results from the software engineering literature**

Feedback?

- **We welcome your feedback**
  - **In the software testing workshop later in the week**
  - **Using our contact information on the last slide**
  - **At a break**

Karen Owens, Senior Project Leader
Software Acquisition and Process Department
Suellen Eslinger, Distinguished Engineer
Software Engineering Subdivision

Computers and Software Division
The Aerospace Corporation
2011-03-30
karen.l.owens@aero.org, suellen.eslinger@aero.org
310.336-5909, 310.336-2906

Backup Charts

AEROSPACE

Mission Assurance Definitions

- **Mission success (MS)** is the achievement by an acquired system (or system of systems) to singularly or in combination meet not only specified performance requirements but also the expectations of the users and operators in terms of safety, operability, suitability and supportability.
  Mission success is typically evaluated after operational turnover and according to program specific timelines and criteria, such as key performance parameters (KPPs). Mission success assessments include operational assessments and user community feedback.
- **Mission assurance (MA)** is the disciplined application of general systems engineering, quality, and management principles towards the goal of achieving mission success, and, toward this goal, provides confidence in its achievement.
  MA focuses on the detailed engineering of the acquired system and, toward this objective, uses independent technical assessments as a cornerstone throughout the entire concept and requirements definition, design, development, production, test, deployment, and operations phases.

Reference: TOR-2007(8546)-6018, Revision A, "Mission Assurance Guide", Edited by Guarro, S. B., and W. F. Tosney, 1 July 2007, The Aerospace Corporation, pg. 1.

Acronyms and Abbreviations

COTS	Commercial-Off-The-Shelf
GOTS	Government-Off-The-Shelf
Integ.	Integration
MIL	Military
NIST	National Institute of Standards and Technology
OSS	Open source software
Qual.	Qualification
S	Standard
SMC	Space and Missile Systems Center
STD	Standard
TOR	Technical Operating Report