White Paper

Development under AC 450.141-1A

Using RTCA DO-178C for Means of Compliance for FAA 14 CFR 450.141

genüen

ideas taking flight

Table of Contents

List of Tables

Genuen's Method for a Means of Compliance of the Federal Aviation Administration's Part 450 Launch and Reentry License Requirements for Software Computing Systems Using DO-178C Processes

On September 30, 2020, the Federal Aviation Administration (FAA) submitted Part 450 of the Title 14 Code of Federal Regulations (14 CFR) for publication, which was driven by the Streamlining Launch and Reentry Licensing Requirements (SLR2) Final Rule. Section 450.141 provides specific requirements regarding licensing of Computing System Safety Items, which are defined as any software or data that implements a capability that, by intended operation, unintended operation, or non-operation, can present a hazard to the public. Shortly after the Part 450 release, on October 15, 2020, the FAA released guidance for an acceptable means of compliance to § 450.141 under AC 450.141-1A, Computing System Safety. On August 16, 2021, the FAA released Revision A updates to AC 450.141-1 under AC.141-1A.

Genuen has significant experience in developing and testing avionics systems utilizing the standard processes defined in RTCA DO-178C – Software Considerations in Airborne Systems and Equipment Certification (and prior versions of DO-178). Pursuant to this knowledge, this paper identifies processes used for DO-178C development that can map to the guidance in AC 450.141-1A. The goal of this analysis is to define an implemental process model for AC 450.141-1A development based on the current Genuen knowledge base that supports fulfillment of the requirements of § 450.141 for licensing submittal for our current and future customers.

Overview of Section 450.141

Part 450 defines the requirements for obtaining and maintaining a license from the FAA for launch and reentry (or both) of a space vehicle. Section 450.141 specifically addresses the prescribed hazard controls for safety-critical computing systems.

Note: Section 450.141 focuses on the safety requirements for software, whereas Section 450.143 describes the requirements of the hazard controls for safety-critical hardware.

Section 450.141 is broken into sections (a) through (d) as described below:

(a) **Identification of Computing System Safety Items**
Identification of the Computing System Safety Items and their associated Level of Criticality (FAA, 2020a, p. 378-380).

(b) **Safety Requirements**
Identification, evaluation, implementation, and verification and validation for all Safety Requirements associated with each Computing System Safety Item (FAA, 2020a, p. 380-384).

(c) **Development Process**
Documentation of the Development Process used for implementation, verification, and validation of the Safety Requirements (FAA, 2020a, p. 384-389)

(d) **Application Requirements**
Defines the minimum set of documentation and data required for license application submittal (FAA, 2020a, p. 389-396).

Overview of Section AC 450.141-1A

In October 2020, the FAA issued Advisory Circular (AC) 450.141-1A Computing Systems to provide a means of compliance to § 450.141. AC 450.141-1A was written to provide flexibility, offering multiple ways to show compliance. Most of these are based upon other standards from the space and defense industries.

AC 450.141-1A defines a means to identify the computing system safety items that present hazards to the public. This is achieved through analysis of all software functions in a way that provides compliance with § 450.141(a). The list of computing system safety items should include all software functions that perform safety-related functions based on functional hazard analysis per § 450.107(b). Appendix B of AC 450.141-1A provides two methods for conducting the computer system hazard analysis using either Software Failure Modes and Effects Analysis (SFMEA) or Software Fault Tree Analysis (SFTA), with corresponding examples for each. There are five methods provided by AC 450.141-1A for assigning criticality levels, all of which are based on the severity of the hazard to the public and the degree of control of the computing system safety item. AC 450.141-1A references several other industry documents for determining the degree of control and severity of hazard categories (FAA, 2020b, p. 16-18, 42-53).

For defining the safety requirements per § 450.141(b), AC 450.141-1A describes a means of compliance by identification, formal inspections, implementation, and verification. This addresses safety requirements specific to the computing system safety item functionality, including power requirement, anomaly/fault detection and responses, interfaces, maintenance, and other functional requirements. Also included are safety requirements relating to process, such as Verification and Validation, Configuration Management, Standards, Security, etc. (FAA, 2020b, p. 29-41)

AC 450.141-1A also provides guidance for compliance to the safety measures required for the development process called out under § 450.141(c). The development process should be based on the level of criticality of each computing system safety item identified within § 450.141(a). Based on the level of rigor required, the process should become accordingly stringent, adding to the confidence level that the safety requirements have been properly implemented and verified. Use of industry standards are encouraged to provide a compelling rationale for acceptance of the development process (FAA, 2020b, p. 18-26).

As defined above, § 450.141(d) addresses the requirements for the licensing application (FAA, 2020a, p. 389-396). AC 450.141-1A merely summarizes these requirements. The application must include the following:

- Computing system safety items with their associated level of criticality
- Safety requirements for each computing system safety item
- Documentation of the development process from requirements through verification and validation
- Evidence of implemented requirements and associated test artifacts (FAA, 2020b, p. 27-28).

Note: The FAA also indicates that there will be a second Advisory Circular released in the third quarter of 2021 to address compliance to § 450.141 for Mission Data Loads (AC 450.141-2).

Overview of DO-178C

RTCA is an organization which creates industry standards that are recognized and referenced by Federal Aviation Administration (FAA) regulations. DO-178C provides comprehensive guidance for the development of airborne software. It has become the universal basis for development of airborne software in avionics applications and is seen as a fundamental process model for both hardware and software standards across multiple industries (RTCA, 2011).

The purpose of DO-178C is to define the lifecycle processes that must be followed and the objectives that must be met to certify airborne software. DO-178C provides graduated levels of process rigor based on Design Assurance Levels (DAL). These span from A–E and are assigned based on the impact of possible software failure (A being catastrophic to the safety of the aircraft, operators, or passengers; E

having no effect on safety). Higher DAL levels have increasingly stringent processes that must be followed to achieve certification and the specifics of these processes are defined as objectives (RTCA, 2011).

The current DO-178C process includes the following fundamental components:

- Planning (RTCA, 2011, p. 25-30).
- Development (RTCA, 2011, p. 31-38).
- Verification (RTCA, 2011, p. 39-52).
- Configuration Management (RTCA, 2011, p. 53-60).
- Quality Assurance (RTCA, 2011, p. 61-64).
- Certification (RTCA, 2011, p. 65-67).

DO-178C is widely considered one of the most rigorous and stringent software product development standards. It requires a comprehensive understanding of not only the process guidelines, but also the intent of the objectives and the required supporting documentation. DO-178C is part of a suite of standards utilized in the aerospace industry that includes:

- DO-254, which sets similar requirements for complex hardware used in mission-critical avionics
- SAE ARP4754A, which addresses system-level concerns
- SAE ARP4761 which addresses the safety assessment process (RTCA, 2011; 2005; SAE, 2010; 1996).

Implementing AC 450.141-1A Using DO-178C

DO-178C can provide a roadmap to creating a certification basis for AC 450.141-1A. DO-178C details necessary software lifecycle processes based on a software safety assessment. With increased hazard comes increased rigor. Performing the activities specified in DO-178C can be used to fulfill the dictates of AC 450.141-1A.

In the tables below, highlight is used to provide readability. In the Levels of Criticality table, different color highlights are used to distinguish between the different comparable criticality levels. For the remaining tables where the AC 450.141-1A and DO-178C processes are defined, the gray highlighting identifies the rows where sections of AC 450.141-1A are described. The light blue highlighting in those tables identifies rows describing the DO-178C processes. The DO-178C light blue rows following the gray rows indicate the processes that roughly map to the AC 450.141-1A processes. Note that some gray rows are not followed by light blue rows, indicating that there is no process in DO-178C which corresponds to that particular AC 450.141-1A process.

Identification of Computing System Safety Items

Assessing the criticality of the software is essential to the software lifecycle processes. Assigning a Design Assurance Level to the computing system item is

the foundation on which the rest of the processes are built. While AC 450.141-1A specifies performance of this activity, DO-178C carries the assumption that it has already been performed, usually in accordance with SAE ARP4761: Guidelines and methods for conduction the safety assessment process on civil airborne systems and equipment. While the details of how to perform the safety assessment are outside the scope of DO-178C, the application of that work product maps directly to AC 450.141-1A.

Table 1 Identification of Computing System Safety Items

Identification of Computing System Safety Items						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
6	2 2.3 Figure 2-1 *SAE ARP4754A* *SAE ARP4761*	Identification of Computing System Safety Items	Each software program with any safety impact is identified and is assigned the appropriate criticality level.	System Design and Software Configuration Index (SCI)	This AC 450.141-1A objective falls **outside of the scope** of DO-178C. 450.141 focuses on **Computing System Safety Items**, whereas DO-178C addresses all software.	Some
6.1	2.3 2.3.1 *SAE ARP4761*	Identify each Computing System Safety Item	Analyze the system requirements, Functional Hazard Analysis, and architecture to identify all software safety items (programs or data). Analysis may include techniques such as Software Failure Modes Effects Analysis (SFMEA) and Software Fault Tree Analysis (SFTA).	System Design and SCI	This AC 450.141-1A objective falls **outside of the scope** of DO-178C, but the **FHA** and **Safety Assessment** processes are indicated as a **prerequisite**.	Some
6.2	2.3 2.3.2 2.3.3 *SAE ARP4761* *RCC 319-19* *MIL-STD-882E* *NASA-GB-8719.13*	Assessment of Computing System Criticality	Specify the level of criticality of each software safety item previously identified using one of the following methods: 1) Assumption of High Criticality 2) RCC 319-19 Section A3.1.1 3) MIL-STD-882E Section 4.4 4) NASA-GB-8719.13 Section 3.1.2 5) AC 450.141 Fault Tolerance Analysis	Software Requirements Document (SRD), Software Plans/Processes	Section 2.3.2 and 2.3.3 of DO-178C address defining the software level (**DAL**) based on **Failure Condition Categorization**, which comes from the system safety assessment process. **Highest criticality level** of any of these approaches identified in AC450.141-1A corresponds roughly to DO-178C **DAL B**. See **Error! Reference source not found.** for a detailed comparison of criticality levels between DO-178C and the methods defined in AC 450.141-1A.	Some

Levels of Criticality

The levels of criticality produced by the AC 450.141-1A and civil aviation safety assessment processes are similar. They ultimately depend on the level of hazard to people, property, and the environment created by a software failure. The results of the AC 450.141-1A safety assessment can then be mapped to the Development Assurance Levels of DO-178C to determine the required rigor of the software lifecycle processes to be used as the certification basis for AC 450.141-1A. Note that the highest DO-178C criticality level that an AC 450.141-1A safety assessment can map to is DAL B.

Table 2 Levels of Criticality

DO-178C		**RCC 319-19**	**MIL-STD-882E**	**NASA-GB-8719.13**	**AC 450.141-1A**
Software Level Definition	Failure Condition Category	Software Categories	Software Safety Criticality Index	Software Safety Effort	Fault Tolerance
Level A: Software whose anomalous behavior, as shown by system safety assessment process, would cause or contribute to a failure of system function resulting in a catastrophic failure condition for the aircraft.	**Catastrophic** - Failure Conditions, which would result in multiple fatalities, usually with the loss of the airplane.	**No equivalent Software Category**	**No equivalent Software Safety Criticality Index**	**No equivalent Software Safety Effort**	**No equivalent Fault Tolerance**
Level B: Software whose anomalous behavior, as shown by system safety assessment process, would cause or contribute to a failure of system function resulting in a hazardous failure condition for the aircraft.	**Hazardous** - Failure Conditions, which would reduce the capability of the airplane or the ability of the flight crew to cope with adverse operating conditions to the extent that there would be: - A large reduction in safety margins or functional capabilities; - Physical distress or excessive workload such that the flight crew cannot be relied upon to perform their tasks accurately or completely, or - Serious or fatal injury to a relatively small number of the occupants other than the flight crew.	**Safety-critical** - Software in a partitioned system shall be assigned to a safety-critical partition when it exercises control over safety systems. Software in a safety-critical partition is necessary to provide public safety functions. This includes, but is not limited to, the following. a. Software that evaluates predefined safety criteria using input tracking data, or lack of tracking data, and recommends or initiates FTS actions; b. Software used to communicate with external systems or other partitions when the communication: may initiate a hazardous state; verify proper operation of software executing in a hazardous state; or direct software executing in a hazardous state to transition to a non-hazardous state.	**SwCI 1** - Program shall perform analysis of requirements, architecture, design, and code, and conduct in-depth safety specific testing. **SwCI 2** - Program shall perform analysis of requirements, architecture, and design; and conduct in-depth safety-specific testing.	**"Full" Software Safety Effort** - Systems and subsystems that have severe hazards which can escalate to major failures in a very short period of time require the greatest level of software safety effort. Some examples of these types of systems include life support, fire detection and control, propulsion/pressure systems, power generation and conditioning systems, and pyrotechnics or ordnance systems. These systems may require a formal, rigorous program of quality and safety assurance to ensure complete coverage and analysis of all requirements, design, code, and tests. Safety analyses, software development analyses, safety design features, and Software Assurance (SA) oversight are highly recommended. In addition, IV&V activities may be required.	**Zero fault tolerant** – a single fault in the computing system safety item could result in an adverse consequence. **Single fault tolerant** – a fault in the computing system safety item requires a second, independent fault in order to result in an adverse consequence. **Critical informational** – a fault in the computing system safety item results in erroneous information that the operator could not readily perceive as erroneous, which can cause an operator to take actions that result in an adverse consequence.

DO-178C		RCC 319-19	MILS-STD-882E	NASA-GB-8719.13	AC 450.141-1A
Software Level Definition	Failure Condition Category	Software Categories	Software Safety Criticality Index	Software Safety Effort	Fault Tolerance
Level C: Software whose anomalous behavior, as shown by system safety assessment process, would cause or contribute to a failure of system function resulting in a major failure condition for the aircraft.	**Major** - Failure Conditions, which would reduce the capability of the airplane or the ability of the crew to cope with adverse operating conditions to the extent that there would be, for example, a significant reduction in the safety margins or functional capabilities, a significant increase crew workload or in conditions impairing crew efficiency or discomfort to the flight crew, or physical distress to passengers or cabin crew, possibly including injuries.	**Support-critical** - Software in a partitioned system shall be assigned to a support-critical partition when it supports execution of safety-critical software. Software in a support-critical partition does not directly provide public safety functions. This includes, but is not limited to, the following. a. Software that performs power-on self-test (POST) operations. b. Software that prepares and configures processors for use. c. Software that initializes input ports, output ports, or communication channels. d. Software that performs memory tests after POST operations. e. Software that loads kernel software or application software. f. Software that verifies successful load of kernel software or application software. g. Software that provides communication services for external systems and vehicle. h. Software that can result in inadvertent activation of the FTS, and does not present hazard to personnel nor affect the ability to issue termination when required. For software of this type, the program must fully accept, in writing, the added mission assurance risk. It is highly recommended that software of this type be allocated to a safety-critical partition.	**SwCI 3** - Program shall perform analysis of requirements and architecture; and conduct in-depth safety-specific testing.	**"Moderate" Software Safety Effort** - Systems and subsystems which fall into this category typically have either 1) a limited hazard potential or 2) the response time for initiating hazard controls to prevent failures is long enough to allow for human operators to respond to the hazardous situation. These systems require a rigorous program for safety assurance of software identified as safety-critical. Non-safety-critical software must be regularly monitored to ensure that it cannot compromise safety controls or functions. Some analyses are required to assure there are no "undiscovered" safety-critical areas that may need software safety features. Some level of Software Assurance oversight is still needed to assure late design changes do not affect the safety criticality. A project of this level may require IV&V. However, it is more likely to require a software Independent Assessment (IA). Software independent assessment (IA) is defined as a review of and analysis of the program/project's system software development lifecycle and products. The IA differs in scope from a full IV&V program in that IV&V is applied over the lifecycle of the system whereas an IA is usually a one time review of the existing products and plans. In many ways, IA is an outside audit of the project's development process and products (documentation, code, test results, and others).	**Dual fault tolerant** – a fault in the computing system safety item requires two or more independent faults in order to result in an adverse consequence. **Informational** – a fault in the computing system safety item results in evidently erroneous information that, if not detected, could cause an operator to take actions that result in an adverse consequence.
Level D: Software whose anomalous behavior, as shown by system safety assessment process, would cause or contribute to a failure of system function resulting in a minor failure condition for the aircraft.	**Minor** - Failure Conditions, which would not significantly reduce airplane safety, and which involve crew actions that are well within their capabilities. Minor Failure Conditions may include, for example, a slight reduction in safety margins or functional capabilities, a slight increase in crew workload, such as routine flight plan changes, or some physical discomfort to passengers or cabin crew.	**No equivalent Software Category**	**SwCI 4** - Program shall conduct safety-specific testing.	**"Minimum" Software Safety Effort** - For systems in this category, either the inherent hazard potential of a system is very low or control of the hazard is accomplished by non-software means. Failures of these types of systems are primarily reliability concerns. Software development in these types of systems must be monitored on a regular basis to ensure that safety is not inadvertently compromised or that features and functions are added which make the software safety-critical. A formal program of software safety is not usually necessary. Of course, good development practices and SA are always necessary.	**No equivalent Fault Tolerance**

DO-178C			RCC 319-19		NASA-GB-8719.13		AC 450.141-1A
Software Level Definition	Failure Condition Category		Software Categories		Software Safety Effort		Fault Tolerance
Level E: Software whose anomalous behavior, as shown by system safety assessment process, would cause or contribute to a failure of system function with no effect on aircraft operational capability or pilot workload.	**No Safety Effect** - Failure Conditions that would have no effect on safety, for example, Failure Conditions that would not affect the operational capability of the airplane or increase crew workload.		**Non-critical** - Software in a partitioned system shall be assigned to a non-critical partition when it does not meet the criteria for safety-critical or support-critical software. This includes, but is not limited to, software used to monitor system performance (e.g., processor temperature, processor throughput, cycle time, memory usage, tasking executive statistics). Note that no safety requirements are imposed upon non-critical software in a partitioned system.		**"Minimum" Software Safety Effort** (see Level D row above) – *Level E may be applied to systems in this category if the proper monitoring mechanisms are in place.*		**Non-safety** – a fault in the computing system safety item has no potential to result in an adverse consequence.

Safety Requirements

AC 450.141-1A calls for specifying the computing system safety item requirements. These safety requirements spell out the necessary function, capability, and attributes for each of the previously identified safety items. Specified requirements must be correct. They must be implemented and that implementation must be verified and validated. The previously determined level of criticality dictates the rigor of these processes.

DO-178C defines these processes for commercial aviation. Development, implementation, and verification of safety requirements are laid out in detail. Checklists are provided for each activity and levels of rigor are defined. The level of independence required for verification at each level is also covered. The evidence produced through these processes can establish a certification basis for the computing system safety items based on Table 1 - Identification of Computing System Safety Items in accordance with AC 450.141-1A.

Table 3 Safety Requirements

Safety Requirements						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
7		Safety Requirements	The safety requirements for each computer safety critical item are developed and tested.	SRD, Software Design Document (SDD), Source Code Files, Executable Files, Software Verification Results (SVR), Trace Data		Extensive
7.1	Table A-2 #1 Table A-2 #2 5.1.1.a 5.1.1.b	Identification of Safety Requirements	Those software requirements that impact safety are identified.	SRD	**Software** requirements are decomposed from System requirements in DO-178C. Similarly, the **safety** requirements are **identified** and **decomposed** from the System requirements per AC 450.141-1A. Any **safety** requirements that are not identified by the System requirements must be considered **derived**, and as such, must undergo the same level of scrutiny within the **system safety assessment** process.	Extensive
	Table A-2 #1 5.1.1.a	Software Requirements are Developed	High-level software requirements are developed from System-level requirements.	SRD, Trace Data	Only **safety** requirements need to be identified for AC 450.141-1A.	Extensive
	Table A-2 #2 5.1.1.b	Derived Software Requirements are defined	Software requirements that don't come directly from system requirements are defined, identified, and provided to the system safety assessment process.	SRD	Only applies to derived **safety** requirements.	Extensive

Safety Requirements						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
7.2	Table A-3 6.3.1	Ensuring Safety Requirements are Complete and Correct	All safety requirements are reviewed against standards for completeness and correctness.	SRD	AC 450.141-1A requires **independence** for validation of requirements for **all levels of criticality**, whereas DO-178C only requires **independence** for **DAL A and B**. Requirement **standards** are not called out in AC 450.141-1A. **Traceability** to system requirements is not required and compatibility with the **target computer** is not identified.	Extensive
	Table A-3 6.3.1	Verification of the Outputs of the Software Requirements Process	All software requirements are reviewed against system requirements to ensure: 1) Software requirements comply with system requirements 2) Software requirements compatible with target computer 3) Software requirements verifiable 4) Software requirements conform to standards 5) Software requirements traceable to the system requirements 6) Algorithms are accurate	SVR	Items 2), 4), and 5) are not required by AC 450.141-1A. **Independence** is required in all reviews/inspections of the safety requirements. Although a **requirement standard** is not required by AC 450.141-1A, referencing a standard may produce a **compelling rationale** for the acceptance of a development process	Extensive
7.3	Table A-2 #3 Table A-2 #4 Table A-2 #5 Table A-2 #6 Table A-2 #7 Table A-4 Table A-5 5.2.1.a 5.2.1.b 5.2.1.c 5.3.1.a 5.4.1.a 6.3.2 6.3.3 6.3.4 6.3.5 6.6 *MIL-STD-882E*	Implementation and Verification of Safety Requirements	Safety and non-safety software requirements are implemented in software. All safety requirements are verified and validated by a team from a different department or organization than the development team. Verification and validation methods are proportional to the level of criticality of each computing system safety item.	SDD, Source Code Files, Executable Files, SVR, Trace Data	Per AC 450.141-1A, Implementation and Verification/Validation **levels of rigor** are based on the level of criticality of the computing system safety item. **Independence** is required for testing of “safety critical” items. AC 450.141-1A defines traceability from **requirements to verification/validation evidence**. DO-178C defines tracing between all **requirement levels** (system-> high-level -> low-level), to **code**, and to **test cases**, and **results**. Per AC 450.141-1A, “There need not be a separate implementation process for safety requirements; the applicant’s normal process for implementing software requirements is sufficient.”	Extensive

Safety Requirements						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	Table A-2 #3 5.2.1.a	Software Architecture is Created	A software architecture is created as a framework for implementing the Software Requirements.	SDD	This is not specifically identified within AC 450.141-1A, but **good designs start with the architecture**. Suggest creating the architecture for **all levels of criticality**.	Extensive
	Table A-2 #4 5.2.1.c	Software Design is Developed	Software Designs are created that implement the software requirements within the defined software architecture.	SDD, Trace Data	This is not specifically identified within AC 450.141-1A but **should be defined for criticality levels** that correspond to **DAL C and above**.	Extensive
	Table A-2 #5 5.2.1.b	Derived Software Design Components are defined	Software design components that don't come directly from software requirements are defined, identified and provided to the system process for safety assessment.	SDD	This is not specifically identified within AC 450.141-1A but **should be defined for criticality levels** that correspond to **DAL C and above**. Any **derived** design or low-level requirement should be **analyzed** under the system safety assessment process.	Extensive
	Table A-4 6.3.2 6.3.3	Verification of the Outputs of the Software Design Process	The review and analysis of software design artifacts confirm that software design components satisfy the following objectives: 1) Software design components comply with the software requirements 2) Software design components are accurate and consistent 3) Software design components are compatible with the target computer 4) Software design components are verifiable 5) Software designs conform to standards 6) Software design components are traceable to software requirements 7) Algorithms are accurate 8) Software architecture is compatible with software requirements 9) Software architecture is consistent 10) Software architecture is compatible with the target computer 11) Software architecture is verifiable 12) Software architecture conforms to standards 13) Software partitioning integrity is confirmed	SVR	Items 3), 5), 6), 10), and 12) from DO-178C and indicated in the Activity column are not required by AC 450.141-1A. **Independence** is required in all reviews/inspections of the safety requirements as identified in AC 450.141-1A. Although a **design standard** is not required by AC 450.141-1A, referencing a standard may produce a **compelling rationale** for the acceptance of a development process	Extensive
	Table A-2 #6 5.3.1.a	Software Code is Developed	Source Code is developed that implements the software design. This may be accomplished with hand coded software or machine generated software, or a combination of both.	Source Code file(s), Trace Data	Traceability to source code is not mentioned in AC 450.141-1A. Instead the focus is on **testing of the safety requirements** defined for the computing system safety items, which implies the source code is properly designed and implemented. Processes utilized to meet DO-178C requirements ensure the code implements the requirements through **tracing to low-level requirements** and **verification of all levels** of software requirements.	Extensive
	Table A-2 #7 5.4.1.a	Object Code is Developed	Object code is compiled from the source code modules.	Executable file(s)	As with the source code, AC 450.141-1A does not provide details regarding object code development, but instead focuses on testing of the safety requirements. DO-178C **integration processes** provide a means to ensure that the object code is correctly implemented.	Extensive
	Table A-2 #7 5.4.1.a	Object Code is Developed	Object code is compiled from the source code modules.	Executable file(s)	As with the source code, AC 450.141-1A does not provide details regarding object code development, but instead focuses on testing of the safety requirements. DO-178C **integration processes** provide a means to ensure that the object code is correctly implemented.	Extensive

Safety Requirements						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	Table A-5 6.3.4 6.3.5 6.6	Verification of the Outputs of the Software Coding and Integration Process	The review and analysis of the software coding process confirms the code satisfies these objectives: 1) Source Code complies with the software design 2) Source Code complies with the software architecture 3) Source Code is verifiable 4) Source Code conforms to standards 5) Source Code is traceable to software design 6) Source Code is accurate and consistent 7) Output of software integration process is complete and correct 8) Parameter Data File Item is complete and correct 9) Verification of Parameter Data File Item is achieved	SVR	Items 4), 5), 8), and 9) from DO-178C and indicated in the Activity column are not required by AC 450.141-1A. **Independence** is required in all reviews/inspections of the safety requirements. Although a **coding standard** is not required by AC 450.141-1A, referencing a standard may produce a **compelling rationale** for the acceptance of a development process. There is **no definition of Parameter Data Item**/File within AC 450.141-1A. It is expected that PDI will be included in AC 450.141-2 when it is released.	Extensive

Development Processes

AC 450.141-1A asks the applicant to describe the development processes used for the computing system safety items. This description includes assigning responsibility for each task, the review and approval process and its level of independence, training required for assigned personnel, tracing the safety requirements to the verification and validation evidence, configuration management, testing processes, how to handle re-use of a computing system safety item, and how to handle 3rd party computing system safety items or item components. Additionally, AC 450.141-1A offers considerations to guide the applicant in formulating these processes, including the use of standards, problem reports, configuration management, quality assurance, and maintainability.

In the world of commercial aviation, the description of the development processes is a set of planning documents that must be accepted by the regulatory authorities at the start authorization of the project. DO-178C specifies the items that must be covered in the planning documents. This list largely mirrors the same aspects of AC 450.141-1A. It covers the requirement and software development processes, as well as the supporting “integral processes” of verification, quality assurance, and configuration management. As the project moves along, evidences are created by following the proscribed processes. The plan and the resulting evidences are made available to the regulators when seeking certification.

Table 4 Development Process

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
8	Table A-10 #1 Table A-10 #2 9.0.a 9.0.b	Development Process	Section 8 describes the creation and implementation of the software development process.	Software Development Plan (SDP) and other accompanying plans	AC 450.141-1A does not indicate any prior approval of plans by the FAA for development of the product to be licensed.	Extensive
	Table A-10 #1 9.0.a	Communication Between the Applicant and the Certification Authority	Establish communication and understanding between the applicant and the certification authority throughout the software lifecycle to assist the certification process.	Plan for Software Aspects of Certification (PSAC)	**Communication** with the FAA **early** and **often** is a **risk mitigator**. In the aviation industry, the FAA has defined Stages of Involvement (SOI) audits at critical times in the program. These SOI audits provide the FAA a means to identify how well the applicant is following their defined processes. It also provides the applicant an assessment of how the FAA views their process and compliance efforts. These SOI audits can be a useful tool in providing confidence to the FAA that the applicant is doing their development right and can accelerate and prepare the process to get a license approval under AC 450.141-1A.	Extensive

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	Table A-10 #2 9.0.b	Means of Compliance	Gain agreement on the means of compliance through approval of the Plan for Software Aspects of Certification.	PSAC	DO-178C defines that through a certification liaison process, the certification authority should provide agreement to the PSAC. Often, all DO-178C plans are submitted to the FAA for approval as soon as possible to gain agreement on all development processes being used. This helps to mitigate risk of rework if the plans are not approved for development. Although it is not required or even suggested under AC 450.141-1A, gaining approval of the development processes early will also help to mitigate risk of future rework if the FAA is not in agreement with the processes.	Extensive
8.1	Table A-1 4.1	Development Process Rigor	Assign development tasks based on assessed criticality level for each computing system safety item (program or data). for	SDP	See **Error! Reference source not found.** for the mapping between AC 450.141-1A **criticality** levels and the **DAL** definition in DO-178C. Based on the determined criticality level for the Computing System Safety Item, the required development processes are defined. In DO-178C, this is accomplished in the planning process.	Extensive
	Table A-1 4.1	Software Planning Process	The software development process is planned and documented. Planning addresses the following activities: 1) Activities of software life cycle processes are defined 2) Software life cycle(s) are defined. 3) Software life cycle environment is defined. 4) Additional considerations are addressed 5) Software Development Standards are defined 6) Software plans comply with targeted certification standard 7) Development and revision of software plans are coordinated	PSAC, SDP, SVP, possibly Software Configuration Management Plan (SCMP) and Software Quality Assurance Plan (SQAP)	Annex A of DO-178C provides tables A-1 through A-10 with a matrix of required processes based on DAL. Although plans are not specifically required by AC 450.141-1A, the process of writing the plans can help to ensure that there are no holes in the process. Since AC 450.141-1A only calls for the development process to be document, it may be useful to **write down the planned process**, then **make updates** as the development process is tweaked, thus eliminating the need for capturing updates in the DO-178C required Software Accomplishment Summary (SAS).	Extensive
8.2		Development Process Requirements	Section 8.2 covers creation of the software development process.	SDP		Extensive
8.2.1	11.1.d 11.3.a 11.4.a 11.5.a	Responsibility Assignments	Define development responsibilities for each task	SDP	AC 450.141-1A indicates “An applicant has met this requirement when the applicant can determine **who conducted and approved each step** in the development of a computing system safety item retrospectively.” Although not stated in DO-178C, the same level of traceability to resources is implied. Capturing the **responsible engineers implementing** the process should be achieved through **automated tool history or process**.	Extensive
	11.1.d 11.3.a 11.4.a 11.5.a	Responsibilities Defined in Plans	Define the organizational and certification liaison process responsibilities throughout the software life cycle, including development, verification, software quality assurance, and configuration management within the plans.	PSAC, Software Verification Plan (SVP), SQAP, SCMP	Defining **responsible engineers** for development, verification, configuration management, and quality assurance during planning provides the certification authority the ability to determine if **proper process** was followed and required **independence** was met.	Extensive

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
8.2.2	6.3 6.4.5 7.0.e 7.0.f 7.1.d 7.1.e 7.2.3.b 7.2.4.c 11.4.b.5 11.3.b 11.5.b	Review and Approval Process	Define the process for internal review and approval such that no person approves their own work.	SDP	For AC 450.141-1A, **independent reviews and approvals** are required for the safety requirements, their implementation, and verification and validation evidence at a minimum.	Extensive
	4.6	Review of Software Planning Process	Review of the plans and standards to show they meet the requirements of DO-178C.	PSAC, SDP, SVP, SQAP, SCMP	Since plans and standards are not required under AC 450.141-1A, the **reviews** of these **plans and standards** are not required, but since plans and standards are recommended, it is **strongly encouraged** to review them.	Extensive
	6.3	Software Reviews and Analyses	Reviews and analyses are applied to all of the outputs of the software development processes. This includes the following: • High-Level Requirements • Low-Level Requirements • Software Architecture • Source Code • Integration Process Outputs	SVR	Under DO-178C, reviews are required at all stages of development, from requirements through implementation to the integrated executable code or data items. AC 450.141-1A contains less detail about the specifics but indicates that **requirements and implementation** artifacts should undergo reviews. It is **advisable** that **reviews** be held for **all stages of development** to ensure **correctness of the implementation**.	Extensive
	6.4.5	Review and Analyses of Test Cases, Procedures, and Results	Review and analysis activities confirm that the test objectives are met by the cases, procedures, and results.	SVR	DO-178C specifically calls out reviews of test cases, test procedures, and test results. Whereas, AC 450.141-1A indicates reviews are needed for **verification and validation evidence**. The AC 450.141-1A expectation could be **interpreted as results**, but without properly **understanding the test cases and procedures**, the **results will not provide sufficient evidence**. Therefore, it is recommended that cases, procedures, and results are reviewed.	Extensive

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	7.0.e 7.0.f 7.1.d 7.1.e 7.2.3.b 7.2.4.c 11.4.b.5	Change Control Review and Approval	Defines objectives and plans regarding review and approval of problem reporting, change control and change reviews	SCMP	Although not specifically calling out review and **approval of problem reports** and changes, it is implied in AC 450.141-1A that these would **require reviews** as part of the **implementation artifacts**.	Extensive
	11.3.b 11.5.b	Independence	Defines independence required for software verification and software quality assurance within the plans.	SVP, SQAP	As with DO-178C, AC 450.141-1A requires independence in reviews.	Extensive
8.2.3		Training	Document required training for all development roles.	SDP	DO-178C does not address training. Per AC450.141-1A, **training should include**, but is not limited to the following: 1) Development tools 2) Development methods 3) Installation and Testing 4) Hazard analysis approaches 5) Computing system use 6) Software maintenance	Extensive
8.2.4	5.5 6.5 11.21	Traceability	Define the processes used for tracing requirements to verification and validation evidence.	SDP	AC 450.141-1A defines that traceability is required from **safety requirements to the verification/validation evidence** but does not specify how the traceability is captured, implying that a single trace from requirement to results can be made. DO-178C indicates that tracing is required between **each layer of requirements**, along with tracing from **test cases** to requirements and test cases to **results** through the **test procedures**.	Extensive
	5.5 6.3.1.f 6.3.2.f 6.3.4.e 11.21.a 11.21.b 11.21.c	Software Development Process Traceability	Defines Trace Data as bi-directional traces between system requirements and high-level requirements, high-level requirements and low-level requirements, and low-level requirements and source code.	Trace Data	Although this level of requirement tracing is not required for AC 450.141-1A, **complete tracing** is **encouraged** for computing system safety items that have a criticality level aligned with **DAL B and DAL C**.	Extensive
	6.5 11.14.c 11.21.d 11.21.e 11.21.f	Software Verification Process Traceability	Defines Trace Data as bi-directional traces between software requirements and test cases, test cases and test procedures, and test procedures and test results.	Trace Data	Although only test evidence tracing is required for AC 450.141-1A, use of **complete tracing** is **encouraged** for computing system safety items that have a criticality level aligned with **DAL B and DAL C**. At a **minimum**, traces are needed between the **requirements and results**.	Extensive

AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
8.2.5	Table A-8 7.1	Configuration Management	Define processes for configuration management that specify the content of each released version of a computing system safety item.	SDP	AC 450.141-1A requires a process that **tracks configurations** of all safety-critical systems and documentation related to the operation, ensures the **use of correct and appropriate versions** of systems and documentation tracked, and **documents their configurations** and versions for each licensed activity.	Extensive
	Table A-8 7.1		The software configuration management process objectives are: 1) Configuration items are identified 2) Baselines and traceability are established 3) Problem reporting, change control, change review and configuration status accounting are established 4) Archive, retrieval and release are established 5) Software load control is established 6) Software life cycle environment control is established	SCM Records, Problem Reports, SCI	CM processes defined under DO-17C **exceed CM process requirements** of AC 450.141-1A. Although **baselines** and **traceability** are not required, they may **provide additional control** of configured items. The Control Categories, CC1 and CC2, are beyond the scope of control needed for AC 450.141-1A. Software load control is not addressed in AC 450.141-1A, but it is expected to be addressed in the upcoming AC 450.141-2.	Extensive
8.2.6		Verification and Validation	Define verification and validation of the safety requirements.	SDP		Extensive
8.2.6.1	Table A-6 6.4	Testing	Define processes for testing that verify and validate all safety requirements	SDP	AC 450.141-1A focuses on demonstrating the **verification and validation evidence sufficiently test** the computing system **safety requirements**. Whereas, DO-178C addresses **verification of all development artifacts**, from high-level requirements, design, and compatibility with the target environment.	Extensive
	Table A-6 6.4	Testing of Outputs of Integration Process	Software testing demonstrates that the software satisfies its safety requirements, and that errors that could impact safety have been removed. The objectives of software testing include the following: 1) Executable Object Code complies with the software requirements 2) Executable Object Code is robust with the software requirements 3) Executable Object Code complies with the software design 4) Executable Object Code is robust with the software design 5) Executable Object Code is compatible with the target computer	Software Verification Cases and Procedures (SVCP), SVR, Trace Data	With the emphasis of AC 450.141-1A on testing of the safety requirements, some programs, where the **level of criticality is lower**, the full gamut of DO-178C testing may go beyond what is required. In all cases, **normal and robust** requirements-based testing should occur on the safety requirements. These test cases should cover **integration testing** of both **hardware/software** and **software.**. For programs with **higher criticality** levels, the use of **low-level testing** and **structural coverage analysis** is used to provide **confidence** needed for licensing of the computing system safety item.	Extensive
8.2.6.2	11.3	Test Plan	Develop the software test plan before verification testing begins.	Software Test Plan	AC 450.141-1A defines a test plan which includes the **scope, approach, resources, and schedule**. It also includes a description of the **test environments** (software tools and equipment). Several test types are listed as options, including **unit, interface, system, stress, and integration tests**. The SVP defined in DO-178C includes everything required by AC 450.141-1A, plus more.	Extensive

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	11.3	Software Verification Plan	This plan defines the process to meet the verification objectives. The plan includes the following: 1) Organizational Responsibilities 2) Independence 3) Verification Methods 4) Verification Environment 5) Transition Criteria 6) Partitioning Considerations 7) Compiler Assumptions 8) Reverification Method 9) Previously Developed Software 10) Multiple Version Dissimilar Software		For AC 450.141-1A development, the SVP defined by DO-178C may be slimmed down to addressing the **Organizational Responsibilities, Independence, Verification Methods, and Verification Environment**. However, although it is not required to be captured in the Test Plan, other verification objectives of the SVP, such as **Partitioning Considerations, Compiler Assumptions, Previously Developed Software, and Multiple Version Dissimilar Software**, should be **considered** when writing test cases.	Extensive
8.2.6.3	6.4.2	Test Cases	Define all test cases with pass and fail criteria.	SVCP	All **safety requirements must have test cases** per AC 450.141-1A. These cases should demonstrate the ability of the software to respond to both **nominal** and **off-nominal** inputs and conditions.	Extensive
	6.4.2	Requirements-Based Test Selection	Activities of requirements-based test section include: 1) Specific test cases developed to include nominal range test cases and robustness. (abnormal range) test cases. 2) The test cases are developed from the software requirements and the error sources inherent with the software development processes. 3) Test procedures are generated from test cases.	SVCP	DO-178C requirements for **nominal** and **robust** test cases maps nicely to the requirements of AC 450.141-1A.	Extensive
8.2.6.4	11.14 ISO/IEC/IEEE 29119-3:2013	Test Log	Record the results of the tests and any anomalies discovered during testing in a test log.	SVR	The Test Log defined by AC 450.141-1A includes records of the **test results**, test **anomalies** found, and the **software version(s)** being tested.	Extensive
	11.14	Software Verification Results	The Software Verification Results includes: 1) For each review/analysis/test, indicate each procedure run that passed/failed during verification and the final pass/fail results. 2) Identify the software version reviewed/analyzed/tested. 3) Include results of tests/reviews /analyses, including coverage and traceability analyses.	SVR	Excluding the coverage analysis, the **Software Verification Results** document defined by DO-178C **meets the Test Log requirements** of AC 450.141-1A.	Extensive
8.2.6.5	6.4 Table A-7	Verification Tests	Use a combination of verification approaches appropriate for each software system (analysis, inspection, and test), including testing to the extent practicable.	SVCP, SVR	AC 450.141-1A allows several methods of testing, all of which are acceptable methods within DO-178C, including **structural coverage** testing.	Extensive

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	Table A-7 6.4.4 6.4.5	Verification of Verification Process Results	Software Verification process traceability activities include: 1) Test procedures are correct 2) Test results are correct and discrepancies explained Objectives for test/coverage analysis are: 1) Test coverage of requirements achieved 2) Test coverage of design achieved 3) Test coverage of decision coverage achieved. 4) Test coverage of statement coverage achieved. 5) Test coverage of data coupling and control coupling achieved.	SVR	DO-178C requires analysis of test coverage, which is also implied within AC 450.141-1A. Structural analysis is not specifically required by AC 450.141-1A, but is listed as a possible test method, including statement coverage and decision coverage. It is strongly encouraged to provide **structural coverage analysis** on programs where the level of criticality is equivalent to **DO-178C DAL C or above**. **DAL C** equivalent programs should meet **statement coverage** requirements and **DAL B** should meet the requirements for **decision coverage**.	Extensive
8.2.7		Previously Developed Software and Computing Systems	Verify and validate the safety requirements for reused computing system safety items.	SVR	Previously developed computing system safety items include commercial off-the-shelf (COTS), government off-the-shelf (GOTS), and “reused” software. AC 450.141-1A **requires verification and validation of the safety requirements of all previously developed software**.	Extensive
	12.1	Use of Previously Developed Software	This section discusses issues with the use of previously developed software, including the assessment of modifications, effect of changing an aircraft installation, application environment, or development environment, upgrading a development baseline, and SCM and SQA considerations.	PSAC	DO-178C provides key criteria/issues that should be evaluated when utilizing previously developed software. It is highly **recommended** that these **criteria/issues be assessed** when utilizing previously developed software in an AC 450.141-1A program.	Extensive
8.3		Development Process Considerations	Section 8.3 covers considerations for formulating the development process	SVR		Extensive
8.3.1	6.3	Analysis	Analysis to verify the software requirements are implemented correctly should include the following types: Logic, Data, Interface, Constraint, Programming style, Non-critical code, Timing and sizing.	SVR	**Analysis** is an **acceptable means** of verification under AC 450.141-1A, as it is also under DO178C.	Extensive
	6.3	Reviews and Analyses	Reviews and analyses are applied to the outputs of the software development processes. One distinction between reviews and analyses is that analyses provide repeatable evidence of correctness and reviews provide a qualitative assessment of correctness. A review may consist of an inspection of the output of a process guided by a checklist or similar aid. An analysis may examine in detail the functionality, performance, traceability, and safety implications of the software component, and its relationship to other components within the system or equipment. The following reviews and analyses are included: • High-Level Requirements • Low-Level Requirements • Software Architecture • Source Code • Outputs of the Integration Process	Software Verification Cases and Procedure, SVR	**Analysis** should be used as a **valid method** of verification. This is especially useful when verifying **performance requirements**.	Extensive

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
8.3.2	Table A-1 #5 4.1.e 4.5 11.6 11.7 11.8	Development Standards	Identify development standards for the following artifact types: requirements, design, coding, and safety.	Software Development Standards	Although not required by AC 450.141-1A, standards may be used and are encouraged to provide a **compelling rationale for acceptance** of the development process.	Extensive
	Table A-1 #5 4.1.e	Software Development Standards are defined	Software Development Standards consistent with system safety objectives are defined.	Software Development Standards	AC 450.141-1A indicates the possible use of requirements, design, code, and **safety standards**. Although use of safety standards within the scope of DO-178C, the requirements, design and code standards should be defined to be consistent with **system safety objectives**.	Extensive
	4.5	Software Development Standards	Software development standards (Software Requirements Standard, Software Design Standard, and Software Code Standards) define the rules and constraints for the software development processes. The verification process uses these standards as a basis for evaluating compliance of actual outputs of a process with intended outputs.	Software Requirements Standard, Software Design Standard, Software Code Standard	The standards defined under DO-178C provide a **solid framework** for the development processes needed for AC 450.141-1A.	Extensive
	11.6	Software Requirements Standards	Software Requirements Standards define the methods, rules, and tools to be used to develop the high-level requirements.	Software Requirements Standard	**Requirements standards** may include methods for developing requirements and a description of how the requirements flow down to coding.	Extensive
	11.7	Software Design Standards	Software Design Standards define the methods, rules, and tools to be used to develop the software architecture and low-level requirements.	Software Design Standard	**Design standards** may include restrictions on the use of scheduling and interrupts, specification of usable code libraries, or rules for conditional branches to reduce complexity.	Extensive
	11.8	Software Code Standards	Software Code Standards define the programming languages, methods, rules, and tools used to code the software.	Software Code Standards	**Coding standards** may include specifications for the programming language; naming conventions for modules, variables, and constants; and constraints on the use of tools.	Extensive
8.3.3	Table A-9 8.1 *NASA-STD-8739.8* *NASA-HDBK-2203*	Quality Assurance	Quality assurance verifies that the objectives and requirements of the software system safety program are satisfied and confirms that deficiencies are detected, evaluated, tracked, and resolved.	SQA Records	Although not required by AC 450.141-1A, an acceptable quality assurance function should include **audits** and **inspections** of **elements** and **processes**. In addition, the quality assurance function may **evaluate the validity of system safety data**.	Extensive
	Table A-9 8.1	Software Quality Assurance Process	Assurance is obtained that the following SQA process objectives are met: 1) Software plans/standards developed and reviewed for compliance and consistency 2) Software life cycle processes comply with software plans 3) Software life cycle processes comply with software standards 4) Transition criteria for the software life cycle processes are satisfied 5) Software conformity review is conducted	SQA Records	SQA plays a significant role in DO-178C monitoring for process compliance. Utilizing an **independent SQA**, even to a lesser extent than described in DO-178C, will provide the FAA confidence when developing a software product under the guidance of AC 450.141-1A.	Extensive
8.3.4	6.3 *NASA-STD-8739.9*	Formal Inspections	Formal technical reviews provide a structured way to find and eliminate defects in documentation products, ranging from a requirements document to the actual source code. A process must be defined for internal review and approval such that no person approves the person's own work	SDP or Review Process Document	Per section 6.3 of DO-178C, "A review may consist of an inspection of an output of a process guided by a **checklist** or similar aid." Use of checklists during review facilitates consistency and provides the reviewers guidance to spot defects in products. To show independence during reviews under AC 450.141-1A, the review documentation should **capture the reviewers and roles**.	Extensive
Development Processes						

AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
8.3.5		Anomaly Reports	Develop a standardized process to document anomalies (problem reports), analyze the root cause, and determine corrective actions. These should cover the following: 1) Anomalous behavior of the computing system and its resolution. 2) Development process non-compliance 3) Deficiencies in Documentation and safety data, including requirements.	SDP	AC 450.141-1A suggests **problem/anomaly reporting** for requirements, implementation, documentation, and process.	Extensive
	7.2.3	Problem Reporting, Tracking, and Corrective Action	Problem Reports should be prepared that describe non-compliances with plans, output deficiencies, or software anomalous behavior, and the corrective actions taken. It should provide for configuration identification of affected item(s) or definition of affected process activities, status reporting, and approval and closure. Corrective actions of the software product or outputs of the life cycle processes should invoke change control activities.	Problem Reports	DO-178C processes define a **framework** that may be easily transferred to be used under AC 450.141-1A for **problem reporting, tracking, and corrective actions**.	Extensive
	11.17	Problem Reports	Problem Reports are a means to identify and record the resolution software product anomalous behavior, process non-compliance with software plans and standards, and deficiencies in software life cycle data. Problem Reports should include: 1) Identification of the configuration item and/or the software life cycle process activity in which the problem was observed. 2) Identification of the configuration item(s) to be modified or a description of the process to be changed. 3) A problem description that enables the problem to be understood and resolved. It should contain sufficient detail to facilitate the assessment of the potential safety or functional effects of the problem. 4) A description of the corrective action taken to resolve the reported problem.	Problem Reports	This DO-178C section identifies the **requirements** of what needs to be recorded in a **problem report**. The recorded data for the problem reports is consistent with the needs of AC 450.141-1A.	Extensive
8.3.6 **8.3.7** **8.3.8**	7.1 7.4 *NASA-GB-8719.13*	Software Maintenance	Develop maintainable software to facilitate changes due to maintenance and reduce the likelihood of introducing new hazards. These maintenance sections address three areas, Maintenance and Repair of Computing System Hardware, Maintenance of Computing System Software, and Building Maintainable Software.	SDD, Source Code Files	AC 450.141-1A identifies the need for software to assist in hardware maintenance/repair. Capabilities such as **fault detection and recording** and **statistical data recording** can assist the maintenance technicians with **troubleshooting** and recognition of a **degraded** computing system safety item. Software maintenance can be provided through **checksums** and **load procedures** that ensure the proper configuration of the computing system safety item. Designing for future, strong CM practices, modular designs, naming conventions, coding standards for consistent and readable code, documentation standards, standardized tools, and well-established regression tests provide a **strong framework for maintainable software**. The software **configuration management** and **change control** in DO-178C provides a solid framework for creating **maintainable software development environment**.	Moderate

Development Processes						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	7.1.h 7.4	Software Load Control	Software load control refers to the process by which programmed instructions and data are transferred from a master memory device into the system. Load control includes procedures for part numbering and media identification that identify software configurations approved for loading into the airborne system or equipment. In addition, records should be kept that confirm software compatibility with the airborne system or equipment hardware.		The processes for **software load control** provided under DO-178C should also be used for AC 450.141-1A development.	Moderate

Application Materials

The final section of AC 450.141-1A lists the materials required for an application for a launch or reentry vehicle license to address 14 CFR § 450.141. These materials include documentation of the safety evaluation process, the criticality decisions, the resulting safety requirements, the development processes, and the evidence that the documented development processes were followed.

DO-178C likewise provides a list of documentation that must be made available to the FAA. Once again, there is a great deal of overlap between the two lists. Using DO-178C throughout the software lifecycle will generate the documentation needed as a certification basis for a 14 CFR § 450.141 project.

Table 5 Application Materials

Application Materials						
AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
9		Application Materials	Section 9 provides information on satisfying the application requirements.	Certification Package		Moderate
9.1	*SAE ARP4754A* *SAE ARP4761*	Computing System Safety Items	Identify and describe all computing system safety items involved in its proposed operation, including the severity of hazards associated with each computing system safety item and the level of criticality regarding each hazard.	SCI, Certification Package	This process falls outside of the scope of DO-178C. It is better mapped to processes in SAE ARP4754A and **SAE ARP4761**.	Moderate
9.2		Safety Requirements	Identify the safety requirements for each computing system safety item and submit all computing system requirements in order to effectively convey the software's intended functionality to the FAA.	Certification Package	Definition of the system requirements is outside of the scope of DO-178C and is defined in **SAE ARP4754A**. Development and tracing of all software requirements are defined within DO-178C.	Moderate
9.3		Development Process	Submit documentation of the development process that meets § 450.141(c), including the minimum attributes required by the regulation. Provide evidence of the execution of the appropriate development process for each computing system safety item.	Certification Package, SDP	Under DO-178C projects, the development process is defined within the plans in the early stages of the project, thus allowing the FAA to approve the process prior to the development. AC450.141-1A only specifies that the process be identified but does not indicate an approval is required. Although, not required, **upfront communication** with the **FAA** will help to ensure that the **process is sufficient** before submitting to the FAA for licensing.	Extensive
9.4	Table A-10 #3 9.0.c	Provide Evidence of Implementation of Each Safety Requirement	Provide evidence of the implementation of each safety requirement. Submit the combination of test descriptions, test plans, test outputs, and analyses that verifies that each computing system safety item meets each applicable safety requirement.	SRD, Software Test Plan, SVCP, SVR, Trace Data	DO-178C requires a SAS to detail the compliance with all governing requirements. This SAS, along with the verification procedures, results, and trace data, provides the necessary compliance data required by the FAA. Additionally, the Accomplishment Summary will identify any non-compliances and any non-conformances of the product being certified. The **SAS** utilized by DO-178C can provide a useful **means to summarize the compliance data** required under AC 450.141-1A.	Extensive
Application Materials						

AC 450.141-1A Reference	DO-178C and Other References	Objective	Activity	Activity Outputs	Applicability (Between AC 450.141-1A and DO-178C)	Genuen Experience
	Table A-10 #3 9.0.c	Compliance Substantiation is Provided	Provide compliance substantiation to the certification authority	Certification Package	Although not all data required by DO-178C is required under AC 450.141 (for example – trace data between requirement, design, and code), much of that data generated during the DO-178C development processes is helpful in guaranteeing compliance and **may be useful** to the FAA for **verification of compliance**.	Extensive

AC 450.141-1A Process Summary

For a Computing System license against the requirements of Title 14 Code of Federal Regulations Part 450.141, experience with DO-178C can be very helpful. This is especially true in regard to parts of § 450.141(b) and all of § 450.141(c). Although there are some minor differences in the processes, the overall structure of the development corresponds nicely. The level of rigor based on the Level of Criticality defined for the Computing System Safety Item in AC 450.141-1A translates closely with the Design Assurance Levels defined in DO-178C, with the exception that there may not be an AC 450.141-1A equivalent to the most critical and stringent DAL A. Following DO-178C processes with the right level of rigor would exceed the requirements described in § 450.141(b) and all of § 450.141(c) (FAA, 2020a, RTCA 2012).

Without detailed analysis, it appears that following the processes of SAE ARP4761 and SAE ARP4754 would fulfill the requirements of § 450.141(a) and the hazard assessment of § 450.141(b). Section 450.141(d) compliance is met by providing the documentation for the computing system safety items, requirements, development process, and test evidence (SAE, 2010; 1993; FAA, 2020a).

Genuen, LLC, with its deep knowledge of and experience with DO-178C, stands as a ready partner for 14 CFR Part 450.141 computing system projects.

References

Department of Defense. (2012). MIL-STD-882E Department of Defense standard practice: system safety. http://everyspec.com/MIL-STD/MIL-STD-0800-0899/MIL-STD-882E_41682/.

Department of Defense. (2010). Joint software systems safety engineering handbook. https://www.acqnotes.com/Attachments/Joint-SW-Systems-Safety-Engineering-Handbook.pdf.

Federal Aviation Administration. (2020a). 14 CFR Part 450.141: Streamlined launch and reentry license requirements. https://www.faa.gov/space/streamlined_licensing_process/media/SLR2_Final_Rule_450.pdf.

Federal Aviation Administration. (2021). Advisory Circular 450.141-1: Computing system safety. https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_450.141-1A_Computing_System_Safety_20210816_v1_(002).pdf.

International Organization for Standardization. (2013). ISO/IEC/IEEE 29119-2-2013: Software and systems engineering — Software testing — Part 3: Test documentation. https://www.iso.org/standard/56737.html.

National Aeronautics and Space Administration. (2004). NASA-GB-8719: NASA software safety guidebook. https://standards.nasa.gov/standard/osma/nasa-gb-871913.

National Aeronautics and Space Administration. (2020a). NASA-STD-8739: Software assurance and software safety standard. https://standards.nasa.gov/standard/osma/nasa-std-87398.

National Aeronautics and Space Administration. (2020b). NASA-HDBK-2203): Software Engineering and Assurance Handbook. https://standards.nasa.gov/standard/oce/nasa-hdbk-2203.

Radio Technical Commission for Aeronautics. (2012). RTCA DO-178C: Software considerations in airborne systems and equipment certification. https://global.ihs.com/doc_detail.cfm?document_name=RTCA%20DO%2D178&item_s_key=00088334.

Range Safety Group. (2019). Flight termination systems: Commonality standard. Range Commanders Council. https://www.wsmr.army.mil/RCCsite/Documents/319-19_FTS_Commonality/319-19_FTS_Commonality.pdf

SAE International. (1996). ARP4761: Guidelines and methods for conduction the safety assessment process on civil airborne systems and equipment. https://www.sae.org/standards/content/arp4761/.

SAE International. (2010). ARP4754A: Guidelines for development of civil aircraft and systems. https://www.sae.org/standards/content/arp4754a/.

Software & Systems Engineering Standards Committee. (2016). IEEE 1012-2016 – IEEE standard for system, software, and hardware verification and validation. IEEE. https://standards.ieee.org/standard/1012-2016.html.

Software & Systems Engineering Standards Committee. (1994). IEEE 11228-1994 – IEEE standard for software safety paln. IEEE. https://standards.ieee.org/standard/1228-1994.html.