ARMSTRONG FLIGHT PROCEDURAL REQUIREMENTS (AFPR)

Directive: AFPR-7150.2-001 B-4
Effective Date: July 23, 2019
Expiration Date: July 23, 2024

Compliance is mandatory.

SUBJECT: Armstrong Software Engineering (SWE) Requirements
(with Change 3, dated 05/10/2021)

RESPONSIBLE OFFICE: Office of the Chief Engineer

Table of Contents

PREFACE

P.1 Purpose

a. This Armstrong Flight Procedural Requirement (AFPR) directive brings the Armstrong Flight Research Center (AFRC) (herein after referred to as the Center) into compliance with the National Aeronautics and Space Administration (NASA) Policy Directive (NPD) 7120.4, NASA Engineering and Program/Project Management Policy, by capturing the requirements in NASA Procedural Requirements (NPR) 7150.2, NASA Software Engineering Requirements, and NASA Standard (NASA-STD)-8719.13, NASA SOFTWARE SAFETY STANDARD. In doing so, this directive provides requirements for the specification, acquisition, development, maintenance, operation, and management of software that supports the Center's flight research mission. It does not prescribe or promote a specific software development life cycle, but instead provides a single set of requirements for Center software engineering (SWE) activities. This will allow organizations at the Center that purchase or develop software the freedom to develop processes tailored for their mission need.

b. In addition to the above, this directive modifies the software classification approach defined in NPR 7150.2 to a hazard-/risk-based system. The approach used in this directive is consistent with the software classification approach defined in NPR 7150.2 for aeronautics applications. This has been done to reduce confusion and improve traceability to other common aeronautics standards and existing Center processes, including Radio Technical Commission for Aeronautics (RTCA) DO-178(B or C), Software Considerations in Airborne Systems and Equipment Certification, and Center-wide procedure AFG-8715.3-018, Hazard Management Guide.

P.2 Applicability

a. This directive is applicable to the Center and on-site support contractors, grant recipients, and other partners to the extent specified in their contracts or agreements.

> Note: The above statement alone is not sufficient to stipulate requirements for the contractor, grant recipient, or agreement. This directive provides requirements for NASA contracts, grant recipients, or agreements to the responsible NASA project managers and contracting officers that are made mandatory through contract clauses, specifications, or statements of work (SOW) in conformance with the NASA Federal Acquisition Regulation (FAR) Supplement or by stipulating in the contracts, grants, or agreements which of the NPR 7150.2 requirements apply.

b. This language applies to Jet Propulsion Laboratory (JPL) (a Federally-Funded Research and Development Center), other contractors, recipients of grants, cooperative agreements, or other agreements only to the extent specified or referenced in the applicable contracts, grants, or agreements.

c. This directive applies to the development, acquisition, and management of flight, flight support, and ground software. This includes flight software, simulation software, range software, and analysis tools. These can be developed by the Center, Inter-centers, maintained by the Center, acquired by the Center, implemented on Center assets (including flight vehicles, simulations, and range assets) or maintained for the Center by another organization (either at the Center or at a remote site).

d. This directive applies to software development, maintenance, retirement, operations, and management, acquisition, and assurance activities. The requirements of this directive cover all software created, acquired, or maintained by or for NASA and apply to all of the Agency's investment areas containing software systems and subsystems. The applicability of these requirements to specific systems and subsystems within the Center's investment areas, programs, and projects is determined through the use of the NASA-wide definition of software classes and safety criticality, as detailed in of this directive, in conjunction with Appendix D, Requirements Mapping and Compliance Matrix, of this directive. Some projects may contain multiple systems and subsystems having different software classes. Using the Appendix D of this directive, the applicable requirements and their associated rigor are adapted according to the classification and safety criticality of the software.

e. This directive will be applied to software development, maintenance, operations, management, acquisition, and assurance activities started after its effective date of issuance.

f. This directive does not supersede more stringent requirements imposed by individual NASA organizations and other Federal government agencies.

g. In this directive, all mandatory actions (i.e., requirements) are denoted by statements containing the term “shall.” The terms “may” or “can” denote discretionary privilege or permission, “should” denotes a good practice and is recommended, but not required, “will” denotes expected outcome, and “are/is” denotes descriptive material.

h. In this directive, refer to Appendix A for the contextual understanding of terms used, such as "software," “software engineering,” “Software Independent Technical Authority (ITA),” and “Software Technical Authority (TA).”

i. In this directive, all document citations are assumed to be the latest version unless otherwise noted.

j. In this directive, the term "simulation" refers to only those simulations that are implemented in software.

P.3 Authority

a. NPD 2800.1, Managing Information Technology

b. NPD 7120.4, NASA Engineering and Program/Project Management Policy

c. NPR 8621.1, NASA Procedural Requirements for Mishap and Close Call Reporting Investigating, and Record Keeping

d. AFPD-1000.0-002, Governance and Strategic Management

e. AFPD-2800.2-001, Managing Information Technology (IT)

f. AFPD-8700.1-001, Organizational & Individual Safety Responsibilities

All authority documents referenced in this section are located on one of the following websites:

- NODIS (https://nodis3.gsfc.nasa.gov/)
- ODIE (https://odie.ndc.nasa.gov/SitePages/Home.aspx)

P.4 Applicable Documents and Forms

a. NPR 7150.2, NASA Software Engineering Requirements

b. NPR 8715.3, NASA Procedural-NASA General Safety Program Requirements

c. NASA-STD-8719.13, Software Safety Standard

d. NASA-STD-8739.8, Software Assurance Standard

e. NASA-HDBK-2203, NASA Software Engineering Handbook (SWEHB)

f. NASA-HDBK-4008, Programmable Logic Devices (PLD) Handbook

g. NASA-HDBK-8739.23, NASA Complex Electronics Handbook For Assurance Professionals

h. AFPD-8040.1-001, Configuration Management

i. AFPR-7123.1-001, Systems Engineering Requirements Document

j. AFPR-7123.2-001, Waivers and Deviations to Technical Requirements and Standards

k. AFOP-1000.3-001, Armstrong Center Management Council (ACMC) Reviews

l. AFOP-7120.5-003, Program and Project Management Manual

m. AFOP-7150.2-004, Software Assurance

n. AFOP-7900.3-023, Airworthiness and Flight Safety Review Process

o. AFOP-7900.3-024, Flight Operational Readiness Review (ORR)

p. AFG-8715.3-018, Hazard Management Guide

q. AFG-8739.8-002, Software Assurance Audit and Corrective Action Handbook

r. Requirements and Standards International Standards Organization (ISO)/IEC Electronics (IEC)/Institute of Electrical and Electronics Engineers (IEEE) 24765:2010 System and Software Engineering - Vocabulary

s. Radio Technical Commission for Aeronautics (RTCA) DO-178(B or C), Software Considerations in Airborne Systems and Equipment Certification

t. AFRC 10117f, Request for Deviation or Waiver

u. AFRC 70010, Configuration Change Request

v. AFRC 80184, Flight Media Release

All documents and forms referenced in this section are located on one of the following websites:

- NODIS (https://nodis3.gsfc.nasa.gov/)
- NASA Technical Standards System (https://standards.nasa.gov/node/1339)
- ODIE (https://odie.ndc.nasa.gov/SitePages/Home.aspx)
- ISO standards (https://www.iso.org/standards.html)
- IEC standards (https://www.iec.ch/dyn/www/f?p=103:6:0)
- IEEE standards (https://standards.ieee.org/standard/index.html)
- Forms (https://nef.nasa.gov/)

P.5 Measurement/Verification

The methods to ensure compliance with this directive and NPR 7150.2 will be documented in the software development implementation procedures and through internal and external assessments and audits.

P.6 Cancellation

AFPR-7150.2-001 A-3, Software Engineering Requirements, dated 9/3/2014.

Hardcopy signed by Center Director or Delegated Official and date

DISTRIBUTION: Approved for release via the Document Library.

Chapter 1: Introduction

1.1 Sources of Requirements

This directive seeks to provide a unified set of process requirements for software development/management activities at the Center. It includes the tailored SWE requirements specified in NPR 7150.2 and the software safety requirements specified in NASA-STD-8719.13. The software document requirements have been removed from NPR 7150.2 and this directive. This information now resides in NASA-HDBK-2203. This directive also includes requirements derived from RTCA DO-178(B or C), as well as Center-unique requirements needed to ensure airworthiness.

1.2 Document Scope

This directive represents the Center's tailored approach to interpreting the NASA requirements specified in NPR 7150.2 and NASA-STD-8719.13. The scope of this directive only includes NPR 7150.2 Classes C, D, and E software requirements. The scope does not include NPR 7150.2 Class A, B, F, G, or H software requirements, since the Center does not currently generate this level of software. For software that is classified into NPR 7150.2 Class A, B, F, G, or H, please refer directly to NPR 7150.2.

1.3 Description of Scope

For the purposes of this directive, the definition of "software" is derived from NPR 7150.2 (See Appendix A of this directive).

Based on this definition, types of software include, but are not limited to:

a. Application software: Software designed to help users perform particular tasks or handle particular types of problems, as distinct from software that controls the computer itself.

b. Custom software: Software product developed for a specific application from a user requirements specification.

c. Embedded software: Software that is part of a larger system and performs some of the requirements of that system.

d. Existing software: Software that is already developed and available, is usable either as-is or with modifications, and that is provided by the supplier, acquirer, or a third party.

e. Firmware: Combination of a hardware device and computer instructions or computer data that reside as read-only software on the hardware device.

f. Programmable Logic Device: Field Programmable Gate Array (FPGA) and Complex Programmable Logic Device (CPLD) (See NASA-HDBK-4008.).

g. Software reuse: A software product developed for one use, but having other, or one developed specifically to be usable on multiple projects or in multiple roles on one project. Examples include, but are not limited to, COTS products, acquirer-furnished software products, software products in reuse libraries, and pre-existing developer software products. Each use may include all or part of the software product, and may involve its modification. This term can be applied to any software, such as requirements and architectures, not just to software code itself. Often, this is software previously written by an in-house development team and used on a different project. GOTS software would come under this category, if the product is supplied from one Government project to another government project.

h. Software tool: A computer program used in the development, testing, analysis, or maintenance of a program or its documentation.

i. Support software: Software that aids in the development or maintenance of other software.

j. System software: Software designed to facilitate the operation and maintenance of a computer system and associated programs. (Reference: ISO/IEC/IEEE 24765:2010)

k. Legacy and heritage: Software products (i.e., architecture, code, requirements) written specifically for one project and, then, without prior planning during its initial development, found to be useful on other projects.

Software can be compiled or interpreted. Interpreted software includes scripting (i.e., shell scripts, test scripts within a simulation, parameter or preference files, spreadsheets used for data analysis, etc.).

Chapter 2: Responsibilities

The designated governance framework for the Center is defined in AFPD-1000.0-002. NPR 7150.2 provides the tailoring, engineering TA, and compliance requirements. Appendix D of this directive identifies the TA for each of the NPR 7150.2 requirements. In the case of NPR 7150.2, the Center-level TA is delegated to the Center Director, or the Center Director's designated Engineering TA. Implementation of the NASA software safety standard requirements is the responsibility of the Safety & Mission Assurance (S&MA) Directorate. This Directorate ensures that the requirements found in NPR 8715.3 are being met by the Center.

The program roles are divided amongst the following personnel listed below and their responsibilities explained in Section 2.1:

a. Project Manager

b. Project Chief Engineer (PCE)

c. Operations Manager/Lead

d. System Safety Manager

e. Software Assurance Manager

f. Software Manager

g. Configuration Manager

h. Flight Systems Lead (FSL)

2.1 Project Manager

The Project Manager is responsible to:

a. Define project priorities, objectives, and milestones in the project plan; and ensure the allocation of resources in the form of aircraft, schedule, staffing, and facilities.

b. Ensure the project plan, request for proposal, contract proposals, and SOW contain appropriate software assurance provisions. The project plan will identify the software manager or their organization and designate a software development agent (SDA). Provisions include:

(1) Safety critical software development/management delegated to the project are produced in accordance with the requirements found in Appendix D of this directive with notation of "SC."

(2) Software deliverables.

(3) Customer surveillance for software assurance in Appendix D of this directive.

(4) How the contractor and customer report and resolve software problems.

(5) Customer agreement for any changes to baselined software elements.

c. Approve all plans generated by the project, including the software development plan/software management plan (SDP/SMP), software assurance plan (SAP), and configuration management plan (CMP).

d. Appoint the configuration control board (CCB) membership.

e. Chair, manage, and preside over the CCB.

f. Approve decisions made by the members of the CCB.

g. Integrate the software safety program with system safety and software development.

h. Work with S&MA to acquire software, resolve conflicts, and implement a process for tracking concerns.

2.2 Project Chief Engineer (PCE)

The PCE has three main areas of responsibility as technical lead, system safety lead, and systems engineering lead. On some projects, the PCE may also have some responsibility for doing the work of a technical discipline on the project. On large projects, the PCE may delegate some of the technical lead responsibilities to a deputy or, some of the systems engineering lead responsibilities, to a systems engineer or a deputy. Regardless of who does the actual work, the PCE is ultimately responsible.

As technical lead, the PCE is accountable for developing technical objectives, giving technical briefings/presentations, leading experiment and system design, managing technical risk, supporting test planning and preparation, directing project flight operations, and ensuring appropriate documentation and reporting. The PCE is also responsible for developing the engineering team organization, providing direction for project engineering activities, supporting schedule maintenance, executing the technical work of the project and working with the Research & Engineering Directorate Branch Chiefs to ensure that the project is adequately staffed.

As system safety lead, the PCE is accountable for ensuring the appropriate hazard analysis efforts and serving as the focal point for safety concerns. The S&MA Directorate typically provides a system safety representative to the project and, while that person often facilitates System Safety Working Group (SSWG) meetings and prepares the hazard reports and associated materials, the PCE is ultimately accountable and responsible to ensure that hazards are identified, documented, and mitigated appropriately.

As systems engineering lead, the PCE is accountable for ensuring the implementation of applicable systems engineering policies, practices, processes, and procedures throughout the project life cycle including reviews and required documentation.

2.3 Operations Manager/Lead

The Operations Engineering Branch has primary responsibility for overall surveillance of aircraft configuration and, in conjunction with quality assurance, is responsible for determining that the aircraft software has been properly generated, verified and validated, and, therefore, acceptable for flight.

An operations engineer's core responsibility is to ensure airworthiness, as defined in Appendix A of this directive, of aircraft and research/science assets by:

a. Approves work orders to load flight and flight support software on board the test vehicle.

b. Develops flight test plans and cards.

c. Ensures operational documents describe all safety-related commands data, input sequences, and options.

d. Ensures operational documents include error message descriptions and corrective actions.

e. Ensures the test vehicle meets its functional or physical configurations, if applicable.

Authority for issues that relate to the application of aircraft airworthiness standards for modification, operation, or maintenance of aircraft are delegated from the Center Director to the Director of Flight Operations down to the operations engineer per AFPR-7123.1-001. This authority is exercised in different ways depending on the focus area an operations engineer is assigned. Aeromechanical design engineers ensure mechanical designs are designed and fabricated using appropriate standards and static and dynamic stress margins. Drawing control and configuration management (CM) operations engineers ensure drawings and CM documents meet Center requirements for proper tracking, documentation, and archival. As for the conventional operations engineer, the delegation of authority makes he/she the Flight Operations Directorate representative for project and aircraft work ensuring that work is planned and accomplished based on maintenance, Center, and project requirements as they relate to aircraft integration.

2.4 System Safety Manager

The Flight Research & Test Safety Branch Chief designates a qualified branch member to fulfill the position of System Safety Manager on each project. The System Safety Manager has the following roles and responsibilities:

a. Support project by attending project meetings and providing an assessment of project compliance with applicable safety requirements to project management and the Flight Research & Test Safety Branch Chief.

b. Develop and maintain System Safety Plan (SSP) and participate with project in development of Risk Management Plan (RMP).

c. Review and/or assist with the Preliminary Hazard Analyses (PHA) (iterative process).

d. Participate in design reviews.

e. Review and/or assist with development of System Hazard Analysis (SHA)/Sub-SHA/Operating and Support Hazard Analysis (iterative process).

f. Assist with development of Hazard Action Matrix and accepted risks.

g. Participate in CCB meeting as required for system safety/software assurance issues from project conception through end of flight activities.

h. Present applicable system safety/software assurance risk analysis documentation portions of Flight Readiness Review project briefing and technical briefs.

i. Verify and document the status of corrective actions in Flight Assurance Matrix (FAM).

j. Participate in crew briefs and mission control room test and training activities based on general responsibilities identified and agreed upon.

k. Provide continual verbal/written feedback to Flight Research & Test Safety Branch Chief on project safety and risk issues, and compliance with standards.

l. Update existing hazard reports with any newly identified flight or ground safety risks, prepare materials to brief senior management of any changes to the current accepted level of residual, and update the FAM accordingly.

m. Assist project in developing, documenting, and processing lessons learned.

2.5 Software Assurance Manager

The Flight Research & Test Safety Branch Chief designates a qualified branch member to fulfill the position of Software Assurance Manager on each project. The Safety Manager has the following roles and responsibilities:

A. Verifies Center directives and NASA standards (or equivalents) in contract/memorandum of agreement (MOA)/memorandum of understanding (MOU), in addition to safety contents.

B. Performs an independent software classification to compare with the software manager's or SDA's classification.

C. Prepares the SAP to establish and implement the software assurance program.

D. Reviews and concurs with the software classification, project plan or SMP, provider SDP, SSP, software requirement document, software design document (SWDD), and software test documentation.

E. Ensures tailoring of software quality, safety, and verification and validation requirements are based on software classification and safety level-of-effort.

F. Supports the CCB as a voting member for software-safety-related matters and impacts, and verifies implementation to configuration item(s).

G. Performs software quality activities, as defined in AFOP-7150.2-004 of this directive, including formal evaluations of process/plan compliance and verifications of product conformance.

H. Performs software safety activities, as defined in AFOP-7150.2-004 of this directive, including identification of potential hazards associated with the software throughout the software development program as part of the SSWG.

2.6 Software Manager (SM)

The Flight Instrumentation & System Integration Branch Chief designates a qualified branch member to fulfill the position of SM on each project. The SM has the following roles and responsibilities:

a. Works to define the generation of the software classification for all configuration item(s), in conjunction with the software quality assurance (SQA) and SDA, the SM.

b. Defines the software development process, modification, maintenance, operations, retirement, management, acquisition, and assurance activities.

c. Establishes a reporting channel and interfaces with the software provider's project management that is independent of the software development function and the software assurance function.

d. Ensures the development requirements specified in this directive are addressed in the SMP/SDP, software test plan, and version description document (VDD) are met.

e. Identifies software configuration items, defines requirements, software or firmware development, and maintains all software configuration items, which also includes requirements tracing for the complete software life cycle, if required.

f. Identifies all COTS, MOTS, and GOTS products for its capability to meet safety critical functions, interface to developed code, and the ability to verify at the same level as developed code.

g. Defines, traces, analyzes, and ensures compliance with all software requirements from one life cycle phase to another.

h. Establishes software configuration baselines and any changes to the baseline, ensuring the smooth transaction of software products from the development baseline to the project baseline.

i. Specifies the flight software to be flown on a designated flight using configuration change requests, flight media releases, NASA Aircraft Management Information System, and work orders for on-aircraft software installations. This includes both informal and formal flight and flight support loads.

j. Serves as a voting member of the CCB for software related matters.

k. Establishes and/or approves procedures for production of flight software media in the SMP/SDP.

l. Defines procedures for physically and electronically controlling flight software media as documented in the SDP and/or CMP.

m. Supports the identification, analysis, and/or generation of software hazards by participating in the SSWG.

n. Evaluates project tools for safety impact and, if necessary, documenting how project tools are selected, approved, and controlled.

o. Supports any NASA Headquarters’ Independent Verification and Validation (IV&V) requirements, if applicable.

2.7 Configuration Manager

The Flight Instrumentation & Systems Integration Branch Chief designates a qualified branch member to fulfill the position of Configuration Manager on each project. The Configuration Manager has the following roles and responsibilities:

a. Establishes a system of software configuration identification of Center software products for all software classifications.

b. Provides a system for software configuration baseline management.

c. Provides a system for reviews and audits of Center software products for Classification I, II, III, and S (See classification guidelines in this directive.).

d. Provides a system of configuration status accounting for Center software products for all software classifications.

2.8 Flight Systems Lead (FSL)

The FSL ensures that the system technically fulfills the defined needs and requirements and that the systems engineering approach is being followed. The FSL oversees the project’s engineering activities as performed by the Flight Instrumentation & System Integration Branch engineer and directs, communicates, monitors, and coordinates tasks, schedules, procurements, staffing levels, and necessary training. The FSL reviews and evaluates the technical aspects of the project to ensure that the systems/subsystems engineering processes are functioning properly and evolves the system from concept to product. The entire technical team is involved in the SE process.

The FSL focuses on the technical characteristics of decisions including technical, cost, and schedule and on providing these to the project manager and chief engineer. The overlap in these responsibilities is natural, with the FSL and chief engineer focused on the success of the engineering of the system (technical, cost, schedule) and the project manager providing constraints on engineering options to maintain a successful delivery of the system within cost and schedule.

The FSL usually plays the key role in leading the development of the concept of operations, system architecture, defining boundaries, defining and allocating requirements, evaluating design tradeoffs, balancing technical risk between systems, defining and assessing interfaces, and providing oversight of verification and validation activities, as well as many other tasks.

The FSL should be involved in project-level planning, scheduling, and staffing exercises and developing higher level project documents. The FSL may provide input to the project level documents, but is usually not responsible for preparing the documents.

Chapter 3: Software Classification

NPR 7150.2 defines eight classifications for NASA software. These classifications are based on:

a. Usage within a NASA system,

b. Criticality of the system to NASA's programs and projects,

c. Extent to which humans depend on the system,

d. Developmental and operational complexity, and

e. The extent of the Agency's investment.

These definitions are assigned an alphabetic classification identification from A-H. Classes A-E covers engineering software while Classes F-H cover business and IT software. In addition, NPR 7150.2 identifies an additional test applied to software defined as Class A, B, C, D, or E. This is the software safety litmus test (Reference NASA-STD-8719.13). The results of this litmus test will change the number of NPR 7150.2 requirements levied on the software. In addition, it will levy the requirements defined in NASA-STD-8719.13. Historically, both the Center and the aeronautics industry have used a hazard-/risk-based software classification system that classifies software based on the effects of the failure of the software to function properly. Classification definitions used for commercial aircraft certification can be found in RTCA DO-178(B or C). The historical software classification system used at the Center is defined in AFOP-7150.2-004. The classification method used in this directive applies the hazard-/risk-based approach, commonly used on aeronautics-based platforms, while meeting the intent of the classification process found in NPR 7150.2, as it applies to aeronautics-based platforms. The scope of this directive only includes NPR 7150.2 Classes C, D, and E. The scope does not include NPR 7150.2 Class A, B, F, G, or H software requirements, since the Center does not currently generate this level of software. For software that is classified into NPR 7150.2 Class A, B, F, G, or H, please refer directly to NPR 7150.2.

3.1 General

Requirements in this directive are assigned to software items according to the criticality of that software. Specifically, software is grouped into one of four different classifications based on the most severe consequence of a software-controlled event. These classifications are closely coupled to the hazard categories described in AFG-8715.3-018. Specifically, these categories are as follows:

a. Class I: Catastrophic

b. Class II: Critical

c. Class III: Minor

d. Class IV: Negligible

The use of Roman Numerals is meant to reduce confusion with other software standards such as NPR 7150.2 and RTCA DO-178(B or C) that use alphabetic classifications and to align with the hazard severity classification in AFOP-7150.2-004. If more than one software class appears to apply, then the higher of the classes is assigned to the system/subsystem.

Since this directive attempts to define both the engineering and business/IT software, the category definitions have been expanded to address other types of consequences, such as a software-related security breach or Agency-wide loss of productivity. For example, software that could cause a critical security breach would be classified as Class II.

Identification/incorporation of the safety-critical software increases the number of requirements levied by NPR 7150.2. It also levies additional requirements called out in NASA-STD-8719.13. Determination of the existence of safety critical software involves performing a system/software safety assessment. If the system and software are determined to be safety critical an additional “S” will be added to the classification to denote the presence of safety critical software. For example, software that could cause a critical injury would be classified as Class II-S.

3.2 The Classification Process

In addition to the Safety Litmus Test, the criticality of a software item should also be determined using a PHA which is performed during system architectural development. The system level PHA will provide an initial assessment of the system/software hazards. From this, preliminary system/software level classifications can be determined. The PHA will be further refined as the software architecture matures until hazards have been reviewed down to the computer software configuration item (CSCI) level. Once this level is reached, the software CM system treats the software as a single entity.

If a CSCI has multiple categories of failures associated with its different functions that item could be further partitioned to limit the interaction between software items. This may allow those items to be developed at different assurance levels, minimizing the volume of code that shall be developed to the more stringent standards.

For CSCIs that support multiple functions, the classification should be based on the most severe of the effects resulting from the failure or malfunction of any supported function or any combination of supported functions.

Analyze software design to ensure that any partitioning or isolation methods used in the design to logically isolate the safety critical design elements from those that are non-safety are effective. Any software that can write or provide data to safety-critical software will also be considered safety critical, unless there is isolation built in, and, then, the isolation design is considered safety critical. (NASA-STD-8719.13, Software Safety Standard (SSS)-004).

Once the software is assessed to the CSCI level, perform a bottom-up review of the software architecture to ensure that CSCIs of differing classifications do not interact in such a way where the failure of a higher classification CSCI (i.e., Class IV) could cause a lower classification CSCI (i.e., Class I) to fail.

> Note: As part of the initial PHA, the assessment should include a check to ensure that the vehicle/project does not fall into the large-scale aeronautics vehicle category. If it does, the program/project need to consult with Center management to discuss the SWE approach to be used. (Exceeding a $250M total life-cycle cost results in software declared Class B per NPR 7150.2.)

3.3 Criteria for Safety Critical Software

Software is considered safety critical, if it resides on a safety-critical system and meets the criteria defined in NASA-STD-8719.13.

In accordance with AFPD-8700.1-001, safety programs are implemented for activities that are internally controlled by the Center or when operations are sponsored or supported by the Center where:

a. Any NASA Center or it’s contractor personnel and it’s equipment are at risk,

b. The Center has an assigned safety responsibility (i.e., flight, ground, range, environmental, etc.), or

c. Any NASA Center owns the asset and are not otherwise excluded by agreement or contract.

d. A contractor owns the asset and is not otherwise excluded by agreement or contract.

This includes the following activities, aviation activity, project activity, and industrial activity. (See AFPD-8700.1-001 for definitions of these activities.)

3.4 Software Classification - Safety Critical

Software considered safety critical using the definition in Appendix A, of this directive, is further classified based on the most severe consequence of a software-controlled event. The classification criteria are as follows per AFOP-7150.2-004:

*Note: Classification of software cannot be waivered. Follow the process in AFPR-7123.2-001 to waive and deviate from Agency-defined requirements. See Appendix A of this directive for definitions of “Deviation” and “Waiver.”

a. Class I-S: Catastrophic

(1) Death or permanently disabling/life-threatening injury, or loss of crew

(2) Destruction of facility on the ground, major system, vehicle, termination of project

b. Class II-S: Critical

(1) Severe/lost time injury or occupational illness

(2) Major loss/damage to facility, system, equipment, flight hardware, vehicle

c. Class III-S *: Not Applicable

d. Class IV-S *: Not Applicable

*Note: A software component that is deemed to be safety-critical software shall, by definition, have a Software Safety Classification of Class I or Class II. Therefore, Class III-S and Class IV-S are not applicable.

3.5 Software Classification – Non-Safety Critical

Software not considered safety critical using the definition in Appendix A, of this directive, is further classified based on the most severe consequence of a software-controlled event. Per AFOP 7150.2-004, the classification criteria are as follows:

a. Class I: Catastrophic

(1) Loss of the only opportunity for critical data

(2) Recovery/replacement cost equal to or greater than $2M

b. Class II: Critical

(1) Long-term project delay

(2) Loss of some project-critical data

(3) Loss of confidentiality, integrity, and/or availability of information with an IT security category of “High” per NPR 2810.1

(4) Recovery/replacement cost equal to or greater than $500K, but less than $2M

(5) Agency-wide productivity impact

c. Class III: Moderate

(1) Loss of mission (sortie, flight, return-to-base, test shut-down, etc.)

(2) Loss of non-critical project data

(3) Loss of confidentiality, integrity, and/or availability of information with an IT security category of "Moderate"

(4) Minor loss/damage to facility, system, equipment, or flight hardware

(5) Recovery/replacement cost equal to or greater than $50K, but less than $500K

(6) Interruptions in the availability of critical data

(7) Center-wide productivity impact

d. Class IV: Minimal

Productivity impact to small number of users

> Note: The monetary values for recovery/replacement costs are found in AFG-8715.3-018. The costs found in this directive should be used only as a reference. If a discrepancy exists between the specified recovery/replacement costs found in this directive, AFG-8715.3-018 takes precedence.

3.6 Classification Guidelines

Software classification is not an exact science and shall be evaluated on a case-by-case basis. Some guidelines are given below:

a. Destruction of facility, major system, or vehicle

The intent of this statement is to capture consequences that would likely lead to a NASA Class A Mishap per NPR 8621.1 hull loss of a crewed aircraft or greater than $2M in property damage to a facility or system. However, in some cases, loss of a test article is either planned or anticipated and, thus, may not drive software criticality to the highest level. Examples include:

(1) Intentional destruction of a vehicle

(2) Vehicles or systems not intended to be recovered once the test is complete

b. Recovery/Replacement Costs

AFG-8715.3-018 and NPR 8621.1 both provide criteria for recovery/replacement costs (for instance, $2M is the threshold for a Category I Hazard in AFG-8715.3-018, and a Type A Mishap in NPR 8621.1). NPR 8621.1 provides guidance as to how to make that assessment.

c. Major Damage vs. Destruction

The distinction between major damage and destruction of a system should be determined by the feasibility of repair. If the system can be repaired within the cost and budget constraints of the project or program, it should be considered damaged. If repair is impossible or the costs prohibitive, it should be considered destroyed.

d. IT-Related Software

When software falls into the category of business and IT infrastructure, as defined in NPR 7150.2, then it should be classified in accordance with the guidance provided by the Center CIO. The Center CIO may decide to apply the business and IT infrastructure software Classifications F-H found in NPR 7150.2 in lieu of the classifications defined in this directive.

e. Long-Term Delay

The definition of long-term delay shall also be project or program specific. A delay that constitutes some significant percentage of the project or program schedule (>5%) would certainly be considered a long-term delay. A delay that could trigger a high-level program review or project cancellation would also be considered long term.

f. Loss of Mission

Defining what constitutes a loss of mission is also highly program or project dependent. In some cases, mission and project are synonymous, and a failure to meet preapproved minimum mission success criteria by definition indicates that project objectives were not met. This is the case where there is only one opportunity to gather the critical data. In other cases, loss of mission may imply loss of a single aircraft sortie, which has a much lower consequence. For the purposes of this directive, loss of mission implies that there will be other opportunities to collect the data.

g. Interruptions in Availability

An interruption in availability occurs when data, stored or real time, is not accessible. This could occur if the system used to access backed-up data fails or if display software becomes inoperative. In those cases where real-time monitoring of data becomes impossible, other impacts may become the driver for criticality determination.

3.7 Software Classification Testing

Software testing is used to demonstrate that the software satisfies its requirements and to demonstrate with a high degree of confidence that errors that could lead to unacceptable failure conditions, as determined by the system safety assessment process, have been removed.

Test coverage analysis is a two-step process involving Requirements-Based and Structural-Based Coverage Analyses. The first step analyzed the test cases in relation to the software requirements to confirm that the selected test cases satisfy the specified criteria. The second step confirms that the requirements-based test procedures exercised the code structure to the applicable coverage criteria. Coverage analysis is the process of determining the degree to which a proposed software verification process activity satisfies its objectives.

The Requirements-Based Coverage Analysis test is used to determine how well the requirements-based testing verified the implementation of the software requirements. This analysis may reveal the need for additional requirements-based test cases.

The Structural-Based Coverage Analysis test determines which code structure, including interfaces between components, was not exercised by the requirements-based test procedures. The requirements-based test cases may not have completely exercised the code structure, including interfaces, so structural coverage analysis is performed and additional verification produced to provide structural coverage. Structural-based testing may be performed on the Source Code, Object Code, or Executable Object Code.

The SMP/SDP will define the structural coverage testing required of the software based on the Software Classification. The structural coverage testing and the required percent coverage will be agreed upon by the Software ITA and acknowledge with their signature on the SMP/SDP.

The current software classifications and associated testing are as follows:

a. Class I: Modified Condition/Decision Coverage (MC/DC)

Every point of entry and exit in a program has been invoked at least once, every condition in a decision in the program has taken all possible outcomes at least once, every decision in the program has taken all possible outcomes at least once, and each condition in a decision has been shown to independently affect that decision’s outcome. A condition is shown to independently affect a decision’s outcome by varying just that condition while holding fixed all other possible conditions or varying just that condition while holding fixed all other possible conditions that could affect the outcome. (RTCA DO-178C)

b. Class II: Decision Coverage

Every point of entry and exit in a program has been invoked at least once and every decision in the program has taken on all possible outcomes at least once. (RTCA DO-178C)

c. Class III: Statement Coverage

Every statement in the program has been invoked at least once. (RTCA DO-178C)

Note: Statement is as defined by the programming language.

3.8 Architectural Considerations

In some cases, mitigations to software hazards can be used to lower the classification of that software, if the following criteria are met:

a. The hazard has been mitigated through system design or through the use of safety devices (See Table 3-1.).

b. These mitigations meet the requirements levied in NPR 8715.3.

c. These mitigations are verifiable and verified.

Note: Warning devices (i.e., a visual or audible alarm to the operator that a hazardous condition exists) or administrative/operational procedures (rules that limit use of the system to areas where the consequence of failure is more benign) alone cannot be used to reduce software classification.

Mitigation Type	Description	Effect on Classification
Design	Other aspects of the system design (hardware or software) prevent the software from generating a hazardous condition.	Can be considered when classifying the criticality of the software.
Safety Devices	Other elements of the system identify and mitigate hazardous conditions before damage can occur.	Can be considered when classifying the criticality of the software.
Caution/Warning Devices	Other elements of the system that warn the operator, if a hazardous condition is detected.	Should not be considered when classifying the criticality of the software.
Operational/ Administrative Procedures	Rules regarding the operation or use of the system to limit the effects of hazardous conditions caused by software.	Should not be considered when classifying the criticality of the software.

Table 3-1: Architectural Considerations in Software Criticality Assessment

Chapter 4: Implementation of SWE Requirements

This section describes the implementation of the requirements from NPR 7150.2 and NASA-STD-8719.13, as well as the Center-specific requirements. Requirements levied by NPR-7150.2 on the Center are captured in the first subsection. The subsequent subsections describe the project activities that are intended to meet the requirements from NASA-STD-8719.13 and NPR 7150.2 for each class of software.

> Note: The scope of this directive only includes NPR 7150.2 Classes C, D, and E. The scope does not include NPR 7150.2 Class A, B, F, G, or H software requirements, since the Center does not currently generate this level of software. For software that is classified into NPR 7150.2 Class A, B, F, G, or H, please refer directly to NPR 7150.2.
>
> Note: The SSS requirements are listed in the SWE Requirements and Appendix D, of this directive, for reference. For more complete details please refer to AFOP-7150.2-004 or NASA-STD-8719.13.

4.1 Requirements Levied on the Center

NPR 7150.2 Requirement		Armstrong Implementation
	Requirements Levied on Center Directors or Designees	
SWE-003	...maintain, staff, and implement a plan to continually advance the Center's in-house SWE capability and monitor the SWE capability of NASA's contractors.	The TA and Software Working Group Lead will establish an informal list of improvements and review the list annually to check progress and relevance
SWE-005	...establish, document, execute, and maintain software processes.	AFPR (Dryden Procedural Requirements (DPR)), AFOP (Dryden Center Procedure (DCP)), Handbook, project records
SWE-006	...maintain a reliable list of their Center's programs and projects containing Class A, B, C, and D software. The list should include a. Project/program name and Work Breakdown Structure (WBS) number. b. Software name(s) and WBS number(s). c. Software size estimate (report in Kilo/Thousand Source Lines of Code (KSLOCs)). d. Phase of development or operations. e. **Safety Critical Software (Yes or No).** f. Software Class or list of the software classes being developed on the project. g. For each CSCI/Major System containing Class A, B, or C software, provide: (1) The name of the software development organization. (2) Title or brief description of the CSCI/Major System. (3) The estimated total KSLOC the CSCI/Major System represents. (4) The primary programing languages used. (5) Primary life-cycle methodology being used on the software project. (6) Name of responsible software assurance organization(s).	The Center participates in the biennial Agency software inventory. The TA maintains a library of active SDPs and SMPs.
SWE-091	For Classes A-C and safety-critical software projects, utilize software measurement data for monitoring SWE capability, improving software quality, and tracking the status of SWE improvement activities.	The Center will collect data on Class I and Class II projects that start coding after 1/1/2017.

NPR 7150.2 Requirement		Armstrong Implementation
	a. Software development tracking data b. Software functionality achieved data c. Software quality data d. Software development effort and cost data	
SWE-092	For Classes A-C and safety-critical software projects, ...utilize software measurement data for monitoring SWE capability, improving software quality, and tracking the status of SWE improvement activities.	
SWE-095	...periodically report on the status of the Center's SWE discipline, as applied to its projects, to the NASA Office of Chief Engineer and relevant TAs, as requested.	The TA participates in capability leadership team and oversees Center participation in Software Working Group
SWE-101	...maintain and implement software training plan(s) to advance its in-house SWE capability and as a reference for its contractors.	Flight Instrumentation & Systems Integration Branch and Sensors & Systems Development Branch training plans include software training for managers and coders
SWE-140	...comply with the requirements in Appendix C, of this directive, that are marked with an "X."	AFPR (DPR)
SWE-142	For Classes A-C software projects, ...establish and maintain a software cost repository(ies) that contains at least one of the following measures: a. Planned and actual effort and cost. b. Planned and actual schedule dates for major milestones. c. Both planned and actual values for key cost parameters that typically include software size, requirements count, defects counts for maintenance or sustaining engineering projects, and cost model inputs. d. Project descriptors or metadata that typically includes software class, software domain/type, and requirements volatility.	The Center will collect data on Class I and Class II projects that start coding after 1/1/2017.
SWE-144	...contribute applicable SWE process assets in use at respective Centers to the Agency-wide process asset library.	The Center will contribute, as required.

	Requirements Levied on the Center Engineering TA	
SWE-002	…lead, maintain, and fund a NASA SWE initiative to advance SWE practices.	NASA Headquarters funds the SWE Capability Leadership Team (CLT). Each Center has a current Center SWE Improvement Plan on file in the NASA Office of the Chief Engineer.
SWE-004	…periodically benchmark each Center's SWE capability against its Center SWE Plan.	NASA Software Working Group and NASA CLT periodically benchmark each center's software capabilities.
SWE-098	…maintain an Agency-wide process asset library of applicable best practices.	NASA Headquarters maintains process asset library.
SWE-100	…provide and fund training to advance SWE practices and software acquisition.	SWE training
SWE-122	...be NASA civil servants (or JPL/CalTech employees) approved by the Center Director.	Center Software TA is appointed by the Director for Research & Engineering Directorate and is a civil servant.
SWE-126	Assess projects' compliance matrices, tailoring, waivers, and deviations from requirements in this directive by: Checking the accuracy of the project's classification of software components against the definitions in NPR 7150.2. Evaluating the project's compliance matrix for commitments to meet applicable requirements in this directive, consistent with software classification. Confirming that requirements marked "Not Applicable" in the project's compliance matrix are not relevant or not capable of being applied. Determining whether the project's risks, mitigations, and related requests for relief from requirements designated with "X" in NPR 7150.2 are reasonable and acceptable.	TA signature on project development and management plans

	Approving/disapproving requests for relief from requirements designated with "X" in Appendix C, which fall under this TA's scope of responsibility. Facilitating the processing of projects' tailoring/compliance matrices, tailoring, waivers, or deviations from requirements in this directive, which fall under the responsibilities of a different TA (See NPR 7150.2 Requirements Mapping Matrix "Technical Authority" column).	
	Ensuring that approved compliance matrices, including any waivers and deviations against this directive, are archived as part of retrievable project records.	Waivers are processed using the Center process per AFPR-7123.2-001.
SWE-129	…authorize appraisals against selected requirements in NPR 7150.2 to check compliance	NASA Headquarters authorizes appraisals.
SWE-145	Indicate their approval by signature(s) in the compliance matrix itself, when the compliance matrix is used to waive/deviate from applicable "X" requirement(s).	TA signature on project development and management plans
SWE-152	…periodically review project compliance matrices.	A software compliance matrix audit checklist added to all future SMPs/SDPs. The audit will be performed by the Software ITA, or delegate, throughout the life cycle of the software. The completed checklist is to be retained in project repository.
SWE-153	...define the content requirements for software documents or records.	Defined in Software Engineering Handbook (SWEHB)

4.2 Class VI Software Implementation

SWE numbers marked with an asterisk (*) are required for ***FLIGHT*** software only

Step	Activity	Applicable SWE Requirements		Waivable by Center Software TA?
	PLANNING			
1	Write down what your software is supposed to do. When the purpose of your software changes, update.	SWE-050	...establish, capture, record, approve, and maintain software requirements, including the software quality requirements, as part of the technical specification.	Yes
		SWE-053*	...track and manage changes to the software requirements.	Yes
2	Use version control (subversion, Git, etc.) as you develop and test.	SWE-080*	...track and evaluate changes to software products.	Yes
		SWE-081*	...identify the software configuration items (e.g., software records, code, data, tools, models, scripts) and their versions to be controlled for the project.	Yes
3	Write simple test plans, perform software test, and record results and track defects	SWE-066*	...perform software testing.	Yes
		SWE-069*	...record defects identified during testing and track to closure.	Yes
4	Write a VDD that includes: a. What the software is supposed to do b. How to install the software c. How to operate the software d. List of software components and version numbers e. List of incorporated and outstanding bug fixes	SWE-063	...provide a software version description for each software release.	Yes
		AFOP-7150.2-004* (Flight Software Media Control 1)	Prior to installation on an aircraft, a VDD will be produced containing a form AFRC 80184, Flight Media Release, and attached a form AFRC 70010, Configuration Change Request, requesting installation.	No - SMA requirement
5	Complete a compliance matrix and get the TA to sign it.	SWE-125	...maintain a compliance matrix or multiple compliance matrices against requirements in NPR 7150.2, including those delegated to other parties or accomplished by contract vehicles or Space Act Agreements.	Yes

		SWE-139	...comply with the requirements in NPR 7150.2 Requirements Mapping Matrix that are marked with a “project” responsibility and an ”X” is consistent with their software classification.	Yes
		SWE-145	When the compliance matrix is used to waive/deviate from applicable “X” requirement(s), the designated TAs ...indicate their approval by signature(s) in the compliance matrix itself.	Yes
		SWE-013	...develop, maintain, and execute software plans that cover the entire software life cycle and, as a minimum, address the requirements of this directive with approved tailoring.	Yes
6	Evaluate software for potential reuse and contribute reuse candidates to Agency Software Catalog.	SWE-148*	...evaluate software for potential reuse by other projects across the Agency and contribute reuse candidates to the Agency Software Catalog.	Yes

Complete the following steps to release your software outside the center:

7	Classify the software per this directive	SWE-020	...classify each system and subsystem containing software in accordance with the highest applicable software classification definitions for ...	Yes
8	Request SQA independent classification and get email from SQA confirming classification and safety criticality	SWE-021	If a system or subsystem evolves to a higher or lower software classification … or there is a change in the safety criticality of the software, the project … update their plan to fulfill the applicable requirements	Yes
		SWE-132	...perform an independent classification assessment.	Yes
		SWE-133	Determine the software safety criticality in accordance with NASA-STD-8719.13.	Yes
		SWE-160	If a software component is determine to be safety critical software then software component classification shall be Software Class D or higher.	Yes
9	Provide requirements, test plan and results, and VDD to software TA for review	SWE-065*	…establish and maintain: a. Software test plan(s). b. Software test procedure(s). c. Software test report(s).	Yes
		SWE-071*	…update software test plan(s) and software test procedures(s) to be consistent with software requirements	Yes

Complete the following steps for software to be installed on a flight vehicle:

10	Classify the software per this directive	SWE-020	...classify each system and subsystem containing software in accordance with the highest applicable software classification definitions for …	Yes
11	Request SQA independent classification and get email from SQA confirming classification and safety criticality	SWE-132	...perform an independent classification assessment.	Yes
		SWE-133	…determine the software safety criticality in accordance with NASA-STD-8719.13.	Yes
		SWE-160	If a software component is determine to be safety critical software then software component classification shall be Software Class D or higher.	Yes
12	Prepare software media as required for installation	AFOP-7150.2-004* (Flight Software Media Control 2)	All software media (tape, disk or chip) … identified and physically labeled at the time of production	No - SMA Requirement
13	Prepare the form AFRC 80184	AFOP-7150.2-004* (Flight Software Media Control 3)	…form AFRC 80184 that uniquely identifies (via checksum(s), file size/modification dates, or other verifiable means) the specific software load that should be installed on the aircraft	No - SMA Requirement
		AFOP-7150.2-004* (Flight Software Media Control 4)	Flight software for a specific flight or block of flights … on a form AFRC 80184	No - SMA Requirement
14	Prepare the software installation procedure	AFOP-7150.2-004* (Flight Software	A procedure … written for flight software installation into the aircraft computer and for verification of correct loading.	No - SMA Requirement

		Media Control 5)		
		AFOP-7150.2-004* (Flight Software Media Control 6)	Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures	No - SMA Requirement

4.3 Class III Software Implementation

Step	Activity	Applicable SWE Requirements		Waivable by Center Software TA?
1	Classify the software per this directive	SWE-020	...classify each system and subsystem containing software in accordance with the highest applicable software classification definitions for NPR 7150.2 Classes A, B, C, D, E, F, G, and H software …	Yes
2	Request SQA independent classification and get email from SQA confirming classification and safety criticality	SWE-132	...perform an independent classification assessment.	Yes
		SWE-133	Determine the software safety criticality in accordance with NASA-STD-8719.13.	Software ITA and SMA
		SWE-160	If a software component is determine to be safety critical software, then software component classification shall be NPR 7150.2 software Class D or higher. [and higher than this directive's Class III (Class III-S does not exist)]	Yes

	Include software activities in the following:			
3	Project Schedule	SWE-016	…document and maintain a software schedule that satisfies the following conditions: a. Coordinates with the overall project schedule. b. Documents the interactions of milestones and deliverables between software, hardware, operations, and the rest of the system. c. Reflects the critical path for the software development activities. d. Adhere to the guidance provided in NASA/SP-2010-3403, NASA Scheduling Management Handbook.	Yes
		SWE-024	…track the actual results and performance of software activities against the software plans. a. Corrective actions are taken, recorded, and managed to closure.	Yes

			b. Changes to commitments (e.g., software plans) that have been agreed to by the affected groups and individuals.	
4	Project Cost Estimates	SWE-015	…establish, document, and maintain one software cost estimate and associated cost parameter(s) for other software projects.	Yes
		SWE-151	…software cost estimate(s) satisfy the following conditions: a. Covers the entire software life cycle. b. Is based on selected project attributes (e.g., assessment of the size, functionality, complexity, criticality, reuse code, modified code, and risk of the software processes and products). c. Is based on the cost implications of the technology to be used and the required maturation of that technology. d. Incorporates risk and uncertainty. e. Includes the cost for software assurance support. f. Includes other direct costs.	Yes
5	Project Reviews/Status Activities	SWE-018	…regularly hold reviews of software activities, status, and results with the project stakeholders and track issues to resolution	Yes
		SWE-019	…select and document a software development life cycle or model that includes phase transition criteria for each life-cycle phase	Yes
6	V&V Planning and Tracking [Following Center V&V process will comply with NPR 7150.2 requirements]	SWE-028	…plan software verification activities, methods, environments, and criteria for the project.	Yes
		SWE-029	…plan the software validation activities, methods, environments, and criteria for the project.	Yes
		SWE-030	…record, address, and track to closure the results of software verification activities.	Yes
		SWE-031	…record, address, and track to closure the results of software validation activities.	Yes
		SWE-068	…evaluate test results and record the evaluation.	Yes
7	Acceptance Criteria	SWE-034	...define and document the acceptance criteria and conditions for the software.	Yes
8	requirements documents	SWE-050	…establish, capture, record, approve, and maintain software requirements, including the software quality requirements, as part of the technical specification.	Yes

	[Note: Following Center requirements management process will comply with NPR 7150.2 requirements]	SWE-052	…perform, record, and maintain bidirectional traceability between the software requirement and the higher-level requirement.	Yes
		SWE-053	…track and manage changes to the software requirements.	Yes
9	Design Documents	SWE-065	...establish and maintain: a. Software test plan(s). b. Software test procedure(s). c. Software test report(s).	Yes
10	Test Plans and Procedures, Test Reports	SWE-071	...update software test plan(s) and software test procedure(s) to be consistent with software requirements.	Yes
11	Operations, Maintenance, Disposal Plans	SWE-075	...plan and implement software operations, maintenance, and retirement activities.	Yes
12	Product Delivery Plans	SWE-085	...establish and implement procedures for the storage, handling, delivery, release, and maintenance of deliverable software products.	Yes
		SWE-080	...track and evaluate changes to software products.	Yes
13	Configuration Control Plans and Activities	SWE-081	...identify the software configuration items (e.g., software records, code, data, tools, models, scripts) and their versions to be controlled for the project.	Yes
		SWE-033	The project manager shall assess options for software acquisition versus development.	Yes
14	Work with Flight Instrumentation & Systems Integration Branch Chief to identify requisite skills for the software effort and provide training, as appropriate.	SWE-121	[where approved] … document and reflect the tailored requirement in the plans or procedures controlling the development, acquisition, and/or deployment of the affected software.	Yes
15		SWE-013	...develop, maintain, and execute software plans that cover the entire software life cycle and, as a minimum, address the requirements of this directive with approved tailoring.	Yes
		SWE-125	...maintain a compliance matrix or multiple compliance matrices against requirements in NPR 7150.2, including those delegated to other parties or accomplished by contract vehicles or Space Act Agreements.	Yes

	Complete a compliance matrix and get the TA to sign it.	SWE-139	... comply with the requirements in NPR 7150.2 Requirements Mapping Matrix that are marked with a "project" responsibility and an "X" consistent with their software classification.	Yes
		SWE-145	When the compliance matrix is used to waive/deviate from applicable "X" requirement(s), the designated TAs ... indicate their approval by signature(s) in the compliance matrix itself.	Yes
16	Develop and document the software architecture.	SWE-057	...develop and record the software architecture. (A/B/C)	Yes
		SWE-058	...develop, record, and maintain a design based on the software architectural design that describes the lower-level units so that they can be coded, compiled, and tested. Note 1: ***Only required for Class III, if structural code coverage is greater than Zero (0) percent***; and can be a tailored version of the SWDD, based on the complexity of the developed software, and submitted in the SDA's SDP with prior approval by the Software ITA. Note 2: A tailored version of the SWDD, based on the complexity of the developed software, can be submitted in the SDA's SDP and shall be approved by the Software Manager.	Yes
17	Plan for coding, compilation, and testing of lower-level units of the architecture.	SWE-022	…plan and implement software assurance per NASA-STD-8739.8. [CAP053:027]	Yes

	If acquiring software:			
18	Determine which processes, documents, electronic products, activities, and tasks are required.	SWE-036	...determine which software processes, software documents, electronic products, software activities, and tasks are required for the project and software suppliers.	Yes
19	Define milestones at which progress will be reviewed and audited.	SWE-037	...define the milestones at which the software supplier(s) progress will be reviewed and audited as a part of the acquisition activities.	Yes
20	Document acquisition planning decisions.	SWE-038	...document software acquisition planning decisions.	Yes

21	Require supplier to allow NASA to: a. Monitor product integration b. Review verification activities c. Review trade studies and source data d. Audit software development process e. Participate in software reviews and Technical Interchange Meetings (TIMs)	SWE-039	...require the software supplier(s) to provide insight into software development and test activities; at a minimum, the software supplier(s) will be required to allow the project manager or designate to: a. Monitor product integration. b. Review the verification activities to ensure adequacy. c. Review trades studies and source data. d. Audit the software development process. e. Participate in software reviews and systems and software TIMs.	Yes
22	Require supplier to provide schedule and schedule updates, as needed.	SWE-046	...require the software supplier(s) to provide a software schedule for the project's review and schedule updates as requested.	Yes
23	Evaluate software for potential reuse and contribute reuse candidates to Agency Software Catalog	SWE-148	...evaluate software for potential reuse by other projects across the Agency and contribute reuse candidates to the Agency Software Catalog.	Yes
24	At design reviews, reaffirm software classification.	SWE-021	If a system or subsystem evolves to a higher or lower software classification … or there is a change in the safety criticality of the software, the project … update their plan to fulfill the applicable requirements.	Yes
25	Code	SWE-060	...implement the software design into software code.	Yes
26	Unit Test	SWE-062	…unit test (statement coverage) the software code per the plans for software testing.	Yes
27	Software Test	SWE-066	...perform software testing.	Yes
		SWE-069	…record defects identified during testing and track to closure.	Yes
28	Write a VDD that includes: a. What the software is supposed to do b. How to install the software c. How to operate the software List of software components and version numbers d. List of incorporated and outstanding bug fixes	SWE-063	…provide a software version description for each software release.	Yes
		AFOP-7150.2-004 (Flight Software Media Control 1)	Prior to installation on an aircraft, a VDD will be produced containing a form AFRC 80184 and an attached a form AFRC 70010 requesting installation.	No - SMA requirement

29	Prepare software media, as required, for installation.	AFOP-7150.2-004 (Flight Software Media Control 2)	All software media (tape, disk or chip) … identified and physically labeled at the time of production.	No - SMA requirement
30	Prepare the form AFRC 80184.	AFOP-7150.2-004 (Flight Software Media Control 3)	…form AFRC 80184 that uniquely identifies (via checksum(s), file size/modification dates, or other verifiable means) the specific software load that should be installed on the aircraft.	No - SMA requirement
		AFOP-7150.2-004 (Flight Software Media Control 4)	Flight software for a specific flight or block of flights … on a form AFRC 80184.	No - SMA requirement
31	Prepare and execute the software installation procedure.	AFOP-7150.2-004 (Flight Software Media Control 5)	A procedure … for flight software installation into the aircraft computer and for verification of correct loading.	No - SMA requirement
		AFOP-7150.2-004 (Flight Software Media Control 6)	Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures.	No - SMA requirement

4.4 Class I/II Combined

Step	Activity	Applicable SWE and SSS Requirements		Waivable by Center Software TA?
1	Classify the software per this directive	SWE-020	...classify each system and subsystem containing software in accordance with the highest applicable software classification definitions for Classes A, B, C, D, E, F, G, and H software in NPR 7150.2.	Yes
		SSS-001	NASA acquirer performs initial Software Safety Critical Assessment (SSCA).	No - SMA Requirement
		SSS-005	Provider performs SSCA.	No - SMA Requirement
2	Request SQA independent classification and get email from SQA confirming classification and safety criticality.	SWE-132	...perform an independent classification assessment.	Yes
		SWE-133	Determine the software safety criticality in accordance with NASA-STD-8719.13.	Yes
		SWE-160	If a software component is determine to be safety-critical software, then the software component classification shall be Software Class D or higher.	Yes
		SSS-006	Acquirer SMA approves provider's SSCA	No - SMA Requirement
3	Disagreements regarding the SSCA should be elevated up the NASA SMA TA chain per AFPR-7123.2-001	SSS-007	Disagreements with software safety-critical assessment	No - SMA Requirement
4	Maintain the results of the approved SSCA as a quality record in accordance with NPR 1441.1	SSS-008	Provider maintains SSCA.	No - SMA Requirement
	Include software and software safety activities in:			

5	Project Schedule	SWE-016	…document and maintain a software schedule that satisfies the following conditions: a. Coordinates with the overall project schedule. b. Documents the interactions of milestones and deliverables between software, hardware, operations, and the rest of the system. c. Reflects the critical path for the software development activities. d. Adhere to the guidance provided in NASA/SP-2010-3403.	Yes
		SWE-024	Project scheduling and tracking will comply with NPR 7150.2 requirements.	Yes
6	Project Cost Estimates	SWE-015	…establish, document, and maintain one software cost estimate and associated cost parameter(s) for other software projects.	Yes
		SWE-151	The project manager's software cost estimate(s) shall satisfy the following conditions: a. Covers the entire software life cycle. b. Is based on selected project attributes (e.g., assessment of the size, functionality, complexity, criticality, reuse code, modified code, and risk of the software processes and products). c. Is based on the cost implications of the technology to be used and the required maturation of that technology. d. Incorporates risk and uncertainty. e. Includes the cost for software assurance support. f. Includes other direct costs.	Yes
7	Safety Planning	SSS-020	Provider includes software in system safety analysis.	No - SMA Requirement
		SSS-030	Provider SMA traces software safe problems to system hazards.	No - SMA Requirement
		SSS-042	Provider makes software requirements analysis available and update in system safety data package.	No - SMA Requirement
		SSS-044	Provider SMA performs software design analysis.	No - SMA Requirement
		SSS-045	Provider updates system safety data package based on software design analysis.	No - SMA Requirement
		SSS-046	Provider SMA concurs on changes.	No - SMA Requirement

		SSS-049	Update system safe data package based on code analysis.	No - SMA Requirement
		SSS-050	Provider SMA concurs on updates.	No - SMA Requirement
		SSS-060	Provider SMA updates system safety data package based on test analysis.	No - SMA Requirement
8	Project Risk Management System	SWE-086	Including software in project risk management process will comply with NPR 7150.2 requirements	Yes
9	Project Reviews/Status Activities	SWE-093	…analyze software measurement data collected using documented project-specified and/or Center/organizational analysis procedures.	Yes
		SWE-094	…provide access to the software measurement data, measurement analyses, and software development status, as requested, to the sponsoring mission directorate, NASA Chief Engineer, Center and NASA Headquarters SMA, and Center repositories.	Yes
		SWE-018	…regularly hold reviews of software activities, status, and results with the project stakeholders and track issues to resolution.	Yes
		SWE-019	…select and document a software development life cycle or model that includes phase transition criteria for each life-cycle phase.	Yes
		SWE-087	...perform and report the results of software peer reviews or software inspections for: a. Software requirements. b. Software plans. c. Any design items that the project identified for software peer review or software inspections according to the SDPs. d. Software code as defined in the software and/or project plans. e. Software test procedures.	Yes
		SWE-089	...for each planned software peer review or software inspection, record basic measurements.	Yes
		SWE-088	Include software in project review cycle. If software is using a different development life cycle than the project, document how the software life-cycle interfaces with the project life cycle.	Yes

			Following Center engineering review process will comply with NPR 7150.2 requirements.	
		SWE-090	...establish, record, maintain, report, and utilize software management and technical measurements.	Yes
		SSS-021	Acquirer SMA approving member	No - SMA Requirement
		SSS-022	Provider develops and maintain evidence of SSS compliance.	No - SMA Requirement
		SSS-023	Provider identifies new or updated software safety requirements.	No - SMA Requirement
		SSS-024	Provider SMA software Safety-Critical Assessment re-evaluation/approval	No - SMA Requirement
10	V&V Planning and Tracking	SWE-028	…plan software verification activities, methods, environments, and criteria for the project.	Yes
		SWE-029	…plan the software validation activities, methods, environments, and criteria for the project.	Yes
		SWE-030	…record, address, and track to closure the results of software verification activities.	Yes
		SWE-031	…record, address, and track to closure the results of software validation activities.	Yes
		SWE-055	…perform requirements validation to ensure that the software will perform as intended in the customer environment.	Yes
		SWE-067	…verify the requirement to the implementation of each software requirement.	Yes
		SWE-068	…evaluate test results and record the evaluation.	Yes
		SSS-049	Update system safety data package based on code analysis.	No - SMA Requirement
		SSS-054	Verify requirements by method of test.	No - SMA Requirement
		SSS-059	Software safety requirements verified by evaluation, inspection, or demonstration.	No - SMA Requirement
		SSS-061	Provider collects objective evidence of acceptance.	No - SMA Requirement
		SSS-062	Acquirer and Acquirer SMA approves acceptance.	No - SMA Requirement

11	Acceptance Criteria	SWE-034	...define and document the acceptance criteria and conditions for the software.	Yes
12	Requirements Documents Note: Following Center requirements management process will comply with NPR 7150.2 requirements.	SWE-050	…establish, capture, record, approve, and maintain software requirements, including the software quality requirements, as part of the technical specification	Yes
		SWE-051	…perform software requirements analysis based on flowed-down and derived requirements from the top-level systems engineering requirements and the hardware specifications and design.	Yes
		SWE-052	…perform, record, and maintain bi-directional traceability between the software requirement and the higher-level requirement.	Yes
		SWE-053	…track and manage changes to the software requirements.	Yes
		SWE-054	…identify, initiate corrective actions, and track until closure inconsistencies among requirements, project plans, and software products.	Yes
		SSS-009	Provider SMA approves documents with safety requirements.	No - SMA Requirement
		SSS-010	Acquirer SMA approves documents with safety regulations.	No - SMA Requirement
		SSS-038	Unique identifiers for software safety requirements.	No - SMA Requirement
		SSS-039	Hardware/software interface constraints documented	No - SMA Requirement
		SSS-040	Provider SMA analyzes the software safety requirements.	No - SMA Requirement
		SSS-041	Provider SMA concurs on software safety requirements tagging.	No - SMA Requirement
		SSS-045	Provider update system safety data package based on software design analysis.	No - SMA Requirement
		SSS-049	Update system safety data package based on code analysis.	No - SMA Requirement
13	Design Documents	SWE-056	...develop, record, and maintain the software design.	Yes
		SSS-043	Provide a unique tag on software design features	No - SMA Requirement

		SSS-045	Provider update system safety data package based on software design analysis.	No - SMA Requirement
		SSS-049	Update system safety data package based on code analysis.	No - SMA Requirement
14	Test Plans and Procedures, Test Reports	SWE-065	... establish and maintain: a. Software test plan(s). b. Software test procedure(s). c. Software test report(s).	Yes
		SWE-071	...update software test plan(s) and software test procedure(s) to be consistent with software requirements.	Yes
		SSS-051	Provider SMA concurs on test procedures.	No - SMA Requirement
		SSS-052	Provider verifies software impacts included in system testing.	No - SMA Requirement
		SSS-053	Provider SMA witnesses verifications.	No - SMA Requirement
		SSS-055	Verifies system hazards are eliminated or controlled.	No - SMA Requirement
		SSS-057	Testing includes loads, stress, and off-nominal conditions.	No - SMA Requirement
		SSS-058	Testing includes correct and safe operations.	No - SMA Requirement
15	Operations, Maintenance, Disposal Plans	SWE-075	...plan and implement software operations, maintenance, and retirement activities.	Yes
		SSS-063	Regression testing required for implementation of new requirements	No - SMA Requirement
		SSS-065	Provider creates a software retirement plan.	No - SMA Requirement
		SSS-066	Acquire SMA concurrence on software retirement plan.	No - SMA Requirement
16	Product Delivery	SWE-077	...complete and deliver the software product to the customer with appropriate records, including as-built records, to support the operations and maintenance phase of the software’s life cycle.	Yes

		SWE-085	...establish and implement procedures for the storage, handling, delivery, release, and maintenance of deliverable software products.	Yes
		SSS-064	Providers SMA evaluates updates to user manuals.	No - SMA Requirement
17	Configuration Control Plans and Activities	SWE-064	...perform, record, and maintain bi-directional traceability from software design to the software code.	Yes
		SWE-080	...track and evaluate changes to software products.	Yes
		SWE-081	...identify the software configuration items (e.g., software records, code, data, tools, models, scripts) and their versions to be controlled for the project.	Yes
		SWE-079	...develop a software CMP that describes the functions, responsibilities, and authority for the implementation of software CM for the project.	Yes
		SWE-082	...establish and implement procedures to: a. Designate the levels of control through which each identified software configuration item is required to pass. b. Identify the persons or groups with authority to authorize changes. c. Identify the persons or groups to make changes at each level.	Yes
		SWE-083	...prepare and maintain records of the configuration status of software configuration items.	Yes
		SWE-084	...perform software configuration audits to determine the correct version of the software configuration items and verify that they conform to the records that define them.	Yes
		SSS-013	Provider SMA verifies adherence to CMP.	No - SMA Requirement
		SSS-014	Provider gives access to acquirer, acquirer SMA, and IV&V.	No - SMA Requirement
		SSS-025	Provider SMA is voting member of provider software change control process.	No - SMA Requirement
		SSS-026	Acquirer SMA is voting member of acquirer software change control process.	No - SMA Requirement

		SSS-027	Provider SMA evaluates software changes.	No - SMA Requirement
		SSS-028	Provider SMA concurs on software changes and discrepancies.	No - SMA Requirement
		SSS-029	Provider reviews all deficiency reports of safety-critical systems.	No - SMA Requirement
		SSS-030	Provider SMA traces software safety problems to system hazards.	No - SMA Requirement
18	Acquisition Activities	SWE-033	The project manager shall assess options for software acquisition versus development.	Yes
19	Complete a compliance matrix and get the TA to sign it.	SWE-121	[Where approved]…document and reflect the tailored requirement in the plans or procedures controlling the development, acquisition, and/or deployment of the affected software.	Yes
		SWE-125	...maintain a compliance matrix or multiple compliance matrices against requirements in NPR 7150.2, including those delegated to other parties or accomplished by contract vehicles or Space Act Agreements.	Yes
		SWE-139	...comply with the requirements in NPR 7150.2 Requirements Mapping Matrix that are marked with a “project” responsibility and an ”X” consistent with their software classification.	Yes
		SWE-145	When the compliance matrix is used to waive/deviate from applicable “X” requirement(s), the designated TAs...indicate their approval by signature(s) in the compliance matrix itself.	Yes
		SWE-013	...develop, maintain, and execute software plans that cover the entire software life cycle and, as a minimum, address the requirements of this directive with approved tailoring.	Yes
20	Write a software safety plan that includes the project's approach for meeting the requirements of NASA-STD-8719.13	SSS-002	Acquirer meets Standard.	No - SMA Requirement
		SSS-031	Acquirer Software Safety Plan and contents	No - SMA Requirement

21	If any of the safety requirements cannot be met, submit a waiver or deviation package in accordance with AFPR-7123.2-001.	SSS-015	Prepares waiver/deviation package.	No - SMA Requirement
		SSS-016	SMA evaluates and submits waiver/deviation package.	No - SMA Requirement
22	Maintain a copy of all waivers and deviations as a quality record in accordance with NPR 1441.1.	SSS-017	Acquirer SMA elevates waiver/deviation package.	No - SMA Requirement
23	When software is deemed safety critical, impose and implement the requirements in the "S" column of this matrix.	SSS-003 SSS-004	Acquirer imposes Standard. Provider adheres to Standard.	No - SMA Requirement No - SMA Requirement
		SWE-023 **[SAFETY CRITICAL ONLY]**	When a project is determined to have safety-critical software…implement the requirements of NASA-STD-8719.13.	ITA and SMA
24	Work with Flight Instrumentation & Systems Integration Branch Chief to identify requisite skills for the software effort and provide training, as appropriate.	SWE-017	...plan, track, and ensure project-specific software training for project personnel.	Yes
25	Develop and document the software architecture.	SWE-057	...develop and record the software architecture.	Yes
26	Plan for coding, compilation, and testing of lower-level units of the software architecture.	SWE-058	...develop, record, and maintain a design based on the software architectural design that describes the lower-level units so that the units can be coded, compiled, and tested.	Yes
27	Plan for documenting and maintaining bi-directional trace between the following: a. Software requirements and software architecture. b. Software architecture and software design. c. Software requirements and software design. d. Software design and code.	SWE-059	...perform, record, and maintain bi-directional traceability between the following: a. Software requirements and software architecture. b. Software architecture and software design. c. Software requirements and software design. The project manager shall perform, record, and maintain bi-directional traceability between the following: a. Software requirements and software architecture. b. Software architecture and software design.	Yes

	e. Software test procedures and requirements. f. software safety requirements and software-related hazards, controls, conditions, and events.		c. Software requirements and software design.	
		SWE-064	…perform, record, and maintain bi-directional traceability from software design to the software code.	Yes
		SWE-072	...provide and maintain bi-directional traceability from the software test procedures to the software requirements.	Yes
		SSS-011	Provider traces between software safety requirements and system hazards.	No - SMA Requirement
28	Plan for SDA SMA to evaluate and concur with the results of the trace between software safety requirements and software-related hazards, controls, conditions, and events.	SSS-012	Provider traces results approved.	No - SMA Requirement
29	Select and verify adherence to software coding methods, standards, and/or criteria.	SWE-061	...select, adhere to, and verify software coding methods, standards, and/or criteria.	Yes
30	Validate and accredit software tools required to develop or maintain software. For safety-critical software development and evaluate tools for impact on system safety.	SWE-136	...validate and accredit software tool(s) required to develop or maintain software.	Yes
		SSS-018	Provider verifies tools impact on safety	No - SMA Requirement
		SWE-070	...use validated and accredited software models, simulations, and analysis tools required to perform qualification of flight software or flight equipment.	Yes

31	Implement safety-critical requirements from the NPR 7150.2.	SWE-134 **[SAFETY CRITICAL ONLY]**	Implement the following items in the software: a. Safety-critical software is initialized, at first start and at restarts, to a known safe state. b. Safety-critical software safely transitions between all predefined known states. c. Termination performed by software of safety critical functions is performed to a known safe state. d. Operator overrides of safety-critical software functions require at least two independent actions by an operator. e. Safety-critical software rejects commands received out of sequence, when execution of those commands out of sequence can cause a hazard. f. Safety-critical software detects inadvertent memory modification and recovers to a known safe state. g. Safety-critical software performs integrity checks on inputs and outputs to/from the software system. h. Safety-critical software performs prerequisite checks prior to the execution of safety-critical software commands. i. No single software event or action is allowed to initiate an identified hazard. j. Safety-critical software responds to an off-nominal condition within the time needed to prevent a hazardous event. k. Software provides error handling of safety-critical functions. l. Safety-critical software has the capability to place the system into a safe state. m. Safety-critical elements (e.g., requirements, design elements, code components, and interfaces) are uniquely identified as safety critical. n. Requirements are incorporated in the coding methods, standards, and/or criteria to clearly identify safety-critical code and data within source code comments.	With Center SMA concurrence

		SSS-047	Unique tagging of safety-critical code	No - SMA Requirement
		SSS-048	Provider SMA performs code analysis on safety critical code.	No - SMA Requirement
	For acquired custom software			
32	Determine which processes, documents, electronic products, activities, and tasks are required.	SWE-036	...determine which software processes, software documents, electronic products, software activities, and tasks are required for the project and software suppliers.	Yes
33	Define milestones at which progress will be reviewed and audited.	SWE-037	...define the milestones at which the software supplier(s) progress will be reviewed and audited as a part of the acquisition activities.	Yes
34	Establish a procedure for software supplier selection, including proposal evaluation criteria.	SWE-035	...establish a procedure for software supplier selection, including proposal evaluation criteria.	Yes
35	Document acquisition planning decisions.	SWE-038	...document software acquisition planning decisions.	Yes

36	Require supplier to allow NASA to: a. Monitor product integration. b. Review verification activities. c. Review trade studies and source data. d. Audit software development process. e. Participate in software reviews and technical interchange meeting (TIMs).	SWE-039	...require the software supplier(s) to provide insight into software development and test activities; at a minimum, the software supplier(s) will be required to allow the project manager or designate to: a. Monitor product integration. b. Review the verification activities to ensure adequacy. c. Review trades studies and source data. d. Audit the software development process. e. Participate in software reviews and systems and software TIMs	Yes
37	Require the software suppliers to provide NASA with software products and process tracking information in electronic format, including software development and management metrics. The number of software defects assessed against mission success risks for proper resolution (vs. total number of software defects) can be used to meet the intent of software quality metric requirements to collect, analyze, and report out to the project per Class I and Class II. [CAP053:023]	SWE-040	...require the software supplier(s) to provide NASA with software products and software process tracking information, in electronic format, including software development and management metrics.	Yes

38	Require the software supplier to provide NASA with electronic access to source code in modifiable format, including MOTS software, non-flight software.	SWE-042	...require the software supplier(s) to provide NASA with electronic access to the source code developed for the project in a modifiable format, including MOTS software and non-flight software (e.g., ground test software, simulations, ground analysis software, ground control software, science data processing software, and hardware manufacturing software).	Yes
39	Require the software provider to track software changes and non-conformances and provide the data for the project's review.	SWE-043	...require the software supplier to track software changes and non-conformances and provide the data for the project's review.	Yes
40	Participate in joint NASA/supplier audits of the software development process and software CM process.	SWE-045	...participate in any joint NASA/supplier audits of the software development process and software CM process.	Yes
41	Require supplier to provide schedule and schedule updates, as required.	SWE-046	...require the software supplier(s) to provide a software schedule for the project's review and schedule updates, as requested.	Yes
42	Require software supplier to make electronically available the software traceability data for the project's review.	SWE-047	...require the software supplier(s) to make electronically available the software traceability data for the project's review.	Yes
43	Specify re-usability requirements that apply to software development activities to enable future reuse of the software, including models used to generate the software.	SWE-147	...specify re-usability requirements that apply to its software development activities to enable future reuse of the software, including models used to generate the software.	Yes
44	Require the software supplier to notify the project, in the response to the solicitation, as to whether or not open source software will be included in code developed for the project.	SWE-041	...require the software supplier(s) to notify the project, in the response to the solicitation, as to whether or not open source software will be included in code developed for the project.	Yes
45	When safety-critical software is to be acquired, include	SSS-032	Acquirer identifies additional software safety requirements	No - SMA Requirement

	implementation of the requirements in “S” column of this matrix in contracts/MOAs/MOUs.	SSS-033	Acquirer SMA verifies contents of contracts/MOAs/MOUs.	No - SMA Requirement
46	When safety-critical software is to be acquired, require the software provider to develop, baseline, and configuration manages a software safety plan to be evaluated and approved the by NASA project SMA.	SSS-034	Acquirer SMA approves provider’s software safety plan.	No - SMA Requirement
		SSS-035	Provider develops, baselines, and configuration manages software safety plan.	No - SMA Requirement
		SSS-036	Provider obtains approval from Acquirer SMA.	No - SMA Requirement
		SSS-037	Provider software safety plan contents.	No - SMA Requirement
47	Plan for software assurance activities.	SWE-022	…plan and implement software assurance per NASA-STD-8739.8.	Yes
48	Evaluate software for potential reuse and contribute reuse candidates to Agency Software Catalog.	SWE-148	...evaluate software for potential reuse by other projects across the Agency and contribute reuse candidates to the Agency Software Catalog.	Yes

49	Document provisions for the use of auto-code in the software development process including: a. Validation and verification of auto-generation tools. b. CM of the auto-generation tools and associated data. c. Identification of the allowable scope for the use of auto-generated software. d. Verification and validation of auto-generated source code. e. Monitoring the actual use of auto-generated source code compared to the planned use. f. Policies and procedures for making manual changes to auto-generated code. g. CM of the input to the auto-generation tool, the output of the auto-generation tool, and modifications made to the output of the auto-generation tools.	SWE-146	...implement the following items in the software: a. Safety-critical software is initialized, at first start and at restarts, to a known safe state. b. Safety-critical software safely transitions between all pre-defined known states. c. Termination performed by software of safety-critical functions is performed to a known safe state. d. Operator overrides of safety-critical software functions require at least two independent actions by an operator. e. Safety-critical software rejects commands received out of sequence, when execution of those commands out of sequence can cause a hazard. f. Safety-critical software detects inadvertent memory modification and recovers to a known safe state. g. Safety-critical software performs integrity checks on inputs and outputs to/from the software system.	Yes

50	Document provisions for the use of open source software components including: a. Annotate requirements that are to be met by the open source software. b. Usage instructions for the open source component in the software documentation package. c. Address proprietary, usage, ownership, warranty, licensing rights, and transfer rights. d. Ensure future support for the software product is planned and adequate for the project needs. e. V&V the software component to the same level required to accept a similar developed software component for its intended use.	SWE-149	...ensure that when an open source software component is acquired or used, the following conditions are satisfied: a. The requirements that are to be met by the software component are identified. b. The software component includes documentation to fulfill its intended purpose (e.g., usage instructions). c. Proprietary, usage, ownership, warranty, licensing rights, and transfer rights have been addressed. d. Future support for the software product is planned and adequate for project needs. e. The software component is verified and validated to the same level required to accept a similar developed software component for its intended use.	Yes
51	Plan for requirements, documentation, rights, support, V&V, and vendor defect reporting of COTS, GOTS, MOTS, or reused software components.	SWE-027	...satisfy the following conditions when a COTS, GOTS, MOTS, or reused software component is acquired or used: a. The requirements to be met by the software component are identified. b. The software component includes documentation to fulfill its intended purpose (e.g., usage instructions). c. Proprietary rights, usage rights, ownership, warranty, licensing rights, and transfer rights have been addressed. d. Future support for the software product is planned and adequate for project needs. e. The software component is verified and validated to the same level required to accept a similar developed software component for its intended use. f. The project has a plan to perform periodic assessments of vendor reported defects to ensure the defects do not impact the selected software components.	Yes

52	Perform a safety analysis on Off The Shelf (OTS) and reused software to evaluate its ability to meet required functions without impacting safety.	SSS-019	Performs SSCA on OTS and reused software.	No - SMA Requirement
53	At design reviews, reaffirm software classification.	SWE-021	If a system or subsystem evolves to a higher or lower software classification…or there is a change in the safety criticality of the software, the project…update their plan to fulfill the applicable requirements.	Yes
54	Code	SWE-060	...implement the software design into software code.	Yes
55	For safety-critical software, perform a code analysis per NASA-STD-8719.13.	SSS-048	Provider SMA performs code analysis on safety-critical code.	No - SMA Requirement
56	Perform static analysis.	SWE-135	...verify the software code by using the results from static analysis tool(s).	Yes
57	Unit Test (For Class I, Modified Condition and Decision Coverage; For Class II, Decision Coverage)	SWE-062	…unit test the software code per the plans for software testing.	Yes
		SSS-056	Unit testing includes safety-critical software	No - SMA Requirement
58	Software Test	SWE-066	...perform software testing.	Yes
		SWE-069	…record defects identified during testing and track to closure.	Yes
59	Perform validation on targeted platform or high-fidelity simulation.	SWE-073	...validate the software system on the targeted platform or high-fidelity simulation.	Yes
60	Write a VDD that includes: a. What the software is supposed to do. b. How to install the software. c. How to operate the software. d. List of software components and version numbers. e. List of incorporated and outstanding bug fixes.	SWE-063	Provide a software version description for each software release.	Yes
		AFOP-7150.2-004 (Flight Software Media Control 1)	Prior to installation on an aircraft, a VDD will be produced containing a form AFRC 80184 and an attached form AFRC 70010 requesting installation.	No - SMA Requirement
61	Prepare software media, as required, for installation.	AFOP-7150.2-004	All software media (tape, disk or chip)…identified and physically labeled at the time of production.	No - SMA Requirement

		(Flight Software Media Control 2)		
62	Prepare the form AFRC 80184.	AFOP-7150.2-004 (Flight Software Media Control 3)	…Form AFRC 80184 that uniquely identifies (via checksum(s), file size/modification dates, or other verifiable means) the specific software load that should be installed on the aircraft.	No - SMA Requirement
		AFOP-7150.2-004 (Flight Software Media Control 4)	Flight software for a specific flight or block of flights…on a form AFRC 80184	No - SMA Requirement
63	Prepare and execute the software installation procedure.	AFOP-7150.2-004 (Flight Software Media Control 5)	A procedure…for flight software installation into the aircraft computer and for verification of correct loading.	No - SMA Requirement
		AFOP-7150.2-004 (Flight Software Media Control 6)	Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures.	No - SMA Requirement

Chapter 5: Documentation from SWEHB

This section explains the artifact requirement for each software life-cycle phase. To use this section, first determine the classification level of the software from Safety. Then select the column that matches the software classification level.

This is a complete list as detailed in NASA-HDBK-2203, NASA Software Engineering Handbook. The SDA within their SDP, or contractor's approved equivalent SMP, are expected to detail how they plan to meet the intent of the required artifacts. Artifacts can be documents, review presentation material, drawings, source code, and executables. Artifacts can be combined or exist as formal review records. For small projects, documents may be combined in order to meet the intent of the SWE/deliverable with the Center Software's TA (or delegate) approval.

Documents marked with an "X*" under Class IV are required for ***FLIGHT*** software only. Items marked as "X(SC)" are required for ***SAFETY-CRITICAL*** software only.

Source SWEHB	Source NPR (SWE)	Description	C	I/II	D	III	E	IV
		Software Plans from SWEHB						
SDP-SMP	013	Software Management Plan (SMP)	X	X	X	X	X	X
SDP-SMP	013	Software Development Plan (SDP) from SDAs	X	X	X	X	X	X (center-wide SDP)
SCMP	079, 084	Software CMP	X	X	X(SC)			
STP	065, 071	Software Test Plan (STP)	X	X	X	X		X*
Maintenance	075	Software Maintenance Plan (includes Retirement)	X	X	X	X		
SAP	022	Software Assurance Plan (SAP)	X	X	X	X		
Safety	023, 134	Software Safety Plan, if safety-critical software (Safety)	X(SC)	X(SC)	X(SC)			

Source SWEHB	Source NPR (SWE)	Description	C	I/II	D	III	E	IV
		Documents from SWEHB						
Risk	086	Records of Continuous RMP	X	X				
Train	017	Software Training Plan	X	X	X(SC)			
SRS	050, 051	Software Requirements Specification (SRS)	X	X	X	X	X	X (informal)
IDD	059	Interface Design Description (IDD)	X	X	X(SC)			
SWDD	057	Software Design Description (Architectural Design) - SWDD – architectural	X	X	X			
SWDD	056, 058	Software Design Description (Detailed Design) - SWDD – detailed	X	X	X			
SDD	058	Software Data Dictionary (SDD)	X	X	X(SC)			
Test	062, 065, 071	Software Test Procedures	X	X	X	X		X*
STR	030,031, 065, 068	Software Test Reports (STR)	X	X	X	X		X*
SUM	SWEHB	Software User's Manual (SUM)	X(SC)	X(SC)	X(SC)			
Metrics	040	Software Metrics / Tech Perf Metrics (TPM)	X	X	X(SC)			
SVD	063,	Software Version Description (SW VDD)	X	X	X	X		X*
Inspect	087, 088, 089	Peer Reviews	X	X	X(SC)			
Entrance/Exit	015, 151	Software Cost Estimate	X	X	X	X		

Entrance/Exit	027, 147, 149, 041	Requirements on OTS Software and Reuse	X	X	X(SC)			
Entrance/Exit	051, 094	Software Analyses: functional, testable, operable, FMEA, reliability, safety/hazards, life cycle, security	X	X	X(SC)			
Entrance/Exit	020, 132, 133, 160	Preliminary Hazard Analysis/Software Classification/Litmus Test for Safety Critical (PHA)	X	X	X	X	X	X
Source SWEHB	Source NPR (SWE)	Description	C	I/II	D	III	E	IV
Entrance/Exit	020, 132, 133, 160	Preliminary Hazard Analysis/Software Classification/Litmus Test for Safety Critical (PHA)	X	X	X	X	X	X
Entrance/Exit	028, 029, 055	V&V Plan	X	X	X	X		
Entrance/Exit	047, 052, 059, 064, 072	Bi-directional traceability matrix	X	X	X(SC)	X		
Entrance/Exit	070, 136	Software Tools	X	X	X(SC)			
Entrance/Exit	033, 035,	Record of trade-off criteria & assessment (make/buy decision)	X	X	X			
Entrance/Exit	090, 093, 094	Measurement Analysis Results	X	X	X(SC)			
Entrance/Exit	034	Acceptance Criteria Conditions	X	X	X	X		
Entrance/Exit	056	Documentation Plan	X	~~X~~				
Entrance/Exit	SWEHB	Data Flow Diagrams	X	X	X	X		
Entrance/Exit	SWEHB	NPR 7150.2 compliance matrix	X	X	X	X		
Entrance/Exit	077	Functional Configuration Audit (FCA)	X	X	X			
Entrance/Exit	077	Physical Configuration Audit (PCA)	X	X	X			
		Additional SWE documents						
	016, 046	Software Schedule	X	X	X	X		
	055, 066, 067, 068	Code Coverage Test/Analysis Results	X	X	X	X		

5.1 Summary of Deliverables by Life-Cycle Phase

D – Draft **P** – Preliminary **B** – Baseline **U** – Updated/Updated as required **F** - Final
X(F) – assume complete (final) not explicit in NPRs or NASA Handbook

MCR = Mission Concept Review, **SRR** = System Requirements Review, **SWRR** = Software Requirements Review,
MDR = Mission Definition, Review, **SDR** = System Definition Review, **PDR** = Preliminary Design Review,
CDR = Critical Design Review, **SIR** = System Integration Review, **I&T** = Integration and Test,
TRR = Test Readiness Review, **SAR** = System Acceptance Review, **SW Deliv** = Software Delivery,
ORR = Operational Readiness Review

Product	MCR	SRR/ SWRR	MDR	SDR	PDR	CDR	I&T SIR	TRR	SW Deliv SAR	ORR
(from NASA HDBK 2203 Section 7.8 Entrance/Exit Criteria)										
Risk Management Plan (RMP)	P	U	U	U	U	U			U	
Software Schedule	D	P	U	U	B	U				
Software Cost Estimate	D	P	U	U	B	U				
Software Management Plan (SMP)		P	P	B		U	U	U	F	
Configuration Management Plan (CMP)		P	P		B	U				
SDA Software Development Plan (SDP)		P			B	U	U	U	F	
Software Requirements Specification (SRS)		P			B	U	U	U	F	
Software Safety Plan (for safety critical)		P			B	U				
Software Assurance Plan		P	P	P	B	U				
Software Test Plans (STP)					P	B	U	U	F	
Software Design Description (Architectural Design)					B	U	U	U	F	

(SWDD - architectural)										
Software Design Description (Detailed Design) (SWDD - detailed)					P	B	U	U	F	
Software Data Dictionary (SDD)					P	B	U	U	F	
Interface Design Description (IDD)					P	B	U	U	F	
Acceptance Criteria and Conditions					P	B				
Record of trade-off criteria and assessment (make/buy decision)					X(F)	X(F)				
Measurement Analysis Results					X(F)	X(F)			F	
Software Test Procedures (Tests)						P	U	B	F	
Software Maintenance Plan						D	P	P	B	U/F
Software Test Reports (STR)									F	
Software User's Manual (SUM)										B
Software Training Plan										
(from NASA HDBK 2203 Section 7.9 Entrance/Exit Criteria)										
Preliminary Hazard Analysis/Software Classification/Litmus Test for Safety Critical (PHA)		P			B	U	U	U	F	
Requirements on OTS S/W and Reuse		P			B	U	U	U	F	
V&V Plan		P			B	U	U	U	F	
Bi-directional traceability matrix		P			B	U	U	U	F	

Product	MCR	SRR/ SWRR			PDR	CDR	I&T SIR	TRR	SW Deliv SAR	ORR
Software Analyses: functional, testable, operable, FMEA, reliability, safety/hazards, life cycle, security		P			B	U	U	U		
Software Tools		P			B	U	U	U	F	
Software Metrics/Tech Perf Metrics (TPM)					P	U	B	F	F	
NPR 7150.2 compliance matrix						B	U	U	F	
Documentation Plan						B	U	U	F	
Data Flow Diagrams						B	U	U	F	
Software Version Description							P	B	F	
Baseline Software Build								B	F	
Functional Configuration Audit (FCA)								F		
Physical Configuration Audit (PCA)									F	
Code Coverage Test/Analysis Results							B	U	F	
Peer Review				Results						

Chapter 6: NPR 7150.2 Compliance/Comparison

This section describes how the Center requirements align with NPR 7150.2 and, in some places, where the Center requirements exceed those of NPR 7150.2.

6.1 Class C to Class I/II

Class I and II Safety Critical map directly to Class C Safety Critical.

Class I and II Non-Safety Critical map directly to Class C Non-Safety Critical.

6.2 Class D to Class III

Class III Non-Safety Critical maps directly to Class D Non-Safety Critical with four exceptions:

RETAIN SWE-052: …perform, record, and maintain bi-directional traceability between the software requirement and the higher-level requirement

- Maintains requirement of a bi-directional traceability matrix document

RETAIN SWE-057: ...develop and record the software architecture.

RETAIN SWE-058: ...develop, record, and maintain a design based on the software architectural design that describes the lower-level units so that the lower-level units can be coded, compiled, and tested.

Note: Only required for Class III, if code coverage is greater than zero (0) percent.

- Maintains requirement of a Software Design Description (Architectural) – SWDD-architectural
- Maintains requirement of a Software Design Description (Detailed Design) – SWDD-detailed

DELETE SWE-077: ...complete and deliver the software product to the customer with appropriate records, including as-built records, to support the operations and maintenance phase of the software's life cycle.

6.3 Class E to Class IV

Class IV is a superset of Class E. Class IV contains all of the items within Class E with the following additional items required for ***FLIGHT*** software and one deletion at the request of SQA:

RETAIN SWE-053: ...track and manage changes to the software requirements.

RETAIN SWE-063: ...provide a software version description for each software release.

RETAIN SWE-065: …establish and maintain:

a. Software test plan(s).
b. Software test procedures(s).
c. Software test report(s)

RETAIN SWE-066: ...perform software testing.

RETAIN SWE-069: ...record defects identified during testing and track to closure.

RETAIN SWE-071: …update software test plan(s) and software test procedures(s) to be consistent with software requirements.

RETAIN SWE-080: ...track and evaluate changes to software products.

RETAIN SWE-081: ...identify the software configuration items (e.g., software records, code, data, tools, models, scripts) and their versions to be controlled for the project.

RETAIN SWE-148: ...evaluate software for potential reuse by other projects across the Agency and contribute reuse candidates to the Agency Software Catalog.

DELETE SWE-077: ...complete and deliver the software product to the customer with appropriate records, including as-built records, to support the operations and maintenance phase of the software's life cycle.

6.4 Center Airworthiness Requirements on All Center Flight Software (Flight Software Media Control – FSWMC)

The following six requirements from the AFOP-7150.2-004 are enforced on all software that is to be used for flight or non-flight software:

a. FSWMC1: Prior to installation on an aircraft, a VDD will be produced containing a form AFRC 80184 and an attached form AFRC 70010 requesting installation.

- Maintains requirement of a VDD

b. FSWMC2: All software media (tape, disk, or chip) identified and physically labeled at the time of production

c. FSWMC3: Form AFRC 80184 that uniquely identifies (via checksum(s), file size/modification dates, or other verifiable means) the specific software load that should be installed on the aircraft.

d. FSWMC4: Flight software for a specific flight or block of flights on a form AFRC 80184.

e. FSWMC5: A procedure written for flight software installation into the aircraft computer and for verification of correct loading

f. FSWMC6: Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures.

6.5 Documents Changed/Deleted

Deleted Concept of Operations from software. This will be created at the project level early in the creation of the program.

Appendix A, Definitions

Accredit. The official acceptance of a software development tool, model, simulation (including associated data) to use for a specific purpose.

Airworthiness. The capability of an aircraft [or research/science asset] to be operated within a prescribed flight envelope in a safe manner per NPR 7900.3.

Analysis. The post-processing or interpretation of the individual values, arrays, files of data, or execution of information. It is a careful study of something to learn about its parts, what they do, and how they are related to each other.

Application Software. Software designed to help users perform particular tasks or handle particular types of problems, as distinct from software that controls the computer itself. (ISO 24765:2010)

Aviation Safety. Safety efforts targeted at hazards associated with aviation activity (AFPD-8700.1-001).

Bi-directional Traceability. Association among two or more logical entities that is discernible in either direction (to/from an entity). (ISO/IEC/IEEE 24765:2010)

Commercial Off-The-Shelf (COTS) Software. Software defined by a market-driven need, commercially available, and whose fitness for use has been demonstrated by a broad spectrum of commercial users; software product available for purchase and use without the need to conduct development activities; and an item that a supplier offers to several acquirers for general use. (ISO/IEC/IEEE 24765:2010)

Computer. Functional unit that can perform substantial computations including numerous arithmetic operations and logic operations.

Computer Software Configuration Item (CSCI). An aggregation of software that is designated for CM and treated as a single entity in the CM process.

Computer System. A system containing one or more computers and associated software. (ISO/IEC/IEEE 24765:2010)

Contracted Software. Software created for a project by a contractor or subcontractor.

Control Flow. The sequence in which operations are performed during the execution of a computer program. (ISO/IEC/IEEE 24765:2010)

Coverage Analysis. The process of determining the degree to which a proposed software verification process activity satisfies its objective. (RTCA DO-178C)

Data. Information for computer processing (e.g., numbers, text, images, and sounds in a form that is suitable for storage in or processing by a computer).

Data Flow. The sequence in which data transfer, use, and transformation are performed during the execution of a computer program. (ISO/IEC/IEEE 24765:2010)

Deactivated Code. Executable object code (or data) that is traceable to a requirement and, by design, is either not intended to be executed (code) or used (data) or is only executed (code) or used (data) in certain configurations of the target computer environment. (RTCA DO-178C)

Dead Code. Executable object code (or data) that exists as a result of a software development error but cannot be executed (code) or used (data) in any operational configuration of the target computer environment. It is not traceable to a system or software requirement. (RTCA DO-178C)

Decision Coverage. Every point of entry and exit in a program has been invoked at least once and every decision in the program has taken on all possible outcomes at least once. (RTCA DO-178C)

Deviation. A documented authorization releasing a program or project from meeting a requirement before the requirement is put under configuration control at the level the requirement will be implemented.

DO (not an acronym). RTCA document identification.

Embedded Computer System. A computer system that is part of a larger system and performs some of the requirements of that system. (ISO/IEC/IEEE 24765:2010)

Embedded Software. Software that is part of a larger system and performs some of the requirements of that system. (ISO/IEC/IEEE 24765:2010)

Establish and Maintain. Formulation, documentation, use/deployment, and current maintenance of the object (usually a document, requirement, process, or policy) by the responsible project, organization, or individual.

Existing Software. Software that is already developed and available; is usable either as is or with modifications; and that is provided by the supplier, acquirer, or a third party. (ISO/IEC/IEEE 24765:2010)

Extraneous Code. Executable code (or data) that is not traceable to any system or software requirement. (RTCA DO-178C)

Facility Safety. Safety efforts targeted at industrial activity associated with the access to and operation of all facilities, including special support capabilities that are resident within these facilities. (AFPD-8700.1-001)

Flight Software. Software that directly modifies or monitors vehicle operation, whether the software is installed in a system on-board an aircraft or installed in a ground-based system that modifies/monitors aircraft operation. (This directive)

Firmware. Combination of a hardware device and computer instructions or computer data that reside as read-only software on the hardware device. (ISO/IEC/IEEE 24765:2010 for programmable firmware see Programmable Logic Device)

Glueware. Software created to connect the OTS software/reused software with the rest of the system. It may take the form of “adapters” that modify interfaces or add missing functionality, “firewalls” that isolate the OTS, or “wrappers” that check inputs and outputs to the OTS software and may modify to prevent failures.

Government Off-The-Shelf Software (GOTS). Software supplied by the government for reuse in another project. This refers to Government-created software, usually from another project. The software was not created by the current developers (see software reuse). Usually, source code is included and documentation, including test and analysis results, is available (e.g., the Government is responsible for the GOTS software to be incorporated into another systems). (ISO/IEC/IEEE 24765:2010)

Ground Safety. Safety efforts targeted at activity not included within the definition of flight safety. (AFPD-8700.1-001)

Ground Test Safety. Ground safety efforts targeted at project-unique equipment. (AFPD-8700.1-001)

Ground Software. Software that could indirectly impact flight or test operations. This includes software supporting simulation, control room, data processing, or verification and validation test operations. (This directive)

Heritage [Legacy] Software. Existing software that was produced/acquired before DPR-7150.2-001 was implemented. This software may have been classified in using DCP-S-007 Revision C (AFOP-7150.2-004) or before. Software products (architecture, code, requirements) written specifically for one project and then, without prior planning during its initial development, found to be useful on other projects. (See Software Reuse.)

Highly Specialized Information Technology. Highly Specialized IT is a part of, internal to, or embedded in a mission platform. The platform's function (e.g., avionics, guidance, navigation, flight controls, simulation, radar, etc.) is enabled by IT, but not driven by IT itself (e.g., computer hardware and software to automate internal functions of a spacecraft or spacecraft support system such as spacecraft control and status, sensor signal and data processing, and operational tasking). Highly Specialized IT acquisitions may include full development (where the information technology is a primary issue) to modification of existing systems (information architecture is firm and demonstrated in an operational environment) where information technology is not an issue. Real time is often critical -- and few opportunities exist to use COTS or GOTS beyond microprocessors and operating systems because these systems are largely unprecedented or largely unique applications. Certain IT considered Mission Critical because the loss of which would cause the stoppage of mission operations supporting real-time on-orbit mission operations is identified as "Highly Specialized" by the Directorate Associate Administrator. Highly Specialized IT is largely custom, as opposed to COTS or commodity IT systems or applications, and includes coding/applications that are integral parts of the research or science requirements (e.g., Shuttle Avionics Upgrade. Common Engineering IT tools such as Product Lifecycle Management systems, Computer-Aided Design systems, and collaborative engineering systems and environments are not Highly Specialized IT. Representative examples of Highly Specialized IT include Avionics software, real-time control systems, onboard processors, Deep Space Network, spacecraft instrumentation software, wind tunnel control system, human physiology monitoring systems, ground support environment, experiment simulators, Mission Control Center, and Launch cameras. (Source: NPR 2800.1)

Independent Verification and Validation. Verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization. (Source: ISO/IEC/IEEE 24765:2010)

Information Technology. Any equipment or interconnected system(s) or subsystem(s) of equipment that is used in the automatic acquisition, storage, analysis, evaluation, manipulation, management, movement, control, display, switching, interchange, transmission, or reception of data or information by the Agency. (Reference: FAR 2.101) (Source: NPR 2800.1)

Insight. An element of Government surveillance that monitors contractor compliance using Government-identified metrics and contracted milestones. Insight is a continuum that can range from low intensity such as reviewing quarterly reports to high intensity such as performing surveys and reviews. (Source: NPR 7123.1)

Interface. A shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical signal exchanges, and other characteristics. (ISO/IEC/IEEE 24765:2010)

Interpreter. A computer program that translates and executes each statement or construct of a computer program before translating and executing the next. (ISO/IEC/IEEE 24765:2010)

Major Engineering/Research Facility. Used in this directive to show research, development, test, or simulation facilities representing a significant NASA investment (facilities with a Current Replace Value (CRV) equal to or greater than $50M), which contains software that supports programs and projects managed under NPR 7120.5, NPR 7120.7, or NPR 7120.8 and that have a Mission Dependency Index value equal to or greater than 70.

Media. Devices or materials that act as a means of transferring or storing software. (RTCA DO-178C)

Mission Critical. Item or function that should retain its operational capability to assure no mission failure (i.e., for mission success - meeting all mission objectives and requirements for performance and safety). (Source: NPR 8715.3)

Model. A description or representation of a system, entity, phenomena, or process. (Source: NASA-STD-7009) Only for the purpose of this directive, the term "model" refers to only those models that are implemented in software.

Modified Off-The-Shelf Software (MOTS). Software product that is already developed and available, usable either ’as is’ or with modification, and provided by the supplier, acquirer, or a third party. (ISO/IEC/IEEE 24765:2010)

When COTS or legacy and heritage software is reused, or heritage software is changed, the product is considered "modified." The changes can include all or part of the software products and may involve additions, deletions, and specific alterations. An argument can be made that any alterations to the code and/or design of an off-the-shelf software component constitutes "modification," but the common usage allows for some percentage of change before the off-the-shelf software is declared to be modified off-the-shelf (MOTS) software. This may include the changes to the application shell and/or glueware to add or protect against certain features and not to the off-the-shelf software system code directly. (See OTS software.)

Modified Condition/Decision Coverage (MC/DC). Every point of entry and exit in a program has been invoked at least once, every condition in a decision in the program has taken all possible outcomes at least once, every decision in the program has taken all possible outcomes at least once, and each condition in a decision has been shown to independently affect that decision’s outcome. A condition is shown to independently affect a decision’s outcome by varying just that condition while holding fixed all other possible conditions or varying just that condition while holding fixed all other possible conditions that could affect the outcome. (RTCA DO-178C)

Off-The-Shelf (OTS) Software. Software not developed in-house or by a contractor for the specific project now underway. The software is generally developed for a purpose different from the current project. Used in practice as umbrella for COTS, GOTS, and MOTS. (NPR 7150.2)

Open-Source Software. Software where its human-readable source code is made broadly available without cost under an OSS license, which provides conditions on use, reuse, modification/improvement, and redistribution; and often where the software development, management, and planning is done publicly, or easily observable by an individual or organization not previously connected with its open source project. (NPR 7150.2)

Operational Software. Software that has been accepted and deployed, has been delivered to its customer, or is deployed in its intended environment. (NPR 7150.2)

P(Center). Review of P(Center) is the short name for the requirement referred to as SWE-127 in NPR 7150.2.

Partitioning. A technique for providing isolation between software components to contain and/or isolate faults. (RTCA DO-178C)

Primary Mission Objectives. Outcomes expected to be accomplished, which are closely associated with the reason the mission was proposed, funded, developed, and operated (e.g., objectives related to top-level requirements or their flow down).

Process Asset Library. A collection of process asset holdings that may be used by an organization or project. (Source: Capability Maturity Model Integration (CMMI®) for Systems Engineering/Software Engineering/Integrated Product and Process Development Supplier Sourcing)

Program. A strategic investment by a Mission Directorate or Mission Support Office that has a defined architecture and/or technical approach, requirements, funding level, and a management structure that initiates and directs one or more projects. A program defines a strategic direction that the Agency has identified as critical.

Programmable Logic Device. FPGA and CPLD. Programmable firmware that is classified as software and controlled as such. (NPR 7150.2).

Project. A specific investment having defined goals, objectives, requirements, life-cycle cost, a beginning, and an end. A project yields new or revised products or services that directly address NASA's strategic needs. They may be performed wholly in-house; by Government, industry, academia partnerships; or through contracts with private industry.

Range Safety. Safety efforts targeted at flight operations that threaten personnel and property to ensure that the risk of casualty/damage from an out-of-control impact is at or below an acceptable threshold. There is a recognized conceptual overlap with flight safety. It is generally recognized that aircrew are not included within the responsibility of range safety. (AFPD-8700.1-001)

Safety Critical. In accordance with NPR 8715.3 and description in NASA-STD-8719.13, any condition, event, operation, process, equipment, or system that could cause or lead to severe injury, major damage, or mission failure, if performed or built improperly or allowed to remain uncorrected.

Scripts. A sequence of automated computer commands embedded in a program that tells the program to execute a specific procedure (e.g., files with monitoring, logic, or commands used by software to automate a process or procedure).

Simulation. The imitation of the characteristics of a system, entity, phenomena, or process using a computational model. (Source: NASA-STD-7009)

Software. Computer programs, procedures, scripts, rules, and associated documentation and data pertaining to the development and operation of a computer system. (NPR 7150.2)

This definition applies to software developed by NASA, software developed for NASA, COTS software, GOTS software, MOTS software, reused software, auto-generated code, embedded software, the software executed on processors embedded in PLD (See NASA-HDBK-4008.), and open-source software components.

Software Architecture. The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the properties of those components, and the relationships between them. The term also refers to documentation of a system's software architecture. Documenting software architecture facilitates communication between stakeholders, documents early decisions about high-level design, and allows reuse of design components and patterns between projects.

Software Assurance. The planned and systematic set of activities that ensure that software life-cycle processes and products conform to requirements, standards, and procedures. For NASA, this includes the disciplines of software quality (functions of software quality engineering, SQA, and software quality control), software safety, software reliability, software verification and validation, and IV&V.

Software Engineering. The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software (i.e., the application of engineering to software). (Source: IEEE 24765)

Software Item. Source code, object code, control code, control data, or a collection of these items.

Software Peer Review and Inspection. A visual examination of a software product to detect and identify software anomalies, including errors and deviations from standards and specifications. (Source: IEEE 1028, IEEE Standard for Software Reviews and Audits) (Refer to NASA-STD-8739.9 for guidelines for software peer reviews or inspections.)

Software Reuse. A software product developed for one use but having other uses or one developed specifically to be usable on multiple projects or in multiple roles on one project. Examples include, but are not limited to, COTS products, acquirer-furnished software products, software products in reuse libraries, and pre-existing developer software products. Each use may include all or part of the software product and may involve its modification. This term can be applied to any software product (such as requirements and architectures), not just to software code itself. Often, this is software previously written by an in-house development team and used on a different project. GOTS software would come under this category if the product is supplied from one Government project to another Government project. (NPR 7150.2)

Software Tool. A software product developed for one use but having other uses, or one developed specifically to be usable on multiple projects or in multiple roles on one project. (ISO/IEC/IEEE 24765:2010)

Software Validation. Confirmation that the product, as provided (or as it will be provided), fulfills its intended use. In other words, validation ensures that "you built the right thing." (Source: IEEE 1012, IEEE Standard for Software Verification and Validation)

Software Verification. Confirmation that work products properly reflect the requirements specified for them. In other words, verification ensures that "you built it right." (Source: IEEE 1012, ISO/IEC/IEEE 24765:2010)

Statement Coverage. Every statement in the program has been invoked at least once. (RTCA DO-178C)

Static Analysis. The process of evaluating a system or component based on its form, structure, content, or documentation. (ISO/IEC/IEEE 24765:2010)

Structural coverage analysis. An evaluation of the code structure, including interfaces, exercised during requirements-based testing. (RTCA DO-178C)

Subsystem. A secondary or subordinate system within a larger system. (ISO/IEC/IEEE 24765:2010)

Support software. Software that aids in the development or maintenance of other software. (ISO/IEC/IEEE 24765:2010)

System. The combination of elements that function together to produce the capability required to meet a need. (NPR 7150.2)

The elements include hardware, software, equipment, facilities, personnel, processes, and procedures needed for this purpose. (NPR 7123.1)

System software. Software designed to facilitate the operation and maintenance of a computer system and associated programs. (ISO/IEC/IEEE 24765:2010)

Tailoring. The process used to adjust or seek relief from a prescribed requirement to accommodate the needs of a specific task or activity (e.g., program or project). The tailoring process results in the generation of deviations and waivers depending on the timing of the request.

Uncertainty. The estimated amount or percentage by which an observed or calculated value may differ from the true value; a broad and general term used to describe an imperfect state of knowledge or a variability resulting from a variety of factors including, but not limited to, lack of knowledge, applicability of information, physical variation, randomness or stochastic behavior, indeterminacy, judgment, and approximation. (NPR 8000.4, Agency Risk Management Procedural Requirements)

Unit testing. A test of individual programs or modules in order to ensure that there are no analysis or programming errors. (ISO/IEC/IEEE 24765:2010)

Testing of individual routines and modules by the developer or an independent tester. (ISO/IEC/IEEE 24765:2010)

A test of individual programs or modules in order to ensure that there are no analysis or programming errors. (ISO/IEC/IEEE 2382-20 Information Technology--Vocabulary--Part 20: System Development, 20.05.05)

Test of individual hardware or software units or groups of related units. (ISO/IEC/IEEE 24765:2010)

Validation. Proof that the product accomplishes the intended purpose. Validation may be determined by a combination of test, analysis, and demonstration. (NPR 7123.1)

Verification. Proof of compliance with specifications. Verification may be determined by test, analysis, demonstration, and inspection. (NPR 7123.1)

Waiver. A documented authorization releasing a program or project from meeting a requirement after the requirement is put under configuration control at the level the requirement will be implemented.

Wrapper. See glueware definition.

Appendix B, Acronyms

AFOP	Operational Procedures
AFPD	Center Policy Directives
AFPR	Center Procedural Requirements
AFRC	Armstrong Flight Research Center
CCB	Configuration Control Board
CIO	Chief Information Officer
CM	Configuration Management
CMMI®	Capability Maturity Model Integration
CMP	Configuration Management Plan
COTS	Commercial-Off-The-Shelf
CPLD	Complex Programmable Logic Device
CSCI	Computer Software Configuration Item
DCP	Dryden Centerwide Procedure
DPR	Dryden Procedural Requirements
FAM	Flight Assurance Matrix
FAR	Federal Acquisition Regulation
FPGA	Field Programmable Gate Array
FSL	Flight Systems Lead
FSWMC	Flight SoftWare Media Control
GOTS	Government-Off-The-Shelf
HDBK	Handbook
IEC	IEC Electronics (company name)

IEEE	Institute of Electrical and Electronic Engineers
ISO	International Standards Organization
IT	Information Technology
ITA	Independent Technical Authority
IV&V	Independent Verification and Validation
JPL	Jet Propulsion Laboratory
MOA	Memorandum of Agreement
MOTS	Modified-Off-The-Shelf
MOU	Memorandum of Understanding
NASA	National Aeronautics and Space Administration
NPD	NASA Policy Directive
NPR	NASA Procedural Requirements
OTS	Off-The-Shelf
PCE	Project Chief Engineer
PHA	Preliminary Hazard Analysis
PLD	Programmable Logic Device
RMP	Risk Management Plan
RTCA	Radio Technical Commission for Aeronautics
S&MA	Safety and Mission Assurance
SAP	Software Assurance Plan
SDA	Software Development Agent
SDP	Software Development Plan

SHA	System Hazard Analysis
SMP	Software Management Plan
SOW	Statements Of Work
SSCA	Software Safety Criticality Assessment
SSP	System Safety Plan
SSS	Software Safety Standard
SSWG	System Safety Working Group
STD	Standard
SWE	Software Engineering
SWEHB	Software Engineering Handbook
TA	Technical Authority
VDD	Version Description Document

Appendix C, Reference Documents

a. Federal Acquisition Regulation (FAR) 2.101

b. NPD 7120.4, NASA Engineering and Program/Project Management Policy

c. NPR 1441.1, NASA Records Management Program Requirements

d. NPR 2800.1, Managing Information Technology

e. NPR 2810.1, Security of Information Technology

f. NPR 7120.5, NASA Space Flight Program and Project Management Requirements

g. NPR 7120.7, NASA Information Technology and Institutional Program and Project Management Requirements

h. NPR 7120.8, NASA Research and Technology Program and Project Management Requirements

i. NPR 7123.1, NASA Systems Engineering Processes and Requirements

j. NPR 7900.3, Aircraft Operations Management

k. NPR 8621.1, NASA Procedural Requirements for Mishap and Close Call Reporting, Investigating, and Recordkeeping

l. NASA-STD-7009, Standard for Models and Simulations

m. NASA-STD-8719.13, NASA SOFTWARE SAFETY STANDARD

n. NASA-STD-8739.9, Software Formal Inspections Standard

o. NASA/SP-2010-3403, NASA Scheduling Management Handbook

p. NASA FAR Supplement

q. ISO/IEC/IEEE 2382-20, Information Technology – Vocabulary – Part 20: System Development

r. IEEE 1012, Standard for Software Verification and Validation

s. IEEE 1028, Standard for Software Reviews

All documents referenced in this Appendix are located on one of the following websites:

- FAR (https://www.acquisition.gov/browse/index/far)
- NODIS (https://nodis3.gsfc.nasa.gov/)
- NASA Technical Standards System (https://standards.nasa.gov/node/1339)
- NASA FAR Supplement (https://www.acquisition.gov/content/supplemental-regulations)
- ISO standards (https://www.iso.org/standards.html)
- IEC standards (https://www.iec.ch/dyn/www/f?p=103:6:0)
- IEEE standards (https://standards.ieee.org/standard/index.html)

Appendix D, Requirements Mapping Matrix

SWE numbers marked with an X and an asterisk (X*) under Class IV are required for ***FLIGHT*** software only.

Items marked as X(SC) are required for ***SAFETY-CRITICAL*** software only

SWE/SSS	C	I/II	D	III	E	IV
NPR 7150.2						
SWE-013	X	X	X	X	X	X
SWE-015	X	X	X	X		
SWE-016	X	X	X	X		
SWE-017	X	X	X(SC)			
SWE-018	X	X	X	X		
SWE-019	X	X	X	X		
SWE-020	X	X	X	X	X	X
SWE-021	X	X	X	X	X	X
SWE-022	X	X	X	X		
SWE-023	X(SC)	X(SC)	X(SC)			
SWE-024	X	X	X	X		
SWE-027	X	X	X(SC)			
SWE-028	X	X	X	X		
SWE-029	X	X	X	X		
SWE-030	X	X	X	X		
SWE-031	X	X	X	X		
SWE-033	X	X	X	X		
SWE-034	X	X	X	X		
SWE-035	X	X	X(SC)			
SWE-036	X	X	X	X		
SWE-037	X	X	X	X		
SWE-038	X	X	X	X		
SWE-039	X	X	X	X		
SWE-040	X	X	X(SC)			
SWE-041	X	X	X(SC)			
SWE-042	X	X	X(SC)			
SWE-043	X	X	X(SC)			
SWE-045	X	X	X(SC)			
SWE-046	X	X	X	X		
SWE-047	X	X	X(SC)			
SWE-050	X	X	X	X	X	X
SWE/SSS	C	I/II	D	III	E	IV
SWE-051	X	X	X(SC)			

SWE-052	X	X	X(SC)	X		
SWE-053	X	X	X	X		X*
SWE-054	X	X	X(SC)			
SWE-055	X	X	X(SC)			
SWE-056	X	X				
SWE-057	X	X		X		
SWE-058	X	X	X(SC)	X (Code Cov. See Sect 4.3)		
SWE-059	X	X	X(SC)			
SWE-060	X	X	X	X		
SWE-061	X	X	X(SC)			
SWE-062	X	X	X	X		
SWE-063	X	X	X	X		X*
SWE-064	X	X	X(SC)			
SWE-065	X	X	X	X		X*
SWE-066	X	X	X	X		X*
SWE-067	X	X	X(SC)			
SWE-068	X	X	X	X		
SWE-069	X	X	X	X		X*
SWE-070	X	X				
SWE-071	X	X	X	X		X*
SWE-072	X	X				
SWE-073	X	X	X(SC)			
SWE-075	X	X	X	X		
SWE-077	X	X	X		X	
SWE-079	X	X	X(SC)			
SWE-080	X	X	X	X		X*
SWE-081	X	X	X	X		X*
SWE-082	X	X	X(SC)			
SWE-083	X	X	X(SC)			
SWE-084	X	X	X(SC)			
SWE-085	X	X	X	X		
SWE-086	X	X				
SWE-087	X	X	X(SC)			
SWE-088	X	X	X(SC)			
SWE-089	X	X	X(SC)			
SWE-090	X	X	X(SC)			
SWE/SSS	**C**	**I/II**	**D**	**III**	**E**	**IV**
SWE-093	X	X	X(SC)			
SWE-094	X	X	X(SC)			

SWE-121	X	X	X	X		
SWE-125	X	X	X	X	X	X
SWE-132	X	X	X	X	X	X
SWE-133	X	X	X	X	X	X
SWE-134	X(SC)	X(SC)	X(SC)			
SWE-135	X	X	X(SC)			
SWE-136	X	X	X(SC)			
SWE-139	X	X	X	X	X	X
SWE-145	X	X	X	X	X	X
SWE-146	X	X	X(SC)			
SWE-147	X	X				
SWE-148	X	X	X	X		X*
SWE-149	X	X	X(SC)			
SWE-151	X	If software cost >$2M	X	If software cost >$2M		
SWE-160	X	X	X	X	X	X
NASA STD 8719.13						
SSS-001	X(SC)	X(SC)	X(SC)			
SSS-002	X(SC)	X(SC)	X(SC)			
SSS-003	X(SC)	X(SC)	X(SC)			
SSS-004	X(SC)	X(SC)	X(SC)			
SSS-005	X(SC)	X(SC)	X(SC)			
SSS-006	X(SC)	X(SC)	X(SC)			
SSS-007	X(SC)	X(SC)	X(SC)			
SSS-008	X(SC)	X(SC)	X(SC)			
SSS-009	X(SC)	X(SC)	X(SC)			
SSS-010	X(SC)	X(SC)	X(SC)			
SSS-011	X(SC)	X(SC)	X(SC)			
SSS-012	X(SC)	X(SC)	X(SC)			
SSS-013	X(SC)	X(SC)	X(SC)			
SSS-014	X(SC)	X(SC)	X(SC)			
SSS-015	X(SC)	X(SC)	X(SC)			
SSS-016	X(SC)	X(SC)	X(SC)			
SSS-017	X(SC)	X(SC)	X(SC)			
SSS-018	X(SC)	X(SC)	X(SC)			
SWE/SSS	**C**	**I/II**	**D**	**III**	**E**	**IV**
SSS-019	X(SC)	X(SC)	X(SC)			
SSS-020	X(SC)	X(SC)	X(SC)			

SSS-021	X(SC)	X(SC)	X(SC)			
SSS-022	X(SC)	X(SC)	X(SC)			
SSS-023	X(SC)	X(SC)	X(SC)			
SSS-024	X(SC)	X(SC)	X(SC)			
SSS-025	X(SC)	X(SC)	X(SC)			
SSS-026	X(SC)	X(SC)	X(SC)			
SSS-027	X(SC)	X(SC)	X(SC)			
SSS-028	X(SC)	X(SC)	X(SC)			
SSS-029	X(SC)	X(SC)	X(SC)			
SSS-030	X(SC)	X(SC)	X(SC)			
SSS-031	X(SC)	X(SC)	X(SC)			
SSS-032	X(SC)	X(SC)	X(SC)			
SSS-033	X(SC)	X(SC)	X(SC)			
SSS-034	X(SC)	X(SC)	X(SC)			
SSS-035	X(SC)	X(SC)	X(SC)			
SSS-036	X(SC)	X(SC)	X(SC)			
SSS-037	X(SC)	X(SC)	X(SC)			
SSS-038	X(SC)	X(SC)	X(SC)			
SSS-039	X(SC)	X(SC)	X(SC)			
SSS-040	X(SC)	X(SC)	X(SC)			
SSS-041	X(SC)	X(SC)	X(SC)			
SSS-042	X(SC)	X(SC)	X(SC)			
SSS-043	X(SC)	X(SC)	X(SC)			
SSS-044	X(SC)	X(SC)	X(SC)			
SSS-045	X(SC)	X(SC)	X(SC)			
SSS-045	X(SC)	X(SC)	X(SC)			
SSS-045	X(SC)	X(SC)	X(SC)			
SSS-046	X(SC)	X(SC)	X(SC)			
SSS-047	X(SC)	X(SC)	X(SC)			
SSS-048	X(SC)	X(SC)	X(SC)			
SSS-048	X(SC)	X(SC)	X(SC)			
SSS-049	X(SC)	X(SC)	X(SC)			
SSS-049	X(SC)	X(SC)	X(SC)			
SSS-049	X(SC)	X(SC)	X(SC)			
SSS-049	X(SC)	X(SC)	X(SC)			
SSS-050	X(SC)	X(SC)	X(SC)			
SWE/SSS	**C**	**I/II**	**D**	**III**	**E**	**IV**
SSS-051	X(SC)	X(SC)	X(SC)			
SSS-052	X(SC)	X(SC)	X(SC)			
SSS-053	X(SC)	X(SC)	X(SC)			
SSS-054	X(SC)	X(SC)	X(SC)			

SSS-055	X(SC)	X(SC)	X(SC)			
SSS-056	X(SC)	X(SC)	X(SC)			
SSS-057	X(SC)	X(SC)	X(SC)			
SSS-058	X(SC)	X(SC)	X(SC)			
SSS-059	X(SC)	X(SC)	X(SC)			
SSS-060	X(SC)	X(SC)	X(SC)			
SSS-061	X(SC)	X(SC)	X(SC)			
SSS-062	X(SC)	X(SC)	X(SC)			
SSS-063	X(SC)	X(SC)	X(SC)			
SSS-064	X(SC)	X(SC)	X(SC)			
SSS-065	X(SC)	X(SC)	X(SC)			
SSS-066	X(SC)	X(SC)	X(SC)			
AFOP 7150						
AFOP 7150.2 (FSWMC 1)		X		X		X*
AFOP 7150.2 (FSWMC 2)		X		X		X*
AFOP 7150.2 (FSWMC 3)		X		X		X*
AFOP 7150.2 (FSWMC 4)		X		X		X*
AFOP 7150.2 (FSWMC 5)		X		X		X*
AFOP 7150.2 (FSWMC 6)		X		X		X*

Appendix E, Verification Matrix

PARAGRAPH	REQUIREMENT	COMPLIANT	NON-COMPLIANT
3.2, Par 2	This may allow those items to be developed at different assurance levels, minimizing the volume of code that shall be developed to the more stringent standards.	□	□
3.4, Note 2	A software component that is deemed to be safety-critical software shall, by definition, have a Software Safety Classification of Class I or Class II.	□	□
3.6	Software classification is not an exact science and shall be evaluated on a case-by-case basis.	□	□
3.6, e	The definition of long-term delay shall also be project or program specific.	□	□
4.2, Step 8	If a software component is determine to be safety critical software then software component classification shall be Software Class D or higher.	□	□
4.2, Step 11	If a software component is determine to be safety critical software then software component classification shall be Software Class D or higher.	□	□
4.2, Step 14	Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures	□	□
4.3, Step 2	If a software component is determine to be safety critical software, then software component classification shall be NPR 7150.2 software Class D or higher.	□	□
4.3, Step 13	The project manager shall assess options for software acquisition versus development.	□	□
4.3, Step 16	A tailored version of the SWDD, based on the complexity of the developed software, can be	□	□

	submitted in the SDA's SDP and shall be approved by the Software Manager.		
4.3, Step 31	Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures.	□	□
4.4, Step 2	If a software component is determine to be safety-critical software, then the software component classification shall be Software Class D or higher.	□	□
4.4, Step 6a	The project manager's software cost estimate(s) shall satisfy the following conditions: a. Covers the entire software life cycle.	□	□
4.4, Step 6b	The project manager's software cost estimate(s) shall satisfy the following conditions: b. Is based on selected project attributes (e.g., assessment of the size, functionality, complexity, criticality, reuse code, modified code, and risk of the software processes and products).	□	□
4.4, Step 6c	The project manager's software cost estimate(s) shall satisfy the following conditions: c. Is based on the cost implications of the technology to be used and the required maturation of that technology.	□	□
4.4, Step 6d	The project manager's software cost estimate(s) shall satisfy the following conditions: d. Incorporates risk and uncertainty.	□	□
4.4, Step 6e	The project manager's software cost estimate(s) shall satisfy the following conditions:	□	□

	e. Includes the cost for software assurance support.		
4.4, Step 6f	The project manager's software cost estimate(s) shall satisfy the following conditions: f. Includes other direct costs.	□	□
4.4, Step 18	The project manager shall assess options for software acquisition versus development.	□	□
4.4, Step 27a	The project manager shall perform, record, and maintain bi-directional traceability between the following: a. Software requirements and software architecture.	□	□
4.4, Step 27b	The project manager shall perform, record, and maintain bi-directional traceability between the following: b. Software architecture and software design.	□	□
4.4, Step 27c	The project manager shall perform, record, and maintain bi-directional traceability between the following: c. Software requirements and software design.	□	□
4.4. Step 63	Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures.	□	□
6.4, f	FSWMC6: Quality inspection shall verify the correct flight software is loaded for the specified flight according to approved procedures.	□	□

Change Log

Baseline, 06-07-10

Baseline-1, Admin Change, 08-17-10

- Formatting changed to comply with Agency standards.

Revision A, 09-03-14

- Answers findings 439-01 and 439-06 OCE
- Updated to address changes associated with NPR 7150.2
  - Minor changes to verbiage in 7150.2 are denoted by an “A” affixed to the DPR requirement number
  - Removed R.0390, R.0400, R.0410, R.0420, R.0430, R.0440
  - Documents the center’s approach to meeting the intent of the CMMI requirements (R.0590)
  - Requires that center software metrics be established and reported
  - Requires a software safety plan for safety-critical software (R.1812, R.1813)
  - Requires an IV&V Project Execution Plan for those projects selected for IV&V (R.0531)
  - Changes the software classification process (R.0531, R.0532)
  - Requires certain features in safety-critical software (R.1271)
  - Requires the use of static analysis tools (R.1001)
  - Requires validation and accreditation of tools (R.2771)
  - Requires peer review of plans (R.1121)
  - Implements compliance matrix (R.0221, R.0222)
- Updated to address changes associated with NASA-STD-8719.13C
  - Implements software safety criticality assessment and software safety litmus test
  - Requires trace of relationships between software safety requirements and software-related system hazards, controls, conditions and events
  - Requires waiver of applicable requirements that are not met
  - Requires safety criticality determination of tools and COTS software
  - Requires that SMA sit on project decision bodies, review discrepancy reports and approve changes to software critical software
  - Requires that software safety organization participate in evaluation of certification process
  - Requires a software safety plan
  - Requires projects to provide proper resources for software safety
  - Removes requirements related to establishing an official certification process for safety-critical software
  - Refines the definition of safety-critical software
  - Removed duplication between NASA-STD-8719.13 and NPR 7150.2
- Updated to address changes associated with RTCA DO-178C
  - Added and refined requirements to ensure correct relationship is maintained between software components across an interface boundary. (R.0926, R.1002, R.1004)

  - Refined dead code requirements to allow analysis to show that dead code can remain as long as it can be removed during the compile/link process (R.1006)
- Updated descriptions associated with software classification III-S
- Revised Systems Engineering and software lifecycle information.
- Added Appendix F Compliance/Reference Information.

Admin Change, A-1, 12-30-14

- Added Section F.5, F.5.1, F.5.2 and F.5.3
- Answers OCE Finding 439-03

Revision B, 05-09-19

- Formatted to current template
- Updated sections P.2, P.4, Chapters 1, 2, 4, 5, and Appendices A, B, C, D
- Updated document references
- Renumbered paragraphs in Chapter 3
- Added Chapter 6
- Changes address audit findings 553-023 and 553-027

Revision B-1, 09-05-19

- Updated cancelled reference AFOP-8739.8-001, Software Assurance Audit and Corrective Action Procedure, to AFG- 8739.8-002, Software Assurance Audit and Corrective Action Handbook

Revision B-2, 12-05-19

- Removed cancelled references AFOP-7900.3-022, Tech brief (T/B) & Mini Tech Brief (Mini T/B), from section P.4 and renumbered

Admin Change B-3, 05-04-21

- Added page break between chapters.
- Updated use of “must” to “shall”.
- Added Appendix E, Requirement Verification Matrix.

Admin Change B-4, 07-03-23

- Updated obsolete reference AFOP-8715.3-005to AFG-8715.3-018.