Comparison Table Between NEI 20-07 and NRC RG's and Endorsed IEEE Standards

White SDO means comparable RG/endorsed IEEE std/NRC review guidance exists.

Yellow SDO means some aspect of the SDO is not met by comparable RG/endorsed IEEE std/NRC review guidance exists.

Red SDO means new criteria developed for NEI 20-07 because comparable RG/endorsed IEEE std/NRC review guidance does not exist.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.1.3.1 Application software requirements are derived from, and backward traceable to, the functional and performance requirements of the affected plant systems and their design and licensing bases.	BTP 7-14 focuses on software tracing but not tracing back to the affected plant system and their design and licensing bases. This requires a level of tracing above software requirements. It entails the System Requirements being traced upward to affected plant system and their design and licensing bases. IEEE Std 1012 Table 1 Requirements V&V Traceability Analysis Trace the software requirements (SRS and IRS) to system requirements (concept documentation) and system requirements to the software requirements. Verify that all system requirements related to software are traceable to software requirements

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.1.3.2 A hazard analysis method is used to identify hazardous control actions that can lead to an accident or loss, and application software requirements and constraints are derived from the identified hazardous control actions.	IEEE Std 7-4.3.2-2016 Clause 5.5.1 A hazard analysis (see Annex D for guidance) shall be performed to identify and address potential hazards of the system. Note: The NRC will soon update RG 1.152 to endorse IEEE Std 7-4.3.2-2016. IEEE Std 7-4.3.2-2016 Annex D Clause D.3.2.1.3 The following additional items should be considered as potential hazards for a PDD: Sequences of actions that can cause the system to enter a hazard state. In addition to the generic list in D.3.2, PDD-specific hazards might include: 1) Early or late outputs 2) No output on demand 3) Inadvertent output without demand 4) Output duration too short 5) Output duration too long 6) Output intermittent 7) Output oscillates or fluctuates rapidly 8) Incorrect response to sensor data (e.g., value too high/low, stuck at previous value, etc.) Could not find explicit requirement for "application software requirements and constraints are derived from the identified hazardous control actions." IEEE 7-4.3.2 Annex D, Clause D.3.4.1, Hazard elimination, provides a guideline that states, "Potential problems include ambiguous statements, unspecified conditions, precision requirements not defined, response to hazards not defined, and requirements that are: incomplete, incorrect, conflicting, difficult to implement, illogical, unreasonable, not verifiable, or not achievable." This guideline does not specifically relate back to the derivation of identified hazards into software requirements.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.1.3.3 The application software requirements resulting from activities performed under SDOs 0 and 0 are sufficiently detailed to support an assessment of functional safety.	RG 1.172 The licensee or applicant should specify the software requirements for fault tolerance and failure modes, derived either from a consideration of system-level hazards analyses or from software internals, for each operating mode. The licensee or applicant should fully specify software behavior in the presence of unexpected, incorrect, anomalous, and improper (1) input, (2) hardware behavior, or (3) software behavior, and should provide software requirements necessary to respond to both hardware and software failures, including the requirements for analysis of, and recovery from, computer system failures.
10.1.3.4 Hardware constraints on the application software are specified and complete.	IEEE Std 830 Clause 4.1 requires the SRS to identify the documented hardware constraints. It does not ensure hardware constraints on the application software are specified and complete.
10.1.3.5 Application software functional and performance requirements are decomposed from I&C system requirements, the I&C system architecture, and any constraints imposed by the I&C system design.	IEEE Std 830 Clause 4.1 The basic issues that the SRS writer(s) shall address are the following: a) Functionality. What is the software supposed to do? b) … c) Performance. What is the speed, availability, response time, recovery time of various software functions, etc.? d) … e) Design constraints imposed on an implementation. Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s) etc.? BTP 7-14, B.3.3.2.1 states, “A review of the software architecture requires a concurrent review of the hardware architecture.”
10.1.3.6 If application software requirements are expressed or implemented via configuration parameters, the specified parameters and their values are consistent and compatible with the I&C platform and the I&C system requirements.	No comparable criteria could be found.

10.1.3.7 If data communications are required between application software elements and/or between application software elements and external systems, data requirements are specified, including best- and worst-case performance requirements.	IEEE Std. 830 An SRS is complete if, and only if, it includes the following elements: b. Definition of the responses of the software to all realizable classes of input data in all realizable classes of situations. Note that it is important to specify the responses to both valid and invalid input values. BTP 7-14 Section B.3.3.3.1 states, "The design should specify the method for determining that the values of input variables are within the proper range, the method by which the software will detect that the values of input variables are not within their proper range, and the actions to be taken in the latter case." BTP 7-14 refers to SRP Chapter 7.9, "Data Communication Systems". There it states, "The review should verify that the protocol selected for the DCS meets the performance requirements of all supported systems. The real-time performance should be reviewed with SRP BTP 7-21, "Guidance on Digital Computer Real-Time Performance." This should include verification that DCS safety system timing is deterministic or bounded. Time delays within the DCS and measurement inaccuracies introduced by the DCS should be considered when reviewing the instrumentation setpoints …. Data rates, data bandwidths, and data precision requirements for normal and off-normal operation, including the impact of environmental extremes, should be reviewed. There should be sufficient excess capacity margins to accommodate likely future increases in DCS demands or software or hardware changes to equipment attached to the DCS. The error performance should be specified. Vendor test data and in situ test results should be reviewed to verify the performance. Analytical justifications of DCS capacity should be reviewed for correctness. The interfaces with other DCSs or other parts of the I&C system should be reviewed to verify compatibility. Note: No RG or IEEE standard could be found that requires, "best- and worst-case performance requirements".

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.2.3.1 When the application software can include or affect a number and/or variety of system elements, and responsibilities for application software design of such elements are split among two or more entities, then a clear division of responsibility (DOR) is developed and agreed upon by all entities, and the DOR is maintained throughout the course of application software development activities.	No comparable criteria could be found.
10.2.3.2 Abstraction and modularity are used to control complexity in the application software design.	BTP 7-14 Section B.3.3.2.1 The hardware and software architecture should be reviewed to verify that the propagation of errors is controlled via a well-structured modular design. No comparable criteria on abstraction can be found.
10.2.3.3 The application software design method aids the expression of functions; information flow; time and sequencing information; timing constraints; data structures and properties; design assumptions and dependencies; exception handling; comments; ability to represent structural and behavioral views; comprehension by entities who need to understand the design; and verification and validation.	No comparable criteria could be found.
10.2.3.4 Testability and modifiability in the operations and maintenance phase of the system lifecycle is considered during application software design.	No comparable criteria could be found.
10.2.3.5 The application software design method has features that support software modification, such as modularity, information hiding, and encapsulation.	No comparable criteria could be found.
10.2.3.6 Application software design notations are clearly and unambiguously defined.	BTP 7-14 Section B.3.3.4.3 In addition to the above, the CL should have sufficient comments and annotations that the intent of the code developer is clear. This is not only so the reviewer can understand and follow the code, but also so future modifications of the code are facilitated.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.2.3.7 The application software design elements are simple to the extent practicable.	No comparable criteria could be found.
10.2.3.8 If a full variability language is used for implementing the application software design, the design includes self-monitoring of control flow and data flow, and on failure detection, appropriate actions are taken.	No comparable criteria could be found.
10.2.3.9 Application software elements of varying safety classifications shall all be treated as the highest safety classification unless adequate independence between elements of different safety classifications is justified.	IEEE Std 7-4.3.2 Clause 5.6 IEEE Std 603-2009 requires that safety functions be separated from non-safety functions such that the non-safety functions cannot prevent the safety system from performing its intended functions. In PDD systems, software performing safety functions and software performing non-safety functions may reside on the same PDD and use the same PDD resources. The non-safety software functions shall be developed in accordance with the safety-related requirements of this standard.
10.2.3.10 When a pre-existing application software element is used to implement a system function, it meets the SDOs in Section **Error! Reference source not found.**	No comparable criteria could be found.
10.2.3.11 When the digital equipment consists of pre-existing functionality that is configured via data to meet application-specific requirements, the applied configuration design is consistent with the design of the equipment. Methods are used to prevent errors during design and implementation of the configuration data using specified configuration data structures.	No comparable criteria could be found.
10.3.3.1 The application software architecture design uses an integrated set of techniques necessary to meet the functional and performance requirements developed via the SDOs in Section **Error! Reference source not found.**.	No comparable criteria could be found.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.3.3.2 Application software architecture design is partitioned into elements or subsystems, and information about each element or subsystem provides verification status and associated conditions.	No comparable criteria could be found.
10.3.3.3 Application software architecture design determines hardware/software interactions unless already specified by the system architecture.	IEEE Std 830 Clause 4.1 The basic issues that the SRS writer(s) shall address are the following: a)... b) External interfaces. How does the software interact with people, the system's hardware, other hardware, and other software?
10.3.3.4 Application software architecture design uses a notation that is unambiguously defined or constrained to unambiguously defined features.	No comparable criteria could be found.
10.3.3.5 Application software architecture design determines the features needed for maintaining the integrity of safety significant data, including data at rest and data in transit.	No comparable criteria could be found.
10.4.3.1 Application software is supported by on-line and off-line support tools. Off-line support tools are classified in terms of their direct or indirect potential impacts to the application software executable code.	IEEE Std 7-4.3.2 Clause 5.3.2 When software tools are selected for use in a safety-related PDD development project, the benefits and risks associated with the tools use should be taken into consideration. The selected tools should limit the opportunity for making errors and introducing faults, while maximizing the opportunity for detecting faults in the resulting safety-related PDD.
10.4.3.2 An application software on-line support tool is an element of the system under design.	IEEE Std 7-4.3.2 Clause 5.3.2 If software tools are used during the life cycle process of safety-related PDD, one or both of the following methods shall be used to confirm outputs of that software tool are suitable for use in safety-related systems: a) The software tool shall be used in a manner such that defects not detected by the software tool will be detected by V&V activities. b) The tool shall be developed using a quality lifecycle process commensurate with the criticality of the safety functions performed by the end product.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.4.3.3 Application software off-line support tools are an element of development activities and are used to reduce the likelihood of errors, and to reduce the likelihood of not detecting errors. When off-line tools can be integrated, the outputs from one tool are suitable for automatic input to a subsequent tool to minimize the likelihood of human error.	No comparable criteria could be found.
10.4.3.4 Offline tools have specified behaviors, instructions, and any specified constraints when 1) they can directly or indirectly contribute to the executable code, or 2) they are used to support the test or verification of the design or executable code where errors in the tool can fail to reveal defects.	See 10.4.3.2
10.4.3.5 Offline tools are assessed for the reliance placed on them and their potential failure mechanisms that may affect the executable application software when 1) they directly or indirectly contribute to the executable code, or 2) they are used to support the test or verification of the design or executable code where errors in the tool can fail to reveal defects.	See 10.4.3.1 and 10.4.3.2
10.4.3.6 Offline tool conformance to its documentation may be based on a combination of history of successful use (in similar environments and for similar applications) and its validation.	In addition to the response in 10.4.3.2, the following is stated in IEEE Std 7-4.3.2 Clause 5.3: The assessment process for software tools should take into account experience from prior use. The experience should include the experience of the developers as well as experience gained from the processes in which the tools are used.
10.4.3.7 Tools are validated with a record of their versions, validation activities, test cases, results, and any anomalies.	No comparable criteria could be found.
10.4.3.8 When a set of tools can function by using the output from one tool as input to another tool then the set is regarded as integrated and they are verified to ensure compatibility.	No comparable criteria could be found.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.4.3.9 The application software design representation or programming language uses a translator that is assessed for suitability at the point when development support tools are selected, uses defined language features, supports detection of mistakes, and supports the design method.	No comparable criteria could be found.
10.4.3.10 If SDO 0 is not fully demonstrated, then the fitness of the language and any measures to address identified shortcomings is justified.	No comparable criteria could be found.
10.4.3.11 Programming languages for developing application software are used per a suitable set of rules which specify good practice, prohibit unsafe features, promote understandability, facilitate verification and validation, and specify code documentation requirements.	No comparable criteria could be found.
10.4.3.12 When offline tools are used, the application software configuration baseline information includes tool identification and version, traceability to the application software configuration items produced or affected by the tool, and the manner in which the tool was used, when 1) the tool can directly or indirectly contribute to the executable code, or 2) the tool is used to support the test or verification of the design or executable code where errors in the tool can fail to reveal defects.	IEEE Std 828 Configuration identification activities shall identify, name, and describe the documented physical and functional characteristics of the code, specifications, design, and data elements to be controlled for the project. The documents are acquired for configuration control. Controlled items may be intermediate and final outputs. These items include outputs of the development process (see IEEE Std 730 [B3]) (such as requirements, design, executable code, source code, user documentation, program listings, databases, test cases, test plans, specifications, and management plans) and elements of the support environment (such as compilers, operating systems, programming tools, maintenance and support items, and test beds). … For each software tool, whether developed within the project or brought in from outside the project, the Plan shall describe or reference its functions and shall identify the configuration controls to be placed on the tool. Note: Not covered is “traceability to the application software configuration items produced or affected by the tool, and the manner in which the tool was used…”

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.4.3.13 Offline tools are under configuration management to ensure compatibility with each other and the system under design, and only qualified versions are used, when 1) the tool can directly or indirectly contribute to the executable code, or 2) the tool is used to support the test or verification of the design or executable code where errors in the tool can fail to reveal defects.	See 10.4.3.12
10.4.3.14 Qualification of each new version of an offline tool may be demonstrated by qualification of an earlier version if the functional differences will not affect compatibility with other tools, and evidence shows that the new version is unlikely to contain significant faults.	No comparable criteria could be found.
10.5.3.1 Information items that describe application software requirements, architecture design, and validation planning are completed prior to application software detailed design and implementation activities and are used to inform the detailed design and its implementation.	No comparable criteria could be found.
10.5.3.2 The application software is modular, testable, and modifiable.	No comparable criteria could be found.
10.5.3.3 For each major element or subsystem identified in the application software architecture design produced via the SDOs provided in Section **Error! Reference source not found.**, further refinement into application software modules is based on partitioning, and modules are designed in sets suitable for integration and integration testing at the software and system levels.	No comparable criteria could be found.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.5.3.4 Application software integration tests and software/hardware integration tests ensure conformance to the requirements produced under the SDOs in Section **Error! Reference source not found.**.	RG 1.171 [GDC] Criterion XI also requires that test results be documented and evaluated to ensure that test requirements have been satisfied. IEEE Std 1012 Table 1 5.4.3 (6) Integration V&V test plan generation 1. Plan integration testing to validate that the software correctly implements the software requirements and design as each software component (e.g., units or modules) is incrementally integrated with each other. 2. Plan tracing of requirements to test design, cases, procedures, and results.
10.6.3.1 Each application software module is reviewed against the goals listed above. • Completeness of module testing with respect to the application software design • Correctness of module testing with respect to the application software design specification • Module testing is repeatable • The module testing configuration is precisely defined	IEEE Std 1012 Table 1 Component V&V test plan generation No comparable criteria could be found for configuration precisely defined.
10.6.3.2 When an application software module is produced by an automatic tool, the SDOs provided in Section **Error! Reference source not found.** are demonstrated.	No comparable criteria could be found.
10.6.3.3 When an application software module consists of reused pre-existing software, SDO **Error! Reference source not found.** is demonstrated.	No comparable criteria could be found.
10.7.3.1 Each application software module is verified (as specified via SDO 0) to perform its intended function and does not perform unintended functions.	IEEE 1012 Clause 5.4.3 The objective of Design V&V is to demonstrate that the design is a correct, accurate, and complete transformation of the software requirements and that no unintended features are introduced.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.7.3.2 Application software module testing results are documented.	IEEE Std 829 Clause 11. Test Summary Report
10.7.3.3 If an application software module test is not successful, corrective actions are specified.	BTP 7-19 B.3.1.12.2 The STP should specify procedures for tracking problem reports, and for ensuring that each problem reported has been correctly resolved.
10.8.3.1 Using the results of activities performed under SDO 0, application software integration testing is performed using specified test cases, and test data; in a specified and suitable environment; with specified acceptance criteria.	IEEE Std 829 Clause 6 and 11.2.6
10.8.3.2 Application software integration tests demonstrate correct interaction between all application software modules and/or application software elements/subsystems.	IEEE Std 1012 Clause 3.1.14 integration testing: Testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them.
10.8.3.3 Application software integration testing information includes whether test acceptance criteria have been met, and if not, the reasons why such that corrective actions are specified.	BTP 7-14 B.3.2.2 Final integration tests should be completed and documented. Reports should be written for each test run. These reports should include any anomalies found and actions recommended.
10.8.3.4 During application software integration, any module changes are analyzed for extent of 1) impact to other modules and 2) rework of activities performed under prior SDOs.	IEEE Std 1012 *Regression analysis and testing.* Determine the extent of V&V analyses and tests that must be repeated when changes are made to any previously examined software products. Assess the nature of the change to determine potential ripple or side effects and impacts on other aspects of the system. Rerun test cases based on changes, error corrections, and impact assessment, to detect errors spawned by software modifications. RG 1.168 7b. Criterion III requires that design changes be subject to design control measures commensurate with those applied to the original design. Regression analysis and testing following the implementation of software modifications is an element of the V&V of software changes and should be part of the minimum set of software V&V activities for safety system software.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.9.3.1 Application software is integrated with the system hardware in accordance with SDO 0.	See 10.9.3.2
10.9.3.2 Using the results of activities performed under SDO 0, system integration testing is performed using specified test types, test cases, and test data; in a specified facility with a suitable environment; using specified software and hardware integration instructions; and with specified acceptance criteria.	IEEE Std 1012 Table 1 Integration V&V test plan generation Integration V&V test design generation Integration V&V test case generation Integration V&V test procedure generation IEEE Std 829 Clause 6.2.5.3 IEEE Std 829 Clause 4.2.11
10.9.3.3 System integration testing information includes whether test acceptance criteria have been met, and if not, the reasons why such that corrective actions are specified. During application software integration, any module changes are analyzed for extent of 1) impact to other modules and 2) rework of activities performed under prior SDOs.	IEEE 1012 Table 1 System V&V test execution Use the V&V system test results to validate that the software satisfies the V&V test acceptance criteria. BTP 7-14 B.3.2.2 Final integration tests should be completed and documented. Reports should be written for each test run. These reports should include any anomalies found and actions recommended. IEEE Std 1012 *Regression analysis and testing.* Determine the extent of V&V analyses and tests that must be repeated when changes are made to any previously examined software products. Assess the nature of the change to determine potential ripple or side effects and impacts on other aspects of the system. Rerun test cases based on changes, error corrections, and impact assessment, to detect errors spawned by software modifications. RG 1.168 7b. Criterion III requires that design changes be subject to design control measures commensurate with those applied to the original design. Regression analysis and testing following the implementation of software modifications is an element of the V&V of software changes and should be part of the minimum set of software V&V activities for safety system software.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.10.3.1 System validation procedural and technical steps are specified in order to demonstrate the application software meets the requirements produced via activities performed under the SDOs in Section **Error! Reference source not found.**.	IEEE Std 1012 Table 1 System V&V test execution Analyze test results to validate that the software satisfies the system requirements.
10.10.3.2 System validation information includes a chronological record of activities; the validated functions; tools and equipment used; results; and any anomalies - including the reasons why so that corrective actions are specified.	IEEE Std 829 Clause 3.12 test log: A chronological record of relevant details about the execution of tests. IEEE Std 1012 Table 1 System V&V test execution Analyze test results to validate that the software satisfies the system requirements. IEEE Std 829 Clause 4.2.6 Specify the major activities, techniques, and tools that are used to test the designated groups of features. IEEE Std 1012 Table 1 System V&V test execution Document the results as required by the System V&V test plan. Document discrepancies between actual and expected test results. BTP 7-14 B.3.2.2 Final integration tests should be completed and documented. Reports should be written for each test run. These reports should include any anomalies found and actions recommended.
10.10.3.3 For application software, system testing is the primary method of validation, and the system is tested by exercising inputs; exercising expected conditions (both normal and abnormal); and exercising hazards that require system action (as identified via activities performed under SDO 0). Analysis, modeling, and simulation may supplement system testing.	IEEE Std 1012 Clause 3.1.31 test case: (A) A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. (B) Documentation specifying inputs, predicted results, and a set of execution conditions for a test item. IEEE Std 1012 Table 1 System V&V test plan generation Performance at boundaries (e.g., data, interfaces) and under stress conditions

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.10.3.4 Tools used for system validation meet the SDOs provided in Section **Error! Reference source not found.**.	See responses to 10.4.3.1 – 10.4.3.14 where applicable for test tools.
10.10.3.5 System validation results demonstrate 1) all application software functions required via activities performed under the SDOS in Section 10.1.3 are met correctly, 2) the application software does not perform unintended functions, 3) test case results information for later analysis or assessment, and 4) successful validation, or if not, the reasons why.	IEEE 1012 Clause 5.4.3 The objective of Design V&V is to demonstrate that the design is a correct, accurate, and complete transformation of the software requirements and that no unintended features are introduced. IEEE Std 1012 Table 1 System V&V test execution Analyze test results to validate that the software satisfies the system requirements. BTP 7-14 B.3.2.2 Final integration tests should be completed and documented. Reports should be written for each test run. These reports should include any anomalies found and actions recommended.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.11.3.1 Application software verification activities are specified: selection of strategies and techniques; selection and utilization of tools; evaluation of results; and corrective action controls.	IEEE Std 1012 Clause 1.2 The purpose of this standard is to — Establish a common framework for V&V processes, activities, and tasks in support of all software life cycle processes, including acquisition, supply, development, operation, and maintenance processes — Define the V&V tasks, required inputs, and required outputs IEEE Std 1012 Clause 7.4 The SVVP shall describe the organization, schedule, software integrity level scheme, resources, responsibilities, tools, techniques, and methods necessary to perform the software V&V. IEEE Std 1012 Clause 7.4.6 The SVVP shall describe documents, hardware and software V&V tools, techniques, methods, and operating and test environment to be used in the V&V process. Acquisition, training, support, and qualification information for each tool, technology, and method shall be included. IEEE Std 1012 Clause 5 The results of V&V activities and tasks shall be documented in task reports, activity summary reports, anomaly reports, V&V test documents, and the V&V final report. IEEE Std 1012 Table 1 Proposed/baseline change assessment Evaluate proposed software changes (i.e., modifications, enhancements, and additions as a result of anomaly corrections or requirement changes) for effects on the systems and previously completed V&V tasks.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.11.3.2 Evidence of application software verification activities is recorded, including verified application software configuration items; information used during verification; and the adequacy of results from activities conducted under prior SDOs, including compatibilities between prior activities.	IEEE Std 1012 Clause 6.1 V&V task reports. The V&V effort shall document V&V task results and status. No comparable criteria could be found for verifying application software configuration items. IEEE Std 1012 Clause 7.5.1 1) V&V tasks The SVVP shall identify the V&V tasks to be performed. Table 1 describes the minimum V&V tasks, task criteria, and required inputs and outputs. IEEE Std 1012 Clause 5 The results of V&V activities and tasks shall be documented in task reports, activity summary reports, anomaly reports, IEEE Std 1012 Clause 5.1 The management process comprises the following generic activities and tasks: — Assessing evaluation results IEEE Std 1012 Clause 5.1.1 Whenever necessary, the V&V management determines whether a V&V task should be reperformed as a result of changes in the software program. IEEE Std 1012 Clause 7.4.2 V&V tasks should be scheduled to be reperformed according to the task iteration policy.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.11.3.3 Application software functional and performance requirements produced via activities under the SDOs in Section **Error! Reference source not found.** are verified against the I&C system requirements that are identified via SDO **Error! Reference source not found.**.	IEEE Std 1012 Clause 5.4.2 The Requirements V&V activity addresses software requirements analysis of the functional and performance requirements… IEEE Std 1012 Table 1 Concept documentation evaluation Analyze system requirements and validate that the following satisfy user needs: Feasibility and testability of the functional requirements Feasibility and testability of the functional requirements Evaluate the requirements (e.g., functional, capability, interface,… The task criteria Are Correctness Verify and validate that the software requirements satisfy the system requirements allocated to software within the assumptions, constraints, and operating environment for the system. Validate that the flow of data and control satisfy functionality and performance requirements.

10.11.3.4 The results of activities performed under the SDOs in Sections **Error! Reference source not found.** through **Error! Reference source not found.** are verified to ensure conformance to the requirements produced via activities performed under the SDOs in Section **Error! Reference source not found.**, as well as completeness, consistency, and compatibility between the results of the activities performed under the SDOs within each Section, and the feasibility, readability, and modifiability of the results produced under the activities of SDOs in each section.	SDO 10.2 Software Quality No comparable criteria could be found. SDO 10.3 Software Architecture Design Quality IEEE Std 1012 Clause 5.4.3 The Design V&V activity addresses software architectural design and software detailed design. SDO 10.4 Application Software Support Tool and Programming Language Quality IEEE Std 1012 Clause 7.4.6 The SVVP shall describe documents, hardware and software V&V tools, techniques, methods, and operating and test environment to be used in the V&V process. Acquisition, training, support, and qualification information for each tool, technology, and method shall be included. IEEE Std 1012 Table 1 Scoping the V&V effort Determine the extent of V&V for tools that insert or translate code (e.g., optimizing compilers, auto-code generators). SDO 10.5 Application Software Detailed Design and Development Quality IEEE Std 1012 Clause 5.4.3 The Design V&V activity addresses software architectural design and software detailed design. IEEE Std 1012 Table 1 Design V&V Process Validate the relationship between each design element and the software requirement(s). Verify that the relationships between the design elements and the software requirements are specified to a consistent level of detail. Verify that all design elements are traceable from the software requirements. Verify that all software requirements are traceable to the design elements.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
	Validate that the flow of data and control satisfy functionality and performance requirements. Validate data usage and format. Verify that all terms and design concepts are documented consistently. Verify that there is internal consistency between the design elements and external consistency with architectural design. Validate that the logic, computational, and interface precision (e.g., truncation and rounding) satisfy the requirements in the system environment. Verify that there are objective acceptance criteria for validating each software design element and the system design. Verify that each software design element is testable to objective acceptance criteria. SDO 10.6 Application Software Implementation Quality IEEE Std 1012 Table 1 Implementation V&V Traceability Analysis Source code and source code documentation evaluation. Interface analysis

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.12.3.1 For each potentially hazardous control action identified via activities performed under SDO 0, causal factor scenarios related to the application software are identified and mitigated.	BTP 7-14 Section B.3.1.9.2 A method should exist to identify hazards caused by software, and to identify hazards whose resolution will be under the control of software. BTP 7-14 Section B.3.1.9.3 It should describe a method for preventing the inadvertent introduction of hazards by the use of project tools. IEEE Std 7-4.3.2 Annex D Clause D.3 The hazards list should include hazards presented to the system from all possible sources including hardware failures, software errors… Clause D.3.2 Examples of techniques that can be used in the identification of hazards include: A multidiscipline team approach should be used for the identification of the hazards at each lifecycle phase... Analyses should then be performed to identify ways that these functions can be degraded; for example: 1) Function not executed, such as data sent on a communication bus is not delivered 2) Function executed when not needed 3) Incorrect state transition 4) Incorrect value provided…
10.12.3.2 Analysis demonstrates that untested combinations of external and internal I&C system states have no impact on achieving the application software functional and performance requirements resulting from the SDOs provided in Section **Error! Reference source not found.**.	IEEE Std 1012 Table 1 Software requirements evaluation, Software design evaluation, and Source code and source code documentation evaluation require the activity to: Validate the sequences of states and state changes using logic and data flows coupled with domain expertise, prototyping results, engineering principles, or other basis.

NEI 20-07 SDO	NRC RG or Endorsed IEEE Standard
10.12.3.3 When equipment under the control of the I&C system is normally in the state needed to perform a safety function, the I&C system design has no inputs that will change state when the EUC is in its normal state, and non-normal states in the EUC are readily detectable via independent means. Administrative controls limit the duration of non-normal EUC states and limit the EUC in a non-normal state to one channel or division.	No comparable criteria could be found.