# Evaluation of Guidance for Tools Used to Develop Safety-Related Digital Instrumentation and Control Software for Nuclear Power Plants

## Task 3 Report: Technical Basis for Regulatory Guidance

Prepared by:
James Servatius

Reviewed by:
William Arcieri
Stephen Alexander
Terry Gitnick

Energy and Space Division
Information Systems Laboratories, Inc.
11140 Rockville Pike, Suite 650
Rockville, MD 20852

January 2015

Prepared for:

U.S. Nuclear Regulatory Commission
Office of Nuclear Regulatory Research
NRC Contract No: NRC-HQ-13-C-04-0004
ISL Project No. 2310

## Executive Summary

Application of software tools in new reactor design, development, and existing reactor upgrade and operations has been increasing in recent years. In general, the use of software tools is more efficient than the traditional development process and may result in fewer faults than a manual engineering process if the tools are well designed, carefully developed and selected, rigorously tested, and appropriately used. However, undetected faults in the automated tools or tool-assisted engineering activities may pose serious risks to nuclear safety. Having a good process and consistent method to assess the safety of software tool use in nuclear safety-significant systems is important to all stakeholders in the nuclear industry from the general public to equipment vendors, utility licensees, and government regulatory organizations. However, there is currently no specific and detailed standard and commonly accepted practice in the U.S. nuclear industry that addresses software tool development, selection, evaluation, or qualification.

For the purposes of this report, the term “safety-significant systems” should be understood to mean systems with safety significance designated important to safety (i.e., defined in the regulations as comprising safety-related systems and certain nonsafety-related systems that could impact safety) through deterministic evaluation and all systems with safety significance or risk importance as determined through probabilistic risk analysis. The focus of this report is on software tools used to develop, operate, and maintain safety-related or safety systems. However, the report addresses software tools used to develop, operate, and maintain other safety-significant systems as necessary when determining the level of rigor and intensity necessary to ensure that software tool use is suitable, reliable, and of sufficiently high quality for the project. Software tools used for developing, operating, and maintaining nonsafety-related or nonsafety systems that have no safety significance are not subject to regulatory oversight and are not discussed in this report except in the context of classifying software or software tools as nonsafety-related rather than safety-related based on meeting certain criteria.

The U.S. Nuclear Regulatory Commission (NRC) and Information Systems Laboratories, Inc. (ISL), the contractor, launched a project focused on the safety assessment of software tool use in nuclear safety systems. The project surveyed more than 150 industrial standards, government positions, and regulatory guidance, which include nuclear, aerospace, civil aviation, railway, and automotive industries, different countries such as U.S., Canada, and Europe, different standard organizations, such as Institute of Electrical and Electronics Engineers (IEEE), International Electrotechnical Commission (IEC), International Organization for Standardization (ISO), International Atomic Energy Agency (IAEA), Electric Power Research Institute (EPRI), and Radio Technical Commission for Aeronautics (RTCA), and involving many government organizations, such as the National Aeronautics and Space Administration (NASA), Federal Aviation Administration (FAA), NRC, National Institute of Standards and Technology (NIST), and Atomic Energy of Canada, Ltd. (AECL). The purpose was to learn from a broad spectrum of expertise and extract the most important and relevant ideas to establish the technical basis for potential regulatory guidance for the safety assessment of software tool use in the development, operation, and maintenance of U.S. nuclear systems with safety significance.

The project was phased in three tasks. Task 1 included a survey of the field of available literature from various industries that make use of safety-significant computer-based systems, software, and logic. The survey found a plethora of information that was applicable to software and logic, directly or indirectly, some that was relevant to software tools, directly or indirectly, and some that, while not written for software tools, could be adaptable and useful in the safety assessment of software tools. This information included regulations or regulatory requirements, regulatory guidance, regulatory information, industry guidance, and industry information. The Task 1 report [1] presented the findings of the research and discussed the various types of software tools that are used in the various life cycle processes of safety-significant software design, construction, implementation, verification and validation (V&V), testing, installation and integration, operation, and maintenance. From this information, Task 1 resulted in a set of broad baselines that have relevance to several aspects of software tool evaluation by users and regulatory oversight of the evaluation processes.

Task 2 of this project included a detailed analysis of the software-related information from Task 1, compared and extracted the requirements and guidelines from those documents, and modified the broad baselines of possible requirements and guidelines for software tool use. Task 2 also included detailed analysis of software and software tool categorization methodologies that determine the appropriate degree of rigor to be applied in the process of verifying the tools' suitability, quality, and reliability through a process of tool qualification.

The analyses of Task 2 also resulted in information that may be used in developing regulatory guidance for software tools as well as internal guidance for the review of applications for approval of software tools to be used in the development and testing of software or logic associated with safety-significant systems in NRC-licensed facilities. The Task 2 report [2] presented the findings of the analysis and discussed the hazards of using the various types of software tools in the various life cycle processes of safety-significant software design, construction, implementation, V&V, testing, installation and integration, operation, and maintenance. From this information, Task 2 refined the set of broad baselines that have relevance to several aspects of software tool evaluation by users and regulatory oversight of the evaluation processes.

This phase (i.e., Task 3), includes the technical basis for software tool regulatory guidance for review and acceptance of software tools based on the Task 1 research and Task 2 analysis. The technical basis is aimed at supporting industry standards, meeting regulatory requirements, improving existing guidance, and having clear relation with baseline requirements covered by industry standards.

This report comprises three major sections. Section 1 of this Task 3 report is an introduction, background discussion, and overview of Task 3. Section 1 discusses the overall research project strategy including the survey scope of Task 1, and the scope of the detailed analysis of documents in Task 2. Section 2 includes a technical basis that might be considered for new regulatory guidance aimed at regulated entities for software tool selection and use in all life cycle processes; software and software tool integrity levels; software tool planning and documentation; software tool quality assurance (QA) and configuration management (CM);

software tool training; software tool development, qualification, and dedication; and software tool review, approval, and acceptance criteria. Section 2 also contains a technical basis that might be considered for improving current regulatory guidance and for influencing changes to industry standards. Section 3 presents a summary of the technical basis that might be considered for new and improved regulatory guidance. Section 4 provides references.

The technical basis developed in Task 3 might be considered in developing regulatory guidance in both a deterministic and risk-informed approach for software tools. The technical basis is structured in a way that might be considered for implementation in a phased approach to minimize the immediate impact of new regulatory guidance on licensees and applicants. The technical basis for the review and approval process might be useful as a two-pronged approach to regulatory oversight of software tools, which could comprise oversight of the software tool acceptance activities by licensees and applicants in conjunction with independent NRC evaluation of selected tools.

# Table of Contents

# Figures



# Tables

## Abbreviations and Acronyms

The following acronyms and abbreviations are used frequently throughout this report. The report convention is to define the acronym the first time it is used in the report and to define all acronyms in the following list.

| | |
|---|---|
| AECL | Atomic Energy of Canada, Ltd |
| BTP | Branch Technical Position |
| CASE | Computer-Aided Software Engineering |
| CCF | Common-Cause Failures |
| CGI | Commercial-Grade Item |
| CM | Configuration Management |
| COTS | Commercial off-the-Shelf |
| DI&C | Digital Instrumentation and Control |
| EPRI | Electric Power Research Institute |
| FAA | Federal Aviation Administration |
| GL | Generic Letter |
| I&C | Instrumentation and Control |
| IAEA | International Atomic Energy Agency |
| IEC | International Electrotechnical Commission |
| IEEE | Institute of Electrical and Electronics Engineers |
| ISG | Interim Staff Guidance |
| ISL | Information Systems Laboratories, Inc. |
| ISO | International Organization for Standardization |
| NASA | National Aeronautics and Space Administration |
| NIST | National Institute of Standards and Technology |
| NRC | U.S. Nuclear Regulatory Commission |
| NUREG | U.S. Nuclear Regulatory Publication |
| PDD | Programmable Digital Device |
| QA | Quality Assurance |
| RG | Regulatory Guide |
| RISC | Risk-informed Safety Class |
| RTCA | Radio Technical Commission for Aeronautics |
| SCM | Software Configuration Management |
| SRS | Software Requirements Specification |
| SSCs | Structures, Systems, and Components |
| Std. | Standard |
| TQL | Tool Qualification Level |
| V&V | Verification and Validation |

## Glossary

For the purposes of this document, the following terms and definitions apply. Definitions without a reference are from software engineering experts in the commercial nuclear power industry.

**Accident:** An unplanned event or series of events that results in death, injury, illness, environmental damage, or damage to, or loss of, equipment or property. (Regulatory Guide (RG) 1.173)

**Assurability:** Ability to confirm the certainties or correctness of a claim, based on evidence and reasoning.

**Basic component:** A safety-related structure, system, or component (SSC), or part thereof, as defined below. (10 CFR 21.3)

(1) (i) When applied to nuclear power plants licensed under 10 CFR Part 50 or Part 52, basic component means a SSC, or part thereof, that affects its safety function necessary to assure:

(A) the integrity of the reactor coolant pressure boundary;

(B) the capability to shut down the reactor and maintain it in a safe shutdown condition; or

(C) the capability to prevent or mitigate the consequences of accidents which could result in potential off-site exposures comparable to those referred to in § 50.34(a)(1), § 50.67(b)(2), or § 100.11, as applicable.

(ii) Basic components are items designed and manufactured under a QA program complying with 10 CFR Part 50, Appendix B, or commercial-grade items (CGIs) that have successfully completed the dedication process.

(2) When applied to standard design certifications under 10 CFR Part 52, Subpart C, and standard design approvals under 10 CFR Part 52, basic component means the design or procurement information approved or to be approved within the scope of the design certification or approval for a SSC, or part thereof, that affects its safety function necessary to assure:

(i) the integrity of the reactor coolant pressure boundary;

(ii) the capability to shut down the reactor and maintain it in a safe shutdown condition; or

(iii) the capability to prevent or mitigate the consequences of accidents which could result in potential off-site exposures comparable to those referred to in § 50.34(a)(1), § 50.67(b)(2), or § 100.11, as applicable.

(3) When applied to other facilities and other activities licensed under 10 CFR Parts 30, 40, 50 (other than nuclear power plants), 60, 61, 63, 70, 71, or 72, basic component means a SSC, or part thereof, that affects their safety function, that is directly procured by the licensee of a facility or activity subject to the regulations in 10 CFR Part 21 and in which a defect or failure to comply with any applicable regulation, order, or license issued by the Commission could create a substantial safety hazard.

(4) In all cases, basic component includes safety-related design, analysis, inspection, testing, fabrication, replacement of parts, or consulting services that are associated with the component hardware, design certification, design approval, or information in support of an early site permit application under 10 CFR Part 52, whether these services are performed by the component supplier or others.

**Commercial-grade item (CGI):** A safety-related SSC as defined below. (10 CFR 21.3)

(1) When applied to nuclear power plants licensed pursuant to 10 CFR Part 50, CGI means a SSC, or part thereof, that affects its safety function, that was not designed and manufactured as a basic component. CGIs do not include items where the design and manufacturing process require in-process inspections and verifications to ensure that defects or failures to comply are identified and corrected (i.e., one or more critical characteristics of the item cannot be verified).

(2) When applied to facilities and activities licensed pursuant to 10 CFR Parts 30, 40, 50 (other than nuclear power plants), 60, 61, 63, 70, 71, or 72, CGI means an item that is:

(i) not subject to design or specification requirements that are unique to those facilities or activities;

(ii) used in applications other than those facilities or activities; and

(iii) to be ordered from the manufacturer or supplier on the basis of specifications set forth in the manufacturer's published product description (e.g., a catalog).

Note that the current rulemaking process for revision of 10 CFR Part 21 is expected to add nuclear power plants licensed pursuant to 10 CFR Part 52 to those for which this definition of CGI is applicable. The Part 21 revision is also expected to clarify definition (1) above. The clarification is expected to make it clearer that items requiring in-process inspections and verifications may be dedicated, provided that all critical characteristics can be verified. Such verification would be done through a commercial-grade survey (EPRI Method 2) that verifies that the manufacturer verifies the critical characteristics in question during manufacture (e.g., at hold points) or by means of source verification (EPRI Method 3) in which a representative of the dedicating entity performs or witnesses the verification directly (e.g., at hold points) on the actual CGIs being manufactured and being dedicated and supplied by or through the dedicating entity.

**Common-cause failure (CCF):** Loss of function to multiple SSCs due to a shared root cause. This is important to an understanding of software tools as the same tool may be used in development of software for multiple systems thus making a fault in the tool a potential source of a CCF. (IEEE Std. 603-2009 [3])

**Complexity:** The degree to which a SSC has a design or implementation that is difficult to understand and verify. (IEEE Std. 100-2000 [4])

**Component:** One part that makes up a system. A component may be hardware or software and may be subdivided into other components. (IEEE Std. 1012-2012 [5])

**Configuration item:** An aggregation of hardware, software, or both, that is designated for CM and treated as a single entity in the CM process. (IEEE Std. 7-4.3.2 [6])

**Configuration management (CM):** A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements. (IEEE Std. 828-2012 [7])

**Critical characteristics:** When applied to nuclear power plants licensed pursuant to 10 CFR Part 50, critical characteristics are those important design, material, and performance characteristics of a CGI that, once verified, will provide reasonable assurance that the item will perform its intended safety function. (10 CFR 21.3)

> Note that the current rulemaking process for revision of 10 CFR Part 21 is expected to add nuclear power plants licensed pursuant to 10 CFR Part 52 to those for which this definition of critical characteristics is applicable.

**Dedication:** An acceptance process as defined below. (10 CFR 21.3)

(1) When applied to nuclear power plants licensed pursuant to 10 CFR Part 30, 40, 50, and 60, dedication is an acceptance process undertaken to provide reasonable assurance that a CGI to be used as a basic component will perform its intended safety function and, in this respect, is deemed equivalent to an item designed and manufactured under a 10 CFR Part 50, Appendix B, QA program. This assurance is achieved by identifying the critical characteristics of the item and verifying their acceptability by inspections, tests, or analyses performed by the purchaser or third-party dedicating entity after delivery, supplemented as necessary by one or more of the following: commercial grade surveys; product inspections or witness at hold points at the manufacturer's facility, and analysis of historical records for acceptable performance. In all cases, the dedication process must be conducted in accordance with the applicable provisions of 10 CFR Part 50, Appendix B. The process is considered complete when the item is designated for use as a basic component.

(2) When applied to facilities and activities licensed pursuant to 10 CFR Parts 30, 40, 50 (other than nuclear power plants), 60, 61, 63, 70, 71, or 72, dedication occurs after receipt when that item is designated for use as a basic component.

Note that the current rulemaking process for revision of 10 CFR Part 21 is expected to add nuclear power plants licensed pursuant to 10 CFR Part 52 to those for which this definition of dedication is applicable.

**Defect:** In the context of equipment, systems, or components (including software) important to safety in NRC-licensed facilities (nuclear power plants in particular), the term defect means: (10 CFR 21.3)

(1) A deviation in a basic component delivered to a purchaser for use in a facility or an activity subject to the regulations in 10 CFR Part 21 if, on the basis of an evaluation, the deviation could create a substantial safety hazard;

(2) The installation, use, or operation of a basic component containing a defect as defined in 10 CFR 21.3;

(3) A deviation in a portion of a facility subject to the early site permit, standard design certification, standard design approval, construction permit, combined license or manufacturing licensing requirements of 10 CFR Part 50 or 10 CFR Part 52, provided the deviation could, on the basis of an evaluation, create a substantial safety hazard and the portion of the facility containing the deviation has been offered to the purchaser for acceptance;

(4) A condition or circumstance involving a basic component that could contribute to the exceeding of a safety limit, as defined in the technical specifications of a license for operation issued under 10 CFR Part 50 or 10 CFR Part 52; or

(5) An error, omission, or other circumstance in a design certification or standard design approval that, on the basis of an evaluation, could create a substantial safety hazard.

**Deviation:** A departure from the technical requirements included in a procurement document, or specified in early site permit information, a standard design certification, or standard design approval. (10 CFR 21.3)

**Error:** The difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. (IEEE Std. 7-4.3.2-2003 [8])

**Failure:** The inability of a system or component to perform its required functions within specified performance requirements. (IEEE Std. 7-4.3.2-2003 [8])

**Fault:** (1) A defect in a hardware device or component, or (2) an incorrect step, process, or data definition in a computer program. (IEEE Std. 7-4.3.2-2003 [8])

**Important to safety:** Comprising equipment that based on deterministic evaluation, (1) is safety-related, (2) is not safety-related, but could impact safety, and (3) certain post-accident monitoring equipment as described in RG 1.97. (10 CFR 50.49(b))

**Independent verification and validation:** V&V performed by an organization that is technically, managerially, and financially independent of the development organization. (IEEE Std. 1012-2012 [5])

**Integrity level:** A value representing project-unique characteristics (e.g., complexity, criticality, risk, safety level, security level, desired performance, and reliability) that define the importance of the system, software, or hardware to the user. (IEEE Std. 1012-2012 [5])

**Partitioning:** A technique for providing isolation between software components to contain and isolate faults. (RTCA DO-178C [9])

**Process:** A set of interrelated or interacting activities which transforms input into output. (IEEE Std. 12207-2008 [10])

**Programmable digital device (PDD):** Any device that relies on software instructions or programmable logic to accomplish a function. Examples include a computer, a programmable hardware device, or a device with firmware. (IEEE Std. 7-4.3.2 [6])

**Qualification:** Process of demonstrating whether an entity is capable of fulfilling specified requirements. (IEEE Std. 12207-2008 [10])

**Reasonable assurance:** A justifiable level of confidence based on objective and measurable facts, actions, or observations from which adequacy may be inferred. (EPRI TR-102260 [11])

**Reliability:** The degree to which a software system or component operates without failure considering only the existence of failure and not the consequences of failure. (U.S. Nuclear Regulatory Publication (NUREG)-0800, Branch Technical Position (BTP) 7-14)

**Requirement:** (1)(a) A characteristic that a system or a software item is required to possess in order to be acceptable to the acquirer, (b) a binding statement in a standard; (2) a statement that identifies a product or process operational, functional, or design characteristic or constraint, which is unambiguous, testable, or measurable, and necessary for product or process acceptability; (3)(a) a condition or capability needed by a user to solve a problem or achieve an objective, (b) a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document, (c) a documented representation of a condition or capability as in definition (3)(a) or (3)(b). (IEEE Std. 100-2000 [4])

> A requirement in this sense is not to be construed as a regulatory requirement (i.e., a provision of regulations). Requirements in this sense, such as mandatory clauses in a standard that must be complied with in order for the standard to be considered met, would normally be characterized as acceptance criteria in NRC review guidance such as NUREG-0800.

**Safety function:** One of the processes or conditions (e.g., emergency negative reactivity insertion, post-accident heat removal, emergency core cooling, post-accident radioactivity removal, and containment isolation) essential to maintain plant parameters within acceptable limits established for a design basis event. (IEEE Std. 603-2009 [3])

**Safety-related:** Designated through deterministic evaluation, a subset of things important to safety; structures, systems, equipment and components that are relied upon to remain functional during and following design-basis events to ensure: (a) the integrity of the reactor coolant pressure boundary, (b) the capability to shut down the reactor and maintain it in a safe shutdown condition, or (c) the capability to prevent or mitigate the consequences of accidents that could results in potential off-site exposures. (10 CFR 21.3)

> Note that in NRC regulations, in the definition of things that are safety-related, the off-site exposure-related regulations have been updated to include 10 CFR 50.34(a)(1), 10 CFR 50.67(b)(2), and specifying § 100.11 of 10 CFR 100.

**Safety-significant systems:** For the purposes of this report, safety-significant systems include all systems designated important to safety (i.e., defined in the regulations as comprising safety-related systems and certain nonsafety-related systems that could impact safety) through deterministic evaluation and all systems with safety significance or risk importance as determined through probabilistic risk analysis.

**Safety system:** A system that is relied upon to remain functional during and following design-basis events to ensure: (a) the integrity of the reactor coolant pressure boundary, (b) the capability to shut down the reactor and maintain it in a safe shutdown condition, or (c) the capability to prevent or mitigate the consequences of accidents that could result in potential off-site exposures comparable to the 10 CFR Part 100 guidelines. (IEEE Std. 603-2009 [3])

> Note that in NRC regulations, in the definition of things that are safety-related, the off-site exposure-related regulations have been updated to include 10 CFR 50.34(a)(1), 10 CFR 50.67(b)(2), and specifying § 100.11 of 10 CFR 100. Also, the definition of a safety system in IEEE Std. 603-2009 [3] is essentially the same as the definition of safety-related systems and basic components in NRC regulations.

**Secure operational environment:** The condition of having appropriate physical, logical, and administrative controls within a facility to ensure that the reliable operation of digital safety systems is not degraded by undesirable behavior of connected systems and events initiated by inadvertent access to the system. (RG 1.152)

**Software:** The programs used to direct operations of a PDD. Examples include computer programs and logic for PDDs, and data pertaining to its operation. Software includes coding structures using ladder logic for programmable logic controllers, hardware description languages (e.g., Very High Speed Integrated Circuit Hardware Description Language and Verilog for Field Programmable Gate Arrays), function blocks (e.g., the Areva Teleperm® XS platform and the Westinghouse Common Qualified Platform), and other types of language constructs. (Adapted from IEEE Std. 7-4.3.2 [6])

**Software tools:** A computer program supporting or used in the design, development, testing, review, analysis, or maintenance of a PDD or its documentation. Examples include compilers, assemblers, linkers, comparators, cross-reference generators, decompilers, editors, flow charters, monitors, test case generators, integrated development environments, timing analyzers, simulators, and thermal-hydraulic analysis programs. (Adapted from IEEE Std. 7-4.3.2 [6])

**State of the practice:** The most advanced documented knowledge in an engineering field in actual use on a regular basis.

(1) A study of the "state-of-the-practice" in a particular field may entail a combination of review of literature such as public consensus standards, survey of leading practitioners, interviews with the best responders, and analysis and organization of the acquired data.

(2) "State-of-the-practice" in an engineering field excludes a process requiring exceptional artistry and craftsmanship (e.g., knowledge not transferable in a written form).

(3) Distinction between "state-of-the-practice" and "common practice": The latter is characterized by breadth of usage in a particular engineering field and industry. The former is distinguished through repeated usage of the most advanced and proven knowledge – also known as "best practice" in some communities.

(4) Distinction between "state of practice" and "state of the art": The latter term characterizes advanced or recently developed methods, techniques and technologies, approaches, etc., that may still be under development or refinement (e.g., "beta testing"), or that may have been demonstrated to be valid or useful, but have not yet been universally adopted in industry codes or standards, or are not yet in common use in the industry.

**Validation:** (1) The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements, or (2) the process of providing evidence that the system, software, or hardware and its associated products satisfy requirements allocated to it at the end of each life cycle activity; solve the right problem (e.g., correctly model physical laws, implement business rules, and use the proper system assumptions); and satisfy intended use and user needs (i.e., the right software was built). (IEEE Std. 1012-2012 [5])

**Verification:** (1) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase, or (2) the process of providing objective evidence that the system, software, or hardware and its associated products conform to requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance); satisfy standards, practices, and conventions during life cycle processes; and successfully complete each life cycle activity and satisfy all the criteria

for initiating succeeding life cycle activities. Verification of interim work products is essential for proper understanding and assessment of the life cycle phase product(s) (i.e., the software was built right). (IEEE Std. 1012-2012 [5])

# 1 Introduction

Current and future nuclear power plant safety systems will rely on digital instrumentation and control (DI&C) systems based on a variety of digital technologies. These safety-significant systems are expected to be developed using various types of software-based development tools. Tool-automated and tool-assisted engineering activities may result in fewer defects than with manual engineering activities if tools are well designed, carefully developed, carefully selected, rigorously tested, and appropriately used. However, unknown deficiencies in automated tools or tool-assisted engineering activities may affect the safety assurability of digital safety systems, dependent on such tools. For the purposes of this report, the term "safety-significant systems" as used in this context should be understood to include all systems with safety significance (i.e., systems deterministically designated as being important to safety, which include both safety-related systems and certain nonsafety-related systems that can impact safety, as well as systems that are safety-significant or risk-important as determined by probabilistic risk analysis).

As defined in IEEE Standard (Std.) 7-4.3.2-2010 [12], software tools are computer programs used in the design, development, testing, review, analysis, or maintenance of a program or its documentation. Based on industry usage, software tools are computer programs or computer-based engineering environments supporting or used in the specification of requirements, architecture, design, implementation, testing, and other forms of verification, review, analysis, validation, and documentation of a system or such components of the system as software, and maintenance of these artifacts or work products. Examples include scripts, compilers, assemblers, linkers, comparators, cross-reference generators, decompilers, editors, flow charters, monitors, test case generators, integrated development environments, timing analyzers, programming generators, simulators, and thermal-hydraulic analysis programs, among many others.

More applications for new nuclear power plants and for modifications to existing nuclear power plants may include an expanded use of software tools in the development of DI&C systems. Some of the design, program generation, design V&V, automated testing, and operation of these DI&C systems may use or expand the use of software tools. Additional regulatory guidance may provide more consistent guidance on the application of the tools. Further, changes to current regulatory guidance and industry standards, or developing new regulatory guidance that will support the industry standards and meet regulatory requirements, might improve the software tool review and approval process. Contract NRC-HQ-13-C-04-0004 is a research project intended to develop a technical basis that might be considered for improved or new regulatory guidance that will follow industry standards, support regulatory requirements, and be consistently implementable. This project documents existing regulatory guidance, standards, practices, and experience concerning review and approval of the use of software tools in engineering systems in industries across the world. Most of the existing documents are not specifically designed for software tool reviews and the guidance is not always directly relevant to software tools.

## 1.1 Background

This research project is phased in three tasks and is depicted in Figure 1.1. In Task 1, the project surveyed standards, regulatory guidance, review and approval practices, and other current industry practices concerning the use of software tools on which the safety assurability of a digital, safety-significant system is dependent. In Task 2, the project performed a detailed analysis of the software-related information identified in Task 1 to determine how FAA, NASA, European nuclear regulatory organizations, and other organizations and industries review and approve software tools used in engineering system design, program generation, design V&V, testing, and operation, and examine how their processes are supported by the industry standards. In Task 3, the project developed a technical basis for software tool regulatory guidance for review and acceptance of software tools based on the Task 1 research and Task 2 analysis.

Figure 1.2 depicts the Task 1 survey process, scope, and products similar to Figure 1.1 but limited to the Task 1 scope. Task 1 surveyed numerous industry standards, government positions, and regulatory guidance covering different industries (i.e., nuclear, aerospace, civil aviation, railway, and automotive), different countries (i.e., U.S., Canada, and Europe), different standard organizations (i.e., IEEE, IEC, ISO, IAEA, EPRI, and RTCA), and involving many government organizations (i.e., NASA, FAA, NRC, NIST, and AECL). The purpose was to learn from a broad spectrum of expertise and extract the most important and relevant criteria to establish the technical basis for potential regulatory guidance for the safety assessment of software tool use in U.S. nuclear safety systems. Although only the most relevant documents are shown in Figure 1.2, Task 1 actually surveyed over more than 150 industry standards, government positions, and regulatory guidance.

The Task 1 survey found an abundance of information that was applicable to software and logic, directly or indirectly, some that was relevant to software tools, directly or indirectly, and some that, while not written for software tools, could be adaptable and useful in the safety assessment of software tools. This information included regulations or regulatory requirements, regulatory guidance, regulatory information, industry guidance, and industry information. The product of Task 1 was a report [1] that summarized the results of the survey. The Task 1 report [1] recommended a broad baseline of requirements based on industry standards and a broad baseline of guidance based on industry guidance. The Task 1 report [1] also recommended review and approval practices based on industry review and approval practices. The baseline requirements for software tools to support regulatory assurance bring about the need for proper selection, evaluation and qualification, adequate CM, thorough V&V, proper use as described by the tool's intended purpose, appropriate assignment of software and software tool integrity levels, thorough up-front planning, continuous QA, adequate reviews, and adequate training on tool use. The baseline guidance for software tools to support regulatory assurance includes selection guidance, evaluation and qualification guidance, CM practices, V&V practices, guidance and limitations on tool use, software and software tool integrity level guidance, planning practices, QA guidance, review and approval guidance, and training practices.

As part of Task 1, the project also researched industry standards, guidance, and practices to determine the types of software tools used in the nuclear and other industries and the general philosophy for software tool use during PDD development. The Task 1 research culminated in a generic list of software tools and a list of software tool categories based upon tools that are widely used within the software development industry. Based on similarities between certain software tools and software tool usage and availability in certain life cycle processes, all software tools were grouped into nine software tool categories as documented in the Task 1 report [1]. These nine software tool categories correspond with their use in certain life cycle processes.

Based on the information in all of the documents surveyed in Task 1, the project found that software tools were used in many of the software and system life cycle processes. However, guidance and review and approval practices were typically limited to software tools used in processes, activities, and tasks that have the potential of directly introducing defects or failing to detect defects in the final safety system software.

Figure 1.3 depicts the Task 2 analysis process, scope, and products. In Task 2, the project performed a detailed analysis of the software-related information identified in Task 1 to determine how FAA, NASA, European nuclear regulatory organizations, and other organizations and industries review and approve software tools used in engineering system design, program generation, design V&V, testing, and operation, and examine how their processes are supported by the industry standards. Task 2 also included an assessment of the hazards associated with the use of each type of software tool in different stages of the engineering life cycle (e.g., challenges to assurability, verifiability, and analyzability) and corresponding criteria to determine the suitability of tools for representative uses. The Task 2 report [2] documented the results of the detailed analysis of 43 industry documents that contained the underlying principles for the baseline requirements, baseline guidance, and review and approval practices.

The Task 2 analysis determined the relevance of the software-related information to software tools and its usefulness in the several aspects of software tool categorization and acceptance, and regulatory review and approval. The several schemes of categorization of tools used in the various industries determine the appropriate degree of rigor to be applied in the process of verifying the tools' suitability, quality, and reliability through a process of tool qualification.

The Task 2 analysis also resulted in information that might be considered in developing new regulatory guidance or revising existing regulatory guidance for software tools as well as internal guidance for the review of applications for approval of software tools to be used in the development and testing of software or logic associated with systems important to safety in NRC-licensed facilities.

## 1.2 Overview

This report is the product of contract NRC-HQ-13-C-04-0004 Task 3. More applications for new nuclear power plants and for modifications to existing nuclear power plants may include an expanded use of software tools in the development of DI&C systems. More software development applications may include the use of software tools, and additional regulatory

guidance may provide more consistent guidance on the application of the tools. Further, changes to current regulatory guidance and industry standards, or developing new regulatory guidance that will support the industry standards and meet regulatory requirements, might improve the software tool review and approval process.

The technical basis in this report is consistent with enhancing regulatory guidance using a phased approach, putting more attention on the deterministic methods currently used by most licensees and describing possible alternative methods, such as risk-informed methods, that may be useful in the future. The technical basis is structured in a way that might be useful for implementation in a phased approach to minimize the immediate impact of new or revised regulatory guidance on licensees and applicants.

The technical basis for the review and approval process might be useful as a two-pronged approach to regulatory oversight of software tools that would support improving the efficiency, effectiveness, and timeliness of regulatory reviews. The two-pronged approach would comprise oversight of the software tool acceptance activities by licensees and applicants in conjunction with independent NRC evaluation of selected tools. This report also discusses a technical basis that might be useful for influencing industry standards and presents a technical basis that might be useful for specifying limitations or other conditions that apply to the use of software tools.

The technical basis in this report represents current industry practices that might be considered by the NRC for use in the domestic commercial nuclear power industry. Some of the industry practices may conflict with current NRC practice while other industry practices might support or might be implemented to supplement existing NRC practice. Consideration of the technical basis in this report for domestic commercial nuclear power that are based on various industry practices is at the discretion of the NRC. The NRC can decide to pursue or not pursue changes to regulatory guidance, new regulatory guidance, or influencing changes to industry standards based on the technical basis in this report.

This Task 3 report contains only a high level discussion of various industry documents sufficient to explain the fundamental principles for the technical basis in this report. The Task 2 report [2] contains a thorough, detailed discussion of these documents and should be referenced for more detailed information on any particular document. The Task 1 report [1] also contains a brief summary of over 150 documents surveyed for this project and should be referenced to gain a general understanding of the content of any particular document.

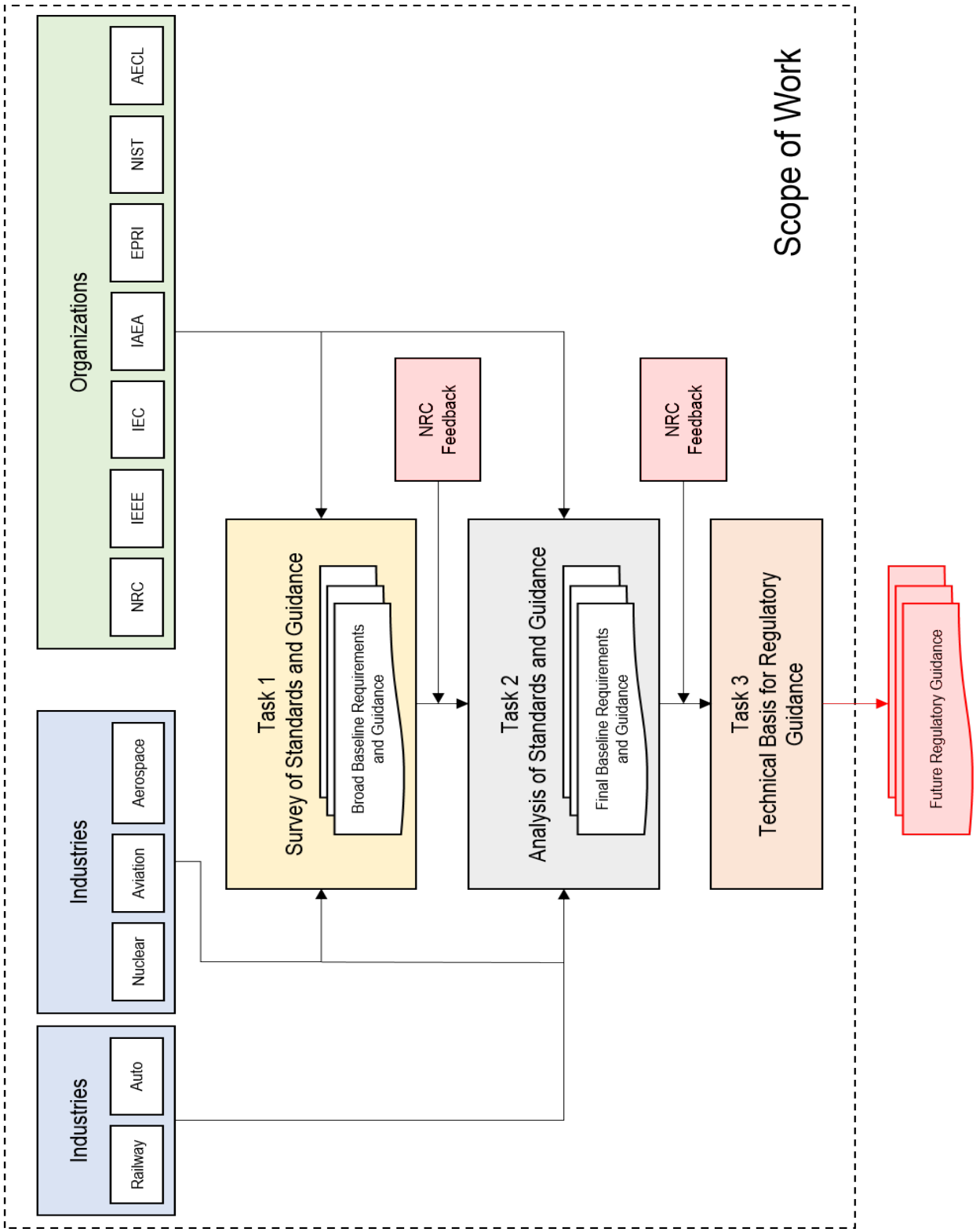


**Figure 1.1 – Overall Research Project Scope**

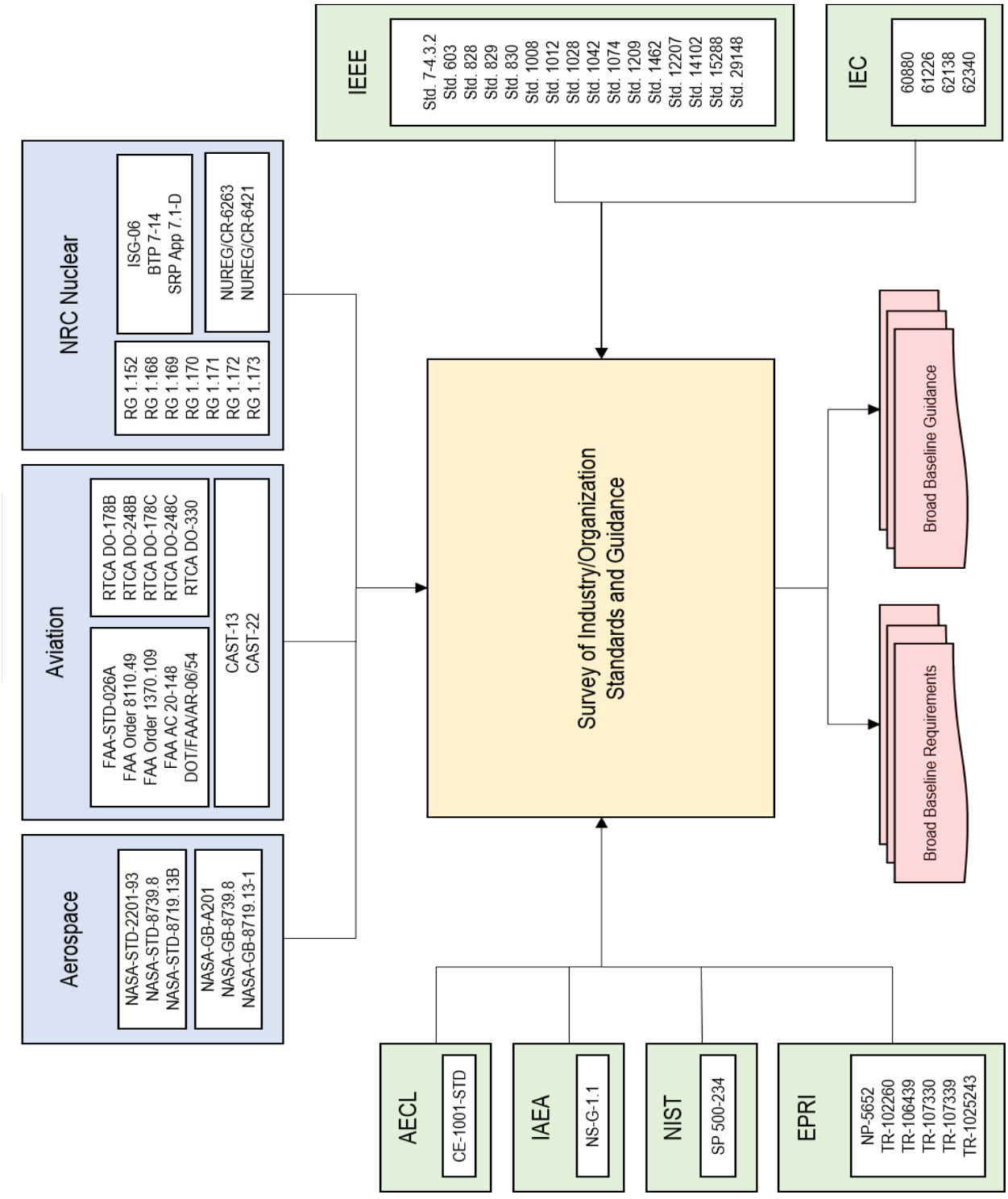


**Figure 1.2 – Task 1 Survey of Industry and Organization Standards and Guidance**

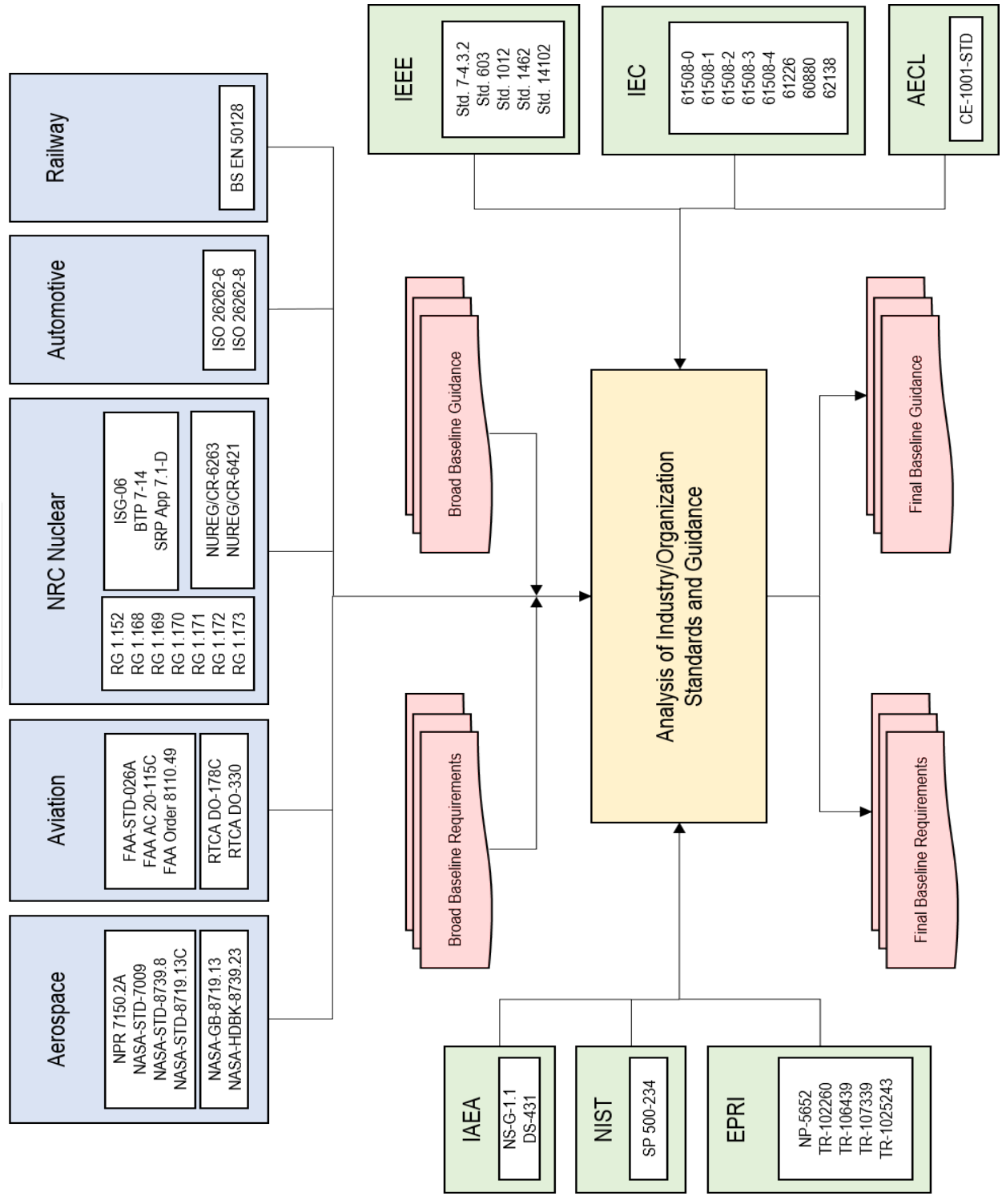


**Figure 1.3 – Task 2 Analysis of Industry and Organization Standards and Guidance**

# 2 Technical Basis

The NRC regulatory framework consists of regulations, other regulatory (legal) requirements (e.g., license technical specifications), regulatory guidance, and regulatory and technical information. Regulatory guidance, which is meant to help implement the regulatory requirements, has historically comprised policy, guidance aimed at the NRC's regulated entities, and guidance for the NRC staff.

Policy takes the form of (1) statements of considerations for the regulations, (2) Commission policy statements (e.g., the safety goals), (3) SECY papers and their associated Staff Requirements Memorandums, (4) the staff's position, with Commission approval, on various issues as expressed in RGs that typically provide qualified endorsements of industry codes and standards, (5) certain NUREGs, and (6) generic communications such as generic letters (GLs), and regulatory issue summaries.

Guidance aimed at licensed and regulated entities has taken the form mainly of RGs that provide guidance directly to licensees and applicants without necessarily referencing industry documents (e.g., RG 1.206, which promulgates guidance on combined license applications under 10 CFR Part 52, Subpart C; or Interim Staff Guidance (ISG)-06, which prescribes the information to be submitted with applications for approval of DI&C equipment and software) as well as RGs that reference or provide qualified endorsements of industry documents. However, some industry documents have been endorsed or approved for limited use by means of NRC safety evaluation reports.

Guidance for the NRC staff includes technical, regulatory, and internal procedural guidance. Examples include technical review guidance (e.g., NUREG-0800, its appendices, and BTPs), inspection guidance, enforcement guidance, and internal procedures for all regulatory activities from rulemaking to review and approval.

The technical basis in Section 2 of this report consists of a set of provisions, including endorsements of documents that Task 2 analysis supports, and specific approaches with options for integrating them into the NRC regulatory framework. The technical basis also contains processes that may be used by licensees, applicants, and third-party contractors or vendors for developing, evaluating, and qualifying tools, whether acceptance of tools is based on the tool being developed under a 10 CFR Part 50, Appendix B, QA program; based on the dedication of commercial-grade tools not developed under an Appendix B QA program using an approach such as EPRI TR-1025243 [13]; developed using another high-quality life cycle approach such as RTCA DO-330 [14], or using the RTCA DO-330 [14] process as a means of dedication (e.g., of commercial off-the-shelf (COTS) software tools). The technical basis also contains specific and detailed information on review and approval processes and acceptance criteria that might be considered for use in NRC license review for the evaluation of proposed equipment and processes involving the use of software tools in their life cycle.

## 2.1 Software Life Cycle Processes

The modern approach to software engineering, including application of software tools, revolves around the implementation of suitable software life cycle processes. As such, any software tool regulatory guidance must consider software life cycle processes. Each software life cycle process should be assessed to determine whether regulatory guidance is necessary and beneficial for software tools used in that process. The assessment of software life cycle processes determines the types of tools used in that process and the tool's impact on the software. The assessment of software life cycle processes allows software tools to be categorized based on similarity of function. Software tools are categorized to determine the impact of software tool failures on the safety-significant software or logic and to determine the necessary software tool development, dedication, or qualification activities and tasks consistent with the safety significance of the software being developed using the tool.

Software tool guidance and review and approval practices are typically limited to software tools used in processes, activities, and tasks that have the potential of introducing defects or failing to detect defects in the final PDD software or logic. These processes, activities, and tasks that have the potential of introducing defects or failing to detect defects are associated with software or logic development and V&V. Expanding software tool regulatory guidance and review and approval practices to cover more PDD life cycle support processes (i.e., QA and CM processes) might be considered when assurability of the safety-significant system software or logic is dependent on a tool used during that life cycle process. Establishing a standard set of system-context and software-specific processes and determining whether a tool used during a specific process impacts the final nuclear plant software might be considered in future regulatory guidance.

Three principal IEEE standards apply to system and software life cycles: IEEE Std. 1074-2006 [15], IEEE Std. 15288-2008 [16], and IEEE Std. 12207-2008 [10]. IEEE Std. 1074-2006 [15], endorsed by RG 1.173, is a standard for developing a software project life cycle process and can be used where software is the total system or where software is part of a larger system. Completing the activities and tasks of IEEE Std. 1074-2006 [15] completes a single process of IEEE Std. 12207-2008 [10], (i.e., Clause 6.2.1); therefore, IEEE Std. 1074-2006 [15] has the narrowest coverage of the three life cycle standards.

IEEE Std. 15288-2008 [16] establishes a common process framework for describing the life cycle of man-made systems. IEEE Std. 15288-2008 [16] does not limit the application to software life cycles and instead opens the application to a broader system life cycle. Of the three standards, IEEE Std. 15288-2008 [16] has the broadest range of coverage.

IEEE Std. 12207-2008 [10], which was written concurrently with IEEE Std. 15288-2008 [16], establishes a framework for software life cycle processes and applies to the acquisition of systems and software products and services; to the supply, development, operation, maintenance, and disposal of software products; and to the software portion of a system whether performed internally or externally to an organization. IEEE Std. 12207-2008 [10] does not have the broad coverage of IEEE Std. 15288-2008 [16] but has more extensive coverage than the narrow focus of IEEE Std. 1074-2006 [15].

Many updates to IEEE software-related standards (e.g., IEEE Std. 828-2012 [7], IEEE Std. 829-2008 [17], IEEE Std. 1012-2012 [5], IEEE Std. 1028-2008 [18], IEEE Std. 14102-2010 [19], and IEEE Std. 29148-2011 [20]) include modifications for harmonization with IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16]. Only two of these six standards (i.e., IEEE Std. 829-2008 [17] and IEEE Std. 1028-2008 [18]) that have been harmonized with IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16] are currently endorsed by RGs. The following guidance might be considered for maintaining consistency with the latest IEEE standards that discuss system and software life cycle processes.

1. *A revision to RG 1.168 might be considered to endorse IEEE Std. 1012-2012 [5] in lieu of IEEE Std. 1012-2004 [21] since the latest version expands the scope of the V&V processes to include systems and hardware as well as software, and the terminology, structure, and mappings have been revised consistent with IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16].*

2. *A revision to RG 1.169 might be considered to endorse IEEE Std. 828-2012 [7] in lieu of IEEE Std. 828-2005 [22] since the latest version expands IEEE Std. 828-2005 [22] beyond just defining the contents of a software configuration management (SCM) plan and supports IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16].*

3. *A revision to RG 1.170 might be considered to reference IEEE Std. 12207-2008 [10] in lieu of IEEE Std. 1074-2006 [15] as an accepted practice for the development of software for safety-related applications including the use of a software life cycle process that incorporates software testing activities.*

4. *A revision to RG 1.170 might be considered to reference IEEE Std. 1012-2012 [5] in lieu of IEEE Std. 1012-2004 [21] for software testing which is a key element in software V&V activities since IEEE Std. 1012-2012 [5] has expanded V&V processes for consistency with IEEE Std. 12207-2008 [10].*

5. *A revision to RG 1.172 might be considered to endorse IEEE Std. 29148-2011 [20] in lieu of IEEE Std. 830-1998 [23] since IEEE Std. 830-1998 [23] has been superseded by IEEE Std. 29148-2011 [20] and IEEE Std. 29148-2011 [20] gives guidelines for applying the requirements and requirements-related processes described in IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16].*

6. *A revision to RG 1.173 might be considered to endorse IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16], which contain software and system development processes, respectively, in lieu of IEEE Std. 1074-2006 [15].*

## 2.2 Software Tool Categories

This section includes a technical basis that might be considered for future regulatory guidance to categorize software tools based on industry practices that meet regulatory requirements. Current regulatory guidance for software tools is limited and does not include guidance for categorizing software tools.

Computerized tools used in the development of software are referred to as software tools or computer-aided software engineering (CASE) tools. Most CASE tools are available as COTS software. Other software tools are specifically developed for the current PDD development project either in house or by an outside vendor. Software tools can be used in almost every phase of the PDD development project life cycle and assist in almost every task.

RG 1.152 endorses IEEE Std. 7-4.3.2-2003 [8] as an acceptable method of satisfying regulations with respect to high functional reliability and design requirements for computers used in the safety systems of nuclear power plants. RG 1.168, RG 1.169, and RG 1.170 all state that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8], as endorsed by RG 1.152. IEEE Std. 7-4.3.2-2003 [8] does not contain guidance for categorizing software tools; however, the proposed draft version of IEEE Std. 7-4.3.2 [6] defines five types of software tools and states that software tools shall be assessed in a manner consistent with the category of tool and the potential that the tool has to introduce faults into the safety-related PDD.

IEEE Std. 14102-2010 [19] is not currently endorsed by regulatory guides; however, IEEE Std. 14102-2010 [19] discusses the attributes of nine CASE tool characteristics that are closely associated with IEEE Std. 12207-2010 [10] software life cycle processes. Table 2.1 identifies IEEE Std. 12207-2008 [10] processes and subprocesses in which CASE tools are used and the software tool category associated with that subprocess. As shown in Table 2.1, CASE tools are not used in all software and system life cycle processes either because CASE tools have not yet been developed to automate, eliminate, or reduce activities and tasks associated with that process or the use of CASE tools in that process would not provide a significant benefit. Also, CASE tools are not always used in the processes for which tools are available because the cost, effort, and risk of using tools in some processes may outweigh the benefits. Although CASE tools are not available for use in all software life cycle processes, new tools are always being developed and the tool categories need to be flexible enough to handle newly-developed tools.

Table 2.2 compares software tool categories from five industry documents including IEEE Std. 14102-2010 [19], the proposed draft of IEEE Std. 7-4.3.2 [6], RTCA DO-178C [9], IEC 60880 [24], and IEC 61508 [25]. The nine CASE tool categories based on the IEEE Std. 14102-2010 [19] CASE tool characteristics related to life cycle processes represent a more detailed categorization of software tools compared to software tool types defined in other documents. The tool categories based on IEEE Std. 14102-2010 [19] also have broader coverage than other documents because IEEE Std. 14102-2010 [19] includes a management support tool category.

**Table 2.1 – IEEE Std. 12207-2008 Processes and Tool Categories**

| Process | Sub-process | Tool Category |
|---|---|---|
| Agreement Processes | Acquisition Process | |
| | Supply Process | |
| Organizational Project-Enabling Processes | Life Cycle Model Management Process | |
| | Infrastructure Management Process | |
| | Project Portfolio Management Process | |
| | Human Resource Management Processes | |
| | Quality Management Process | |
| Project Management Processes | Project Planning Process | Management Support Tools |
| | Project Assessment and Control Process | Management Support Tools |
| Project Support Processes | Decision Management Process | |
| | Risk Management Process | QA Tools |
| | Configuration Management Process | |
| | Information Management Process | |
| | Measurement Process | |
| System Technical Processes | Stakeholder Requirements Definition Process | Implementation Tools |
| | System Requirements Analysis Process | |
| | System Architectural Design Process | |
| | Implementation Process | |
| | System Integration Process | |
| | System Qualification Testing Process | |
| Software Technical Processes | Software Installation Process | |
| | Software Acceptance Support Process | |
| | Software Operation Process | |
| | Software Maintenance Process | Maintenance Tools |
| | Software Disposal Process | |
| Software Implementation Processes | Software Implementation Process | Implementation Tools |
| | Software Requirements Analysis Process | Implementation Tools |
| | Software Architectural Design Process | Software Development Modeling Tools |
| | Software Detailed Design Process | Software Development Modeling Tools |
| | Software Construction Process | Software Development Modeling Tools and Construction Tools |
| | Software Integration Process | Construction Tools |
| | Software Qualification Testing Process | |
| Software Support Processes | Software Documentation Management Process | Documentation Tools |
| | Software Configuration Management Process | CM Tools |
| | Software Quality Assurance Process | QA Tools |
| | Software Verification Process | V&V Tools |
| | Software Validation Process | V&V Tools |
| | Software Review Process | |
| | Software Audit Process | |
| | Software Problem Resolution Process | Maintenance Tools |
| Software Reuse Processes | Domain Engineering Process | |
| | Reuse Asset Management Process | Documentation Tools |
| | Reuse Program Management Process | |

**Table 2.2 – Software Tool Category Comparisons**

<table>
<tr><th>IEEE Std. 14102-2010</th><th>IEEE Std. 7-4.3.2 (Draft)</th><th>RTCA DO-178C</th><th>IEC 60880</th><th>IEC 61508</th></tr>
<tr><td>Management Support Tools</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Maintenance Tools</td><td>Type II<br>Surveillance and Maintenance Tools</td><td></td><td>Diagnostic Tools</td><td></td></tr>
<tr><td>QA Tools</td><td></td><td></td><td rowspan="2">Infrastructure Tools</td><td rowspan="2">Off-Line<br>Class T1</td></tr>
<tr><td>Documentation Tools</td><td rowspan="2">Type III<br>Development Process Support Tools</td><td></td></tr>
<tr><td>CM Tools</td><td></td><td>CM Tools</td><td rowspan="5">Off-Line<br>Class T3</td></tr>
<tr><td>Implementation Tools</td><td></td><td rowspan="3">Criteria 1<br>Development Tools</td><td rowspan="4">Transformation Tools</td></tr>
<tr><td>Software Development Modeling Tools</td><td></td></tr>
<tr><td rowspan="2">Construction Tools</td><td>Type I<br>Software Development Tools</td></tr>
<tr><td rowspan="2">Type IV<br>In-Line V&V Tools</td><td rowspan="2">Criteria 2<br>V&V Tools</td></tr>
<tr><td rowspan="2">V&V Tools</td><td rowspan="2">V&V Tools</td><td rowspan="2">Off-Line<br>Class T2</td></tr>
<tr><td>Type V<br>Off-Line V&V Tools</td><td>Criteria 3<br>V&V Tools</td></tr>
</table>

As shown in Table 2.2, IEEE Std. 14102-2010 [19] includes a software development modeling tool category that refers to tools used in the software development process to directly support the software architectural design, detailed design, and construction processes (e.g., data flow diagrams, diagram analyzers, and specification construct modeling). However, this software development modeling tool category does not include plant modeling, analysis, and simulation tools as described in NASA-STD-7009 [26] and EPRI TR-1025243 [13]. Plant modeling, analysis, and simulation tools serve an important role in the design of the plant safety-significant software and logic and are used to calculate the behavior of the entire nuclear power plant, including the safety-significant system that contains safety-significant software, in response to normal and abnormal transients. The output of plant modeling, analysis, and simulation tools help to determine instrumentation and control (I&C) system parameters and to define the functionality and specific features of the safety-significant software and logic. The output of these tools also help to verify and validate the functionality and specific features of the safety-significant software and logic. Plant modeling, analysis, and simulation tools include piping stress and flexibility analysis programs, steady-state thermal-hydraulic and pipe flow analysis programs, transient thermal-hydraulic analysis programs, finite element analysis programs, electrical power system computer programs, and plant simulation programs. Plant

modeling, analysis, and simulation tools might be considered as a software tool type in future regulatory guidance.

As shown in Table 2.2, the draft version of IEEE Std. 7-4.3.2 [6] does not contain an explicit plant modeling, analysis, and simulation tool type. The working group for the proposed draft of IEEE Std. 7-4.3.2 [6] considers plant modeling, analysis, and simulation tools, as described in NASA-STD-7009 [26] and EPRI TR-1025243 [13], as either Type III or Type IV V&V tools because they are used in either an “in-line” or “off-line” manner to support various software development activities. However, the descriptions of Type III and IV tools do not specifically mention plant modeling, analysis, or simulation tools. The proposed draft of IEEE Std. 7-4.3.2 [6] also does not address these types of tools when used to support the generation of a software requirements specification (SRS) or to support design decisions.

As shown in Table 2.2, the draft version of IEEE Std. 7-4.3.2 [6] uses the terms “in-line” and “off-line” to categorize Type IV and Type V V&V tools. The use of the term “in-line” in the draft version of IEEE Std. 7-4.3.2 [6] refers to V&V tools where the safety-related software or logic can be automatically or manually changed by the tool user based on the results of the V&V process. The use of the term “off-line” in the draft version of IEEE Std. 7-4.3.2 [6] refers to V&V tools that do not modify or manipulate the software or logic that is being tested. IEEE Std. 7-4.3.2 [6] “off-line” software tools produce test results that provide a tool user with information needed to validate that the software or logic is performing its required design functions. If the V&V activities identify the need for software or logic changes, then that information is provided as feedback to the software developer who uses a Type I software development tool to make the change.

The IEEE Std. 7-4.3.2 [6] treatment of “in-line” and “off-line” differs from the treatment of the same terms in IEC 61508 [25]. IEC 61508 [25] defines an “in-line” support tool as a software tool that can directly influence the safety-related system during its run time. IEC 61508 [25] defines an “off-line” support tool as a software tool that supports a phase of the software development life cycle that cannot directly influence the safety-related system during its run time. IEC 61508 [25] categorizes “off-line” support tools as Class T1, T2, or T3 depending on the function of the tool. Consistent definitions of “in-line” and “off-line” might be considered in future versions of IEEE and IEC standards to avoid confusion.

Tool categories or types, as defined in industry documents, are based on the tool functionality and the tool’s impact on the programming of the safety-significant system, software, or logic. Most industry documents only address single-purpose tools when defining tool types or categories. However, the proposed draft of IEEE Std. 7-4.3.2 [6] addresses multi-function V&V tools that have the ability to correct or change programming based on V&V results by defining a separate “in-line” V&V tool type. However, other multi-function tools are not covered by the tool types in the draft version of IEEE Std. 7-4.3.2 [6]. A better approach to categorizing multi-function tools might be considered that depends on the independence of the multiple functions.

The following might be considered in new regulatory guidance or to influence industry standards with respect to categorizing tools and defining tool types.

1. *Influencing the IEEE might be considered to expand the definition of Type I software development tools in the proposed draft version of IEEE Std. 7-4.3.2 [6] to include software development modeling and implementation tools.*

2. *Influencing the IEEE might be considered to expand the definition of Type III development process support tools in the proposed draft version of IEEE Std. 7-4.3.2 [6] to include QA tools and plant modeling, analysis, and simulation tools.*

3. *Influencing the IEEE might be considered to expand the number of tool types in the proposed draft version of IEEE Std. 7-4.3.2 [6] to include a management support tool type and to expand the definition of Type IV "in-line" V&V tools to include plant modeling, analysis, and simulation tools.*

4. *Influencing the IEC and IEEE might be considered to improve the consistency of the use of the terms "in-line" and "off-line" used in the draft version of IEEE Std. 7-4.3.2 [6] and in IEC 61508 [25].*

5. *Influencing the IEC might be considered to replace the types of tools identified in IEC 60880 [24] with assigned tool classes of non-classified, T1, T2, or T3 for consistency with IEC 61508 [25] or with the tool types identified in the draft version of IEEE Std. 7-4.3.2 [6] to maintain international consistency.*

6. *Influencing the IEEE might be considered to delete the "in-line" V&V tool type which defines a multi-function tool that performs both V&V and software development functions. Multi-function tools might be assigned to more than one category based on their functionality. If the tool functions are independent, the independent functions of the tool might be assigned different types and treated independently. However, if all tool functions are integrated and not independent, then the entire tool might be assigned to the type that has greatest impact on the safety-significant system, software, or logic.*

Use, adoption, or endorsement of software tool categories might be considered in future regulatory guidance so the rigor and intensity of software tool development, dedication, or qualification can be tailored based on the type of tool and the safety significance of the software or logic that is to be developed using the tools. A phased implementation of regulatory guidance that specifies software tool categories as described in Sections 2.2.1 and 2.2.2 of this report might be considered to minimize the impact on industry and to harmonize with other industry standards.

### 2.2.1 Phase 1

After the new version of IEEE Std. 7-4.3.2 [6] is published for use, phase 1 might consider new regulatory guidance or revisions to existing RGs that endorse the IEEE Std. 7-4.3.2 [6] tool types with some possible enhancements. The first enhancement that might be considered is expanding the definition of Type I tools to include software development implementation and modeling tools. The second enhancement that might be considered is expanding the definition of Type III tools to include QA tools and plant modeling, analysis, and simulation tools. The third enhancement that might be considered is expanding the definition of Type IV tools to

include plant modeling, analysis, and simulation tools. The fourth and final enhancement that might be considered is adding an additional tool type to cover management support tools. The following tool types would result from this expanded endorsement of the IEEE Std. 7-4.3.2 [6] tool types.

*Type I – Software Development Tools – Type I tools are used by programmers for generating software or logic to be loaded into a computer system or PDD. Type I tools generate object code or logic synthesis that will be executed within the target system hardware. Type I tools include transformation tools including compilers, linkers, loaders, tools that transform requirements into lower-level requirements, and auto-code generators. Type I tools also include software development modeling tools including diagram development tools, diagram translators, diagram analysis tools, requirements specification support tools, design specification support tools, specification construct modeling tools, software or system simulation tools, software or system prototyping tools, and human interface modeling tools.*

*Type II – Surveillance and Maintenance Tools – Type II tools are used for surveillance testing and changing configuration data of safety systems. Type II tools may generate output that includes the safety system response to tool use or modified configuration of the safety system.*

*Type III – Development Process Support Tools – Type III tools are used in an "off-line" manner to support various software development activities. They are considered "off-line" because they do not have a direct impact or interaction with the software or logic development processes. Type III tools cannot directly introduce errors into the safety-related software or logic; however, the output of Type III tools can result in inaccurate data to be used in the safety-related software or logic or inaccurate requirements specifications. Type III tools are used to create and maintain documentation, CM, and QA of the development process and are used to simulate, model, and analyze the safety system and nuclear power plant to support the development of system requirements. The output of a Type III tool is documentation or specific file or data sets for the safety system. Type III tools can also be used to maintain configuration control over Type I, II, IV, and V tools.*

*Type IV – In-Line V&V Tools – Type IV tools use software or logic input from the output of a Type I tool to perform or assist software or logic verification or validation activities. Type IV tools are considered "in-line" tools because the safety-related software or logic can be automatically or manually changed by the tool user based on the results of the V&V process. Type IV tools include plant modeling, analysis, and simulation tools that are used to model, simulate, and analyze the behavior of a subset or the entire nuclear power plant in response to normal and abnormal transients for the purpose of safety-related software or logic V&V. The output of a Type IV tool is adjusted or modified software or logic that has been corrected to meet the requirements programmed into the tool itself.*

*Type V – Off-Line V&V Tools – Type V tools use software or logic input from the output of a Type I tool to perform or assist software or logic verification or validation activities. However, they differ from Type IV tools in that they do not modify or manipulate the software or logic that is being tested. Type V tools produce test results which provide a tool user with information needed to validate that the software or logic is performing its required design functions. If the V&V activities identify the need for software or logic changes, then that information is provided as feedback to the software developer who uses a Type I tool to make the change.*

*Type VI – Management Support Tools. Type VI tools are resource and project tracking tools and do not generate output that can directly or indirectly contribute to the programming of the safety system. Type VI tools include cost and schedule estimating tools, tools that track project activity, project status analysis and reporting tools, and tools that define detailed work items (e.g., resources, personnel, deadlines).*

### 2.2.2 Phase 2

Phase 2 might consider endorsing the tool types based on the IEEE Std. 14102-2010 [19] CASE tool characteristics because of the expanded number of tool types and because the tool types are closely tied to the IEEE Std. 12207-2008 [10] software life cycle processes. These tool types are also sufficiently general to cover most new tools developed to automate, eliminate, or reduce activities and tasks performed in the software development life cycle processes. However, IEEE Std. 14102-2010 [19] does not recognize plant modeling, analysis, and simulation tools; so phase 2 regulatory guidance might consider augmenting the IEEE Std. 14102-2010 [19] tool types with an additional tool type that includes these types of tools. In Phase 2, tool types are categorized using a graduated scale based on the importance of the tool and the importance of the life cycle process in which the tool is used. The tool types cover all tools used in the development of safety-significant software or logic. The following tool types would result from this expanded endorsement of the IEEE Std. 14102-2010 [19] tool categories.

*Type I – Construction Tools. These tools generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools include compilers, linkers, loaders, automated report generators, syntax-directed editors, debugging tools, and auto-code generators.*

*Type II – Software Development Modeling Tools. These tools generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools include diagram development tools, diagram translators, diagram analysis tools, requirements specification support tools, design specification support tools, specification construct modeling tools, software or system simulation tools, software or system prototyping tools, and human interface modeling tools.*

*Type III – Plant Modeling, Analysis, and Simulation Tools. These tools generate output that can indirectly contribute to the programming of the safety-significant*

*system and support the design of the safety-significant system, software or logic. These types of tools are used to model, analyze, and simulate the behavior of a subset or the entire nuclear power plant, including the safety-significant systems that contain safety-significant software or logic, in response to normal and abnormal transients. The output of these tools help to define the functionality and specific features of the safety-significant software. These types of tools include piping stress and flexibility analysis programs, steady-state thermal-hydraulic and pipe flow analysis programs, transient thermal-hydraulic analysis programs, finite element analysis programs, electrical power system computer programs, and plant simulation programs.*

*Type IV – Implementation Tools. These tools generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools include requirement elicitation support tools, requirement analysis support tools, requirements specification support tools, requirement V&V support tools, and requirement management support tools.*

*Type V –Maintenance Tools. These tools generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools support problem understanding and resolution, localization of the portion of the software requiring modification, identification of potential consequences of modifications, reverse engineering of data from source code or logic, source code or logic restructuring, and source code or logic translation.*

*Type VI – V&V Tools. These tools do not generate output that can directly or indirectly contribute to the programming of the safety-significant system but support the test or verification of the design or programming, where errors in the tool can fail to reveal defects but cannot directly create errors in the programming. These types of tools include specification traceability analysis tools, specification analysis tools, source code or logic analysis tools, test harness generators, test coverage measurement tools, static analysis tools, test case generators, failure and defect analysis tools, and tools that automate the execution and traceability of testing activities and data.*

*Type VII – Documentation Tools. These tools do not generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools include tools that automate, eliminate, or reduce the effort associated with the software or logic development documentation including text editing tools; graphical editing tools; forms-based editing tools; desktop publishing tools; and tools that accept, store, and retrieve specifications of the content, format, and layout of textual and graphical data and extract and produce data in compliance with a specification.*

*Type VIII – CM Tools. These tools do not generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools control access to data elements, track modifications, define and*

*manage multiple versions of a system that may share common components, generate reports defining the history, contents, and status of the various configuration items being managed, support user definition of steps required to create a version of the software for release, and to automatically place data elements in secondary storage for subsequent retrieval.*

*Type IX – QA Tools. These tools do not generate output that can directly or indirectly contribute to the programming of the safety-significant system. These types of tools support user entry of quality data; analyze quality data; generate information to support quality management; and support risk identification, estimation, impact assessment, monitoring, and controlling.*

*Type X – Management Support Tools. These tools are resource and project tracking tools and do not generate output that can directly or indirectly contribute to the programming of the safety-significant system including cost and schedule estimating tools, tools that track project activity, project status analysis and reporting tools, and tools that define detailed work items (e.g., resources, personnel, deadlines).*

## 2.3 Software and Software Tool Integrity Levels

This section includes a technical basis that might be considered for guidance on assigning software and software tool integrity levels. Integrity levels quantify the complexity, risk, and safety level of a system, hardware, software, or software tool based on its safety significance. The technical basis considers current regulatory guidance and regulations associated with assigning an integrity level or a risk-informed safety class (RISC) to safety-significant software and identifies an approach for implementing changes that might be considered based on the best industry practices.

The integrity level or RISC of software tools considers their use in or with safety-significant software or logic, with a degree of rigor and level of detail appropriate to the safety significance of the software or logic, and its vulnerabilities to tool-related problems in all life cycle processes. Assigning a software tool integrity level or RISC might consider a method consistent with the graded approach discussed in 10 CFR 21.3, EPRI TR-102260 [11], and EPRI TR-1025243 [13]. A graded approach might also be considered for consistency with the importance to safety prescribed in Criterion II of 10 CFR Part 50, Appendix B, which states:

> "The quality assurance program shall provide control over activities affecting the quality of the identified structures, systems, and components, to an extent consistent with their importance to safety."

### 2.3.1 Software Integrity Levels

The technical basis for assigning software integrity levels allows the software development processes to be tailored based on the importance of the software or logic to the safety of the system, whether evaluated deterministically or through a risk-informed probabilistic risk analysis. The software integrity level is used for determining a tool qualification level (TQL) that

is used to determine the appropriate degree of rigor and intensity to be applied in the process of verifying the tools' suitability, quality, and reliability. Tool qualification might be considered as the basis for licensee or applicant acceptance of tools acquired as CGIs and dedicated as prescribed in EPRI TR-1025243 [13]; tools developed using another high-quality life cycle approach such as that prescribed in RTCA DO-330 [14]; or tools dedicated using the RTCA DO-330 [14] process, conducted under appropriate Appendix B QA controls as a means of dedication (e.g., for COTS software tools).

IEEE Std. 603-1991 [27] and RG 1.152 discuss the safety classification of software and require that equipment that performs both safety and nonsafety functions on the same computer be classified as part of the safety system. Specifically, IEEE Std. 603-1991 [27] states:

> "Equipment that is used for both safety and nonsafety functions shall be classified as part of the safety systems."

RG 1.152 states:

> "… any software providing nonsafety functions that resides on a computer providing a safety function must be classified as a part of the safety system."

The IEEE Std. 603-1991 [27] and RG 1.152 conservative classification of equipment providing nonsafety functions as part of the safety system might be reconsidered in future regulatory guidance. Software providing nonsafety functions that resides on a computer providing a safety function might be considered as nonsafety if the functions can be demonstrated to be sufficiently independent.

RG 1.168 endorses IEEE Std. 1012-2004 [21] for software V&V requirements, which uses a four-level integrity schema to define the minimum V&V requirements. IEEE Std. 1012-2012 [5] is the latest version of the standard but this version is not yet endorsed by the NRC. Both IEEE Std. 1012-2004 [21] and IEEE Std. 1012-2012 [5] contain a method for assigning integrity levels to software based on the four-level schema. However, other integrity schemas are acceptable. For any selected integrity schema, the selected integrity levels are mapped into the IEEE Std. 1012-2004 [21] or IEEE Std. 1012-2012 [5] four-level schema to demonstrate that the minimum V&V requirements are satisfied.

RG 1.170 endorses IEEE Std. 829-2008 [17] for software and system test documentation requirements. IEEE Std. 829-2008 [17] also uses four integrity levels to describe the importance of the software and software-based system. The integrity levels define the minimum recommended testing tasks to be performed. However, other integrity schemas are acceptable. For any selected integrity schema, the selected integrity levels are mapped into the IEEE Std. 829-2008 [17] four-level schema to demonstrate that the minimum testing requirements are satisfied.

RG 1.168, RG 1.170, and all versions of IEEE Std. 7-4.3.2 [6][8][12] assign all software used in nuclear power plant safety systems to integrity level 4 or equivalent. Not using a graded approach for assigning software integrity levels prevents software development activities from being tailored based on the importance to safety of the various elements of the safety-significant

system. Consistency with Criterion II of 10 CFR Part 50, Appendix B, requires that any graded approach for assigning software integrity levels must be custom tailored to the integrity level schema being used by the developer.

With respect to software classification and software integrity levels, the following might be considered in future regulatory guidance and to influence industry standards.

1. *Future regulatory guidance might consider endorsing the expanded classification system of NITSL-SQA-2005-02 [28] for commercial nuclear power to recognize that not all safety-related software has an equal impact on safety-related SSC functionality.*

2. *A revision to RG 1.168 and RG 1.170 might be considered to endorse the IEEE Std. 1012-2004 [21] or IEEE Std. 1012-2012 [5] graduated integrity levels instead of assigning integrity level 4 to all software because the potential consequences of failures of safety-significant software are not always catastrophic and can vary from minor to catastrophic depending on the software functionality.*

3. *A revision to RG 1.168 and RG 1.170 might be considered to recognize the RISC-based categorization of SSCs in 10 CFR 50.69.*

4. *Influencing the IEEE might be considered to change the IEEE Std. 7-4.3.2 [6] requirement that software V&V satisfy the IEEE Std. 1012-2012 [5] requirements for the highest integrity level (i.e., integrity level 4) so the V&V effort can be tailored based on the software's importance to safety as described in 10 CFR Part 50, Appendix B, Criterion II.*

5. *Future regulatory guidance might consider a risk-informed approach that decides and assesses the required software integrity level at a system level, on the basis of the system integrity level and the level of risk associated with the use of the software in the system.*

6. *Future regulatory guidance might consider that partitioning, diverse software, and safety monitoring are acceptable justifications for changing the integrity level of the software component.*

7. *Future regulatory guidance might consider that where a safety-related system implements both safety and non-safety functions, all software must be treated as safety-related unless the implementation of the safety-related and nonsafety-related functions can be demonstrated to be sufficiently independent.*

### 2.3.2 Software Tool Integrity Levels

The technical basis for assigning software tool integrity levels allows the software tool development processes to be tailored based on the importance of the software tool to the safety of the software and system. The safety-significance of the software tool can be expressed in terms of a software tool integrity level or a TQL. Analogous to the software integrity level, a software tool integrity level might be used to determine the minimum V&V requirements and

recommended testing tasks for software tools developed under a 10 CFR Part 50, Appendix B, QA program. A software tool integrity level is not necessary for software tools that are acquired as CGIs and dedicated or for software tools developed using the high-quality life cycle approach of RTCA DO-330 [14] because the safety-significance of the tool and the rigor and intensity of development and qualification is determined by the TQL based on the software integrity level or RISC and the type of tool as described in Sections 2.7.1 and 2.7.2 of this report.

RG 1.168 endorses IEEE Std. 1012-2004 [21] for software V&V requirements, which could be applied as software tool V&V requirements using the software tool integrity level in place of the software integrity level. IEEE Std. 1012-2004 [21] and IEEE Std. 1012-2012 [5] both use a four integrity level schema to define the minimum V&V requirements; however IEEE Std. 1012-2012 [5] is not yet endorsed by regulatory guides. Assigning a software tool integrity level and satisfying the minimum V&V requirements of IEEE Std. 1012-2004 [21] or IEEE Std. 1012-2012 [5] might be considered for software tools developed under a 10 CFR Part 50, Appendix B, QA program.

RG 1.170 endorses IEEE Std. 829-2008 [17] for software and system test documentation requirements. IEEE Std. 829-2008 [17] also uses four integrity levels to describe the importance of the software and software-based system. The integrity levels define the minimum recommended testing tasks to be performed and might be considered for application to software tool testing using the software tool integrity level in place of the software integrity level. Assigning a software tool integrity level and satisfying the minimum recommended testing tasks of IEEE Std. 829-2008 [17] might be considered for software tools developed under a 10 CFR Part 50, Appendix B, QA program.

The four-level schema used in IEEE Std. 1012-2004 [21], IEEE Std. 1012-2012 [5], and IEEE Std. 829-2008 [17] is not required. Other integrity schemas are acceptable. For any selected integrity schema, the selected integrity levels shall be mapped into the IEEE Std. 1012-2004 [21], IEEE Std. 1012-2012 [5], or IEEE Std. 829-2008 [17] four-level schema to demonstrate that the minimum V&V or software testing requirements are satisfied.

IEEE Std. 1012-2004 [21], IEEE Std. 1012-2012 [5], and IEEE Std. 829-2008 [17] only contain a single requirement for assigning integrity levels to software tools. Tools that insert or translate code (e.g., compilers or auto-code generators) or logic are assigned the same integrity level as the integrity level assigned to the software element that the tool affects. IEEE Std. 1012-2004 [21], IEEE Std. 1012-2012 [5], and IEEE Std. 829-2008 [17] do not provide guidance for assigning software tool integrity levels for tools that do not insert or translate code or address tools that are used to generate logic used in other PDDs.

With respect to software tool integrity levels, the following might be considered in future regulatory guidance and to influence industry standards to assign a software tool integrity level to define the minimum V&V requirements and minimum recommended testing tasks for software tools developed under a 10 CFR Part 50, Appendix B, QA program. The technical basis implements a graded approach for assigning software tool integrity levels based on the impact of the software tool on the software and system. Any graded approach for assigning software

tool integrity levels must be custom tailored for the integrity level schema being used by the developer.

1. *Future regulatory guidance might consider that software development implementation, modeling, and construction tools that insert or translate requirements, code (e.g., compilers and auto-code generators), or logic should be assigned the same integrity level as the integrity level assigned to the software element that the tool affects.*

2. *Future regulatory guidance might consider that V&V, CM, and maintenance tools should be assigned a lower integrity level than software development implementation, modeling, and construction tools.*

3. *Future regulatory guidance might consider that QA and documentation tools should be assigned a lower integrity level than V&V, CM, and maintenance tools.*

4. *Future regulatory guidance might consider that management support tools should be classified as nonsafety-related since these tools have no safety significance.*

5. *Future regulatory guidance might consider that if the results of a software tool are independently verified each time the tool is used, the tool should be classified as nonsafety-related because the tool is not relied upon as the sole basis for the calculation.*

## 2.4 Software Tool Planning

This section includes a technical basis that might be considered for future regulatory guidance for software tool planning based on industry practice. Current regulatory guidance for software tools is limited and does not include guidance for software tool planning or for the content of software development planning documents.

RG 1.152 states that conformance with the requirements of IEEE Std. 7-4.3.2-2003 [8] is a method acceptable for satisfying regulations with respect to high functional reliability and design requirements for computers used in the safety systems of nuclear power plants. RG 1.168, RG 1.169, and RG 1.170 state that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8].

IEEE Std. 7-4.3.2-2003 [8] contains computer-specific requirements to supplement the criteria and requirements of IEEE Std. 603-1998 [29]. However, neither IEEE Std. 7-4.3.2-2003 [8] nor IEEE Std. 603-1998 [29] contain software tool planning requirements. The proposed draft version of IEEE Std. 7-4.3.2 [6] contains requirements for tool selection and use but does not contain requirements for software tool planning.

BTP 7-14 discusses software tool planning in the context of the review of planning document content. BTP 7-14 defines software tools as resource characteristics of planning documents. BTP 7-14 states that planning documents are required to describe tools used, define the process by which tools are selected, and define the tool operational environment and any special controls necessary to execute the tools. BTP 7-14 states that if the output of any tool

cannot be proven to be correct, such as may occur if the tool produces machine language software code, the tool itself should be developed or dedicated as safety-related, with all the attendant requirements.

BTP 7-14 identifies 12 planning documents that may contain software development information to be reviewed:

1. Software Management Plan
2. Software Development Plan
3. Software QA Plan
4. Software Integration Plan
5. Software Installation Plan
6. Software Maintenance Plan
7. Software Training Plan
8. Software Operations Plan
9. Software Safety Plan
10. Software V&V Plan
11. Software CM Plan
12. Software Test Plan

BTP 7-14 states that all plans should include a description of the tools to be used and should identify suitable tools to carry out the activities of the specific life cycle process. BTP 7-14 also states that the software development plan, software integration plan, software installation plan, and software maintenance plan should require tools in these life cycle phases to be qualified with a degree of rigor and level of detail appropriate to the safety significance of the software developed using the tool. BTP 7-14 also states that tools that produce results that cannot be verified or that are not compatible with safety requirements should be prohibited, unless analysis shows that the alternative would be less safe.

With respect to software tool planning, the following might be considered in future regulatory guidance to augment the planning guidance in BTP 7-14. The suggested changes to BTP 7-14 listed below might alternatively be implemented by means of an additional BTP specifically applicable to software tools that could be referenced in BTP 7-14 and related guidance.

1. *Changes to BTP 7-14 might be considered to expand the guidance for tools that cannot be proven to be correct, such as may occur if the tool produces machine language software code. In addition to the guidance that such a tool be developed or dedicated as safety-related, BTP 7-14 might consider the acceptability of the tool being developed using the RTCA DO-330 [14] high-quality software tool development process.*

2. *Changes to BTP 7-14 might be considered to state that the software verification plan should describe the functionality of any verification tool, including expectations and limitations on how it is to be used (e.g., domain, language, process).*

3. *Changes to BTP 7-14 might be considered to state that the software verification plan should require V&V tools to be qualified or commercially dedicated since the tools may fail to detect defects in the safety-significant software.*

4. *Changes to BTP 7-14 might be considered to state that the software operating plan should require that tools used to operate safety-significant software should be qualified.*

5. *Future regulatory guidance might consider that the entire project should be planned in advance to select an integrated set of tools.*

6. *Future regulatory guidance might consider that if approval is sought for use of the tools in combination, the sequence of the options should be examined and specified in the appropriate plan.*

7. *Future regulatory guidance might consider that if optional features of software tools, especially compilers and auto-code generators, are chosen for use in a project, the effects of the options should be examined and specified in the appropriate plan.*

8. *Future regulatory guidance might consider that the software planning process should provide and define the means to detect object code or logic synthesis that is not directly traceable to the source code or logic, for example, initialization, built-in error detection, or exception handling, and to ensure verification coverage.*

9. *Future regulatory guidance might consider that if a new compiler, linkage editor, or loader version is introduced, or compiler options are changed during the software life cycle, previous tests and coverage analyses may no longer be valid and verification planning should provide a means of re-verification.*

## 2.5 Software Tool Selection and Use

This section includes a technical basis that might be considered for future regulatory guidance for software tool selection and use based on industry practice. Current regulatory guidance for software tools is limited and does not include guidance for software tool selection and use.

RG 1.152 states that conformance with the requirements of IEEE Std. 7-4.3.2-2003 [8] is a method that is deemed acceptable for satisfying regulations with respect to high functional reliability and design requirements for computers used in the safety systems of nuclear power plants. Also, RG 1.168, RG 1.169, and RG 1.170 state that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8].

Neither IEEE Std. 7-4.3.2-2003 [8] nor IEEE Std. 7-4.3.2-2010 [12] address the selection of software tools or acceptance criteria for compilers, operating systems, and libraries. The proposed draft version of IEEE Std. 7-4.3.2 [6] provides more specific criteria on the use of software tools but is not yet recognized by any regulatory guide. Specifically, the proposed draft version of IEEE Std. 7-4.3.2 [6] requires that the benefits and risks associated with selecting software tools be taken into consideration, that the selected tools limit the opportunity for making errors and introducing faults while maximizing the opportunity for detecting faults in the

resulting safety-related PDD, and that tools and their output not be used outside their documented functionality or limits of application without prior justification. IEEE Std. 7-4.3.2-2003 [8] and the proposed draft version of IEEE 7-4.3.2 [6] also require that a software tool be used in a manner such that defects not detected by the tool will be detected by V&V activities. Neither IEEE Std. 7-4.3.2-2003 [8] or the proposed draft version of IEEE 7-4.3.2 [6] explain how software tools can be used in different manners such that, when used in one manner, defects not detected by the tool will be detected by V&V activities and when used in another manner, defects not detected by the tool will **not** be detected by V&V activities. Also, the use of the word "will" in the requirement is a strong statement of compliance. Neither IEEE Std. 7-4.3.2-2003 [8] or the proposed draft version of IEEE 7-4.3.2 [6] explain how tools can be used in such a manner as to guarantee that defects not detected by the tool will be detected by V&V activities.

The purpose of IEEE Std. 14102-2010 [19] is to assist organizations in the proper evaluation and selection of CASE tools. IEEE Std. 14102-2010 [19] contains few requirements and instead describes a "best practices" methodology for selecting CASE tools rather than a requirements-based methodology. The sections of IEEE Std. 14102-2010 [19] describing the characteristics and sub-characteristics of the various tool types can be shaped into guidance for evaluation of tools that have already been selected. Furthermore, by recognizing this standard as an acceptable means of selecting reliable and safe tools, tool planners and users can be directed toward an easily evaluated set of documents describing the software tool selection process, relieving reviewers of much of the burden of sorting through non-standard documentation of the tools used.

The use of compiler optimization in the development of safety-significant software is a complex issue and not well addressed in IEEE standards. None of the versions of IEEE Std. 7-4.3.2 [6][8][12] address compiler optimization; however, the RTCA and IEC address its use. RTCA DO-178C [9] states that the use of optimization is acceptable provided proper planning and test case coverage determine the impact of the optimization. IEC 60880 [24] states that compiler optimization should be avoided unless absolutely necessary to meet performance requirements due to constraints in hardware speed and storage limits. IEC 60880 [24] states that optimization shall not be used if it produces object code that is excessively difficult to understand, debug, test, and validate. If optimization is used, IEC 60880 [24] requires that tests, verification, and validation be performed on the optimized code. IEC 60880 [24] also states that, in lieu of optimization, the design should consider the use of assembly language and changing the hardware platform.

With respect to software tool selection and use, the following might be considered in future regulatory guidance to augment the software tool selection guidance in the draft version of IEEE Std. 7-4.3.2 [6] and to provide guidance on the use of compiler optimization based on industry practice.

1. *Future regulatory guidance might consider endorsing IEEE Std. 14102-2010 [19] for evaluating and selecting CASE tools.*

2. *Future regulatory guidance might consider that the selection of software tools should be justified, known tool problems and limitations should be assessed, and issues that can adversely affect software or logic should be addressed.*

3. *Future regulatory guidance might consider that well established tools for PDD development should be used and tools that are still at the research stage should not be used.*

4. *Future regulatory guidance might consider that a version history may provide assurance of maturity of a tool, and a version history may provide a record of the errors and ambiguities that should be taken into account when the tool is used in a new development environment.*

5. *Future regulatory guidance might consider that the selection of tools should consider the degree to which the tool supports the production of software or logic with the required properties, the clarity of the operation and functionality of the tool, and the correctness and repeatability of the output of the tool.*

6. *Future regulatory guidance might consider that the use of tools or combinations of tools and parts of the PDD development environment should be chosen such that an error introduced by one part would be detected by another. An acceptable environment is produced when both parts are consistently used together.*

7. *Future regulatory guidance might consider that tools used in the software development process should be consistent across all the sub-phases of the software life cycle, should remain available throughout the software's service life, should be compatible with the system and hardware development phases, and should be selected as an integral set so they work co-operatively such that the output from one tool have suitable content and format for automatic input to a subsequent tool, thus minimizing the possibility of introducing human error in the reworking of intermediate results.*

8. *Future regulatory guidance might consider that the safety-significant system hardware may impose compatibility constraints on software tools (e.g., a processor emulator needs to be an accurate model of the real processor electronics).*

9. *Future regulatory guidance might consider that the competence of the users of the selected tools should be a consideration when selecting software tools.*

10. *Future regulatory guidance might consider that a well-known tool with extensive operating experience is preferable to an untried tool with little or no operating experience as long as the well-known tool meets other selection criteria.*

11. *Future regulatory guidance might consider that independent V&V should use or develop independent test and analysis tools separate from the developer's tools.*

12. *Future regulatory guidance might consider that different tools should be used on redundant systems to reduce the possibility of CCF.*

13. *Influencing the IEEE might be considered to clarify the requirement of the draft version of IEEE Std. 7-4.3.2 [6] that a software tool shall be used in a manner such that defects not detected by the tool will be detected by V&V activities.*

14. *Future regulatory guidance might consider providing guidance on how software tools can be used in different manners such that, in one manner, defects not detected by the tool will be detected by V&V activities, and in another manner, defects not detected by the tool will **not** be detected by V&V activities.*

## 2.6 Software Tool Documentation

This section includes a technical basis that might be considered for future regulatory guidance for software tool documentation based on industry practice that meets regulatory requirements. Current regulatory guidance for software tools is limited but does include some guidance for documenting the use, suitability, or characteristics of software tools.

RG 1.152 endorses IEEE Std. 7-4.3.2-2003 [8] as an acceptable method of satisfying regulations with respect to high functional reliability and design requirements for computers used in the safety systems of nuclear power plants. RG 1.168, RG 1.169, and RG 1.170 all state that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8], as endorsed by RG 1.152.

IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 7-4.3.2-2010 [12], and the proposed draft version of IEEE Std. 7-4.3.2 [6] require software tools to be controlled under CM and provides requirements for determining the suitability of software tools; however, neither IEEE Std. 7-4.3.2-2003 [8] nor IEEE Std. 7-4.3.2-2010 [12] contain requirements for documenting the use, suitability, or characteristics of software tools. The proposed draft of IEEE Std. 7-4.3.2 [6] requires that the intended functionality and limitations of applicability for all software tools be identified and documented. The proposed draft of IEEE Std. 7-4.3.2 [6] also requires that measures be put into place to ensure that only approved and documented software tools are used to support the development of a safety-related PDD; however the specifics of software tool documentation are not covered in the proposed draft of IEEE Std. 7-4.3.2 [6].

The only regulatory guidance applicable to software tool documentation is contained in RG 1.170, RG 1.171, and RG 1.172 and involve guidance for software testing activities. RG 1.170 endorses IEEE Std. 829-2008 [17] which contains requirements for manual or automated documentation of testing activities. IEEE Std. 829-2008 [17] requires that various test plans contain a summary and description of required testing tools including information regarding acquisition, training, support, and qualification for each tool. However, the scope of IEEE Std. 829-2008 [17] is restricted to software tool documentation used for software testing activities and IEEE Std. 829-2008 [17] does not apply to the documentation of software tools used in other software development processes and activities.

RG 1.171 states that the documentation to support software unit testing should include tools needed for the accomplishment of unit testing; however, RG 1.171 does not provide guidance on the specific content of the documentation. RG 1.171 endorses IEEE Std. 1008-1987 [30] as

an acceptable approach for meeting regulatory requirements for unit testing of safety system software. However, IEEE Std. 1008-1987 [30] does not contain requirements for documenting information regarding acquisition, training, support, or qualification of software tools used for software unit testing analogous to the software tool documentation requirements of IEEE Std. 829-2008 [17] for software testing activities.

RG 1.172 states that if requirements are generated using specification or representation tools, traceability should be maintained between these representations and the natural language descriptions of the software requirements that are derived from systems requirements and system safety analyses. RG 1.172 endorses IEEE Std. 830-1998 [23] which is a recommended practice for writing a SRS. IEEE Std. 830-1998 [23] does not acknowledge the use of any software tool for preparing an SRS and does not recommend documentation of any software tool used for that purpose.

With respect to software tool documentation, the following might be considered in future regulatory guidance and to influence industry standards so that software tool documentation requirements might be expanded to cover all software development tools and not just software tools used in software testing activities.

1. *Influencing the IEEE might be considered to revise IEEE Std. 830-1998 [23] to recommend that the software project plan include documentation of information regarding development or acquisition, training, support, or qualification of any software tool used to develop a SRS.*

2. *Influencing the IEEE might be considered to revise IEEE Std. 1008-1987 [30] to require that the unit testing plan include documentation of information regarding development or acquisition, training, support, or qualification of any software tool used for software unit testing.*

3. *Future regulatory guidance might consider that documentation associated with CASE tools and data in these tools should include:*

    a. *Identification and version number of the software tool*
    b. *Configuration of the software tool including tool parameters*
    c. *Rationale for selecting the software tool and compatibility with other tools*
    d. *Software development life cycle process in which the tool is used*
    e. *Details on the suitability of the software tool*
    f. *Sequence in which the tool is used, if applicable*
    g. *Environment in which the software tool is executed*
    h. *Reference to user or operator manuals or instructions for the software tool*
    i. *Maximum integrity level of the SSC that can be violated if the software tool is malfunctioning and producing corresponding erroneous output*
    j. *Details on tool acquisition*
    k. *Details on tool training*
    l. *Details about vendor support, licensing, and data rights, including whether the item is currently supported by the vendor, whether it is expected to be supported at the time*

*of delivery, whether licenses will be assigned to the support organization, and the terms of such licenses*

m. *Methods and results of software tool qualification and the TQL, if required*

## 2.7 Software Tool Development, Qualification, and Dedication

This section includes a technical basis that might be considered for future regulatory guidance and used by licensees, applicants, and third-party contractors or vendors for developing, dedicating, and qualifying tools, whether acceptance of tools is based on the tool being developed under a 10 CFR Part 50, Appendix B, QA program; based on the dedication of commercial-grade tools not developed under an Appendix B QA program; based on the tool being developed using another high-quality life cycle approach such as RTCA DO-330 [14]; or based on the dedication of tools using the RTCA DO-330 [14] process, conducted under appropriate Appendix B QA controls.

Basic components are defined in 10 CFR 21.3, in part, as items designed and manufactured under a QA program complying with 10 CFR Part 50, Appendix B, or CGIs that have successfully completed the dedication process, which itself must be performed under the applicable elements of an Appendix B-compliant QA program. The dedication process in general involves a technical evaluation phase, in which the dedication process is developed and defined including critical characteristic verification methods and acceptance criteria, and an acceptance phase, in which the approved verification methods are implemented.

RTCA DO-330 [14] presents an entire qualification and development process for software tools that recommends the application of a life cycle process with associated objectives and activities to meet those objectives. The RTCA DO-330 [14] process with appropriate Appendix B QA oversight or controls might be endorsed as an acceptable method of dedication for software tools with safety significance requiring that level of rigor. Thus, a software tool designed and developed using RTCA DO-330 [14] could be used as or with a basic component. Tool qualification using the RTCA DO-330 [14] process might therefore be considered as a basis for licensee or applicant acceptance of a tool regardless of the methods used to develop the tool.

RTCA DO-330 [14] also addresses COTS software tool qualification with tool qualification activities split between the tool developer and the tool user. RTCA DO-330 [14] further provides guidance on tool qualification related to the use of multi-function tools, previously qualified tools, service history, exhaustive input testing, formal methods, and dissimilar tools for achieving compliance with objectives of RTCA DO-330 [14]. The objectives and activities for COTS tool qualification require considerable effort on the part of the COTS vendor. Since the tool was developed by a vendor independent of any specific application and without knowledge of what TQL would be assigned to the tool, it is unlikely that the developer would have performed the required activities or produced the required data. Thus, a more practical approach to COTS tool qualification might be considered that does not rely on vendor participation and generation of qualification data beyond what is in the user's manual. COTS tool qualification should concentrate on the use of service history, exhaustive input testing, formal verification methods, or dissimilar tools.

RG 1.152 endorses IEEE Std. 7-4.3.2-2003 [8] as an acceptable method of satisfying regulations with respect to high functional reliability and design requirements for computers used in the safety systems of nuclear power plants. RG 1.168, RG 1.169, and RG 1.170 all state that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8], as endorsed by RG 1.152. However, the provisions of RG 1.168 only apply to software tools when V&V activities cannot detect flaws in the software produced by the tools; otherwise, the V&V tasks of witnessing, reviewing, and testing of the software tool are not required.

IEEE Std. 7-4.3.2-2003 [8] requires that a test tool validation program be developed to provide confidence that the necessary features of the software tool function as required and that tool operating experience may be used to provide additional confidence in the suitability of a tool, particularly when evaluating the potential for undetected defects. IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 7-4.3.2-2010 [12], and the proposed draft version of IEEE Std. 7-4.3.2 [6] state that validation of a software tool is not necessary if the tool is 100% tested.

IEEE Std. 7-4.3.2-2010 [12] requires that the output of the software tool be verified and validated to the same integrity level as the safety-related software, that the software tool be developed using the same or an equivalent high quality life cycle process as required for the software upon which the tool is being used, or that the software tool be commercially dedicated. The proposed draft version of IEEE Std. 7-4.3.2 [6] does not include an option that a software tool can be commercially dedicated and instead requires that the tool be developed using a quality life cycle process commensurate with the criticality of the safety functions performed by the end product. The high-quality life cycle process development option in IEEE Std. 7-4.3.2-2010 [12] and the quality life cycle process commensurate with the criticality of the safety functions performed by the end product option in the proposed draft version of IEEE Std. 7-4.3.2 [6] are not usually viable for COTS tools since COTS tools are typically developed for widespread use without regard for their application (i.e., nuclear safety-related software development).

The proposed draft version of IEEE Std. 7-4.3.2 [6] also requires that software tools be assessed in a manner consistent with the category of tool, and the potential that the tool has to introduce faults into the safety-related PDD. The proposed draft version of IEEE Std. 7-4.3.2 [6] also states that the assessment process for software tools should take into account experience from prior use including the experience of the developers as well as experience gained from the processes in which the tools are used.

IEC 60880 [24] acknowledges that automation increases the amount of testing that can be performed in a given period. IEC 60880 [24] states that automated validation tools that generate test data, transport or transform test data and test results, and evaluate test results should record a complete test log. Further, the tools should be appropriate to simulate the behavior of the programming on the target system, and the tools should be appropriate to verify that the right software or logic is loaded correctly on the target system. IEC 60880 [24] does not define the details of what constitutes a complete test log; however, the requirement can be

interpreted to mean that logs must be a thorough, entire, and full description of all tests performed by automated validation tools.

With respect to software tool development, qualification, and dedication, the following might be considered in future regulatory guidance to supplement the software tool assessment requirements of the proposed draft version of IEEE Std. 7-4.3.2 [6] and to influence industry standards.

1. *Future regulatory guidance might consider that automated validation tools that generate test data; transport, or transform test data and test results; and evaluate test results should record a thorough, entire, and full test log. This is applicable to module tests as well as to plant simulations.*

2. *Future regulatory guidance might consider that tools should be verified and assessed consistent with the type of tool, the potential of the tool to introduce faults or fail to make the user aware of existing faults, and the extent to which the tool may affect redundant elements of a system or diverse systems. Examples of situations that can affect the degree of verification and assessment needed include, for example:*

    a. *tools that have the ability to introduce faults need to be verified to a greater degree than tools that are demonstrated to not have that capability,*
    b. *tools that can fail to make the user aware of existing faults need to be verified to a greater degree than tools that do not have that capability,*
    c. *verification is not necessary for tools when the output of the tool is systematically and independently verified, and*
    d. *less rigor in tool verification may be accepted if there is mitigation of any potential tool faults (e.g. by process diversity or system design).*

3. *Future regulatory guidance might consider that when automatically generated code is safety-significant or resides within an unprotected partition with safety-significant code, the automatically generated code should be subject to the same inspection, analysis, and testing as hand-generated code.*

4. *Future regulatory guidance might consider that software tool diversity can be demonstrated by showing that:*

    a. *each tool was obtained from a different supplier (e.g., one tool could be developed and the other tool could be purchased off the shelf),*
    b. *the tools have different input or output languages, or*
    c. *the tools have dissimilar requirements and design processes.*

5. *Future regulatory guidance might consider that each new version of a software tool should be qualified and that qualification may rely on evidence provided for an earlier version provided that the functional differences will not affect tool compatibility with the rest of the toolset and the new version is unlikely to contain significant, new, unknown faults based on:*

a. *clear identification of the changes made,*
b. *an analysis of the V&V actions performed on the new version, and*
c. *any existing operational experience from other users that is relevant to the new version.*

6. *Future regulatory guidance might consider that translators should be thoroughly tested. If the translator used is not thoroughly tested, additional analysis and verification, manual or using another tool, should demonstrate that the translator is correct.*

7. *Future regulatory guidance might consider that when software tools are modified or upgraded, the change should be subject to an impact assessment and the level of rigor applied in justifying and executing the change should be pre-determined and based upon the integrity level or qualification level of the tool, the type of tool, and the integrity level of the system or function on which the software tool is used.*

8. *Influencing the IEEE might be considered to modify the IEEE Std. 7-4.3.2 [6] requirement that a software tool does not have to be verified if it is 100% tested. An alternate requirement that might be considered is that verification of software tools is not necessary when the output of the tool is systematically (i.e., every time the tool is used) and independently (i.e., by someone other than the developer) verified.*

9. *Future regulatory guidance might consider that an independent V&V team should execute qualification tests on software tools shared with the development environment to ensure that the common tools do not mask errors in the programming being analyzed and tested.*

10. *Influencing the IEEE might be considered to change the IEEE Std. 1012-2012 [5] position that off-the-shelf tools with an extensive history of use do not need to be qualified but the input data for these tools should be verified. History of use should not be solely relied upon to assure that off-the-shelf tools are suitable for use in independent V&V activities. History of use should be augmented with testing, source evaluation, or surveys of the tool provider development process.*

11. *Influencing the IEEE or future regulatory guidance might consider extending the IEEE Std. 7-4.3.2 [6] CCF design criterion to software tools by providing a requirement that states a tool should be sufficiently verified and validated to ensure that a defect in the tool does not cause a fault that can result in a CCF in the safety-significant software or logic if a single software tool is used during several phases of the PDD development life cycle or a single software tool is used in the same development life cycle for software or logic in multiple safety-significant systems or the safety-significant system's internal redundancies.*

12. *Future regulatory guidance might consider that the qualification process of a tool should take into account experience from prior use, including experience of the developers and experience gained from the processes in which the tools are used.*

13. *Future regulatory guidance might consider that V&V should be done using different techniques and tools from those used to develop the software to ensure technical independence and diversity.*

14. *Future regulatory guidance might consider that changes to an application or development environment should be identified, analyzed, and re-verified and that if a new development environment uses software tools, the tool may need to be qualified.*

15. *Future regulatory guidance might consider that the impact of using a different auto-code generator or a different set of auto-code generator options should be analyzed since they may change the source code, logic, or object code generated.*

16. *Future regulatory guidance might consider that if a different compiler or options are used, previous verification may not be valid and should not be used for the new application.*

17. *Future regulatory guidance might consider endorsing, adopting, or adapting the methods of EPRI TR-1025243 [13] for the dedication of commercial-grade design and analysis tools that have been procured as a CGI from a supplier or developer that does not maintain a QA program that meets the requirements of 10 CFR Part 50, Appendix B.*

18. *Future regulatory guidance might consider adapting the EPRI TR-1025243 [13] commercial-grade dedication methodology to the commercial-grade dedication of CASE tools and other software development tools not covered by EPRI TR-1025243 [13].*

19. *Future regulatory guidance might consider that if the tool eliminates, reduces, or automates software life cycle processes and its output is not verified, the tool should be qualified.*

20. *Future regulatory guidance might consider that if a tool is a commercially available product, source code or logic review and analysis, verification of low-level requirements used to create the source code or logic, verification of the software architecture, and verification of the degree of test coverage may not be possible due to the proprietary nature of the tool development; therefore, the tool must be qualified.*

21. *Future regulatory guidance might consider that tools not developed under a QA program complying with 10 CFR Part 50, Appendix B, should be assigned a TQL determined from the software integrity level or RISC and the type of tool and qualified or dedicated by endorsing the high-quality life cycle processes of RTCA DO-330 [14] with exception to reduce the COTS software tool qualification requirements imposed on the COTS tool developer.*

22. *Future regulatory guidance might consider clarifying that COTS software tool qualification, as described in RTCA DO-330 [14], should be performed using a combination of performance history review and analysis in conjunction with one or more alternate methods of qualification such as exhaustive input checking or the use of dissimilar tools in lieu of relying mostly on COTS vendor participation.*

Tool qualification might be considered as the basis for licensee or applicant:

- acceptance of what is expected to be the majority of tools used, that were not developed under such a nuclear QA program, through the process of commercial-grade dedication; or
- acceptance of tools developed under another high-quality life cycle approach.

The TQL determination in Sections 2.7.1 and 2.7.2 of this report is a phased implementation of tool qualification consistent with the phased implementation of tool categories discussed in Sections 2.2.1 and 2.2.2 of this report. The TQL determination is necessary to select the minimum qualification activities specified in RTCA DO-330 [14]. The TQL determination supports up to a four-level integrity level schema and, therefore, supports current regulatory guidance which assigns an integrity level 4 to all software, supports the use of RISC-1 to RISC-4 specified in 10 CFR 50.69, and supports future use of integrity levels beyond the current integrity level 4.

### 2.7.1 Phase 1

After the new version of IEEE Std. 7-4.3.2 [6] is published for use, phase 1 might consider a determination of the TQL as shown in Table 2.3 based on the tool categories defined in Section 2.2.1 of this report and the integrity level of the software or logic as determined from IEEE Std. 1012-2012 [5] or the RISC of the SSC containing the software or logic as determined from 10 CFR 50.69. Consistent with RTCA DO-178C [9] and RTCA DO-330 [14], Table 2.3 uses five TQLs where TQL-1 is the most rigorous level and TQL-5 is the least rigorous level. The objectives, activities, guidance, and life cycle data required for each TQL to qualify the software tool are described in RTCA DO-330 [14]. The TQL in Table 2.3 is determined from two independent parameters: (1) the type of tool based on the tool categories of IEEE Std. 7-4.3.2 [6] and (2) the integrity level or RISC of the software or logic being developed using the tool. Based on the software integrity level or RISC, Table 2.3 assigns a higher TQL that requires less qualification rigor to software tools that are used to develop software or logic that is less safety significant. Based on the type of tool, Table 2.3 assigns a lower TQL that requires more qualification rigor to software tools that have a higher potential of introducing errors in the PDD. Table 2.3 assigns a slightly higher TQL that requires slightly less qualification rigor to software tools that can only fail to detect errors in the PDD. Table 2.3 assigns the highest TQL that requires the minimum qualification rigor to software tools that can neither introduce nor fail to detect errors in the PDD.

Based on the definitions of the tool types in Section 2.2.1 of this report, Type I, II, and IV tools have a potential direct impact on the PDD through requirements specification, transformation of requirements into source code or logic, or automatic source code or logic generation. Therefore, Type I, II, and IV software tools are assigned the most rigorous TQL of 1 through 4 commensurate with the software integrity level or the RISC. Type III tools have a potential indirect impact on the PDD through the use of inaccurate data generated by a plant modeling, analysis, or simulation support tool, or by a failure in the CM process. Type III software tools do not have a direct impact on the PDD; therefore, Type III software tools are assigned a slightly

less rigorous TQL of 2 through 5 than Type I, II, and IV tools commensurate with the software integrity level or the RISC. Type V "off-line" V&V tools can fail to detect errors in the PDD but cannot introduce errors either directly or indirectly. Therefore, Type V tools do not require the same level of qualification rigor and intensity as Type I, II, III, and IV tools and are assigned a slightly less rigorous TQL than the Type III tools. Tool type VI has little or no impact on the software or logic and are assigned the highest TQLs consistent with their importance to safety.

**Table 2.3 – Phase 1 Tool Qualification Level Determination**

| Risk-Informed Safety Class | Software Integrity Level | Tool Type | | | | | |
|---|---|---|---|---|---|---|---|
| | | Type I – Software Development Tools | Type II - Surveillance and Maintenance Tools | Type IV – In-Line V&V Tools | Type III - Development Process Support Tools | Type V – Off-Line V&V Tools | Type VI - Management Support Tools |
| RISC-1 | 4 | TQL-1 | | | TQL-2 | TQL-3 | TQL-4 |
| RISC-2 | 3 | TQL-2 | | | TQL-3 | TQL-4 | TQL-5 |
| RISC-3 | 2 | TQL-3 | | | TQL-4 | TQL-4 | TQL-5 |
| RISC-4 | 1 | TQL-4 | | | TQL-5 | TQL-5 | TQL-5 |

### 2.7.2 Phase 2

Phase 2 might consider a determination of the TQL as shown in Table 2.4 based on the tool categories defined in Section 2.2.2 of this report and the integrity level of the software or logic as determined from IEEE Std. 1012-2012 [5] or the RISC of the SSC containing the software or logic as determined from 10 CFR 50.69. Similar to the method described in Section 2.7.1 and Table 2.3 of this report, Table 2.4 uses five TQLs where TQL-1 is the most rigorous level and TQL-5 is the least rigorous level. The objectives, activities, guidance, and life cycle data required for each TQL to qualify the software tool are described in RTCA DO-330 [14]. The TQL in Table 2.4 is determined in the same manner as in Table 2.3 except that different tool types are used. Instead of the tool types defined by IEEE Std. 7-4.3.2 [6], Phase 2 and Table 2.4 use tool types based on IEEE Std. 14102-2010 [19] CASE tool characteristics that are consistent with the life cycle processes of IEEE Std. 12207-2008 [10]. Based on the software integrity level or RISC, Table 2.4 assigns a higher TQL that requires less qualification rigor to software tools that are used to develop software or logic that is less safety significant. Based on the type of tool, Table 2.4 assigns a lower TQL that requires more qualification rigor to software tools that have a higher potential of introducing errors in the PDD. Table 2.4 assigns a slightly higher TQL that requires slightly less qualification rigor to software tools that can only fail to detect errors in the PDD. Table 2.4 assigns the highest TQL that requires the minimum qualification rigor to software tools that can neither introduce nor fail to detect errors in the PDD.

Based on the definitions of the tool types in Section 2.2.2 of this report, Type I, II, III, IV, and V tools have a potential impact on the PDD software or logic and are assigned a TQL of 1 through 4 commensurate with the software integrity level or the RISC. Type VI V&V tools can fail to detect errors in the PDD software or logic but cannot introduce errors. Therefore, Type VI tools do not require the same level of qualification rigor and intensity as Type I, II, III, IV, and V tools and are assigned a higher TQL. Tool types VII, VIII, IX, and X have less impact on the PDD and are assigned successively higher TQLs consistent with their safety significance.

**Table 2.4 – Phase 2 Tool Qualification Level Determination**

| Risk-Informed Safety Class | Software Integrity Level | Tool Type | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Type I - Construction Tools | Type II – Software Development Modeling Tools | Type III - Plant Modeling, Analysis, and Simulation Tools | Type IV - Implementation Tools | Type V - Maintenance Tools | Type VI – V&V Tools | Type VII - Documentation Tools | Type VIII – CM Tools | Type IX – QA Tools | Type X - Management Tools |
| RISC-1 | 4 | TQL-1 | | | | | TQL-2 | TQL-3 | | | TQL-4 |
| RISC-2 | 3 | TQL-2 | | | | | TQL-3 | TQL-4 | | | TQL-5 |
| RISC-3 | 2 | TQL-3 | | | | | TQL-4 | TQL-5 | | | TQL-5 |
| RISC-4 | 1 | TQL-4 | | | | | TQL-5 | TQL-5 | | | TQL-5 |

## 2.8 Software Tool Quality Assurance

This section includes a technical basis that might be considered for future regulatory guidance for implementing QA processes based on industry practice that meet regulatory requirements. Current regulatory guidance for software tools is limited and includes some high-level guidance for software tool QA.

Appendix B to 10 CFR Part 50 establishes QA requirements for the design, manufacture, construction, and operation of SSCs. The requirements of 10 CFR Part 50, Appendix B, apply to software tools used in designing, testing, operating, maintaining, and modifying safety-related software. Criterion II of 10 CFR Part 50, Appendix B, states that the QA program shall provide control over activities affecting the quality of the identified SSCs, to an extent consistent with their importance to safety and that the QA program shall take into account the need for special controls, processes, test equipment, tools, and skills to attain the required quality.

RGs 1.152, 1.168, 1.169, 1.170, 1.171, 1.172, and 1.173 describe acceptable methods of satisfying portions of the 10 CFR Part 50, Appendix B, requirements with respect to the use of computers in safety systems of nuclear power plants. These RGs endorse several IEEE standards that contain methods for promoting high functional reliability and design quality in safety system software. However, the endorsed IEEE standards contain few requirements for software tool QA.

RG 1.152 states that COTS systems and tools are likely to be proprietary and generally unavailable for review; therefore, RG 1.152 considers that it is likely that there is no reliable method to determine security vulnerabilities for operating systems and tools. In such cases, unless such systems or tools are modified by the application developer, the security effort should be limited to ensuring that the features within the operating system or tool do not compromise the secure operational environment that would degrade the reliability of the safety system.

The proposed draft version of IEEE Std. 7-4.3.2 [6] requires that measures be put into place to ensure that only approved and documented software tools are used to support the development of safety-related PDD and that these tools are utilized within their design capabilities.

With respect to software tool QA, the following might be considered in future regulatory guidance to supplement the guidance in RG 1.152 and the requirements of IEEE Std. 7-4.3.2 [6] and to provide additional guidance for meeting the 10 CFR Part 50, Appendix B, QA criteria.

1. *Future regulatory guidance might consider that the selection, verification, validation, and qualification of tools should be justified and documented and that any tools used for software or logic development should be shown to be suitable for the purpose.*

2. *Future regulatory guidance might consider that all tools used should be developed, dedicated, or qualified to a level commensurate with their importance to safety.*

3. *Future regulatory guidance might consider that software or logic implementation should be established using a predetermined combination of techniques commensurate with the system's importance to safety, covering languages, tools, coding or logic generation practices, analysis, review, and testing.*

4. *Future regulatory guidance might consider that in those cases where the QA requirements of an original tool cannot be determined, an engineering evaluation should be conducted by qualified individuals to establish the requirements and controls including assurance that interfaces, interchangeability, safety, fit, and function are not adversely affected or contrary to applicable regulatory or code requirements.*

5. *Future regulatory guidance might consider that when dealing with COTS software tools, operating experience is often the deciding factor in determining a tool's quality; however, additional evidence may be based on tool qualification, validation of the suppliers, and tests.*

6. *Future regulatory guidance might consider that evidence should be provided regarding the quality of the software tools that might introduce faults in software or in system design, and regarding their ability to produce correct results.*

7. *Future regulatory guidance might consider that where the results of tool validation are unavailable, there should be effective measures to control failures of the programmed safety-significant system that result from faults that are attributable to the tool (e.g., generation of diverse redundant programming which allows the detection and control of failures of the programmed safety-significant system as a result of faults that have been introduced into the programmed safety-significant system by a translator).*

8. *Future regulatory guidance might consider that translators and compilers should not remove, without warning, defensive programming or error-checking features introduced by the programmer.*

9. *Future regulatory guidance might consider that an assessment should be carried out for V&V, software development construction, modeling, implementation, and CM tools to determine the level of reliance placed on the tools, and the potential failure mechanisms of the tools that may affect the programmed software or logic. Where such failure mechanisms are identified, appropriate mitigation measures should be taken (e.g., avoid known bugs, restrict use of the tool functionality, check the tool output, or use diverse tools for the same purpose).*

10. *Future regulatory guidance might consider that for each software development construction, modeling, implementation, and CM tool, evidence should be available that the output of the tool conforms to its specification, or documentation of the output or failures in the output are detected based on:*

    a. *a suitable combination of history of successful use in similar environments and for similar applications (within the organization or other organizations),*
    b. *tool validation,*
    c. *diverse redundant programming which allows the detection and control of failures resulting in faults introduced by a tool, and*
    d. *other appropriate methods for avoiding or handling failures introduced by tools such as version history or a record of the errors and ambiguities associated with tool use in the environment.*

11. *Future regulatory guidance might consider that where automatic code generation or similar automatic translation takes place, the suitability of the automatic translator for safety-significant system development should be assessed at the point in the development life cycle where development support tools are selected.*

12. *Future regulatory guidance might consider that software tool usage, environmental and functional constraints, and general operating conditions should be shown to comply with the tool's qualification through the use of identical version and configuration settings for the same use cases together with the same implemented*

*measures for the prevention or detection of malfunctions and their corresponding erroneous output, as documented in the qualification report for the software tool.*

13. *Future regulatory guidance might consider that if the qualification of a software tool is performed independently from the development of a particular safety-significant SSC, the validity of this predetermined qualification should be confirmed prior to the software tool being used for the development of a particular safety-significant SSC.*

## 2.9 Software Tool Training

This section includes a technical basis that might be considered for future regulatory guidance for software tool training based on industry practice. Current regulatory guidance for software tools is limited and does not include guidance for software tool training.

RG 1.152 states that conformance with the requirements of IEEE Std. 7-4.3.2-2003 [8] is a method that is deemed acceptable for satisfying regulations with respect to high functional reliability and design requirements for computers used in the safety systems of nuclear power plants. Also, RG 1.169 states that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8]. Neither IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 7-4.3.2-2010 [12], nor the proposed draft of IEEE Std. 7-4.3.2 [6] address software tool training. However, 10 CFR Part 50, Appendix B, Criterion II, states that the QA program shall provide for indoctrination and training of personnel performing activities affecting quality as necessary to assure that suitable proficiency is achieved and maintained.

With respect to software tool training, the following guidance might be considered in future regulatory guidance to address personnel training when developing and using software tools.

1. *Future regulatory guidance might consider that software developers, software QA personnel, and independent V&V personnel should be trained in relevant software engineering tools so they stay current with the engineering environment and products they are using or assuring and know when tools are appropriate for use.*

2. *A revision to BTP 7-14 might be considered to require the software QA plan to require that tools used are known to, and mastered by, the persons concerned.*

## 2.10 Software Tool Configuration Management

This section includes a technical basis that might be considered to improve current regulatory guidance and for developing future regulatory guidance for software tool CM based on industry practice. Current regulatory guidance covers software tool CM.

RG 1.169 endorses IEEE Std. 828-2005 [22] for SCM methods that apply to the maintenance and control of appropriate records of software development activities. Although RG 1.169 applies primarily to software, RG 1.169 states that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8] and that an SCM program operated by the organization using the tool that complies with IEEE Std. 828-2005 [22] should treat tools as configuration items. RG 1.169 also states that SCM plans should identify

that software tools used in the production of safety system software be considered as configuration items. RG 1.169 further states that for safety system software, support software (i.e., tools) used in the development of safety system software and testing tools used to test safety system software must be identified and controlled as configuration items or discussed in controlled documents. Also, RG 1.169 states that items that may not change but are necessary to ensure correct software production, such as compilers, should also be configuration items, thereby ensuring that all factors contributing to the executable software are understood.

IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 7-4.3.2-2010 [12], and the proposed draft of IEEE Std. 7-4.3.2 [6] all require that software tools used to support the software or logic life cycle processes of safety-related PDD be maintained under CM.

With respect to software tool CM, the following guidance might be considered to improve current regulatory guidance or in future regulatory guidance to expand software tool CM guidance.

1. *A revision to RG 1.169 might consider that an SCM program should ensure that only qualified tools are used, only tools compatible with each other are used, and that information on tools that generate configuration items is documented.*

2. *Future regulatory guidance might consider that all I&C items including software tools that are used to produce, control, configure, verify, or validate I&C components and the tool parameter settings should be designated, given a unique identification, and placed under configuration control.*

3. *Future regulatory guidance might consider that software tool use shall be traced so that the tools, if any, that were used to generate a given item or information may be identified.*

4. *Future regulatory guidance might consider that where automatic or semi-automatic tools are used for the production of documentation, specific procedures may be necessary to ensure effective measures are in place for the management of versions or other control aspects of the documents.*

5. *Future regulatory guidance might consider adopting the Table 2.5 minimum set of CM process activities that should be performed when developing, dedicating, or qualifying a software tool in accordance with RTCA DO-330 [14]. RTCA DO-330 [14] specifies the use of control category 1 (CC1) or control category 2 (CC2) as a function of TQL for data items generated during the tool development and qualification life cycle processes.*

**Table 2.5 – SCM Process Objective Control Categories**

| SCM Process Objective | CC1 | CC2 |
|---|---|---|
| Configuration Identification | ● | ● |
| Baselines | ● | |
| Traceability | ● | ● |
| Problem Reporting | ● | |
| Change Control – integrity and identification | ● | ● |
| Change Control – tracking | ● | |
| Change Review | ● | |
| Configuration Status Accounting | ● | |
| Retrieval | ● | ● |
| Protection against Unauthorized Changes | ● | ● |
| Media Selection, Refreshing, Duplication | ● | |
| Release | ● | |
| Data Retention | ● | ● |

## 2.11 Software Tool Review, Approval, and Acceptance Criteria

This section includes a technical basis that might be considered for future regulatory guidance for the review, approval, and acceptance criteria of software tools used in the design of new, or upgrades to, domestic commercial nuclear power plants (or other NRC-licensed facilities as applicable). The technical basis might be used by licensees, applicants, and third-party contractors or vendors for software tool acceptance based on the tool being developed under a 10 CFR Part 50, Appendix B, QA program; based on the dedication of commercial-grade tools not developed under an Appendix B QA program; or developed or dedicated using the high-quality life cycle approach of RTCA DO-330 [14].

Appendix C of NASA-STD-8739.8 [31] is a requirements compliance matrix that lists all the requirements of NASA-STD-8739.8 [31] along with personnel roles and responsibilities required for satisfying each requirement. Such a matrix can be used as a checklist to ensure coverage of all requirements. The use of a requirements compliance matrix might be considered as a review and approval aid to formally document compliance to software tool requirements. A requirements compliance matrix can be generated after software tool requirements and guidance have been generated or finalized in regulatory guidance or endorsed industry standards.

With respect to software tool review, approval, and acceptance criteria, the following general guidance might be considered to improve current regulatory guidance or in future regulatory guidance that applies regardless of the method of software tool development.

1. *Future regulatory guidance might consider that special care should be taken when reviewing software tool usage, especially compilers, linkers, and logic generators, to ensure that the software tool cannot introduce a CCF.*

2. *Future regulatory guidance might consider that a general acceptance criterion for tools is that tools should be developed or dedicated as safety-significant or qualified using a quality lifecycle process if the output of the tool cannot be proven to be correct, such as may occur if the tool produces machine language software code.*

3. *Pending final regulatory guidance and updated industry standards, a compliance checklist might be considered similar to that used in NASA-STD-8739.8 [31] for use as an aid in the review and approval of software tools used to develop safety-significant software for use in a commercial nuclear power plant.*

4. *Future regulatory guidance might consider that the use of tools that are not common or extensively used within the industry should increase the level of NRC involvement in approving the tool.*

Sections 2.11.1 through 2.11.3 include a technical basis for future regulatory guidance for the review, approval, and acceptance criteria for software tools. Section 2.11.1 includes a technical basis for accepting software tools developed under a 10 CFR Part 50, Appendix B, QA program. Section 2.11.2 includes a technical basis for accepting commercial-grade software tools through the commercial-grade dedication process. Section 2.11.3 includes a technical basis for accepting software tools developed or dedicated using the RTCA DO-330 [14] high-quality life cycle approach.

### 2.11.1 10 CFR Part 50, Appendix B, Development

This section includes a technical basis for reviewing, approving, and accepting software tools developed under a 10 CFR Part 50, Appendix B, QA program. Appendix B to 10 CFR Part 50 contains QA criteria for the design, manufacture, construction, and operation of SSCs. The pertinent requirements of 10 CFR Part 50, Appendix B, apply to all activities affecting the safety-related functions of SSCs including, but not limited to, designing, testing, operating, maintaining, and modifying. The requirements of 10 CFR Part 50, Appendix B, also apply to software tools used in designing, testing, operating, maintaining, and modifying safety-related software.

RGs 1.152, 1.168, 1.169, 1.170, 1.171, 1.172, and 1.173 describe acceptable methods of satisfying portions of the 10 CFR Part 50, Appendix B, requirements with respect to the use of computers in safety systems of nuclear power plants. These RGs endorse several IEEE standards that contain methods for promoting high functional reliability and design quality in safety system software including IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 1012-2004 16], IEEE Std. 1028-2008 [18], IEEE Std. 828-2005 [22], IEEE Std. 829-2008 [17], IEEE Std. 830-1998 [23], IEEE Std. 1008-1987 [30], and IEEE Std. 1074-2006 [15]. These RGs also recognize detailed information in a few other industry standards including EPRI TR-106439 [32] that have been accepted. These RGs also incorporate recommendations and are consistent with the basic principles provided in IAEA NS-G-1.1 [33].

NUREG-0800, Appendix 7.1-D states the following with respect to software tool reviews. First, reviewers should thoroughly evaluate tool usage. Also, tools used for software development may reduce or eliminate the ability to evaluate the output of those tools, and therefore, rely on

the tool, or on subsequent testing, to show the software will perform as intended. Testing alone can only show that those items tested will operate as intended, and cannot be relied upon to show that no unintended functions exist, or that the software will function in conditions other than those specifically tested. The use of software tools should be evaluated in the overall context of the quality control and V&V process, and there should be a method of evaluating the output of the tool.

BTP 7-14 and ISG-06 reiterate the IEEE Std. 7-4.3.2-2003 [8] requirement that software tools shall be designed as safety-related unless the tools are used in a manner such that defects not detected by the tool are detected by V&V activities. Neither BTP 7-14 nor ISG-06 acknowledge that the software tool requirements of IEEE Std. 7-4.3.2-2003 [8] have been updated in IEEE Std. 7-4.3.2-2010 [12] and the proposed draft of IEEE Std. 7-4.3.2 [6]. The software tool requirements in IEEE Std. 7-4.3.2-2010 [12] state that the software tool output shall be verified and validated to the same level of rigor as the safety-related software or that the tool shall be developed to the same level of quality as the safety-related software or be commercially dedicated. The software tool requirements in the proposed draft of IEEE Std. 7-4.3.2 [6] state that software tools should be used in a manner such that defects not detected by the software tool will be detected by V&V activities or that the tool shall be developed using a quality life cycle process commensurate with the criticality of the safety functions performed by the end product.

The revisions that might be considered to the RGs and IEEE standards noted in Sections 2.1 through 2.10 of this report along with future regulatory guidance that might be considered would expand the coverage of the RGs and IEEE standards to the planning, selection, use, documentation, development, qualification, dedication, QA, training, and CM of safety-significant software and software tools analogous to the coverage of safety-related software. The acceptance criterion that might be considered for software tools developed under a 10 CFR Part 50, Appendix B, QA program includes following the guidance and satisfying all the pertinent requirements of IEEE standards endorsed by RGs 1.152, 1.168 through 1.173, and any future regulatory guidance based on an assigned software tool integrity level.

With respect to software tool review, approval, and acceptance criteria for software tools developed under a 10 CFR Part 50, Appendix B, QA program, the following guidance might be considered to improve current regulatory guidance or in future regulatory guidance.

1. *Future regulatory guidance might consider that the acceptance criteria of software tools developed under a 10 CFR Part 50, Appendix B, QA program would be the assignment of a software tool integrity level, following the guidance appropriate to that integrity level, and satisfying all the pertinent requirements of IEEE standards endorsed by RG 1.152, and RGs 1.168 through 1.173 for the assigned software tool integrity level.*

2. *Revisions to BTP 7-14 and ISG-06 or new BTPs and ISGs might be considered for consistency with the assignment of a software tool integrity level.*

3. *Revisions to BTP 7-14 and ISG-06 might be considered to acknowledge the changes in software tool acceptance criteria in IEEE Std. 7-4.3.2-2010 [12] or the proposed draft version of IEEE Std. 7-4.3.2 [6].*

### 2.11.2 Commercial Dedication

This section includes a technical basis that might be considered for future regulatory guidance for accepting software tools acquired as CGIs and dedicated.

Part 21 of Title 10 of the CFR (10 CFR Part 21) describes the commercial-grade dedication process and establishes that a CGI properly dedicated under the applicable controls of a nuclear QA program compliant with 10 CFR Part 50, Appendix B, may be considered equivalent to a component designed and manufactured under an Appendix B QA program, and thus may be designated as a basic component. The NRC has endorsed the acceptance methods of certain IEEE standards and EPRI documents (some generic and some applicable to software), with qualifications and restrictions (e.g., such as those in NRC GL 89-02 and GL 91-05), for commercial-grade dedication through certain generic communications and safety evaluations.

Basic components are defined in 10 CFR 21.3 as including CGIs that have successfully completed the dedication process that includes the identification and verification of critical characteristics. Critical characteristics are defined in 10 CFR 21.3 as those important design, material, and performance characteristics of a CGI that, once verified, provide reasonable assurance that the CGI will perform its intended safety function. Dedication is defined in 10 CFR 21.3 as an acceptance process undertaken to provide reasonable assurance that a CGI to be used as a basic component will perform its intended function and, in this respect, is deemed equivalent to an item designed and manufactured under a 10 CFR Part 50, Appendix B, QA program. As discussed in 10 CFR 21.3, reasonable assurance is achieved by identifying the critical characteristics of the item and verifying their acceptability by inspections, tests, or analyses performed by the purchaser or third-party dedicating entity after delivery, supplemented as necessary by one or more of the following:

- commercial-grade surveys,
- product inspections or witness at hold points at the manufacturer's facility, and
- analysis of historical records for acceptable performance.

RG 1.152 identifies that IEEE Std. 7-4.3.2-2003 [8] provides general guidance for commercial-grade dedication of safety system designs that use computers but have not been specifically designed for nuclear power plant applications. IEEE Std. 7-4.3.2-2003 [8] states that the dedication process for computers shall include identification of the physical, performance, and development process requirements necessary to provide adequate confidence that the proposed digital system or component can achieve the safety function. The dedication process for software and firmware shall, whenever possible, include an evaluation of the design process.

IEEE Std. 7-4.3.2-2003 [8] describes the preliminary and detailed phases of the dedication process. In the preliminary phase, risks and hazards are evaluated, the safety functions are

identified, CM is established, and the safety category of the system is determined. In the detailed phase, the CGI is evaluated for acceptability using detailed acceptance criteria. The critical characteristics of a CGI shall be identified by a technical evaluation and shall be verifiable by a combination of inspection, analysis, or testing. Annex C of IEEE Std. 7-4.3.2-2003 [8] provides guidance for the dedication of commercial computers; however, RG 1.152 does not endorse the annex because it provides inadequate guidance. RG 1.152 endorses EPRI TR-106439 [32] for commercial-grade dedication guidance.

IEEE Std. 7-4.3.2-2010 [12] and the draft version of IEEE Std. 7-4.3.2 [6] both contain similar guidance for commercial-grade dedication. These standards describe four acceptance methods for evaluating traditional COTS equipment based on EPRI NP-5652 [34]. These four acceptance methods are defined in the following paragraphs.

Method 1: Use of special tests, inspections, or some combination of tests and inspections, after receipt of the commercial items.

Method 2: Use of a commercial grade survey to take credit for the commercial controls the supplier exercises.

Method 3: Source verification, in which the acceptance and dedication is based on direct verification of critical characteristics and their control by the vendor.

Method 4: Evaluation of commercial equipment based on the confidence achieved through proven performance of that item in equivalent service and configuration.

IEEE Std. 7-4.3.2-2010 [12] and the draft version of IEEE Std. 7-4.3.2 [6] state that proven performance (i.e., Method 4) shall not be used as the sole means of assessing digital system quality since even small changes in configuration or environmental stressors can change the behavior of digital systems. IEEE Std. 7-4.3.2-2010 [12] and the draft version of IEEE Std. 7-4.3.2 [6] also state that commercial-grade survey of the supplier (i.e., Method 2) shall not be employed as the basis for accepting items from suppliers with undocumented commercial quality control programs or with programs that do not effectively implement their own necessary controls. Also, commercial-grade surveys (i.e., Method 2) shall not be used as the basis for accepting items from distributors unless the survey includes the part manufacturer(s) and the survey confirms adequate controls by both the distributor and the part manufacturer(s).

Additional commercial-grade dedication guidance is provided in GL 89-02 and GL 91-05. GL 89-02 addresses methods to detect substandard, counterfeit, or fraudulently marketed products. GL 89-02 conditionally endorses EPRI NP-5652 [34]; however, GL 89-02 states that the acceptance method of proven performance should not be used alone unless the historical record is based on industry-wide performance data that is directly applicable to the item's critical characteristics and the intended safety-related application; and the manufacturer's measure for the control of design, process, and material changes have been adequately implemented as verified by audit. GL 91-05 clarifies the NRC staff's position on licensee implementation of commercial-grade dedication programs with respect to critical characteristics. GL 91-05 clarifies

that not all design requirements must be critical characteristics; however, the licensee is responsible for identifying the important design, material, and performance characteristics for each part, material, and service intended for safety-related applications; establishing acceptance criteria; and providing reasonable assurance of the conformance of items to the acceptance criteria.

ISG-06 addresses the commercial-grade dedication of software through reference to requirements in IEEE Std. 7-4.3.2-2003 [8] and guidance in EPRI TR-106439 [32] and EPRI TR-107330 [35]. ISG-06 clarifies that the NRC staff recognizes two different ways a component can be approved for use in safety-related applications:

1. If a basic component has critical characteristics that cannot be verified, then it must be designed and manufactured under a QA program.

2. If a basic component has critical characteristics that can be verified, then it may be designed and manufactured as a CGI and then commercially dedicated under a QA program.

EPRI NP-5652 [34] and EPRI TR-102260 [11] discuss a generic process for the acceptance of commercial-grade hardware components and parts for use in a nuclear safety-related application. A fundamental concept of commercial-grade dedication discussed in EPRI TR-102260 [11] is in the application of a graded approach that is consistent with 10 CFR Part 50, Appendix B, Criterion II, which requires that a QA program control activities affecting the quality of the identified SSCs, to an extent consistent with their importance to safety. The most important parts of the generic process are identifying the critical characteristics and accepting the item by verification of those critical characteristics. EPRI NP-5652 [34] states that an item's critical characteristics can include a part number, identification markings, or performance characteristics and that verification of the CGI's critical characteristics provides assurance of acceptability. There are four acceptance methods:

Method 1: Special tests and inspections,
Method 2: Commercial-grade survey of the supplier,
Method 3: Source verification, and
Method 4: Acceptable supplier or item performance record.

The generic process and acceptance methods were written for applicability to hardware components and parts, not software or software tools.

EPRI TR-106439 [32] and EPRI TR-107339 [36] contain guidance on the evaluation and acceptance of commercial-grade digital equipment as part of the commercial-grade dedication process for nuclear safety applications. EPRI TR-106439 [32] relies on the existing commercial-grade dedication process described in EPRI NP-5652 [34] and EPRI TR-102260 [11] with supplemental guidance to address digital-specific issues. The primary focus of EPRI TR-107339 [36] is on the differences in the technical evaluation and acceptance process required for digital, software-based equipment and systems. The guidance of EPRI TR-106439 [32] and the supplemental guidance of EPRI TR-107339 [36] are relevant to commercial-grade

software but not particularly relevant to software tools. The NRC safety evaluation for EPRI TR-106439 [32] states that V&V activities common to software development in digital systems are a critical characteristic that can be verified as being performed correctly following the completion of the software development by conducting certain dedication activities such as audits, examinations, and tests.

EPRI TR-1025243 [13] provides guidance for the safety classification of design and analysis tools that are not resident or installed as part of plant SSCs. EPRI TR-1025243 [13] defines functional safety classifications of computer programs based on the discussion in 10 CFR Part 21. There are only two major classifications: safety-related and nonsafety-related. A subset of nonsafety-related computer programs may be classified as augmented quality. The simple use of two classifications results in all safety-related computer programs being subject to the same requirements and level of QA even though computer programs have various degrees of potential impact on safety-related components.

EPRI TR-1025243 [13] also provides guidance for using a commercial-grade dedication process to accept safety-related, commercial-grade design and analysis tools used in the design and analysis of safety-related plant SSCs. EPRI TR-1025243 [13] extends the commercial-grade dedication process described in EPRI NP-5652 [34] from hardware components to design and analysis software tools used to support the hardware design and development. EPRI TR-1025243 [13] guidance applies to steady-state and transient thermal-hydraulic and mechanical design and analysis tools, administrative support tools, operations support tools, software integral to measurement and test equipment, and software used in the manufacturing of SSCs. EPRI TR-1025243 [13] guidance does not apply to software development construction, modeling, implementation, V&V, CM, and QA tools used to design and develop safety-related DI&C software. EPRI TR-1025243 [13] states that computer programs involved in QA aspects of the program (e.g., document control, records management, corrective action tracking, maintenance of training records, emergency response preparedness) are nonsafety-related but augmented quality controls should be considered.

EPRI TR-1025243 [13] commercial-grade dedication involves a technical evaluation followed by an acceptance process to provide a reasonable assurance that the software tool procured meets specified requirements. The technical evaluation for a design or analysis computer program consists of the following activities:

1. Identify the scope of use and function of the computer program
2. Determine the safety classification of the computer program
3. Identify the safety functions of the computer program
4. Perform a failure modes and effects analysis to identify characteristics of the computer program
5. Identify critical characteristics and acceptance methods
6. Specify technical, quality, and documentation requirements
7. Document the technical evaluation

As defined in 10 CFR Part 21, critical characteristics are those important design, material, and performance characteristics of a CGI that, once verified, provide reasonable assurance that the

CGI will perform its intended safety function. Tool qualification might be considered as a critical characteristic of a software tool and completion of tool qualification might be considered as an acceptance criterion. Use of a TQL and using a four level schema to categorize safety-related, nonsafety-related, and augmented quality software tools might be considered as a method of expanding the scope of EPRI TR-1025243 [13] from design and analysis software tools to all software development and support tools.

Four current methods of accepting CGIs that are recognized by IEEE Std. 7-4.3.2-2010 [12], the proposed draft version of IEEE Std. 7-4.3.2 [6], and EPRI TR-1025243 [13] are the same as the four methods previously discussed from EPRI NP-5652 [34]. However, most of the software tools used to develop safety-related software are COTS software tools. Commercial-grade supplier surveys and source verification are not viable options for verifying critical characteristics of COTS computer programs because dedication of the computer programs occur well after the programs have been developed and because of the unwillingness of COTS program vendors to allow outside access to proprietary processes and documentation.

With respect to software tool dedication, the following might be considered in future regulatory guidance to address software tool review, approval, and acceptance criteria that meets regulatory requirements.

1. *Future regulatory guidance might consider defining tool qualification as a critical characteristic that must be verified during the acceptance process described in EPRI TR-1025243 [13].*

2. *Future regulatory guidance might consider that the critical characteristics of CASE tools and other software development tools should be verified using extensive testing and performance history of the COTS software tool.*

3. *Future regulatory guidance might consider that, as part of the commercial-grade dedication process, all critical characteristics must be verified to commercially dedicate a software tool; therefore, the list of critical characteristics should be limited to only those characteristics strictly necessary for dedication of the software tool.*

4. *Future regulatory guidance might consider that review and approval of software tools that have been commercially dedicated should rely on the verification that the commercial-grade dedication process has been followed and completed successfully.*

5. *Future regulatory guidance might consider that the acceptance criteria of commercial-grade software tools that have been dedicated would be the assignment of a TQL and tool qualification as a critical characteristic and that acceptance would be based on verifying completion of all RTCA DO-330 [14] activities and tasks required by that TQL.*

### 2.11.3 RTCA DO-330 Development

This section includes a technical basis that might be considered for future regulatory guidance for the review, approval, and acceptance of software tools not developed under a 10 CFR Part

50, Appendix B, QA program or commercially dedicated, but developed using the high-quality life cycle approach of RTCA DO-330 [14]. Alternatively, the RTCA DO-330 [14] tool qualification process, conducted under appropriate Appendix B QA controls, might be viewed as an acceptable approach to dedication of software tools.

RTCA DO-178C [9] and RTCA DO-330 [14] provide a comprehensive tool qualification process with RTCA DO-178C [9] defining the need and rigor necessary for the tool qualification while RTCA DO-330 [14] provides the processes, activities, and tasks for tool qualification. The NRC may find that RTCA DO-178C [9] cannot be endorsed or adopted by regulatory guidance because of the industry-specific software levels. However, regulatory guidance that includes the key elements of RTCA DO-178C [9] might be considered including the specification of a TQL, as discussed in Section 2.7 of this report, so that the processes, activities, and tasks defined in RTCA DO-330 [14] can be endorsed, adapted, or adopted for incorporation into possible future regulatory guidance.

RTCA DO-330 [14] presents an entire development, review, and approval process for software tools being developed as part of a software project, either in-house or through a specific outside vendor or contractor, where the responsibility of tool qualification is shared by both the tool developer and the tool user. The tool qualification liaison process described in RTCA DO-330 [14] contains similar activities and tasks as ISG-06 to facilitate the review and approval of software tools. The tool qualification liaison process identifies the activities that are required to obtain approval of the tool qualification as a function of TQL. These activities are concerned with identifying the need for tool qualification, analyzing all known problems and functional limitations of the tool, and reporting the results of the analysis. All objectives should be satisfied for all five TQLs. Frequent communication with approval personnel is paramount for obtaining approval of tools. The essential elements of the tool qualification liaison process include submitting evidence that tool problems and functional limitations have been analyzed, resolved, and properly reported; the tool qualification process has been completed successfully; and all objectives that should have been satisfied in the tool qualification process have actually been satisfied.

With respect to software tool review, approval, and acceptance using the RTCA DO-330 [14] software tool development processes, the following might be considered in future regulatory guidance to address software tool review, approval, and acceptance criteria that meets regulatory requirements.

1. *Future regulatory guidance might consider endorsing, with exceptions, the RTCA DO-330 [14] tool qualification liaison process as part of the software tool review and approval process. Exceptions might be considered in the future regulatory guidance to translate terms from civil aviation terminology to NRC terminology (e.g., certification to approval, certification authority to NRC, etc.).*

2. *Future regulatory guidance might consider that review and approval of software tools should be dependent on following all the objectives, executing all activities that satisfy each objective, and developing all the associated life cycle data required by RTCA DO-330 [14].*

# 3 Summary and Conclusions

More applications for new nuclear power plants and for modifications to existing nuclear power plants may include an expanded use of software tools in the development of DI&C systems. Some of the design, program generation, design V&V, automated testing, and operation of these DI&C systems may use or expand the use of software tools. Additional regulatory guidance may provide more consistent guidance on the application of the tools. Further, changes to current regulatory guidance and industry standards, or developing new regulatory guidance that will support the industry standards and meet regulatory requirements, might improve the software tool review and approval process. Endorsing, adopting, or adapting other industry standards in future regulatory guidance might improve the consistency of software tool treatment in the U.S. commercial nuclear power industry with other industries.

The technical basis in this report is consistent with enhancing regulatory guidance using a phased approach to minimize the immediate impact of new regulatory guidance on licensees and applicants, putting more attention on deterministic methods currently used by most licensees and describing possible alternative methods, such as risk-informed methods. The technical basis also contains processes that may be used by licensees, applicants, and third-party contractors or vendors for developing, evaluating, and qualifying tools, whether acceptance of tools is based on the tool being developed under a 10 CFR Part 50, Appendix B, QA program; based on the dedication of commercial-grade tools not developed under an Appendix B QA program using an approach prescribed in EPRI TR-1025243 [13]; developed using another high-quality life cycle approach prescribed in RTCA DO-330 [14]; or using the RTCA DO-330 [14] process as a means of dedication (e.g., of COTS software tools).

Future regulatory guidance and revisions to existing regulatory guidance might consider all aspects of software tool use during the development of safety-significant nuclear plant software including software tool integrity level determination; software tool planning; software tool selection and use; software tool documentation; software tool development, dedication, and qualification; software tool QA; software tool training; software tool CM; and software tool review, approval, and acceptance criteria. All of these aspects of software tool use might be addressed through regulatory guidance, endorsement of industry standards, or eventually through other policy documents or regulations.

The overall conclusions and summary of the key aspects of this report are presented in Sections 3.1 through 3.11 of this report.

## 3.1 Software Life Cycle Processes

Software tool guidance and review and approval practices are typically limited to software tools used in processes, activities, and tasks that have the potential of introducing defects or failing to detect defects in the final PDD software or logic. These processes, activities, and tasks that have the potential of introducing defects or failing to detect defects are associated with software or logic development and V&V. Expanding the software tool regulatory guidance and review and approval practices to cover more PDD life cycle support processes (i.e., QA and CM

processes) might be considered when assurability of the safety-significant system software or logic is dependent on a tool used during that life cycle process.

Three principal IEEE standards apply to system and software life cycles: IEEE Std. 1074-2006 [15], endorsed by RG 1.173; IEEE Std. 15288-2008 [16]; and IEEE Std. 12207-2008 [10]. Many updates to IEEE software-related standards include modifications for harmonization with IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16]. Revisions to existing regulatory guides might be considered to endorse the latest versions of IEEE standards that have been updated for harmonization with IEEE Std. 12207-2008 [10] and IEEE Std. 15288-2008 [16].

## 3.2 Software Tool Categories

Current regulatory guidance for software tools is limited and does not include guidance for categorizing software tools. Software tools can be used in almost every phase of the PDD development project life cycle and assist in almost every task.

IEEE Std. 7-4.3.2-2003 [8], endorsed by RG 1.152, does not contain guidance for categorizing software tools; however, the proposed draft version of IEEE Std. 7-4.3.2 [6] defines five types of software tools and states that software tools shall be assessed in a manner consistent with the category of tool and the potential that the tool has to introduce faults into the safety-related PDD.

IEEE Std. 14102-2010 [19] is not currently endorsed by regulatory guides; however, IEEE Std. 14102-2010 [19] discusses the attributes of nine CASE tool categories that are closely associated with IEEE Std. 12207-2010 [10] software life cycle processes. IEEE Std. 14102-2010 [19] includes a software development modeling tool category but does not include a plant modeling, analysis, and simulation tool category.

Although the definitions of the five tool types in the draft version of IEEE Std. 7-4.3.2 do not specifically discuss plant modeling, analysis, and simulation tools, the working group for the proposed draft of IEEE Std. 7-4.3.2 [6] considers plant modeling, analysis, and simulation tools as either Type III or Type IV V&V tools because they are used in either an “in-line” or “off-line” manner to support various software development activities. The proposed draft of IEEE Std. 7-4.3.2 [6] also does not address these types of tools when used to support the generation of a requirements specification or to support design decisions.

Use, adoption, or endorsement of software tool categories might be considered in future regulatory guidance so the rigor and intensity of software tool development, dedication, or qualification can be tailored based on the type of tool and the safety significance of the software or logic that is to be developed using the tools. A phased implementation of regulatory guidance that specifies software tool categories might be considered to minimize the impact on industry and to harmonize with other industry standards.

After the new version of IEEE Std. 7-4.3.2 [6] is published for use, phase 1 might consider new regulatory guidance or revisions to existing RGs that endorse the IEEE Std. 7-4.3.2 [6] tool types with some possible enhancements. These enhancements include expanding the definition of Type I tools to include software development implementation and modeling tools;

expanding the definition of Type III tools to include QA tools and plant modeling, analysis, and simulation tools; expanding the definition of Type IV tools to include plant modeling, analysis, and simulation tools; and adding an additional tool type to cover management support tools.

Phase 2 might consider endorsing the IEEE Std. 14102-2010 [19] tool types that are closely tied to the IEEE Std. 12207-2008 [10] software life cycle processes. These tool types are also sufficiently general to cover most new tools developed to automate, eliminate, or reduce activities and tasks performed in the software development life cycle processes. However, IEEE Std. 14102-2010 [19] does not recognize plant modeling, analysis, and simulation tools; so phase 2 regulatory guidance might consider augmenting the IEEE Std. 14102-2010 [19] tool types with an additional tool type that includes these types of tools.

## 3.3 Software and Software Tool Integrity Levels

Software and software tool integrity levels quantify the complexity, risk, and safety level of a system, hardware, software, or software tool based on its safety significance. The integrity level or RISC of software tools considers their use in or with safety-significant software or logic, with the degree of rigor and level of detail appropriate to the safety significance of the software or logic, and its vulnerabilities to tool-related problems in all life cycle processes.

### 3.3.1 Software Integrity Levels

The technical basis for assigning software integrity levels allows the software development processes to be tailored based on the importance of the software or logic to the safety of the system, whether evaluated deterministically or through a risk-informed probabilistic risk analysis. The software integrity level is used for determining a TQL that is used to determine the appropriate degree of rigor and intensity to be applied in the process of verifying a software tools' suitability, quality, and reliability. Tool qualification might be considered as the basis for licensee or applicant acceptance of tools acquired as CGIs and dedicated as prescribed in EPRI TR-1025243 [13]; tools developed using the high-quality life cycle approach prescribed in RTCA DO-330 [14]; or tools dedicated using the RTCA DO-330 [14] process, conducted under appropriate Appendix B QA controls as a means of dedication (e.g., for COTS software tools).

The IEEE Std. 603-1991 [27] and RG 1.152 conservative classification of equipment providing nonsafety functions as part of the safety system might be reconsidered in future regulatory guidance. Software providing nonsafety functions that resides on a computer providing a safety function might be considered as nonsafety if the functions can be demonstrated to be sufficiently independent.

RG 1.168 endorses IEEE Std. 1012-2004 [21] for software V&V requirements, and RG 1.170 endorses IEEE Std. 829-2008 [17] for software and system test documentation requirements. Both IEEE Std. 1012-2004 [21] and IEEE Std. 829-2008 [17] use a four-level integrity schema to define the minimum V&V requirements and minimum recommended testing tasks. Other integrity schemas are acceptable and for any selected integrity schema, the selected integrity levels are mapped into the IEEE Std. 1012-2004 [21] or IEEE Std. 829-2008 [17] four-level schema to demonstrate that the minimum V&V requirements are satisfied and the minimum recommended testing tasks are completed.

RG 1.168, RG 1.170, and all versions of IEEE Std. 7-4.3.2 [6][8][12] assign all software used in nuclear power plant safety systems to integrity level 4 or equivalent. Not using a graded approach for assigning software integrity levels prevents software development activities from being tailored based on the importance to safety of the various elements of the safety-significant system. Consistency with Criterion II of 10 CFR Part 50, Appendix B, requires that any graded approach for assigning software integrity levels must be custom tailored to the integrity level schema being used by the developer.

### 3.3.2 Software Tool Integrity Levels

The technical basis for assigning software tool integrity levels allows the software tool development processes to be tailored based on the importance of the software tool to the safety of the software and system. The safety significance of the software tool can be expressed in terms of a software tool integrity level. Analogous to the software integrity level, a software tool integrity level might be considered as a method to determine the minimum V&V requirements and recommended testing tasks for software tools developed under a 10 CFR Part 50, Appendix B, QA program.

IEEE Std. 1012-2004 [21], endorsed by RG 1.168, and IEEE Std. 829-2008 [17], endorsed by RG 1.170, use a four-level integrity schema to describe the importance of the software and software-based system that might be considered for applicability to software tools. However, IEEE Std. 1012-2004 [21] and IEEE Std. 829-2008 [17] only contain a single requirement for assigning integrity levels to software tools. Tools that insert or translate code (e.g., compilers or auto-code generators) or logic are assigned the same integrity level as the integrity level assigned to the software element that the tool affects. Future regulatory guidance might consider either influencing the IEEE to revise standards to assign integrity levels for all software tools or provide regulatory guidance to supplement the IEEE requirements.

## 3.4 Software Tool Planning

Current regulatory guidance for software tools is limited and does not include guidance for software tool planning or for the content of software development planning documents.

RG 1.152 and RG 1.169 either endorse or state that tools should be handled according to IEEE Std. 7-4.3.2-2003 [8]. IEEE Std. 7-4.3.2-2003 [8] contains computer-specific requirements to supplement the criteria and requirements of IEEE Std. 603-1998 [29]. However, neither IEEE Std. 7-4.3.2-2003 [8] nor IEEE Std. 603-1998 [29] contain software tool planning requirements. The proposed draft version of IEEE Std. 7-4.3.2 [6] contains requirements for tool selection and use but does not contain requirements for software tool planning.

BTP 7-14 defines software tools as resource characteristics of planning documents and states that planning documents are required to describe tools used, identify suitable tools to carry out the activities of the specific life cycle process, define the process by which tools are selected, and define the tool operational environment and any special controls necessary to execute the tools. BTP 7-14 also states that the software development plan, software integration plan, software installation plan, and software maintenance plan should require tools in these life cycle

phases to be qualified with a degree of rigor and level of detail appropriate to the safety significance of the software developed using the tool.

Revisions to BTP 7-14 or alternately, implementing changes by means of an additional BTP specifically applicable to software tools, might be considered to state that the software verification plan and the software operating plan should require V&V software tools and tools that operate safety-related equipment be qualified. Future regulatory guidance might be considered to identify the benefits of more thorough and up-front planning and selecting integrated tool sets.

## 3.5 Software Tool Selection and Use

Current regulatory guidance for software tools is limited and does not include guidance for software tool selection and use.

RG 1.152 and RG 1.169 either endorse or state that tools should be handled according to IEEE Std. 7-4.3.2-2003 [8]. IEEE Std. 7-4.3.2-2003 [8] does not address the selection of software tools or acceptance criteria for compilers, operating systems, and libraries. The proposed draft version of IEEE Std. 7-4.3.2 [6] provides more specific criteria on the use of software tools but is not yet recognized by any regulatory guide. None of the versions of IEEE Std. 7-4.3.2 [6][8][12] address compiler optimization; however, the RTCA and IEC address its use.

IEEE Std. 7-4.3.2-2003 [8] and the proposed draft version of IEEE 7-4.3.2 [6] require that a software tool be used in a manner such that defects not detected by the tool will be detected by V&V activities. Neither IEEE Std. 7-4.3.2-2003 [8] or the proposed draft version of IEEE 7-4.3.2 [6] explain how software tools can be used in different manners such that, when used in one manner, defects not detected by the tool will be detected by V&V activities; however, when used in another manner, defects not detected by the tool will **not** be detected by V&V activities. In addition, neither IEEE Std. 7-4.3.2-2003 [8] nor the proposed draft version of IEEE 7-4.3.2 [6] explain how tools can be used in such a manner as to guarantee that defects not detected by the tool will be detected by V&V activities.

The purpose of IEEE Std. 14102-2010 [19] is to assist organizations in the proper evaluation and selection of CASE tools. IEEE Std. 14102-2010 [19] contains few requirements and instead describes a “best practices” methodology for selecting CASE tools rather than a requirements-based methodology. The sections of IEEE Std. 14102-2010 [19] describing the characteristics and sub-characteristics of the various tool types can be shaped into guidance for evaluation of tools that have already been selected.

Future regulatory guidance might consider providing guidance on how software tools can be used in different manners such that, in one manner, defects not detected by the tool will be detected by V&V activities, and in another manner, defects not detected by the tool will **not** be detected by V&V activities. Future regulatory guidance might also consider endorsing IEEE Std. 14102-2020 [19] for evaluating and selecting software tools, provide guidance on the use of compiler optimization based on RTCA and IEC practices, provide guidance on the use of tools

in combination, provide guidance on software tool CCF concerns, and provide guidance on software tool diversity and independence.

## 3.6 Software Tool Documentation

Current regulatory guidance for software tools is limited but does include some guidance for documenting the use, suitability, or characteristics of software tools.

RG 1.152 endorses IEEE Std. 7-4.3.2-2003 [8] and RG 1.168, RG 1.169, and RG 1.170 all state that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8]. IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 7-4.3.2-2010 [12], and the proposed draft version of IEEE Std. 7-4.3.2 [6] require software tools to be controlled under CM and provides requirements for determining the suitability of software tools; however, neither IEEE Std. 7-4.3.2-2003 [8] nor IEEE Std. 7-4.3.2-2010 [12] contain requirements for documenting the use, suitability, or characteristics of software tools. The proposed draft of IEEE Std. 7-4.3.2 [6] requires that the intended functionality and limitations of applicability for all software tools be identified and documented.

The only regulatory guidance applicable to software tool documentation is contained in RG 1.170, RG 1.171, and RG 1.172 and involve guidance for software testing activities. RG 1.170 endorses IEEE Std. 829-2008 [17], which contains requirements for manual or automated documentation of testing activities. RG 1.171 states that the documentation to support software unit testing should include tools needed for the accomplishment of unit testing; however, RG 1.171 does not provide guidance on the specific content of the documentation. RG 1.172 states that if requirements are generated using specification or representation tools, traceability should be maintained between these representations and the natural language descriptions of the software requirements that are derived from systems requirements and system safety analyses.

Future regulatory guidance and an attempt to influence industry standards might be considered so that software tool documentation requirements might be expanded to cover all software development tools and not just software tools used in software testing activities. Future regulatory guidance might also consider providing guidance on the desired content of software tool documentation (e.g., version, configuration, suitability, operating environment, acquisition, qualification, training, and vendor support).

## 3.7 Software Tool Development, Qualification, and Dedication

A technical basis that might be considered for future regulatory guidance includes an alternative method for developing or dedicating software tools for use in safety-significant software or logic development. Basic components are defined in 10 CFR 21.3 as items designed and manufactured under a QA program complying with 10 CFR Part 50, Appendix B, or CGIs that have successfully completed the dedication process.

The alternative method for developing or dedicating software tools would use the processes of RTCA DO-330 [14]. RTCA DO-330 [14] presents an entire qualification and development process for software tools that recommends the application of a life cycle process with associated objectives and activities to meet those objectives. The RTCA DO-330 [14] process

with appropriate Appendix B QA oversight or controls might be endorsed as an acceptable method of dedication for software tools with safety significance requiring that level of rigor. RTCA DO-330 [14] determines the processes, activities, and tasks for software tool qualification based on an assigned TQL. The TQL for use in RTCA DO-330 [14] is currently determined in a higher-level document, RTCA DO-178C [9]. Since the NRC may find that RTCA DO-178C [9] cannot be endorsed or adopted by future regulatory guidance because of industry-specific software levels, future regulatory guidance might consider including a determination of TQL based on tool type and software integrity level as determined from IEEE Std. 1012-2012 [5] or the RISC of the SSC containing the software as determined from 10 CFR 50.69. The TQL determination is necessary to select the minimum qualification activities specified in RTCA DO-330 [14].

A TQL determination that might be considered in future regulatory guidance is a phased implementation consistent with the phased implementation of tool categories. Future regulatory guidance might consider assigning a TQL-1 to software tools that have the highest potential impact on safety-related software with the highest integrity level or RISC. Lower integrity level software or lower RISC-based software are assigned lower values of TQL (i.e., TQL-2 through TQL-5) to tool types that do not require the highest level of qualification rigor and intensity. In Phase 1, the TQL determination might consider tool categories based on an expanded definition of tool types in the draft version of IEEE Std. 7-4.3.2 [6]. In Phase 2, the TQL determination might consider the expanded tool categories based on IEEE Std. 14102-2010 [19].

IEEE Std. 7-4.3.2-2010 [12] requires that the output of the software tool be verified and validated to the same integrity level as the safety-related software or that the software tool be developed using the same or an equivalent high quality life cycle process as required for the software upon which the tool is being used or commercially dedicated. The proposed draft version of IEEE Std. 7-4.3.2 [6] does not include an option that a software tool can be commercially dedicated and instead requires that the tool be developed using a quality life cycle process commensurate with the criticality of the safety functions performed by the end product.

Future regulatory guidance might consider that a software tool developed using the high-quality life cycle process of RTCA DO-330 [14] satisfies the requirement in the proposed draft of IEEE 7-4.3.2 [6]; however, the definition of basic component is inconsistent with the IEEE 7-4.3.2 [6] requirement. Future regulatory guidance might also consider that a software tool that has been commercially dedicated using the guidance of EPRI TR-1025243 [13] satisfies the definition of a basic component.

## 3.8 Software Tool Quality Assurance

Current regulatory guidance for software tools is limited and includes some high-level guidance for software tool QA.

The requirements of 10 CFR Part 50, Appendix B, apply to software tools used in designing, testing, operating, maintaining, and modifying safety-related software. Criterion II of 10 CFR Part 50, Appendix B, states that the QA program shall provide control over activities affecting the quality of the identified SSCs, to an extent consistent with their importance to safety and that

the QA program shall take into account the need for special controls, processes, test equipment, tools, and skills to attain the required quality.

RG 1.152 states that COTS systems and tools are likely to be proprietary and generally unavailable for review. The proposed draft version of IEEE Std. 7-4.3.2 [6] requires that measures be put into place to ensure that only approved and documented software tools are used to support the development of safety-related PDD and that these tools are utilized within their design capabilities.

Future regulatory guidance might consider the need to provide additional guidance based on industry practice for meeting the 10 CFR Part 50, Appendix B, QA criteria. Some of the industry guidance that might be considered includes QA guidance for COTS tools; guidance for situations where tool validation results are unavailable; guidance for ensuring the selection, verification, validation, and qualification of tools is documented; guidance for using techniques commensurate with the software's safety significance; and guidance for determining the suitability of software tools.

## 3.9 Software Tool Training

Current regulatory guidance for software tools is limited and does not include guidance for software tool training.

Neither IEEE Std. 7-4.3.2-2003 [8], endorsed by RG 1.152, IEEE Std. 7-4.3.2-2010 [12], nor the proposed draft of IEEE Std. 7-4.3.2 [6] address software tool training. However, 10 CFR Part 50, Appendix B, Criterion II, states that the QA program shall provide for indoctrination and training of personnel performing activities affecting quality as necessary to assure that suitable proficiency is achieved and maintained.

Future regulatory guidance might consider that software developers, software QA personnel, and independent V&V personnel should be trained in the use of tools used to develop safety-significant software.

## 3.10 Software Tool Configuration Management

Current regulatory guidance covers software tool CM. RG 1.169 states that tools used in the development of safety system software should be handled according to IEEE Std. 7-4.3.2-2003 [8] and that an SCM program operated by the organization using the tool that complies with IEEE Std. 828-2005 [22] should treat tools as configuration items. RG 1.169 also states that SCM plans should identify that software tools used in the production of safety system software be considered as configuration items and that items that may not change but are necessary to ensure correct software production, such as compilers, should also be configuration items, thereby ensuring that all factors contributing to the executable software are understood.

IEEE Std. 7-4.3.2-2003 [8], IEEE Std. 7-4.3.2-2010 [12], and the proposed draft of IEEE Std. 7-4.3.2 [6] all require that software tools used to support the software or logic life cycle processes of safety-related PDD be maintained under CM.

Future regulatory guidance might consider endorsing, adopting, or adapting the practice of RTCA DO-330 [14] that uses control categories as a function of TQL for determining the minimum set of CM process activities that should be performed when developing or qualifying software tools.

## 3.11 Software Tool Review, Approval, and Acceptance Criteria

A technical basis that might be considered for future regulatory guidance for the review, approval, and acceptance criteria of software tools is dependent on how the software tool is developed (i.e., developed under a 10 CFR Part 50, Appendix B, QA program; acquired as a CGI and dedicated; or developed or dedicated using the high-quality life cycle approach of RTCA DO-330 [14]). Within the regulatory framework of 10 CFR Part 21, the RTCA DO-330 [14] process might be endorsed as an acceptable means of acceptance of software tools requiring that level of rigor through dedication, regardless of the development process. Also, the use of a requirements compliance matrix might be considered as a review and approval aid to formally document compliance to software tool requirements once regulatory guidance and requirements in IEEE standards endorsed by RGs are finalized.

### 3.11.1 10 CFR Part 50, Appendix B, Development

Appendix B to 10 CFR Part 50 contains QA criteria for the design, manufacture, construction, and operation of SSCs. The pertinent requirements of 10 CFR Part 50, Appendix B, apply to all activities affecting the safety-related functions of SSCs including, but not limited to, designing, testing, operating, maintaining, and modifying. The requirements of 10 CFR Part 50, Appendix B, also apply to software tools used in designing, testing, operating, maintaining, and modifying safety-related software.

RGs 1.152, 1.168, 1.169, 1.170, 1.171, 1.172, and 1.173 describe acceptable methods of satisfying portions of the 10 CFR Part 50, Appendix B, requirements with respect to the use of computers in safety systems of nuclear power plants. These RGs endorse several IEEE standards that contain methods for promoting high functional reliability and design quality in safety system software, recognize detailed information in a few other industry standards, and incorporate recommendations and are consistent with the basic principles provided in IAEA NS-G-1.1 [33].

NUREG-0800, Appendix 7.1-D states that reviewers should thoroughly evaluate tool usage, should rely on testing as a method of evaluating the output of tools, and should be evaluated in the overall context of the quality control and V&V process.

The revisions that might be considered to the RGs and IEEE standards noted in this report along with future regulatory guidance that might be considered would expand the coverage of the RGs and IEEE standards to the planning, selection, use, documentation, development, qualification, dedication, QA, training, and CM of safety-significant software and software tools analogous to the coverage of safety-related software. The review, approval, and acceptance criteria for software tools developed under a 10 CFR Part 50, Appendix B, QA program might consider relying on existing regulatory guidance, future regulatory guidance that specifically addresses software tools, and confirmation that all the pertinent requirements of IEEE standards

endorsed by RGs 1.152, 1.168 through 1.173, and any future regulatory guidance have been satisfied based on an assigned software tool integrity level.

### 3.11.2 Commercial Dedication

Part 21 of Title 10 of the CFR (10 CFR Part 21) describes the commercial-grade dedication process and establishes that a CGI properly dedicated under the applicable controls of a nuclear QA program compliant with 10 CFR Part 50, Appendix B, may be considered equivalent to a component designed and manufactured under an Appendix B QA program, and thus may be designated as a basic component. The NRC has endorsed the acceptance methods of certain IEEE standards and EPRI documents (some generic and some applicable to software), with qualifications and restrictions (e.g., such as those in NRC GL 89-02 and GL 91-05), for commercial-grade dedication through certain generic communications and safety evaluations.

RG 1.152 identifies that IEEE Std. 7-4.3.2-2003 [8] provides general guidance for commercial-grade dedication of safety system designs that use computers but have not been specifically designed for nuclear power plant applications. IEEE Std. 7-4.3.2-2003 [8] describes the preliminary and detailed phases of the dedication process. In the preliminary phase, risks and hazards are evaluated, the safety functions are identified, CM is established, and the safety category of the system is determined. In the detailed phase, the CGI is evaluated for acceptability using detailed acceptance criteria. IEEE Std. 7-4.3.2-2010 [12] and the draft version of IEEE Std. 7-4.3.2 [6] both contain similar guidance for commercial-grade dedication. These standards describe four acceptance methods for evaluating traditional COTS equipment based on EPRI NP-5652 [34].

Additional commercial-grade dedication guidance is provided in GL 89-02 and GL 91-05. GL 89-02 addresses methods to detect substandard, counterfeit, or fraudulently marketed products. GL 89-02 conditionally endorses EPRI NP-5652 [34]. GL 91-05 clarifies the NRC staff's position on licensee implementation of commercial-grade dedication programs with respect to critical characteristics.

ISG-06 addresses the commercial-grade dedication of software through reference to requirements in IEEE Std. 7-4.3.2-2003 [8] and guidance in EPRI TR-106439 [32] and EPRI TR-107330 [35]. ISG-06 clarifies that the NRC staff recognizes two different ways a component can be approved for use in safety-related applications. If basic component critical characteristics can be verified, then it may be designed and manufactured as a CGI and commercially dedicated. Otherwise, it must be designed and manufactured under a QA program.

EPRI NP-5652 [34] and EPRI TR-102260 [11] discuss a generic process, including a graded approach, for the acceptance of commercial-grade hardware components and parts for use in a nuclear safety-related application. EPRI TR-106439 [32] and EPRI TR-107339 [36] contain guidance on the evaluation and acceptance of commercial-grade digital equipment as part of the commercial-grade dedication process for nuclear safety applications. The primary focus of EPRI TR-107339 [36] is on the differences in the technical evaluation and acceptance process required for digital, software-based equipment and systems. EPRI TR-1025243 [13] provides

guidance for using the commercial-grade dedication process to accept safety-related, commercial-grade design and analysis tools used in the design and analysis of safety-related plant SSCs. EPRI TR-1025243 [13] extends the commercial-grade dedication process described in EPRI NP-5652 [34] from hardware components to design and analysis software tools used to support the hardware design and development.

EPRI TR-1025243 [13] commercial-grade dedication involves a technical evaluation followed by an acceptance process to provide a reasonable assurance that the software tool procured meets specified requirements. The technical evaluation for a design or analysis computer program consists of several activities that include the identification of critical characteristics and choosing the acceptance methods. As defined in 10 CFR Part 21, critical characteristics are those important design, material, and performance characteristics of a CGI that, once verified, provides reasonable assurance that the CGI will perform its intended safety function. Tool qualification might be considered as a critical characteristic of a software tool and completion of tool qualification might be considered as an acceptance criterion. Use of a TQL and using a four level schema to categorize safety-related, nonsafety-related, and augmented quality software tools might be considered as a method of expanding the scope of EPRI TR-1025243 [13] from design and analysis software tools to all software development and support tools.

Four current methods of accepting CGIs include: (1) special tests and inspections, (2) survey of commercial-grade supplier, (3) source verification, and (4) supplier and item performance history. Most of the software tools used to develop safety-related software are COTS software tools. Commercial-grade supplier surveys and source verification are not viable options for verifying critical characteristics of COTS computer programs because dedication of the computer programs occurs well after the programs have been developed and because of the unwillingness of COTS program vendors to allow outside access to proprietary processes and documentation. Therefore, future regulatory guidance might consider that the acceptance of COTS software tools should rely on special tests and inspections, and supplier item performance history.

The review, approval, and acceptance criteria for software tools acquired as a CGI and commercially dedicated under an Appendix B QA program might consider relying on the commercial-grade dedication process of EPRI TR-1025243 [13] and confirmation that the tool qualification requirements of RTCA DO-330 [14] have been satisfied as part of the critical characteristic verification.

### 3.11.3 RTCA DO-330 Development

The review, approval, and acceptance criteria for software tools developed or dedicated using the high-quality life cycle approach of RTCA DO-330 [14] might consider relying on future regulatory guidance that defines the TQL based on the software tool type and the importance of the safety-significant software, the confirmation that the tool qualification requirements of RTCA DO-330 [14] have been satisfied as part of the critical characteristic verification, and successful completion of the software tool liaison process of RTCA DO-330 [14]. Within the existing regulatory framework, development of software tools under the RTCA DO-330 [14] process, when conducted under appropriate Appendix B QA controls, could be viewed as an acceptable

or even preferred basis for acceptance for use of software tools in safety-significant applications.

Future regulatory guidance for software tools might consider endorsing RTCA DO-330 [14] which presents an entire development, review, and approval process for software tools. The tool qualification liaison process described in RTCA DO-330 [14] contains similar activities and tasks as ISG-06 to facilitate the review and approval of software tools. The tool qualification liaison process identifies the activities that are required to obtain approval of the tool qualification as a function of TQL. The essential elements of the tool qualification liaison process include submitting evidence that tool problems and functional limitations have been analyzed, resolved, and properly reported; the tool qualification process has been completed successfully; and all objectives that should have been satisfied in the tool qualification process have actually been satisfied.

# 4 References


[1] J. Servatius, P. Butchart, "*Evaluation of Guidance for Tools Used to Develop Safety-Related Digital Instrumentation and Control Software for Nuclear Power Plants – Task 1 Report: Survey of the State of Practice"*, Information Systems Laboratories Inc., ISL-ESD-TR-13-05, July 2013

[2] J. Servatius, S. Alexander, T. Gitnick, "*Evaluation of Guidance for Tools Used to Develop Safety-Related Digital Instrumentation and Control Software for Nuclear Power Plants – Task 2 Report: Analysis of the State of Practice"*, Information Systems Laboratories Inc., ISL-ESRD-TR-14-03, August 2014

[3] IEEE, "*IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations"*, IEEE Std. 603-2009, November 2009

[4] IEEE, "*IEEE 100 The Authoritative Dictionary of IEEE Standards Terms*", IEEE Std. 100-2000, Seventh Edition, February 2007

[5] IEEE, "*IEEE Standard for System and Software Verification and Validation"*, IEEE Std. 1012-2012, May 2012

[6] IEEE, "*IEEE Standard Criteria for Programmable Digital Devices in Safety Systems of Nuclear Power Generating Stations"*, IEEE Std. 7-4.3.2 (DRAFT), Issue Pending

[7] IEEE, "*IEEE Standard for Configuration Management in Systems and Software Engineering"*, IEEE Std. 828-2012, March 2012

[8] IEEE, "*IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations"*, IEEE Std. 7-4.3.2-2003, December 2003

[9] RTCA, "*Software Considerations in Airborne Systems and Equipment Certification"*, RTCA DO-178C, December 2011

[10] IEEE, *"Systems and Software Engineering - Software Life Cycle Processes",* IEEE Std. 12207-2008, Second Edition, February 2008

[11] EPRI, "*Supplemental Guidance for the Application of EPRI Report NP-5652 on the Utilization of Commercial Grade Items"*, EPRI TR-102260, March 1994

[12] IEEE, "*IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations",* IEEE Std. 7-4.3.2-2010, August 2010

[13] EPRI, "*Plant Engineering: Guideline for the Acceptance of Commercial-Grade Design and Analysis Computer Programs Used in Nuclear Safety-Related Applications"*, EPRI TR-1025243, Revision 1, December 2013

[14] RTCA, "*Software Tool Qualification Considerations"*, RTCA DO-330, December 2011

[15] IEEE, "*IEEE Standard for Developing a Software Project Life Cycle Process"*, IEEE Std. 1074-2006, July 2006

[16] IEEE, "*Systems and Software Engineering - System Life Cycle Processes"*, IEEE Std. 15288-2008, Second Edition, February 2008

[17] IEEE, "*IEEE Standard for Software and System Test Documentation"*, IEEE Std. 829-2008, July 2008

[18] IEEE, "*IEEE Standard for Software Reviews and Audits"*, IEEE Std. 1028-2008, August 2008

[19] IEEE, "*IEEE Standard for Adoption of ISO/IEC 14102:2008, Information Technology - Guideline for the Evaluation and Selection of CASE Tools"*, IEEE Std. 14102-2010, September 2010

[20] IEEE, "*Systems and Software Engineering - Life Cycle Processes - Requirements Engineering"*, First Edition, IEEE Std. 29148-2011, December 2011

[21] IEEE, "*IEEE Standard for Software Verification and Validation"*, IEEE Std. 1012-2004, June 2005

[22] IEEE, "*IEEE Standard for Software Configuration Management Plans"*, IEEE Std. 828-2005, August 2005

[23] IEEE, "*IEEE Recommended Practice for Software Requirements Specifications"*, IEEE Std. 830-1998, 1998

[24] IEC, "*Nuclear Power Plants - Instrumentation and Control Systems Important to Safety - Software Aspects for Computer-Based Systems Performing Category A Functions"*, IEC 60880 Ed. 2.0, Second Edition, May 2006

[25] IEC, "*Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems"*, Second Edition, IEC 61508 Ed. 2.0, April 2010

[26] NASA, "*Standard for Models and Simulations",* NASA-STD-7009, July 2008

[27] IEEE, "*IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations"*, IEEE Std. 603-1991, 1991

[28] National Energy Institute, "*Guidance Document to Implement Policy for Software Quality Assurance in the Nuclear Power Industry",* NITSL-SQA-2005-02, Revision 1, January 2009

[29] IEEE, "*IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations"*, IEEE Std. 603-1998, 1998

[30] IEEE, "*IEEE Standard for Software Unit Testing"*, ANSI/IEEE Std. 1008-1987, 1986, Reaffirmed 1993

[31] NASA, "*Software Assurance Standard",* NASA-STD-8739.8, Change 1, July 2004

[32] EPRI, "*Guideline on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications"*, EPRI TR-106439, October 1996

[33] IAEA, "*Software for Computer Based Systems Important to Safety in Nuclear Power Plants – Safety Guide"*, IAEA NS-G-1.1, 2000

[34] EPRI, "*Guideline for the Utilization of Commercial Grade Items in Nuclear Safety Related Applications (NCIG-07)",* EPRI NP-5652, June 1988

[35] EPRI, "*Generic Requirements Specification for Qualifying a Commercially Available PLC for Safety-Related Applications in Nuclear Power Plants*", EPRI TR-107330, December 1996

[36] EPRI, "*Evaluating Commercial Digital Equipment for High Integrity Applications"*, EPRI TR-107339, December 1997