AEROSPACE REPORT NO.
TR-RS-2020-00012

Software Development Standard for Space Systems

July 10, 2020

Vale T. Sather[1], Heesoon E. Kim[2], and Gloria E. Pugliese-Rosillo[3]
[1]Enterprise Systems Engineering, Engineering and Integration Directorate
[2]Data and Architecture Modeling, Software Architecture and Engineering Department
[3]Software Product Quality and Testing, Software Systems Quality and Analysis Department

And

Delores J. Harralson
Formerly of The Aerospace Corporation

Prepared for:

Space and Missile Systems Center
United States Space Force
483 N. Aviation Blvd.
El Segundo, CA 90245-2808

Contract No. FA8802-19-C-0001

Authorized by: Space Systems Group

Distribution Statement A: Approved for public release; distribution unlimited.

This report was submitted by The Aerospace Corporation, El Segundo, CA 90245-4691, under Contract No. FA8802-19-C-0001 with the Space and Missile Systems Center, 483 N. Aviation Blvd., El Segundo, CA 90245. It was reviewed and approved for The Aerospace Corporation by Anthony T. Salvaggio, Jr., Principal Director. Steven J. Martin was the project officer for the SMC Atlas Corps Directorate of Engineering.

This report has been reviewed by the Public Affairs Office (PAS) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication. Publication of this report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas until adopted or otherwise implemented by the government.

Steven J. Martin, NH-04
SMC Atlas Corps Directorate of Engineering

All trademarks, service marks, and trade names are the property of their respective owners.

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden, estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**			
1. REPORT DATE (*DD-MM-YYYY*) 10-07-2020	**2. REPORT TYPE**		**3. DATES COVERED *(From - To)***
4. TITLE AND SUBTITLE Software Development Standard for Space Systems			**5a. CONTRACT NUMBER** FA8802-19-C-0001
			5b. GRANT NUMBER
			5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Vale T. Sather, Heesoon E. Kim, Delores J. Harralson, and Gloria E. Pugliese-Rosillo			**5d. PROJECT NUMBER**
			5e. TASK NUMBER
			5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The Aerospace Corporation 2310 E. El Segundo Blvd. El Segundo, CA 90245-4691			**8. PERFORMING ORGANIZATION REPORT NUMBER** TR-RS-2020-00012
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Space and Missile Systems Center United States Space Force 483 N. Aviation Blvd. El Segundo, CA 90245			**10. SPONSOR/MONITOR'S ACRONYM(S)** SMC
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			
13. SUPPLEMENTARY NOTES			
14. ABSTRACT The requirements herein enable the implementation of software development best practices for space systems acquisition, based upon lessons learned from experience with software-intensive space programs and other government programs. These requirements apply to the following space systems software: on board space vehicles, on board launch and upper-stage vehicles, ground operations software, user equipment software, test software used to verify or validate requirements or mission data, software used in performing or supporting operations, and firmware.			
15. SUBJECT TERMS Software Development; Space systems			

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Vale T. Sather
a. REPORT UNCLASSIFIED	**b. ABSTRACT** UNCLASSIFIED	**c. THIS PAGE** UNCLASSIFIED		18	**19b. TELEPHONE NUMBER *(include area code)*** 310.336.2644

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

CHANGE HISTORY

Rev No	Description of Change	Effective Date
Initial release	TOR-2004(3909)-3537, Rev. B "Software Development Standard for Space Systems" • *Established a best practice for space system software development* • *Leveraged MIL-STD-498, "Software Development and Documentation," dated December 5, 1994 (cancelled) and EIA/IEEE Interim Standard J-STD-016-1995, "Standard for Information Technology, Software Lifecycle Processes, Software Developer Acquirer-Supplier Agreement," dated September 30, 1995* Released by SMC as SMC-S-012 (2008)	2004
-	TOR-2013-00083/TR-RS-2015-00012 "Software Development Standard for Mission Critical Systems" • *Established a prescriptive process for space system software development* • *Increased the number of requirements (i.e. "shall" statements) to 975* • *Included requirements for format and content of multiple CDRLs and software work products* • *Assumed traditional software development* Released by SMC as SMC-S-012 (2015)	2015
-	This document • *Streamlined* TR-RS-2015-00012/*SMC-S-012(2015) to essential software development process requirements* • *Reduced the number of requirements to 95* • *Re-worded requirements to be agnostic of the software development method* • *Aligned with, and leveraged, related industry standards* • *Accommodated newer acquisition methods such as Agile or DevOps* Released by SMC as SMC-S-012 (2020)	2020

CONTENTS

1. Scope

1.1 Purpose

The requirements herein enable the implementation of software development best practices for space systems acquisition, based upon lessons learned from experience with software-intensive space programs and other government programs.

The requirements encapsulate common basic principles that have supported successful software development efforts, and which tend to remain constant. What has changed in this revision, and will continue to evolve, is how the requirements are implemented, a process which must be defined by and agreed upon between the contractor and the acquirer. The non-prescriptive approach used to derive the recommended requirements supports contractor innovation and creativity in the development method.

1.2 Application

The requirements apply to the following space systems software:

- Software on board space vehicles
- Software on board launch and upper-stage vehicles
- Ground operations software
- User equipment software
- Test software used to verify or validate requirements or mission data
- Software used in performing or supporting operations
- Firmware

1.3 Tailoring

These requirements should be tailored by the acquirer, together with the developer, based on the chosen software life cycle, the software development method, program-unique software risks, mission risk classification level, and any program-imposed contractual requirements on software.

Table K-2 in Appendix K of TOR-2011(8591)-5 [12] provides guidance for tailoring the software development requirements contained in this document based upon the mission risk classification.

1.4 Compliance

The requirements in this standard or its tailored contents must be placed on contract for compliance.

1.5 Order of Precedence

In the event of conflict between the requirements of this standard and other mandatory standards, the acquirer-developer agreement shall prevail.

2. Applicable Documents

2.1 Normative References

[1] Department of Defense, *Standard Practice System Safety*, MIL-STD-882E, May 2012.

[2] Institute of Electrical and Electronics Engineers, *IEEE Standard for Application of Systems Engineering on Defense Programs*, IEEE STD 15288.1-2014, May 2015.

[3] Institute of Electrical and Electronics Engineers, *IEEE Standard for Software Quality Assurance Processes*, IEEE STD 730-2014, June 2014.

[4] Institute of Electrical and Electronics Engineers, *IEEE Standard for Technical Reviews and Audits on Defense Programs*, IEEE STD 15288.2-2014, May 2015.

[5] International Organization for Standardization, International Electrotechnical Commission, and Institute of Electrical and Electronics Engineers, *Systems and software engineering – Measurement process*, International Standard ISO/IEC/IEEE 15939:2017, April 2017.

[6] SAE International, *Configuration Management Requirements for Defense Contracts*, Technical Report SAE-EIA 649-1, November 2014.

2.2 Other References

[7] Air Force Space Command, *Space and Missile Systems Center Standard: Software Development*, SMC Standard SMC-S-012. January 2015.

[8] Defense Acquisition University, *GLOSSARY: Defense Acquisition Acronyms and Terms*, February 2017.

[9] Department of Defense, *Clarifying Guidance Regarding Open Source Software (OSS),* Memorandum from the DoD Chief Information Officer, October 2009.

[10] Department of Defense Chief Information Officer, *Software Test Description Data Item Description*, DI-IPSC-81439A, December 1999.

[11] International Organization for Standardization, International Electrotechnical Commission, and Institute of Electrical and Electronics Engineers, *Systems and software engineering – Vocabulary*, International Standard ISO/IEC/IEEE 24765-2017, August 2017.

[12] Johnson-Roth, G. A., and W. F. Tosney, *Mission Risk Planning and Acquisition Tailoring Guidelines for National Security Space Vehicles*, Aerospace Report No. TOR-2011(8591)-5, The Aerospace Corporation, El Segundo, CA, September 2010.

[13] L. Bass, P. Clements, R. Kazman, *Software Architecture in Practice*, Third Edition. Addison-Wesley Professional. October 2012.

[14] Space and Missile Systems Center, *Guidance Memorandum to SMCI 63-1205: The SMC System Safety Program*, SMCI 63-1205, December 2015.

[15] International Organization for Standardization, International Electrotechnical Commission, and Institute of Electrical and Electronics Engineers, *Systems and software engineering – Software Lifecycle Processes*, International Standard ISO/IEC/IEEE 12207:2017, November 2017.

[16] Institute of Electrical and Electronics Engineers, *IEEE Standard Glossary of Software Engineering Terminology*, IEEE STD 610.12-1990, December 1990.

3. Definitions

Acquirer	An organization that procures software products for itself or another organization [7].
Build	A version of software that meets a specified subset of requirements that the completed software will meet [7].
Commercial off-the-shelf (COTS) software	A software product available for purchase and use without the need to conduct development activities [11].
Database	A collection of related data stored in one or more computerized files in a manner that can be accessed by users or computer programs via a database management system [7].
Design	The process by which the software requirements are translated into a representation of software architecture, components, and interfaces, necessary for the implementation phase [11].
Detailed design	The process of refining and expanding the preliminary design of a system or component to the extent that the design is sufficiently ready to be implemented [11].
Developer	An organization that develops software products. "Develops" may include new development, modification, reuse, maintenance, or any other activity that results in software products [7].
Document/documentation	A collection of data, regardless of the medium on which it is recorded, that generally has permanence and can be read by humans or machine [7].
Evaluation	The process of determining whether an item or activity meets specified criteria [7].
Firmware	Computer instructions and/or computer data that reside as read-only software on the hardware device [7].
Functional requirement	A requirement that defines a specific behavior or function of the system or software [7].
Interface	A relationship in which two or more entities share, provide, or exchange data (such as software to software, software to hardware, software to user) [7].
Maintenance organization	The organization that is responsible for modifying and otherwise sustaining the software after transition from the development organization [7].

Measurement	The set of operations having the object of determining a value of a measure [11].
Non-functional requirement	A system or software requirement that specifies criteria that can be used to judge the operation of the system or software, rather than specific behaviors. Non-functional requirements include for example safety, security, dependability, reliability, maintainability, availability, resiliency, and human systems integration [7].
Open Source Software (OSS)	Open Source Software is software for which the human-readable source code is publicly available for use, study, reuse, modification, enhancement, and redistribution by the users of that software [9].
Preliminary design	The process of analyzing design alternatives and defining the architecture, components, interfaces, and timing and sizing estimates for a system or component [11].
Qualification testing	1. Testing, conducted by the developer and witnessed by the acquirer (as appropriate), to demonstrate that a software product meets its specifications and is ready for use in its target environment or integration with its containing system [15] 2. Testing conducted to determine whether a system or component is suitable for operational use [11].
Regression testing	Selective retesting of a system or component to determine whether modifications to the system or its environment (e.g., operating system, software, or hardware) have caused unintended effects and to determine whether the system or component still complies with its specified requirements. Source: Adapted from [16].
Requirement	1. A characteristic that a system or software item must possess 2. A mandatory statement in this standard or another portion of the contract [7].
Safety	1. Functionality that protects the user from a hazardous system state (one that could result in unintended death, injury, loss of property, or environmental harm) [7]. 2. Expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered [11]. 3. Freedom from conditions that can cause death; injury; occupational illness; damage to or loss of mission, equipment or property; or damage to the environment [1].

Software	Computer programs, procedures (e.g., interpreted scripts), and data (e.g., information in databases, rule bases, or configuration data) pertaining to the operation of a computer system [7].
Software architecture	A set structures of the system needed to reason about the system, which comprise software elements, externally visible properties of the elements, and the relationships among the elements [13].
Software development	A set of activities that results in software products. Software development may include new development, modification, reuse, maintenance, or any other activities that result in software products [7].
Software development life cycle	The software development lifecycle begins after the system acquisition with software development planning and software requirement activities to software acceptance for the transition to operations and to maintenance [7].
Software development method	A framework that controls the way the software is developed. [Derived from multiple sources]
Software development process	An organized set of activities performed to translate user needs into software products [7].
Software engineering	In general usage, a synonym for software development. As used in this standard, a subset of software development consisting of all activities except qualification testing. The standard makes this distinction for the sole purpose of giving separate names to the software engineering and software test environments [7].
Software item	An aggregation of software that satisfies an end-use function and is designated for purposes of specification, interfacing, qualification testing, configuration management, or other purposes [7].
Software life cycle	The full software life cycle begins in the conceptual stage of the program and its acquisition through the retirement and disposal of the system from operational use. Within the software life cycle, a software development life cycle is defined for the activities that develop the software [7].
Software maintenance	The set of activities that takes place to ensure that software installed for operational use continues to perform as intended and fulfill its intended role in system operation. Software maintenance includes sustaining support, aid to users, and related activities [7].
Software product	Software or associated information created, modified, or incorporated to satisfy a contract. Examples include plans, requirements, design, code, databases, test information, and manuals [7].

Software reuse product	A software product developed for one use but having other uses, or the use of existing software to build new software [11].
Software system safety	The application of system safety principles to software [1].
Software test description	Description of the test preparations, test cases, and test procedures to be used to perform qualification testing of a software item [10].
Software test plan	A document that describes the technical and management approach to be followed to test a software system or component [11].
Software transition	The set of activities that enables responsibility for software to pass from one organization to another [7].
Software unit	An element in the design of a software item, for example, a major subdivision of a software item, a component of that subdivision, a class, object, module, function, routine, or database [7].
Subsystem	A functional grouping of components that combine to perform a major function within a system [8].
System	A composite of hardware and software elements used together in the intended operation or support environment, to perform a given task or achieve a specific production, support or mission requirement [14].
Test	The terms "test" and "testing," as used in this standard, refer to the activities of verifying that the implementation meets the design (i.e., unit and integration testing) and verifying that the requirements are satisfied (i.e., qualification testing). These terms should not be confused with the verification method "Test," which is only one of the four standard methods used for verification (Inspection, Analysis, Demonstration, and Test) [7].
User	As used in this standard, includes operators of the system in addition to the end users of the data produced by the system [7].
Work product	Software artifacts produced by means of the activities in this standard, including the deliverable and non-deliverable products, documentation, data, and other technical information [7].

4. General Requirements for Software Development

This section specifies the general requirements that apply to the software development activities through the software development life cycle when implementing the detailed software development requirements in section 5 of this standard.

4.1 Software Work Products

The software work products need to be captured, retained, and maintained to facilitate the development, subsequent maintenance, and future reuse of the delivered software.

1. The developer shall retain the software work products resulting from each software development activity identified in section 5 of this standard.

2. The developer shall make the software work products available, viewable, and retrievable to the acquirer throughout the software development life cycle.

3. The developer shall develop the following software work products for conducting all software testing (e.g., software unit test, software integration test, software qualification test, software regression test):

   a. procedures for preparing the test environment

   b. test cases for all requirements whose verification method is test

   c. test procedures

   d. test drivers or test scripts, or both

   e. test data, including test databases.

5. Detailed Requirements for Software Development

This section provides requirements for the software activities performed in a software development life cycle.

The planning of the software activities depends on the selected software development life cycle model and the software development method. The selected software development life cycle model (e.g., evolutionary, incremental, iterative, spiral, waterfall, and rapid application development) provides the framework that can be used to organize and implement the applicable software activities or software support of system activities. For example, the software activities may be performed sequentially, each one completing before the next activity can be started until the entire software product is completed. Or the software activities may be performed by iteratively cycling through the sequence of software engineering activities with multiple builds of functional software subsets until the entire software product is completed.

5.1 Software Development Planning

1. The developer shall plan to support system engineering activities, such as system level requirements analysis, architecture and design, integration and testing.

2. The developer shall document the environments needed for software development and for software test.

5.1.1 Planning Software Development Activities and Processes

1. The planning for the activities required by this standard and by the contract shall be recorded in the software development plan or other plans cited by the software development plan.

2. The developer shall record the selected software development life cycle model(s) appropriate to the software being developed in the software development plan.

3. The developer shall describe the development methods for performing the activities required by this standard in the software development plan.

4. The developer shall describe the processes, including roles and responsibilities, to be used for performing the software activities in the software development plan or other process documents as cited in the software development plan.

5. The developer shall define, in the software development plan, the work products and other information to be developed as required in this standard.

6. The developer shall ensure software test planning includes:

    a. Specialty engineering requirements, including cybersecurity

    b. All internal and external interfaces

    c. All end-to-end functional capabilities of the software item

    d. Performance testing, including timing and accuracy requirements

   e. Stress testing

   f. All startup, termination, and restart conditions, as applicable

   g. All fault detection, isolation, recovery (e.g., failover), and data capture and reporting

   h. Resource utilization requirements (e.g., CPU, memory, storage, bandwidth)

7. The developer shall describe the process assessment and improvement opportunities in the software development plan.

5.1.2 Following and Updating Plans

1. The developer shall update the software development plan to reflect the actual plans and processes as changes occur.

5.2 Software Configuration Management

1. The developer shall implement software configuration identification in accordance with SAE-EIA 649-1 Section 3.2 Configuration Identification [6].

2. The developer shall implement software configuration control in accordance with SAE-EIA 649-1 Section 3.3 Configuration Change Management [6] .

3. The developer shall implement software status accounting in accordance with SAE-EIA 649-1 Section 3.4 Configuration Status Accounting (CSA) [6] .

4. The developer shall implement software configuration audits in accordance with SAE-EIA 649-1 Section 3.5 Configuration Verification and Audits [6].

5.3 Technical Reviews

1. The developer shall perform all contractually required technical reviews in accordance with the applicable section(s) in IEEE STD 15288.2:2014 [4].

5.4 Software Risk Management

1. The developer shall perform software risk management in accordance with IEEE STD 15288.1-2014 Section 6.3.4 Risk Management Process [2].

5.5 Software Measurement

1. The developer shall perform software measurement in accordance with the ISO/IEC/IEEE 15939:2017, Sections 6.3.2 and 6.3.3[5].

5.6 Software Team Management

1. The developer shall ensure that all software development teams comply with the requirements in this standard.

5.7 Software Requirements Analysis

The software requirements analysis may define all requirements for a software item or may define requirements by iteration beginning with an initial set of requirements.

1. The developer shall define and record for each software item:

    a. the software functional requirements

    b. the software non-functional requirements

    c. the software interface requirements

    d. the methods and levels for verifying each software requirement

2. The developer shall evolve and maintain the software requirements consistent with the software development method.

3. The developer shall ensure bi-directional traceability of the software requirements and each of the following:

    a. system and subsystems

    b. software architecture

    c. software design

    d. software code

    e. software qualification test cases

5.8 Software Architecture

The software architecture may be defined completely for a software item or may be defined by iteration beginning with a preliminary architecture.

5.8.1 Software Item Architecture

1. The developer shall describe, in a software architecture description document, the software architecture for the software item, including

    a. the structure of the software item including software units

    b. software unit interfaces, and

    c. the interactions among the software units

    d. description of software architecture from three perspectives: the static, dynamic, and physical

    e. architecturally significant requirements that are quality attribute needs of the system, and their priorities

f. architecture constraints, both non-technical (e.g., cost, schedule) and technical constraints (e.g., tools, languages, infrastructures).

g. architecture trade studies and decisions and trace to architecturally significant requirements

h. design constraints and guidelines

2. The developer shall evolve and maintain the software architecture consistent with the software development method.

5.8.2 Software Reuse Products

1. The developer shall define the criteria for evaluation of software reuse products.
2. The developer shall identify software reuse products based on the criteria and record the results.
3. The developer shall ensure software reuse products selected for use comply with the data rights specified in the contract.

5.8.3 Commercial Off-The-Shelf Software and Open Source Software

1. The developer shall identify and record all Commercial Off-the-Shelf (COTS) software products.
2. The developer shall identify and record all Open Source Software (OSS) products.

5.9 Software Design

The software design may be defined completely for a software item or may be defined by iteration.

1. The developer shall develop and record in the software design description the design of each software item.
2. The developer shall record the designs pertaining to interfaces in the interface design description or software design description.
3. The developer shall record the designs of software units that are databases or are service units that access or manipulate databases in the database design description or software design description.

5.10 Software Code and Unit Testing

The source code for a software item may be defined completely or by iteration accompanied with unit testing.

5.10.1 Software Coding

Software coding activities can include the following, as applicable:

a. coding computer instructions and data definitions

    b. populating databases and other data files with data values

    c. configuring and/or modifying reused software

    d. employing automated code generators

1. The developer shall develop the code for each software unit as specified in the software item design.

2. The developer shall adhere to the coding standards defined in the software development plan.

5.10.2 Automated Software Analysis

1. The developer shall incorporate automated software analysis tools to identify and evaluate the quality of the code (e.g., readable, maintainable, testable, extensible, efficient, consistent, commented/documented).

5.10.2.1 Preparing for Unit Testing

1. The developer shall ensure the work products needed to test each software unit, including reused software, are available

2. The test cases shall verify the unit's design, including

    a. the correct execution of statements and branches

    b. error and exception handling

    c. software unit interfaces

    d. limits and boundary conditions

    e. algorithms

3. The developer shall define the code coverage requirements of the software units.

4. The developer shall automate software unit testing to the maximum extent practicable.

5.10.2.2 Performing Unit Testing

1. The developer shall execute software unit testing in accordance with the software development plan.

2. The developer shall record the test results for each software unit.

5.11 Software Integration and Testing

Software integration and testing is often an iterative process that completes when all software units that make up a software item are successfully integrated and tested.

5.11.1 Preparing for Software Integration and Testing

1. The developer shall ensure the software work products needed for conducting software integration and testing are available.
2. The developer shall identify and record all software units to be integrated and tested.
3. The developer shall ensure that all software units to be integrated have passed unit testing.
4. The developer shall automate software integration and testing, including regression testing, to the maximum extent practicable.

5.11.2 Performing Software Integration and Testing

1. The developer shall execute software integration and testing in accordance with the software test procedures.
2. For all software integration and testing performed, the developer shall record all software defects and software changes in accordance with the software development plan or the configuration management plan.
3. The developer shall regression-test any subsequent changes made to the integrated software units.

5.12 Software Qualification Testing

Software qualification testing addresses the verification of software requirements.

5.12.1 Independence in Software Qualification Testing

1. The developer shall ensure that person(s) responsible for the software qualification testing are not the person(s) who performed detailed design or implementation of the software item.

5.12.2 Software Qualification Testing Environment

1. The developer shall conduct all software qualification testing in an operationally equivalent environment.
2. The developer shall perform software qualification testing using actual interfaces or operationally equivalent simulations of the interfaces.
3. The developer shall describe the complete configuration of the software qualification testing environment in the software test plan.

5.12.3 Preparing for Software Qualification Testing

1. The developer shall ensure the software work products needed to verify all software requirements, are available.
2. The developer shall automate software qualification testing to the maximum extent practicable.

3. The developer shall conduct a software test readiness review before performing the software qualification test.

5.12.4 Performing Software Qualification Testing

1. The developer shall execute software qualification testing in accordance with the software test plan and software test description.
2. The developer shall report the results of performing the software qualification test.
3. The developer shall make the necessary revisions to the software qualification testing work products based on the results of software qualification testing.
4. The developer shall perform regression testing for any subsequent changes made to the software item during software qualification testing.

5.13 Software Peer Reviews

1. The developer shall perform peer reviews on software work products as specified in the software development plan.

5.13.1 Preparing for Software Peer Reviews

1. The developer shall define and record:
    a. What type of peer review will be conducted on the software work products
    b. Which roles are required to participate in which types of peer reviews and for which type of software work product
    c. Peer review entry and exit criteria
2. The developer shall identify key reviewers who must participate in the peer review.
3. The developer shall ensure the entry criteria are met prior to the peer review.

5.13.2 Conducting Peer Reviews

1. The developer shall record all software defects and action items in accordance with the software development plan or the configuration management plan.
2. The developer shall ensure that the exit criteria are met.
3. The developer shall analyze and report the findings of the peer reviews, including trends (e.g., the most frequent types of discrepancies).

5.14 Software Quality Assurance

1. The developer shall implement the software quality process activities in accordance with IEEE STD 730-2014 Section 5.3 Software Quality Assurance Process Implementation Activities [3].

2. The developer shall implement the software quality product assurance activities in accordance with IEEE STD 730-2014 Section 5.4 Product Assurance Activities [3].

3. The developer shall implement the software quality process assurance activities in accordance with IEEE STD 730-2014 Section 5.5 Process Assurance Activities [3].

5.15 Corrective Action

1. The developer shall implement and maintain software defect management and software change processes that cover:

    a. Definition of the content of the software defects and software changes

    b. Recording and tracking of software defects and software changes

    c. Disposition of software defects and software changes (e.g., evaluation, assignment, resolution)

    d. Corrective action for the reported discrepancies

    e. Prioritization of the software defects and software changes

5.16 Software Security

1. The developer shall use secure programming practices, including secure coding standards.

2. The developer shall incorporate automated software vulnerability analysis tools throughout the life cycle to identify and evaluate software vulnerabilities.

3. The developer shall ensure that the program protection implementation plan defines how it will implement the software protection requirements specified in the acquirer's program protection plan.

4. The developer shall ensure that the supply chain risk management plan includes:

    a. all developed software

    b. software reuse products

    c. software development tools, and

    d. software environment

5.17 Software System Safety

1. The developer shall plan and implement software safety requirements in accordance with MIL-STD-882E [1].

5.18 Software Transition to Operations

Software transition to operations is defined as transferring the software to the user(s) for operation.

5.18.1 Preparing for Software Transition to Operations

1. The developer shall generate and prepare the executable software for delivery to the user(s) from the configuration-managed versions of source files and other software executables such as Commercial Off-the-Shelf (COTS) software or open-source software (OSS).

2. The developer shall prepare any additional files (e.g., batch files, command files, configuration files, data files, databases) needed to install or operate the software in the acquirer-designated operational environment.

3. The developer shall prepare the software version description for the acquirer-designated operational environment.

4. The developer shall prepare the software installation procedures for each user or acquirer-designated operational environment.

5. The developer shall include the software version description information, including the version numbers and license information of all COTS or OSS used in the prepared software, in the software version description document.

6. The developer shall prepare the information needed for the user to operate the software in the software user manual.

7. The developer shall prepare the information needed for the proper execution of the software in the computer operation manual, if not provided by the computer vendor.

8. The developer shall prepare the user training and associated training materials (e.g., use case models, tutorial scripts, guidance or help manuals).

5.18.2 Transition to Operations

1. The developer shall perform software transition to operations as defined in the software transition plan.

5.19 Software Transition to Maintenance

Software transition to maintenance is defined as the transfer of the software to the maintenance organization who will perform needed corrections and improvements during the software sustainment phase.

5.19.1 Preparing for Software Transition to Maintenance

1. The developer shall generate and prepare the executable software delivery for the maintenance site(s) from the configuration-managed versions of source files and other software executables such as COTS software.

2. The developer shall prepare the software installation procedure for the software executables.

3. The developer shall include the software source code and other related files needed to modify, build, install, and test the software by the maintenance organization(s).

4. The developer shall prepare the software version description for the maintenance organization.

5. The developer shall include license information for all COTS or OSS in the delivered software.

6. The developer shall ensure the software work products (e.g., software requirements, software interface requirements, software architecture, software design) match the “as built” software.

7. The developer shall specify all necessary tools, dependent hardware, and configuration information necessary to build, test (unit, component, integration, and qualification in an operationally equivalent environment) the software item.

8. The developer shall record all information needed to program and reprogram any firmware devices in which the software will be installed.

5.19.2 Transition to Maintenance

1. The developer shall perform software transition to maintenance as defined in the software transition plan.

6. Acronyms

6.1 Scope

This appendix provides a list of acronyms used in this standard. This appendix is not a mandatory part of the standard. The information provided is intended for guidance only.

6.2 Acronyms

COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
CSA	Configuration Status Accounting
DOD	Department of Defense
OSS	Open Source Software
SIQT	Software Item Qualification Testing
SMC	Space and Missile Systems Center
SMC/EN	SMC Engineering Directorate
TOR	Technical Operating Report
TR	Technical Report

External Distribution

REPORT TITLE		
Software Development Standard for Space Systems		
REPORT NO.	PUBLICATION DATE	SECURITY CLASSIFICATION
TR-RS-2020-00012	July 10, 2020	UNCLASSIFIED

Adkisson, Robert
The Boeing Company
Robert.w.adkisson@boeing.com

Baldwin, Mark
Raytheon Company - Space and Airborne Systems
Mark_L_baldwin@raytheon.com

Briseno, Naomi
Raytheon Company
Naomi.Briseno@raytheon.com

Brown, Wayne
United Launch Alliance
Wayne.brown@ulalaunch.com

Crownover, Christopher Blue Origin
Ccrownover@blueorigin.com

Carnahan, Chris, Director of Standardization
Aerospace Industries Association (AIA)
Chris.carnahan@aia-aerospace.org

Clark, John
International Council on Systems Engineering (INCOSE)
John.clark@incose.org

Clark, John
International Council on Systems Engineering (INCOSE)
John.clark@ngc.com

Croll, Paul R., Chair
Institute of Electrical and Electronics Engineers (IEEE) Software and Systems Engineering Standards Committee
pcroll@computer.org

Day, Craig, Director of Business Development
American Institute of Aeronautics and Astronautics (AIAA)
craigd@aiaa.org

Ess, Bob
Blue Origin
Bess@blueorigin.com

Evers, John, Chair
SAE International G-47 Systems Engineering Committee
drjohneusa@aim.com

Fenimore, Todd
Lockheed Martin Space Systems Co.
Todd.w.fenimore@lmco.com

Floyd, Michael
General Dynamics
Mike.Floyd@gdc4s.com

Guisinger, Dan
Aerojet Rocketdyne
danny.guisinger@rocket.com

Gustufson, Scott
Lockheed Martin Space Systems Co.
scott.gustafson@lmco.com

Hopkins, Paul
Lockheed Martin Space Systems Co.
Paul.c.hopkins@lmco.com

Hurst, Kyle
Departmental Standardization Officer, Air Force
james.k.hurst9.civ@mail.mil

Jacobs, Scott
Blue Origin
Sjacobs@blueorigin.com

Jensen, Mike
United Launch Alliance
Mike.jensen@ulalaunch.com

Koenigsmann, Hans SpaceX
Hans.Koenigsmann@spacex.com

Lawson, Aaron
Northup Grumman Innovation Systems
aaron.lawson@ngc.com

Levine, Leonard Departmental Standardization Officer, Defense Information Systems Agency
Leonard.f.levine.civ@mail.mil

Lillie, Wendy
Ball Aerospace
wlillie@ball.com

Linder, Katherine
Blue Origin
Klinder@blueorigin.com

Lockwood, Harry Lockheed Martin Space Systems Co.
harry.lockwood@lmco.com

Loman, James
MAXAR
james.loman@sslmda.com

Maginnis, Laura
Blue Origin
Lmaginnis@blueorigin.com

Mueller, Tom
Space Exploration Technologies Corporation
tom.mueller@spacex.com

Nafus, Cindy
United Launch Alliance
cynthia.l.nafus@ulalaunch.com

Paquette, Chris Departmental Standardization Officer, Navy
Christopher.paquette@navy. mil

Pinkley, David
Ball Aerospace
dpinkley@ball.com

Ponce, Luis
Northup Grumman Innovation Systems
luis.ponce@ngc.com

Rassa, Bob, Chair emeritus National Defense Industrial Association (NDIA) -Systems Engineering Division
rcrassa@raytheon.com

Rentsch, Rusty, Vice President ofTechnical Operations National Security Aerospace Industries Association (AIA)
Rusty.rentsch@aia-aerospace.org

Roedler, Garry, Chair International Standards Organization (ISO) - ISO/IEC JTC1/SC7 U.S. Technical Advisory Group
gjrjar@gmail.com

Sanneman, Paul
Northup Grumman Innovation Systems
Paul.Sanneman@ngc.com

Saunders, Gregory
Defense Standardization Program Office
gregory.saunders@dla.mil

Schubring, Wade Departmental Standardization Officer, Army
Wade.j.schubring.civ@mail.mil

Schultz, James
The Boeing Company
James.w.schultz@boeing.com

Sedmak, Aileen
ASD(R&E)/SE/MA
aileen.g.sedmak.civ@mail.mil

Serna, Frank, Chair National Defense Industrial Association (NDIA) Systems Engineering Division fserna@draper.com

Swanson, David
Northup Grumman Innovation Systems
david.swanson@ngc.com

Tongson, Nick, Director of Standards
American Institute of Aeronautics and Astronautics (AIAA)
nickt@aiaa.org

Violet, Mike
Northup Grumman Innovation Systems
Michael.Violet@ngc.com

Wade, James
Raytheon Company
James.w.wade@raytheon.com

Wood, Bob
Blue Origin
Bwood@blueorigin.com

APPROVED BY____ DATE______________
(AF OFFICE)

AEROSPACE REPORT NO.
TR-RS-2020-00012

Software Development Standard for Space Systems

Approved Electronically by:

Minh N. Nguyen, DIRECTOR DEPT
SOFTWARE ENGINEERING SUBDIVISION
INFORMATION SYSTEMS & CYBER
DIVISION
ENGINEERING & TECHNOLOGY GROUP

Dewanne M. Phillips, PRINC DIRECTOR
INFORMATION SYSTEMS & CYBER
DIVISION
ENGINEERING & TECHNOLOGY GROUP

Cognizant Program Manager Approval:

Anthony T. Salvaggio, PRINCIPAL DIRECTOR
ENTERPRISE GROUND & LAUNCH DIVISION
LAUNCH & ENTERPRISE OPERATIONS
SPACE SYSTEMS GROUP

Aerospace Corporate Officer Approval:

Randolph L. Kendall, VICE PRESIDENT
SPACE SYSTEMS GROUP
OFFICE OF EVP

Content Concurrence Provided Electronically by:

Vale T. Sather, PROJECT LEADER SR
ENTERPRISE SYSTEMS ENGINEERING
ENGINEERING & INTEGRATION
SPACE SYSTEMS GROUP

© The Aerospace Corporation, 2020.

All trademarks, service marks, and trade names are the property of their respective owners.

SQ0429

AEROSPACE REPORT NO.
TR-RS-2020-00012

Software Development Standard for Space Systems

Technical Peer Review Performed by:

Vale T. Sather, PROJECT LEADER SR
ENTERPRISE SYSTEMS ENGINEERING
ENGINEERING & INTEGRATION
SPACE SYSTEMS GROUP

Heesoon E. Kim, PROJECT LEADER SR
DATA & ARCHITECTURE MODELING
SOFTWARE ARCHITECTURE & ENGINEERING DEPT
ENGINEERING & TECHNOLOGY GROUP

Delores J. Harralson, ENGRG SPCLST SR
SOFTWARE SYSTEMS QUALITY & ANALYSIS DEPT
SOFTWARE ENGINEERING SUBDIVISION
ENGINEERING & TECHNOLOGY GROUP

Gloria E. Pugliese-Rosillo, MEMBER-TECH STF
SOFTWARE PRODUCT QUALITY AND TESTING
SOFTWARE SYSTEMS QUALITY & ANALYSIS DEPT
ENGINEERING & TECHNOLOGY GROUP

Brian E. Shaw, PROJECT LEADER SR
ENTERPRISE SYSTEMS ENGINEERING
ENGINEERING & INTEGRATION
SPACE SYSTEMS GROUP

© The Aerospace Corporation, 2020.

All trademarks, service marks, and trade names are the property of their respective owners.

SQ0429