Digital Systems Software Requirements Guidelines

Vol. 1. Guidelines

Contract RES-00-037

M. Hecht
H. Hecht

SoHaR Incorporated
Beverly Hills, CA

for

Nuclear Regulatory Commission
Washington, D.C.

Abstract

This document identifies guidelines that review high-integrity software requirements documents in nuclear power plants and addresses item 4 (Adequacy of System Functions and Commitments) of the seven review requirements for digital systems listed in NUREG-0800. The framework used for developing the guidelines was taken from the Nuclear Regulatory Commission Standard Review Plan and Branch Technical Position (HICB-14), which was adapted to the software. The topic areas are conformance with the SRP, accuracy, functionality, reliability, robustness, maintainability, security, timing, and human-computer interaction. A checklist for the requirements and their ordering for a Software Requirements Specification in accordance with IEEE Std-830 is discussed.

Table of Contents

List of Tables

Executive Summary

Software requirements specification is an important source of errors in system development (NUREG-0800, USNRC, 1997c, p. A-7). A significant proportion (if not the majority) of all accidents in which software was involved could be traced to requirements flaws—that is, incomplete or wrong assumptions about how the system operated. Not only do missing, inaccurate, or incomplete requirements lead to flaws in software development, they also prevent these flaws from being detected during verification and validation. For example, functional testing is based on the requirements; a missing or inaccurate requirement will therefore not be detected. Structural testing is based on the developed code; an unstated requirement is unlikely to be implemented and will therefore not be detected. Integration testing sometimes detects the omissions or inaccuracies, but more frequently it is only through failures in actual operation that these defects are made manifest.

This is the first of two volumes. This volume contains guidelines for software requirements. The guidelines, together with a methodology for their application, are intended to provide reviewers with a tool to assess the correctness, completeness, and accuracy of requirements for safety-related digital systems.

The guidelines address the following nine major areas:

! *Conformance with the Standard Review Plan (SRP) NUREG-0800:* Specifically, conformance with the Review Process for Digital Instrumentation Systems in Appendix 7.0-A, the Acceptance Criteria for I&C Systems Important to Safety in Appendix 7.1-A, the Guidance for Evaluation of Conformance with ANSI/IEEE Std. 279 in Appendix 7.1-B, and the Guidance for Evaluation of Conformance with IEEE Std. 603 in Appendix 7.1-C

! *Accuracy :* Guidelines associated with numerical accuracy and including precision

! *Functionality:* Guidelines for the specification of functions that must be performed for each mode of operation with emphasis on completeness

! *Reliability:* Guidelines related to system-level requirements for failure rates and recovery times

! *Robustness:* Guidelines related to the specification of behavior of the software in the presence of unexpected, incorrect, or anomalous and improper (1) input, (2) hardware behavior, or (3) software execution

! *Maintainability:* Guidelines related to both online, in-service testing and diagnostics and to the means by which the source code reduces the likelihood that faults will be introduced during changes made after delivery

! *Security:* Guidelines related to requirements of the software to detect, prevent, or mitigate threats, including access control restrictions

! *Timing:* Guidelines related to functions that must operate within specific timing constraints

! *Human-Computer Interaction:* Guidelines related to software requirements that affect the operator displays, annunciators, and controls

This organization was derived from the NRC Human Factors and Instrumentations and Control Branch (HICB) Branch Technical Position BTP-14. A set of checklists based on these topics and their ordering for a Software Requirements Specification (SRS) in accordance with IEEE Std 830 is also contained in this report.

This second volume in this project contains a set of 45 failures that illustrate the need for and the importance of specific requirements. Cross-reference tables link the requirements guidelines to the failure descriptions and the failure descriptions to the requirements guidelines.

Acknowledgments

We wish to acknowledge the support and interest of the NRC Office of Research and in particular, that of Roman Shaffer, the project manager. His active participation and guidance were essential contributions to the results of this project.. We also wish to acknowledge the former project manager, Mr. Robert Brill, whose vision initiated this project. Finally, we wish to acknowledge the helpful and energetic comments from the panel of reviewers whose input significantly impacted the design guidelines and examples.

Acronyms

A/D	Analog to Digital
APU	Auxiliary Power Unit
ARTCC	Air Route Traffic Control Center
CFR	Code of Federal Regulations
CCCH	Central Computing Complex-Host
CCU	Central Control Unit
COTS	Commercial, Off-the-Shelf
CPU	Central Processing Unit
D/A	Digital to Analog
DHCP	Dynamic Host Configuration Protocol
EPRI	Electric Power Research Institute
ESFAS	Engineered Safety Function Actuation System
FADEC	Full Authority Digital Engine Controller
FDIO	Flight Data Input/Output
FMEA	Failure Modes and Effects Analysis
FMECA	Failure Modes and Effects and Criticality Analysis
HICB	Division of Reactor Controls and Human Factors, Instrumentation and Controls Branch
HVAC	Heating, Ventilation, and Air Conditioning System
I&C	Instrumentation and Control
IEEE	Institute of Electrical and Electronics Engineers
LER	Licensee Event Report
LSB	Least Significant Bit
NASDAQ	National Association of Securities Dealers Automated Quotation
NRC	U. S. Nuclear Regulatory Commission
P&ID	Process and Instrumentation Diagram
PLC	Programmable Logic Controller
RTS	Reactor Trip Systems
PDS	Previously Developed Software
PID	Proportional Integral Derivative Control
UPM	Uninterruptible Power Module
SAR	Safety Analysis Report
SoW	Statement of Work
SRP	Standard Review Plan (NUREG-0800)
SRS	Software Requirements Specification
TRACON	Terminal Radar Approach Control Center
TTL	Transistor-Transistor Logic
V&V	Verification and Validation

1. Introduction

This report provides guidance to the NRC for reviewing high-integrity software requirements documents in nuclear power plants. The term "requirements" is used here in the same sense as a *critical characteristic* in NUREG-0800 (USNRC, 1997c).[1] Requirements specification and allocation activities, particularly for software, have proven to be an important source of errors in system development (NUREG-0800, USNRC, 1997c, p. A-7). At least one author has unequivocally stated that the vast majority of accidents in which software was involved could be traced to requirements flaws—that is, incomplete or wrong assumptions about how the system operated (Leveson, 1995, p. 359). Correction of requirements errors can consume 25 to 40 percent of the project effort and budget (Leffingwell and Widrig, 1999). Not only do missing, inaccurate, or incomplete requirements lead to flaws in software development, they also prevent these flaws from being detected during V&V. For example, functional testing is based on the requirements; a missing or inaccurate requirement will therefore not be detected. Structural testing is based on the developed code; an unstated requirement is unlikely to be implemented and will therefore not be detected. Integration testing sometimes detects the omissions or inaccuracies, but more frequently it is only through failures in actual operation that these defects are made manifest.

This report was prepared as an account of work sponsored by the Nuclear Regulatory Commission, an agency of the United States government. Neither the United States government, nor any agency thereof, nor any employee, makes any warranty, expressed or implied, or assumes legal liability or responsibility for any information, apparatus, product, or process disclosed in this report, or represents that its use by a third party would not infringe privately owned rights. The opinions, findings, conclusions, and recommendations expressed herein are those of the authors and do not necessarily reflect the views of the NRC.

1.1 Scope

This report focuses on software requirements and addresses Item 4 (Adequacy of System Functions and Commitments) of the seven review requirements for digital systems listed in NUREG-0800 (USNRC, 1997c, Appendix 7.0-A, p. A-5). Issues related to design, implementation, verification and validation, and the development process are covered in other industry standards and NRC reports (see reference list). In this document, these topics are covered only to the extent that they affect requirements.

This report contains guidelines and the technical basis for the review of software requirements for safety systems and is written under the requirements of Contract NRC-04-00-037. The guidelines, together with

[1] The distinction between a requirement and a characteristic in NUREG-0800 is that the former refers only to a functional attribute and the latter refers to a functional attribute plus the additional attributes of robustness, testability, and dependability. Our use of the term "requirement" reflects general usage in software development to encompass these additional attributes.

a methodology for their application, are intended to provide reviewers with a tool to assess the correctness, completeness, and accuracy of requirements for safety-related digital systems.

1.2 Methodology of This Investigation

This work is divided into the following two tasks:

1. Methodology for Specification Review: Attributes related to safety were identified in relevant standards and the current literature. Table 1–1 lists the sources from which the majority of the attributes were extracted.

2. Catalog of Complex Software Failures Based on Field Data: In this task, we gathered accounts of failures from a variety of sources. In some cases, the requirements of those systems were revised as a result of the analysis of the failures.

The following subsections discuss the methodology in greater detail.

1.2.1 Task 1 Methodology

In Task 1, generic attributes were defined through the following three-step process:

1. Identify a useful classification for top-level safety-related requirements.

2. Identify lower-level requirements from guidelines taken from computer systems and general systems safety-related literature pertinent to software.

3. Rank guidelines by importance to safety to set review priorities.

The most appropriate basis for the development of the guidelines was found in the Standard Review Plan, SRP, (USNRC, 1997c) and in Branch Technical Position HICB-14 (USNRC, 1997a). Because the objective of the SRP is the system-level review, some provisions had to be interpreted to be applicable for the review environment of the current guideline, which is software.

Once the top-level classification guidelines were defined, the next step was to compile, edit, and classify safety-related requirements from previous work. Table 1–1 lists the sources used for this work. They include both NRC-suggested sources, as well as other sources that the authors consider significant to safety.

Table 1-1. Sources Used for the Identification of Software Safety Attributes

NUREG/CR-6680, "Review Templates for Computer Based Reactor Protection Systems," (Johnson, 1998)
NUREG/GR-0019, "Software Engineering Measures for Predicting Software Reliability in Safety Critical Systems," (Smidts, 2000)
NUREG-1709, "Selection of Sample Rate and Computer Word Length in Digital Instrumentation and Control Systems," (USNRC, 2000)
NUREG-0800, Standard Review Plan Chapter 7, "Instrumentation and Controls," (USNRC, 1997c)
IEEE Std 830-1993, "IEEE Recommended Practice for Software Requirements Specifications,"
Branch Technical Position HICB-14, "Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems," (USNRC, 1997a)
Regulatory Guide 1.172, "Software Requirements Specifications for Digital Computer Software Used in Safety Systems of Nuclear Power Plants," (USNRC, 1997b)
NUREG/CR-6113, "Class 1E Digital System Studies," (Hecht, 1993)
NUREG/CR-6293, "Verification and Validation Guidelines for High Integrity Systems," (Hecht, 1995)
NUREG/CR-6101, "Software Reliability and Safety in Nuclear Reactor Protection Systems," (Lawrence, 1993)
NUREG/CR-6463, "Review Guidelines on Software Languages for Use in Nuclear Power Plant Safety Systems," (Hecht, 1996)

The guideline ranking was performed using engineering judgment to the importance of each criterion in a generic safety system. However, these should be considered representative rather than absolute rankings. The actual importance of the guidelines will vary based on the functions, complexity, and architecture of the actual systems. Thus, the user of this document should consider the guidelines in light of specific system characteristics.

1.2.2 Task 2 Methodology

In Task 2, we examined failure reports and other indirect accounts of failures, edited them, and related them back to the requirements identified in Task 1. Table 1–2 shows the failure data sources used for this task.

Table 1-2. Failure Data Sources for Validation of Attributes

Data Source	System Characteristics	Period
Industrial Automation and Control Mailing List	Various industrial computing platforms including PLCs, PCs, and instrumentation buses. Mailing list on problems and issues.	1995–present
Risks to the Public electronic newsletter compiled by P. Neumann, SRI (Neumann, 2001)	Compilation of accounts of failures from a variety of systems. Those used in this study include flight control and ground transportation.	1986–present
Eagle 21 LERs	Reactor safety system	1990–present
FAA failure reports	Safety critical air traffic control systems	1995–present

Although many hundreds of failure reports were examined, only 45 were included in the listing of failures. The criteria used for inclusion were:

a. The failure resulted from multiple causes (this was specified in the SOW).
b. The proper definition and implementation of a software requirement would have mitigated or prevented the failure.
c. The failure had safety significance.
d. The failure was adequately documented to determine cause(s) and effect(s).

1.2.3 Technical Basis

Five criteria for a technical basis for the review of digital systems were defined in NUREG/CP-0136 (Beltracchi, 1994, p. 39). Table 1–3 shows how these criteria have been addressed in this document.

The guidelines developed in this work provide a basis for auditing software requirements in safety systems, but they are not exhaustive because they are written without knowledge of the specific systems and environments to which they may be applied. On the other hand, not all guidelines included in this document may be applicable to a specific project because of the presence or absence of design constraints, the specific functional and performance characteristics of the system under development, or other factors. Use of these guidelines will assist auditors in identifying problems in the software specification of safety systems, but it does not guarantee that such problems can be completely eliminated.

Systematic collection of reviewers' comments on ease of use and results achieved with this edition of the guidelines will facilitate periodic updating. Generation of these guidelines is an empirical undertaking, and iterative evaluation is essential for perfecting the review activities.

Table 1-3. Technical Basis Criteria and How They Are Addressed

Technical Basis Criterion	How Addressed
1. The topic has been clearly coupled to safe operations.	The rationale for each guideline has been stated in this document.
2. The scope of the topic is clearly defined.	Section 1.1 describes the scope of safety system concerns.
3. A substantial body of knowledge exists, and the preponderance of the evidence supports a technical conclusion.	Extensive literature on systems safety and safety related to digital systems has been reviewed and applied in this study. Guidelines were reviewed by independent subject matter experts.
4. A repeatable method exists to correlate relevant characteristics with performance.	This topic is not addressed in this document. Due to the paucity of failure data on digital nuclear safety systems and the rarity of events resulting in challenges to such systems, a repeatable method for correlating the identified attributes with safe operation is not possible at this time. However, we did include data from other systems involving safety significant failures where correct definition and implementation of requirements could have mitigated the impact.
5. A threshold for acceptance can be established.	This topic is not directly addressed in this study. The guidelines identify qualitative attributes rather than quantitatively measurable parameters. Substantial progress in research on the quantitative failure behavior of high integrity software is necessary to formulate a threshold.

1.3 Contents Overview

The next chapter lists guidelines to evaluate the completeness of software requirements important to safety. In that chapter the guidelines are arranged by topics largely derived from BTP-14. The conduct of a review will be constrained by the structure and ordering of the requirements documentation, usually in the form of a software requirements specification (SRS). The methodology for conducting a review based on the topics of Chapter 2 within the framework of a typical SRS is discussed in Chapter 3. Chapter 4 is a brief concluding section that also contains recommendations for further research. Appendix A is a glossary of significant technical terminology used in this report, together with the source of the definitions. Appendix B contains checklists for applying the guidelines in a review. Appendix C summarizes the "Importance to Safety" ratings and explains the rationale for their use. Appendix D links standards and related documents to the guidelines. Appendix E presents the backgrounds of the authors and technical reviewers of this document. Volume 2 lists failures of complex systems that relate to the requirements guidelines listed in Chapter 2.

2. Review Guidelines for Safety and Safety-Related Software Requirements

This chapter describes guidelines for reviewers of requirements for safety and safety-related systems. The guidelines described in this chapter are not requirements in themselves, but a description of what should be included within the requirements. Although these guidelines are specific to nuclear reactor safety systems, they are generic in the sense that they are independent of the architecture and implementation.

The guidelines were defined hierarchically by subject. The choice of a hierarchy was determined by what would be meaningful and relevant to NRC reviewers and licensees. Hence, the general topic areas identified in NRC Division of Reactor Controls and Human Factors, Instrumentation and Controls Branch (HICB), Branch Technical Position (BTP)-14 were used as a starting point and were elaborated, where necessary, to the needs of software requirements. The end result includes nine top-level guideline categories of which the first contains requirements directly derived from the Standard Review Plan, while the latter eight cover topics in HICB BTP-14.

- *Conformance with the Standard Review Plan (SRP) NUREG-0800:* Specifically, conformance with the Review Process for Digital Instrumentation Systems in Appendix 7.0-A, the Acceptance Criteria for I&C Systems Important to Safety in Appendix 7.1-A, the Guidance for Evaluation of Conformance with ANSI/IEEE Std. 279 in Appendix 7.1-B, and the Guidance for Evaluation of Conformance with IEEE Std. 603 in Appendix 7.1-C.

- *Accuracy :* Guidelines associated with numerical accuracy and including precision

- *Functionality:* Guidelines for the specification of functions that must be performed for each mode of operation with emphasis on completeness

- *Reliability:* Guidelines related to system level requirements for failure rates and recovery times

- *Robustness:* Guidelines related to the specification of behavior of the software in the presence of unexpected, incorrect, or anomalous and improper (1) input, (2) hardware behavior, or (3) software execution

- *Maintainability:* Guidelines related to both online, in-service testing and diagnostics and to the means by which the source code reduces the likelihood that faults will be introduced during changes made after delivery

- *Security:* Guidelines related to requirements of the software to detect, prevent, or mitigate threats, including access control restrictions

! *Timing:* Guidelines related to functions that must operate within specific timing constraints

! *Human-Computer Interaction:* Guidelines related to software requirements that affect the operator displays, annunciators, and controls

The relationship between these top-level attributes and those defined in BTP-14 are shown in Table 2–1.

Table 2-1. Relationship of Terminology Used Here and BTP-14 Terminology

BTP-14 terminology	Guideline Categories in This Report
[N/A]	Conformance with NUREG-0800 Criteria
Accuracy	Accuracy
Functionality	Functionality
Reliability	Reliability
Robustness	Robustness
Safety	Safety
Security	Security
Timing	Timing
[N/A]	Human-Computer Interaction

The following sections discuss each of these guideline categories in greater detail.

2.1 Conformance with the Standard Review Plan (SRP)

Importance to safety ranking: HIGH (all guidelines in this section)

Software requirements should conform to the criteria specified in the SRP and the documents referenced therein. Since these are system-level documents, the criteria (requirements) cannot be used directly for software requirements. However, these criteria can be tailored. The following criteria from Appendix 7.1-B (USNRC, 1997c) have been modified in the following ways:

1. *Completeness with respect to design basis:* The software requirements should address all system functions allocated to software from the design basis specified in Appendix 7.1-B as necessary to fulfill the system's safety intent. Examples of the need for this fundamental requirement can be seen in Failure Description Nos. 9 and 29.[2] Neither one of these was in an environment that invoked NUREG-0800.

[2] See Volume 2 for all failure descriptions.

2. *Consistency*: The software requirements should be internally consistent and consistent with the design basis and the plant safety analysis, including the design basis event analysis (Chapter 15 of the Safety Analysis Report); the mechanical and electrical system designs; and other plant system designs. All requirements should be traceable to criteria in IEEE Std 7-4.3.2-1993 and IEEE Std 603-1991, including, most notably, the single failure criterion. Requirements should also be mutually consistent with respect to function, performance, security, and safety.

3. *Correctness:* The software requirements should be technically accurate and up-to-date. The correctness requirement is amplified in the discussion of precision in Section 2.2. An example of the violation of this seemingly most obvious of guidelines is shown in Failure Description No. 5, where incorrect system requirements resulted in the locking out of a backup diesel generator system in a hospital. Description No. 26 discusses errors in a centralized British train status monitoring system.

4. *Traceability:* All requirements should be traceable to the system-level requirements, the system architecture, and the software design. It should be possible to trace the information in each requirement back to the safety analyses (Chapter 15 of the SAR), plant system design documents, regulatory requirements, applicant/licensee commitments, or other plant documents.

5. *Unambiguity:* The information provided in the software requirements, taken alone and in combination, should have one and only one interpretation.

6. *Verifiability:* The information provided in the software requirements should be stated or provided in such a way as to facilitate the establishment of verification criteria and the performance of analyses and reviews of the various safety systems. The information should include:

    ! The means by which meeting the requirement is satisfied (test, analysis, demonstration of executable requirements, inspection, automated verification)

    ! Acceptance criteria (data to be recorded, analysis procedures on the data, expected results)

    ! Test procedures (including specification of data by which acceptance can be determined, test configuration, data simulation and recording equipment, input value sets, limits, number of repetitions, initial conditions, data rates, etc.)

7. *Prioritization:* Software functions, operating procedures, input, and output should be classified according to their importance to safety. Requirements important to safety should be identified as such in the SRS. The identification of safety items should include safety analysis report requirements, as well as abnormal conditions and events as described in Regulatory Guide 1.152 (USNRC, 1995).

2.2 Guidelines Related to Accuracy

This guideline includes attributes associated with numerical accuracy, precision, and range of sensor values, actuator outputs, and internal variables. Accuracy and precision are important to safety when:

! Thresholds are approached.

! The algorithms result in taking a small difference between two large numbers.

! The algorithms can result in the possibility of a denominator being sufficiently close to zero to cause an overflow condition in the safety system.

! Iterative algorithms can cause cumulative effects from initially small errors.

The rationale for inclusion of these guidelines is based on both existing standards and inherent attributes of digital systems. SRP Appendix 7.1-B, Section 1 (referring to IEEE Std. 279) requires, in part, the identification of system accuracies, ranges, and rates of change of sensed variables to be accommodated. Consequences of improperly specified precision of input, output, and stored variables include insufficient accuracy or resolution in the monitoring of plant parameters and resultant control (or actuation of safety systems), improperly calculated results for display, or insufficient dynamic range for variables resulting in overflow conditions with unpredictable results.

Accuracy and precision guidelines affect software requirements related to input and output variables, variables stored internally in the processor in which the software is executed, or all of them. The following subsections discuss these issues by general attributes (affecting both input and output and internally stored variables), input and output precision, and precision related to internally stored variables.

2.2.1 General Guidelines

Importance to safety ranking: HIGH (all guidelines in this section)

This section addresses attributes of requirements for accuracy that cover both input and output and stored variables. These attributes include:

1. *Explicit precision definition:* Precision requirements should be stated explicitly and numerically for *all* input, output, and internal variables in terms of both relative values (e.g., 0.1%) and storage size (e.g., 16 bits). These requirements should be based on an analysis of the underlying physical processes, as well as the attributes of the hardware involved in the sensing, conversion, transmission, and storage of such variables (from the system-level requirements and hardware specifications). The requirements should explicitly identify the minimum precision necessary for the execution of algorithms under all possible conditions (including spurious instrumentation readings). In some cases, the precision requirements may be implemented (programmed) by defining data types or by using the defaults within the runtime environment (including hardware and software), effectively making an implicit definition of precision requirements. However, an explicit statement of requirements is necessary in order to allow for subsequent V&V activities to ensure that each variable meets the minimum required precision, even if they are implicitly defined using general type definitions.

2. *Explicit upper and lower bounds definition:* Requirements should identify upper and lower bounds for each process variable and parameter. For variables associated with physical measurements, upper and lower bounds should be set on the basis of what could conceivably be encountered by the digital control system, rather than what the physical limits might be. Setting ranges on variables can facilitate early detection of computing anomalies (e.g., a temperature, flow rate, or pressure reading being negative). An example, reported in Failure Description No. 12, was the GPS rollover that occurred in August 1999. The nature of the problem was that the date was stored as a fixed width integer and the software in the satellites' clocks had been configured to deal with 1024 weeks. Consequently, on 22 August 1999 (which is week 1025), some GPS receivers reverted to week one (i.e., 6 January 1980).

3. *Ensure data types appropriate to the variables*: A reviewer of requirements should ensure that the appropriate data type has been selected for each process variable. For example, fixed or floating point data types should not be used for integer quantities (e.g., counters) or discrete (on/off) sensors or actuators. Lack of requirements to ensure the use of appropriate data types may have contributed to the decimalization error reported in Failure Description No. 2 (see 2.1-3 above)

2.2.2 Specification of Physical Input and Output Quantity Accuracy

Importance to safety ranking: HIGH (all guidelines in this section)

These guidelines relate to requirements that define the specification of input and output quantities. Where the output of one algorithm becomes the input of another, the cumulative effect of error build-up must be considered. A/D precision relates to the precision of inputs from sensors and outputs to actuators. Precision requirements arise out of NUREG-0800 App. 7.1-C, Section 5, that states that safety systems shall "...with precision and reliability maintain plant parameters..."

1. *There must be consistency with hardware capacities:* Precision requirements must be consistent with the constraints imposed by word length, least significant bit error, and headroom (defined under Item 2) for each input and output quantity. For quantities that are input and output directly to the I&C system, word length is governed not by the CPU, but by the analog to digital converters that handle the sensor and transducer values. As a result, a least significant bit (LSB) error of half of the lowest bit is introduced. In addition, plant signals pass through a variety of electrical interfaces. The measured value may also be affected by power fluctuations and electromagnetic interference. The resulting error may affect the lowest two or three bits of the converted value.

   The quantity "dynamic range" can be used as a measure of consistency of the software requirements with the capabilities of the underlying hardware. It is defined by the following relation.[3]

   $$\text{Dynamic Range} = (2^n - 1) / \text{LSB Error}$$

2. *Accuracy requirements for monitoring and controlling the process must not be greater than the resolution capabilities of the hardware and must allow for headroom:* Resolution describes the minimum error. The resolution should be sufficient such that setpoints, comparisons, and calibrations meet the criteria and system requirements imposed by safety considerations, e. g., the ability to specify a trip value that assures that the technical specification is adhered to.

   Resolution is the inherent error simply in the process of transforming a continuous analog signal into a stair-step digital signal. The size of these steps should be such that the inherent linearity error does not significantly affect the process outcome. Resolution is determined by the following relation (NUREG-1709, USNRC, 2000):

   $$\text{Resolution} = \text{Range} / (2^n - 1)$$

[3] In the following equations *n* refers to the number of bits in the computer word (word length).

The range should be defined in consideration of both the anticipated maximum physical range of the measured quantity and additional "headroom" that will be necessary to encounter system upsets, noise, or other anomalies. "Headroom" is the ability of the A/D converter to represent analog values beyond the normal operating range. For example, for anticipated maximum physical range of 10 V and an A/D word length of 20 bits, the resolution would be 10 V/(2^{20}-1), or 9.54 ìV. However, the anticipated headroom might double the range, so that the minimum sensed resolution would have to be limited to 19.1 ìV.

3. *Accuracy requirements must account for conversion error*s: The precision required for operation of the I&C function should not be greater than the maximum errors within the A/D conversion. Factors affecting A/D conversion (in addition to the errors discussed above) include:

   ! *Linearity error* is the maximum deviation of the A/D converter from the ideal to the actual across the range of the A/D converter. A second type of linearity error, dynamic non-linearity, or DNL, reflects the maximum deviation in going from one step to the other and may be quite significant where measurements are being done on the difference between values (e.g., flux level changes).

   ! *Gain error* is the deviation between the full scale actual change in the input signal and the output of the A/D converter.

   ! *Offset error* is a consistent shift (higher or lower) between all values of the A/D converter and the input value.

   ! *Non-monotonicity* is the deviation of the direction (sign) of the A/D conversion relative to a change in the input signal (i.e., decreasing when the input signal is increasing or increasing when the signal is decreasing).

   ! *Missing code* is a deviation of the A/D conversion in which one or more increments of the input signal are not reflected in the A/D output.

   A conservative way of accounting for all of these errors is to sum their absolute values. The sum of all such errors should be less than that required for acceptable operation of the I&C function for each.

4. *Accuracy requirements should reflect the presence and characteristics of anti-aliasing filters:* Anti-aliasing filters can either increase or reduce the accuracy of the signals, depending on the signal spectrum and the filter characteristics. Further information on this subject can be found in NUREG-1709 (USNRC, 2000).

5. *Accuracy requirements should properly reflect the accuracy and bias of the sensors and actuators:* The requirements and algorithms should not depend on data precision that is any greater than the capability of the sensors from which the data are derived, and the safety of the plant should not require outputs to be any more accurate than that which actuators are capable of delivering.

   Items that affect the limits of input precision data include (NUREG/CR-6101, Lawrence, 1993, p. 72):

   - Sensor maximum range and anticipated headroom requirements for sensor signals (see above)
   - Units of measurement
   - Error bounds on sensor measurement
   - Calibration accuracy, including hysteresis and non-linearity of analog signals
   - Repeatability of analog signals
   - Temperature sensitivity
   - Drift
   - Power supply variations
   - Conversion algorithms

   Items that affect the limits of output precision include:

   - Actuator or display device maximum range of values
   - Units of measurement of actuator or display device
   - Resolution of actuator or display device
   - Conversion properties if actuator is analog
   - Calibration accuracy of actuator
   - Repeatability
   - Temperature sensitivity
   - Power supply variations

2.2.3 Internal Accuracy

Importance to safety ranking: HIGH (all guidelines in this section)

Internal accuracy refers to the precision of process parameters and derived quantities within the memory of the computer that is monitoring or controlling the process.

1. *Requirements for fixed point variables should be set to eliminate the possibility of truncation, round-off errors, and overflow errors.* Computers specialized for process control (e.g., PLCs) frequently use fixed point representations for storage of variables and fixed point arithmetic for manipulation of such parameters. Numbers represented as fixed can be stored as signed fractions (s.mmmmmmmm) in 8-, 16-, or 32-bit storage locations (NUREG/CR-6463, Hecht, 1996, p. 5-9). This representation ensures that multiplication will not result in overflow (the product of such a multiplication is always smaller than each of the multiplicands). However, it can result in truncation error or round-off error as the least significant bits are eliminated. Ignoring round-off and truncation errors can have two consequences: (1) Such errors can result in a loss of precision in the displayed or monitored parameter; and (2) Such errors can cause oscillation in closed loop control systems because the control error (the difference between the measured value and the control setpoint) is inaccurately represented and the output signal is either set too high or too low, depending on the error.

   On the other hand, the addition of two numbers, each of which is greater than 0.5, can result in an overflow condition in which the most significant bit is ignored. Undetected overflow errors can result in spikes, called *overflow noise* (NUREG-1709, USNRC, 2000, p. 27), at the output of a digital I&C system.

   Truncation, round-off, and overflow errors are eliminated by scaling (i.e., multiplying by a constant less than 1) the variables appropriately such that the products are not too small and the sums are not too large. However, it is important to ensure that this scaling does not impact the precision and dynamic range necessary to perform the control or monitoring function. Another approach is to select control algorithms that minimize the effects of round-off and truncation.

2. *Requirements for floating point operations should be set to ensure sufficient precision:* Round-off and truncation errors are possible in arithmetic operations when quantities are represented as floating point numbers, particularly when there is a large magnitude difference in the operands. Most current computers have a floating point processor that conforms to the IEEE 754 (1985) floating point standard. Its single precision specification allows for only 7 decimal digit precision in the mantissa (NUREG 1709,.USNRC, 2000, p. 27). The double precision standard allows for 15 decimal digits of precision in the mantissa. When referring to the IEEE standard, the requirements should be explicit about which precision is being used. In special purpose processors, such as those that are found in some PLCs and other embedded systems, the precision may be lower. The requirements should reflect the limitations of such processors.

3. *Requirements should be set for precision in type conversions:* Requirements should specify the precision-related conversion between data types, particularly between fixed point and floating point forms. For example, if the floating point representation becomes too large, scaling might cause a loss of precision needed to evaluate an input or to output a result. If the scaling is not proper, there is a possibility of overflow if the floating point value is too large. Such type conversions have been implicated in two high-visibility rocket accidents: the failure of a U.S. Patriot missile to intercept an Iraqi-launched Scud missile during the Gulf War; and the failure of the Ariane 5 launch vehicle during its maiden flight (GAO, 1992; SIAM, 1996).

2.3 Guidelines Related to Functionality

Guidelines in this section discuss requirements related to completeness of the specification of functions that must be performed for each mode of operation. The first subsection identifies general guidelines (i.e., those common to all phases of system operation). The following three subsections cover initialization, input and output, and processing.

2.3.1 General Guidelines

Importance to safety ranking: HIGH (all guidelines in this section)

1. *Complete definition of the hardware and software runtime environment:* The requirements should describe all aspects of the software runtime and the physical/operational environment. Failure Description No. 1 is an example of how a control system malfunctioned because of the incomplete definition of the hardware and software environment in the requirements. Fortunately, the consequences in this case were minor. However, Failure Description No. 22 was less benign. The release mechanism on a fighter aircraft was not inhibited when the aircraft was upside down, thereby allowing the pilot to release bombs or fuel tanks that would have destroyed the aircraft.

2. *Complete definition of the design basis:* As a minimum, each of the design basis aspects identified in SRP Appendix 7.1-B, Section 1, should be addressed. These are:

   ! *Completeness*
   ! *Consistency*
   ! *Correctness*
   ! *Traceability*
   ! *Unambiguity*
   ! *Verifiability*

3. *Traceability to system requirements:* The software requirements must be consistent with the system requirements and overarching design basis, including consistency with the mechanical and electrical system designs and other plant system designs.

4. *Functional completeness of software requirements:* All functions that are allocated to software from the system requirements should be documented in the software requirements. For such functions, the software requirements should cover all operations necessary to fulfill the system's safety intent. Information provided for each software requirements should be sufficient to enable the design, coding, and integration of the function to be carried out. The importance of this provision is illustrated in several failure descriptions for nuclear power plants (FD 0033, 0040, 0042, and 0044). The common thread in these events was lack of a complete listing of all requirements and consequent inability to test for the presence of all required functions. A contributing factor in some of these events was the deletion of functions during maintenance, which also could have been avoided by accessing a list of required functions.

5. *Unambiguity:* The software requirements, taken alone and in combination, should have one and only one interpretation.

6. *Operating Modes:* Requirements for software should cover all operating modes. The software requirements should also specify what mode transitions are allowed and under what circumstances they are prohibited. Such requirements will be reflected in:

   - Permissives
   - Interlocks
   - Resets
   - Restarts

   Failure Description No. 6 illustrates how a lack of considering the "clutch down" mode caused a potentially very hazardous condition in Saturn automobiles. Failure Description No. 21, in which a test pilot was able to retract the landing gear while the test aircraft was parked, is an example of inadequate specification of requirements for interlocks in all operating modes.

2.3.2 Initialization

Importance to safety ranking: HIGH (all guidelines in this section)

1. *Startup into safe state:* Requirements should specify that the system shall be initialized into a safe state and any additional detail to support this requirement. A safe state is an operational state in which the reactor system will not cause injury or damage in the presence or absence of positive control. The absence of the definition of a safe state in which the software begins operation could result in unanticipated computer-initiated actions, in turn resulting in unstable operation. There can be more than one safe state, and requirements should identify the one that is to be selected under given conditions.

   Requirements for system startup should specify:

! Initialization values of all variables
! Synchronization of time and replicated values
! Items to be logged and recorded upon startup
! Check or setup interlocks and permissives
! Initialization monitoring and diagnostics
! Sequence of software and other processing at startup
! States to avoid
! Means of ensuring that startup does not induce transients or exacerbate transients already underway at the time of startup
! Differences in the startup sequence from a "cold start" and a restart after interruption of operations
! Re-initialization operations if the computer remains operational but the I/O interface fails and is restarted
! Differences in the startup sequence from a "cold start" and a restart after interruption of operations
! Re-initialization operations if the computer remains operational but the I/O interface fails and is restarted

These requirements should include failure indications (including time-outs) and the course of action to be taken for each failure indication. The requirements should specify that an initialization failure will not result in an unsafe system state.

2. *Startup from reset, degraded, or failed state:* Requirements should specify the course of initialization (including all items identified above) when initialization occurs from a reset, degraded, or failed state. Failure Description No. 31, a security computer failure, is an example. Response times and transition times should be included for all significant steps in the sequence.

3. *System response to input during non-operational status*: Requirements should specify that the system shall not cause any unsafe conditions when not operating.

4. *Allowable transitions to and from degraded states:* The system requirements should specify how transitions will be made between all partially operational (degraded) states to a stable safe state. Response times and transition times should be included for all significant steps in the sequence. Such transitions include reset and termination.

2.3.3 Input and Output

Importance to safety ranking: HIGH (all guidelines in this section)

1. *Completeness of each parameter description:* Requirements should completely identify each process parameter including P&ID identification ("tag"), instrument type, input signal type (continuous or discrete), precision-related quantities (range, headroom, resolution, etc.; see section 2.2), and anticipated off-normal conditions and system responses. Adequately addressing this guideline may be complex. Failure Description No. 3 discusses how a barricade control system malfunctioned twice within two years, resulting in injuries. A requirement for handling off-normal conditions should have been formulated after the first event.

2. *Completeness of parameter (process variables) list*: All information from the parameters should be referenced (i.e., utilized) in the specification, and all essential parameters from the system specification should appear in the parameter list. The absence of complete correspondence may be an indication of missing requirements. The SRP Appendix 7.1-B, Section 1, requires in part the identification of variables that are monitored in order to provide protective action. The tables in Sections 7.2 and 7.3 of the SAR should provide this information. SRP Appendix 7.1-B also requires the identification of the minimum number and location of sensors for those parameters that have a spatial dependence. In the course of reviewing software requirements, it should be demonstrated that data from all sensors needed for monitoring and control of safety functions are identified and the number and location of sensed parameters is adequate.

3. *Permissive parameters*: Permissive parameters should be utilized only in the manner identified in the system specification.

4. *Identification of measurement locations*: The requirements should address the identification of measurement locations and how they are to be handled in the processing.

5. *Specification of hardware dependencies:* The requirements documents should explicitly identify all hardware dependencies and minimum requirements. This applies specifically where the software functions require specific inputs to "data highway" interfaces, discretes at 10 V, TTL discretes at TTL voltage, A/D interfaces, and D/A interfaces, all of which must be supported by the hardware computing platform.

6. *Bypass:* Requirements should clearly identify how the software will meet the mandate of 10 CFR 50.34(f)(2)(v) that requires that the I&C system provide for an automatic indication of bypassed and operable status of reactor systems. Requirements should also specify how the safety function will operate in the presence of bypasses, and how bypasses will be monitored and reported.

2.3.4 Processing

Importance to safety ranking: HIGH (all guidelines in this section)

1. *Completeness of requirements specifying action for a single condition:* The requirements should cover all possible values of a variable used in evaluating a condition. For example, if a threshold is being used to determine whether a safety action should be taken, the requirements should indicate what happens when the parameter is at less than the threshold value, at the threshold value, and greater than the threshold value. If the requirements specify three or more different actions or states for that variable, the requirements should indicate what is to happen when the variable is below, at, and above the threshold for each of those states. As a check on the completeness of the requirements, it should be possible to determine what will happen to the variable at any point over the entire range of values for that variable. Failure Description No. 7 demonstrates a classic case of a "fencepost" error, in which a requirement expressed as a "greater than" should have also included "equal." This requirements error was revealed in an e-commerce context, where the consequences were not severe.

2. *Completeness of requirements specifying multiple conditions:* Any requirement containing multiple logical conditions that are used for a decision purpose should be checked with respect to any subset of these component conditions. For example, if there is a condition with two parameters, then the requirements should indicate what happens when either parameter is at the threshold and another is below, both parameters are at the threshold, either parameter is above the threshold and the second is at the threshold, either parameter is below the threshold, and the second is above the threshold, and both parameters are above the threshold. Failure Description No. 6 relates to an automobile overspeed prevention system that caused the engine to stop when the clutch was disengaged. The proper response was to slow down the engine (which did happen when the clutch was engaged). Tables in which columns specify the conditions and rows specify the values for variables are sometimes useful for expressing such requirements.

3. *Completeness of the requirements in specifying the transition from the transient to a safe state for each set of conditions (relating or describing a system state) and defining a safe state:* The requirements should completely indicate the sequence of actions to restore the system to a safe state for each set of parameters outside of the threshold values.

4. *Non-ambiguity of processing:* There should be only one sequence of actions for each state transition.

5. *Accounting for spatial dependency*: Where multiple sensors are used at different locations, the software requirements should explicitly identify the effect (if any) of spatial dependency. This is of particular importance for core power sensing devices such as flux monitors.

6. *Auxiliary feedwater system control software requirements:* The software requirements should be traceable to BTP HICB-4, which provides guidance on the failures and malfunctions that initiate and control auxiliary feedwater systems.

7. *Reactor protection system software requirements:* The software requirements should be traceable to BTP HICB-5, which provides guidance on conditions requiring protective action. The malfunctions and resultant software requirements should be traceable to control system failure modes described in Section 7.7 of the SAR and the reactivity control interlock functions described in Section 7.6 of the SAR.

8. *Software requirements and setpoints:* Setpoints are the levels of monitored parameters that define the onset of unsafe conditions (setpoint), and limits are levels that require protective action (safety limit, i.e., value assumed in the safety analysis) for each variable. The software requirements should address the margin that exists between operating limits and setpoints, such that (a) a low probability exists for inadvertent actuation of the system, and (b) the system will be actuated when required. BTP HICB-12 provides guidance on the establishment of safety system setpoints. BTP HICB-6 provides specific guidance for determining if the timing margins for changeover from injection to recirculation mode are sufficient to allow manual initiation of the transition.

9. *Manual controls*: SRP Appendix 7.1-B requires manual initiation of all plant protective functions. The software requirements should allow for a manual initiation, they should allow for an interruption of those functions that can be interrupted when such an initiation occurs, and they should address how previously initiated functions that are uninterruptible will be handled when a manual intervention occurs. Failure Description No. 5 is an example of inadequately specified manual control requirements. An operating theater was left dark because there was no way to activate the backup power as a result of an outage.

2.4 Guidelines Related to Reliability

Reliability is defined alternatively as the predictable and consistent performance of the software under conditions specified in the design basis (SRP Appendix 7.1-B, Section 1) or the probability of successful operation over a given time interval (IEEE Std. 100-1977). The first definition is qualitative, and as such, it is related to either development process issues or product qualities (possibly controlled by the development process) such as robustness and specificity and completeness. The latter are covered in separate headings of this report and the development process is outside the scope of this document.

The second definition of reliability, however, is not covered under any other heading and yet is a relevant requirements issue. In the simplest case, quantitative requirements for individual channel reliability can be used to obtain high assurance that at least the minimum required number of channels will be operative at all times. The risk-informed approach specified in Regulatory Guide 1.174 also depends on quantitative reliability specifications for all parts of the system, including software. Regulatory Guide 1.174 states in part:

> The use of PRA technology should be increased in all regulatory matters to the extent supported by the state of the art in PRA methods and data in a manner that complements the NRC's deterministic approach and supports the NRC's traditional defense-in-depth philosophy.

There may be greater emphasis on quantitative reliability requirements for software—particularly for COTS or PDS—where there is an operational history that can be used to substantiate conformance with the requirement.

IEEE Std. 7-4.3.2-1993 specifies that where quantitative requirements for system reliability involve software, quantitative requirements should be specified for both the hardware *and* the software. Software reliability requirements are derived from the system architecture and redundancy. Originally, redundancy was implemented as sense and command channels with voting elements attached to execute elements (actuators) in the manner conceived of by IEEE Std 603-1991. This architecture had a relatively small impact on the application software level. However, with the advent of more sophisticated instrumentation buses, greater architectural flexibility was possible through the distribution of multiple channels of sensor signals to multiple processors. Architectures such as active/standby redundancy were introduced for PLCs (EPRI, 1996, p. 1-8). These architectures will have an impact on the allocated reliability requirements for the constituent software components that are given a total system-level requirement. In setting quantitative reliability requirements for redundant components, reliability requirements for individual software components can be allocated properly if the following are appropriately accounted for (Hecht, 1997; Tang, 1995).The following excerpts from Regulatory Guide 1.152 are pertinent:

> The [NRC] staff does not endorse the concept of quantitative reliability as the sole means of meeting the Commission's regulations for reliability of digital computers used in safety systems. The NRC staff's acceptance of reliability of the computer system is based on deterministic criteria for both the hardware and software rather than on quantitative reliability goals.

9. *Correlated failure probability:* When redundant channels are being used to perform a safety function, the reliability or probability of success on demand is limited by the likelihood of multiple channels failing simultaneously due to a single event that affects multiple channels. There are several types of correlated failures, including *common mode* failures in which multiple channels simultaneously fail in the same way (e.g., a software "crash"); *common cause* failures, in which multiple failures in different channels can be attributed to a single event (e.g., an out-of-range or unexpected value that the software exception handling capability does not adequately address); or both a *common cause/common mode* failure (frequently referred to simply as a "common mode failure"), in which a single initiating event causes multiple channels to fail in the same manner by the same cause. Requirements for maximum acceptable correlated failure probability should be defined for the software and verified through an assessment of testing and, where possible, comparable operational results. Methods for a measurement-based assessment of correlated failures are discussed in references by Tang (Tang, 1995) and Hecht (Hecht, 1997 and 2000).

Importance to safety of above guideline: HIGH

10. *Detection probability:* Where active/standby redundancy is used and the reliability of system operation depends in part upon the ability of a failed software component to restart and resume operation as part of the system, a failure detection probability should be specified. Two qualitatively different detection probabilities are applicable:

    - *Plant conditions:* Detection of off-normal conditions within the plant that require system intervention—this is a functional requirement and is conditional on the system being operable.

    - *System failure conditions:* Detection of conditions within the system resulting in the need for reconfiguration and recovery—this detection probability relates to the likelihood that the software will be able to detect and initiate recovery from such conditions.

    Failure Description No. 19, in which all control circuit cards failed simultaneously due to electrostatic discharge, is an example of a correlated failure (in this case a common cause failure).

    Importance to safety of above guideline: HIGH

1. *Recovery probability:* Where active/standby redundancy is used, and the reliability of system operation depends in part upon the ability of a failed software component to restart and resume operation as part of the system, a recovery probability should be specified and should be verifiable. The recovery probability should be such that the system will be able to meet its overall requirements of reliability or probability of success on demand. Recovery probabilities should be specified for all anticipated operating modes of the system and for all anticipated failures that are to be detected.

   Importance to safety of above guideline: HIGH

2. *Recovery time:* Recovery time relates to the time needed for the system to return to operation after a failure. This is distinct from response time, which is discussed below. When the recovery time requirement is less than the response time, then the system can be considered to have failed *transparently*, that is, without any impact on the system. However, even when the recovery time is greater than the response time, there is still a benefit as a restored system will be able to resume functioning (or provide a greater level of redundancy). Recovery time requirements (together with confidence limits) should be specified for all anticipated failure modes.

   Importance to safety of above guideline: HIGH

3. *Failure Rates:* Requirements for failure rates are of particular importance for the underlying "platform" software, including operating systems, data acquisition, communications, and related functions. Such software is often reused or "off-the-shelf," and previous operating history can be used to verify conformance with failure rate requirements. Failure Description Nos. 14 and 17 are examples of "crash" failures that could be used, together with information on the operating time, to determine the failure rate and hence reliability as part of a larger model. The use of prior operating data requires evaluation of the run-time environment and all conditions of use. Failure Description No. 27, describing an outage of the NASDAQ computer system, also indicates the general importance of quantitative information on reliability and availability for critical systems.

   Importance to safety of above guideline: MEDIUM

4. *Degraded Operation:* Requirements for operation in a degraded state should be identified where this is appropriate. Key functions that must remain operational in the degraded state must be identified. A time limit for operation in the degraded state may have to be specified.

   Importance to safety of above guideline: MEDIUM

Enhancements in processor and network speed will encourage the introduction of lower-cost, general purpose ruggedized embedded computers and PDS operating system components with operating histories into safety systems. For such systems, assessing conformance with reliability and related requirements can be achieved in part by evaluating the operating history. For such evaluations, not only the mean value of the reliability-related quantities, but also the upper and lower confidence bounds are important and should be specified in the requirements. An important purpose of the confidence intervals is to distinguish between a mean obtained from a few observations (wide confidence intervals) and one obtained from many observations (narrow confidence intervals, particularly if most of the results are in a close range).

Similarly, when evaluation of parameters such as detection and recovery probability are based on test data, upper and lower confidence limits should also be specified. For example, a detection probability should be stated in terms of "a minimum probability of detection of 0.99 at the 95% level of confidence." A requirement stated in such a manner allows a test plan to be developed, and allows quantitative reliability-related software requirements to be verified. The confidence level for each quantity should be developed based on the system safety analysis and the design basis of the SAR.

As such systems are placed into operation, it is possible to refine estimates of reliability with actual experience. Approaches can include both conventional and Bayesian methods (Hecht, 1997; Bouissou, 1999). The resulting refinements can benefit requirements for system upgrades or additional applications of the same technology.

2.5 Guidelines Related to Robustness

Importance to safety ranking: HIGH (all guidelines in this section)

Robustness guidelines cover requirements related to behavior of the software in the presence of unexpected, incorrect, anomalous, and improper input; hardware behavior; or behavior of other software components. Of particular concern is the behavior of the software in the presence of unexpectedly high or low rates of message traffic. Requirements for fault tolerance and failure modes should be fully specified for each operating mode as part of the system level design (and derived from the design basis). Software requirements for handling both design basis hardware and software failures should be provided, including requirements for detection of and recovery from computer system failures. Failure detection can comprise up to 80% of the system (including the software), and thus the robustness of the failure detection provisions needs careful review. The topics of Section 2.5.1, particularly those dealing with exception and error handling and with the independence (from the monitored system) of failure detection and recovery provisions, are related to this topic. The first subsection lists guidelines common to all aspects of system operation, the second lists guidelines related to input, the third covers processing, and the final section covers output.

2.5.1 General Guidelines

1. *Requirements for software exceptions and error handling*: Requirements should identify all foreseeable exceptions and system errors and specify how they are handled (detection, damage confinement, and recovery). Requirements should be traceable to safety analyses, FMECAs, or other parts of the design basis to indicate I&C failures that would generate system exceptions (e.g., I&C interface noise, data corruption, data overloads, power failures) and should state how they would be handled. Requirements for handling of such exceptions may include:

    - *Fail stop:* When software encounters an error or exception condition, it stops instantly and causes all output registers to be flushed and the channel to be shut down without output of any signal. If such requirements are in place, there should be other means of mitigating a common software failure (e.g., diverse means of achieving the function).

    - *"Lifeboat" or alternate routine:* When the software exception is encountered, the software switches to a diverse alternate routine that allows for graceful shutdown to a safe state or minimal functioning.

    - *Restart:* This is similar to a fail stop, except that the software may restart. Requirements for a "clean" shutdown and reset also need to be established (see below).

Additional analyses of the computer system itself should be performed to indicate the anticipated failure modes, resultant failure conditions, and exceptions generated as a result of these failure conditions, and the state of the system after the exceptions. Analysis techniques such as the Failure Modes and Effects Analysis (FMEA) can be used for this purpose.[4] Dynamic analyses can be run with "hardware in the loop" or with computer, sensor and plant models. In the latter case the validation of models used to support analyses is required. Failure Description 15 is an example of "crash" failures that might have been avoided had better requirements been defined for exception handling. Failure Description No. 19 is a hardware analog to inadequate exception handling, in which the failure condition defeated the massive redundancy that was installed.

2. *Requirements for independence of failure detection and recovery mechanisms*: All safety systems should include requirements for independence of all internal detection and recovery mechanisms from those of the monitored system. Independence does not, by itself, guarantee safety, but it removes the possibility that a single fault disables both the monitored system and the recovery provisions. Failure detection and recovery provisions must, almost by definition, operate in less predictable environments than the monitored system and the review must establish that verifiable requirements for meeting the failure detection and recovery objectives exist (see Section 2.5.2–2.5.4). Examples of independence requirements are:

    - If a watchdog timer is used, that watchdog timer should not use the same time source as the CPU clock.

    - If communication of heartbeats and checkpoints is used as part of the fault tolerance design of an active/standby system, then the communication channel over which those recovery critical data are transferred should not be the same as the normal data channel.

    If communications with external systems are required, then both system and software requirements should specify redundancy, as well as protocols to be used for redundant message channels. The service interruptions in Failure Description Nos. 15 and 16 might have been mitigated by the existence of requirements for redundancy in both the communication channels and the software to utilize them appropriately.

3. *Active/standby software requirements:* As noted in the section on reliability guidelines, redundant computing is increasingly based on an active/standby design, rather than the passive redundancy of systems that were designed using voting redundancy. Where such systems are used, requirements should be in place to *completely* address the following issues:

[4] The FMEA is defined by MIL-STD-1629. A description of the application of this technique to digital systems can be found in multiple works (e.g., Leveson, 1995; Lutz, 1999).

- Periodic status monitoring ("heartbeats"): This should include the length of the period (relative to the required response and recovery times), the consequences of missed heartbeats, protocols for avoiding spurious recovery, and time synchronization.

- Data consistency: Active/standby systems have separate replicated data stores that must be kept synchronized and consistent. Requirements should address what data are updated, the update frequency, how loss of consistency is detected, and protocols for restoring consistency after it is lost.

- Switchover protocols: Requirements should explicitly address how the standby becomes active, including shutting down the previously active node to prevent spurious signals from being transmitted into the plant.

2.5.2 Input Processing

1. *Handling of input and output hardware and communication failures:* NUREG-0800 requires that the design of an I&C system be such that all failures result in the system transitioning into another operationally acceptable or non-operational safe state in the presence of any losses of system integrity. This requirement should be reflected in software requirements. As suggested in NUREG-0800 Appendix 7.1-B (p. B-7), assessment of the completeness and correctness of these robustness requirements can be performed with the help of a Failure Modes and Effects Analysis (FMEA). Related software requirements include:

   ! *Minimum data required for safe operation of a function:* Requirements should specify the minimum set of data and associated accuracies that are required for operation of the safety function. The requirements should specify (a) what actions should be taken if the minimum data requirements are not met, (b) the maximum allowable time duration for falling below a minimum data level, and (c) the maximum allowable rate of interruptions. These requirements should have references to external documentation in the design basis or elsewhere for substantiation.

   ! *Minimum operational configuration:* Requirements should specify the minimum operational configuration of the digital control system for each function and how it transitions from other degraded states to the minimum operational configuration.

- *Handling of sensor failures:* Software requirements should specify how the software is able to detect the failure of sensors—both those that are detected and "announced" by existing surveillance and diagnostics provisions (see the section on maintainability) and those that are not. The latter category requires additional logic and functionality that should be specified in the requirements. For example, software to provide loss of flow protection would normally derive its signal from flow sensors. However, the sensors may fail "silently" and, in particular, may give an indication of a normal flow when, in fact, there is none (this failure mode should appear in an FMEA). In order to handle this failure mode, requirements should be defined to look at indirect measurements such as pump speed, pressure, level, or temperature.

The importance of handling sensor failures can be seen in Failure Description No. 3, in which a gate failed to properly function due to sensor failures. The result was injury. ailure Description No. 4 shows how back-up provisions for a major communication network overlooked a single point failure condition that affected all channels. In several automatic commuter train door failures (Failure Description Nos. 28, and 29 are the most obvious ones), the results varied from benign to fatal because of a failure to account for the inherent inaccuracy of door position sensors. Failure Description No. 23 indicates how incorrect sensor data processing resulted in the spurious shutdown of aircraft engines, which could have resulted in a major air disaster. Failure Description No. 25 indicates "sensitivities" in the engine startup computer aboard the 747 that prevented the unit from functioning when the aircraft switched from ground to auxiliary power in the wrong sequence.

2. *Accounting for multiplexed signals:* Reactor protection systems typically consist of multiple groups of sensors or instruments. Traditionally, these connections were made with one circuit per sensor or actuator, and both the wiring and the sensor were replicated when redundancy was required. In newer designs and upgrades, it can be anticipated that multiple sensors will be attached to digital communications networks (Prekshott, 1993, p. 57; Fabio, 2000). While there are advantages to such installations, they potentially introduce more complex failure modes that must be handled by the software. Among these are:

   - Data errors affecting multiple sensors (typically resulting in missing or detectably corrupted data rather than corrupted data that is not detected)

   - Changes in the message mix or formats

   - Babbling resulting in the delivery of excessive data

   - Permanent or intermittent loss of multiple signals

Handling these failure modes is an architecture and design issue that falls outside of the scope of this document (and the task of the reviewer of software requirements). However, the software requirements should completely reflect the design and architecture provisions that are implemented. Among the items that should be reflected in the software requirements are:

1. Signal handling from multiple, redundant buses

2. Algorithms for detecting missing signals from one or multiple devices from one or multiple redundant data networks

3. Proper interaction with the error detection and correction capabilities of the underlying instrumentation buses (i.e., software should not time out before the bus times out)

Care must be taken to ensure that recovery requirements address timing and performance. Failure Description No. 4, although relating to a much larger and less deterministic system, demonstrates that recovery requirements that do not adequately take into account the characteristics of the network can result in a long-term denial of service. Failure Description No. 20 demonstrates how inadequate requirements for communication exception handling contributed to the failure of the Boston Air Route Traffic Control Center.

3. *Validity checks on inputs:* The requirements should specify that all incoming values are checked and that a response is provided for each out of range condition (greater than maximum expected value, less than minimum expected value, greater than expected increment, totally invalid value, etc.). In upgrades from analog to digital I&C systems, input signals can have failure modes that were not present in analog I&C systems and, hence, may not have been reflected in the system requirements if they were a direct translation of the original analog system’s functional specifications. Perhaps the most significant examples are sudden changes in values due to the “flipping” of high order bits (due to transient failures in the A/D conversion or transmission in a manner that it is not detected by cyclic redundancy checks). Requirements for detection of sensor failures described in the maintainability section may be developed in common with these requirements. Failure Description No. 9 describes how invalid sensor data resulted in a near fatal accident related to subway doors. Failure Description No. 30 reports how inadequate sensor data processing requirements led to the shutdown of 747-400 engines during takeoff because of spurious sensing of the presence of thrust reversal or other signals. Failure Description No. 26 indicates how incorrect status was displayed as a result of invalid input.

4. *Data age requirements:* The requirements should specify the maximum acceptable interval between parameter updates (measurement age), how the system will verify data age, and what actions the system will take if the data become “stale” (data updates are older than the allowed limits).

5. *Missing data requirements:* Requirements should specify how the software will respond to missing data items (this is related to issue no. 2 for multiplexed signals).

6. *Corrupt data handling requirements:* Data can be corrupted by a number of mechanisms. Digital network failure modes were cited above. Mechanisms from analog methods (as well as from, to a certain extent, digital data transmission methods) that can corrupt signals include degradation of isolation (causing cross talk and other interference), inadequate noise immunity, and permanent or intermittent contact degradation. Requirements should specify the limits on inaccuracy of data, how corrupt data will be detected, and what actions the control system will take in the presence of the corrupt data. Such requirements should be checked for consistency with the underlying hardware and communication system attributes for error rates and reliability.

7. *Handling buffer overflows:* Software requirements should identify the system response in the event of input overflows. Requirements should be specified for each input signal, and, in all cases, should ensure that the plant enters or remains in a safe state. Failure Description No. 18 is an example of a buffer overflow condition where more accurate estimation of the queue size and a more complete specification of software behavior when a buffer overflow occurs could have averted or mitigated this failure.

2.5.3 Operation

1. *Interrupts:* Requirements should specify to what interrupts the software will respond, what states will be saved prior to handling the interrupt, the maximum allowable time for the interrupt, and how the system will restore states after the interrupt. Interrupt-driven processing should be minimized in a safety grade system in order to permit complete verification and validation. Because of the stochastic nature of interrupts, it is very difficult to verify that a particular response will occur under all timing conditions. Note that the “interrupts” referred to in this paragraph are normal processing interrupts, not exceptions related to data, software, or system failures, which are discussed in a later paragraph on exception handling.

2. *Uninterruptible control and safety actions:* Requirements should specify which functions must be completed or not run at all (“atomic,” or indivisible, actions), the consequences of interrupting an uninterruptible function (due to a failure or exception), and the means of recovering from such a failure (perhaps by shutting down that channel or processor and letting another processor resume or continue the function).

3. *Canceling a partially completed action:* Requirements should specify which (if any) functions can be canceled prior to completion, what system response will be taken, and how the operator will be notified of canceled actions. For example, in a PLC, a watchdog timer may reset the device if its timeout is reached. The software requirements would have to specify how the software responds to this event (e.g., to shut down in an orderly fashion, send a message to the operator interface, and execute an exit) without affecting safety. The subway automatic door incidents described in Failure Description Nos. 28 and 29 illustrate the importance of establishing proper requirements for responses to partially completed actions. In these cases, the requirements had to balance avoiding nuisance alarms and maintaining safety.

4. *Requirements limiting sensitivity of monitoring and control algorithms:* Requirements should be set on the sensitivity of process monitoring and control algorithms to internal and I/O precision errors. As noted above, control algorithm coefficients can be affected by the precision of input and stored variables. Such considerations may be important not only in safety systems, which generally do not rely on PID feedback control (Prekshott, 1993, p. 57), but also in systems important to safety or systems that might initiate a challenge to the safety system. The sensitivity of polynomial roots in control or filtering algorithms increases with the order of the polynomials. The output of algorithms implemented as large order polynomials can be significantly affected by errors in measurement or precision of control variables. Thus, requirements should be set on the sensitivity of such algorithms, based on the anticipated measurement errors and precision limitations of the A/D conversion. It is possible to decrease the sensitivity of such control algorithms by implementing them as a series of small polynomials rather than one large polynomial (Phillips and Nagel, 1995; Franklin, et al., 1994).

2.5.4 Output

1. *Limits on step size in outputs:* Requirements should specify limits to conform with maximum tolerable step changes for output equipment. For example, if a digitally calculated signal is converted to an analog signal to control a hydraulic actuator, the step must not be too large to either damage the equipment or cause a pressure transient in the plant (NUREG-1709, USNRC, 2000, p. 28).

2. *Output conversion errors:* Requirements should be written to ensure that no digital to analog conversion errors affect safety. The considerations related to error, precision, and resolution are similar to those for input (see subsection 2.5.2) and are therefore not discussed further. Software requirements should identify the system response in the event of output overflows. Requirements should be specified for each output signal, and, in all cases, should ensure that the plant enters or remains in a safe state.

2.6 Guidelines Related to Maintainability

Maintainability guidelines are related to both online, in-service testing and diagnostics and to the means by which the source code reduces the likelihood that faults will be introduced during changes made after delivery. An important quality is functional partitioning (avoiding multiple functions in one module) because it reduces the likelihood that a change made to one function will affect another one. Maintainability can also be understood to refer to the ease with which software can be maintained by the appropriate practices in software design, coding, debugging, and configuration management. However, this latter type of maintainability is influenced primarily by the *development process* and not by the system-level requirements and as such is not in the scope of this document.

Requirements related to maintainability should address:

1. *Online system status monitoring and reporting requirements:* Requirements should specify what aspects of the system status and configuration are reported. These requirements should include:

   - What items (sensors, processors, buses, firmware, software, etc.) will be included in configuration reports
   - How the operational status of these items will be checked
   - How the revision number and serial numbers will be reported (manually, during automated system status checks, etc.)
   - What data will be recorded about the system configuration and at what frequency
   - What plant functions are checked by each surveillance or monitoring function
   - What (if any) functionality is disabled during surveillance and monitoring
   - The frequency of execution of each monitoring function, by sensor or channel, if applicable
   - How successful operation of the system monitoring function will be recorded and reported
   - How failure of the system monitoring function will be reported, recorded, and alarmed
   - The impact of failure of execution of the system monitoring function
   - What actions are to be taken for each anomaly detected by the system monitoring function

   Importance to safety of above guideline: HIGH

   Four failure reports indicate deficiencies in this area of requirements. The air-conditioning failure described in Failure Description No. 1 was aggravated by lack of maintenance and diagnostic provisions in the computer part of the system. Three failures in air traffic control installations (Nos. 16, 17 and 18) resulted in prolonged outages because of weak requirements for diagnostics.

3. *Sensor and communication system checks:* Software requirements should be established for the sensor and communication system. These requirements should include provisions for continuously or periodically checking that the software receives valid input signals. Such checks should include but are not limited to comparing values of redundant sensors, using known physical relations to compare diverse sensors at a given point (e.g., pressure, temperature, volume), or performing time series analyses.

   Importance to safety of above guideline: MEDIUM

4. *Off-line system monitoring and diagnostic:* Requirements should specify each of the system surveillance and monitoring operations, including:

   ! The functions that are checked by offline operations

   ! Interlocks to prevent operation when systems are being maintained (or are in a maintenance mode)—in Failure Description No. 36, a nuclear plant emergency was created by omission of a step during the revision of a maintenance procedure.

   ! The frequency of execution of offline monitoring functions, by sensor or channel, if applicable—Failure Description No. 34 shows how ambiguous statements about inspection frequency can cause serious problems in a nuclear power plant.

   ! How results (indications of both normal and failed operation) will be reported

   ! The actions to be taken for each anomaly detected by the system monitoring function

   ! How data related to system crashes will be recorded (this generally requires that data be collected during operation to enable a post-mortem "dump")—Failure Description No. 14 describes an incident in which an air traffic control weather radar experienced a transient failure and was simply reset by the technician. Diagnostic information was not gathered on the source of the problem to enable an assessment of whether the equipment status was suitable for resumed operation. In contrast, in Failure Description No. 18, the presence of trace recordings enabled a diagnosis of the buffer overflow problem and enabled a requirements deficiency to be detected during operation and fixed.

   Importance to safety of above guideline: MEDIUM

5. *Requirements to allow technician maintenance:* Requirements should indicate:

   ! How the software will allow and support system maintenance—Failure Description No. 20 indicates that technician action to reconfigure an air traffic control computer at the same time as the system was trying to reconfigure itself resulted in an exacerbation of the problem.

   ! How software will verify that parameter changes, diagnostics, and other system functions are performed as specified

   ! How technician actions are recorded and reported

   Importance to safety of above guideline: MEDIUM

6. *Upgrade support:* Software requirements should address how upgrades will be performed and verified so that crashes or other system anomalies will not occur upon initial loading and operation of system revisions and upgrades. The importance of considering upgrades can be demonstrated by Failure Description Nos. 10 and 20.

   Importance to safety of above guideline: HIGH

7. *PDS checking of operational environment:* For PDS or COTS software, requirements should be defined to ensure checking of the runtime environment (including the platform and operating system) for compatibility. Checks should ensure that the software will run properly in the new environment. Failure Description No. 1 is an example of how an HVAC system was not properly controlled by software, which was apparently developed for multiple configurations. Failure Description No. 11 is an example of an interface failure for an elevator annunciator function.

   Importance to safety of above guideline: MEDIUM

2.7 Guidelines Related to Security

NUREG-0800 requires that security threats to the computer system be identified and classified according to severity and likelihood. In earlier designs, physical and administrative controls were the primary security mechanism. However, it is quite possible that many functions related to authentication, auditing, and access control will be migrated to software because of the increasing sophistication of software-based security functionality. These ongoing developments suggest that specialized research into security related requirements might be beneficial. This section discusses guidelines for security-related requirements of the software to detect, prevent, or mitigate security threats.

1. *System Changes:* NUREG-0800 Appendix 7.1B requires control of access to setpoint adjustments, calibrations, and test points (p. B-9). These requirements can be implemented either by physical or software security measures. If implemented in software, requirements should specify how the software will authenticate the identities of individuals accessing the system to make alterations of setpoints, calibration parameters, program changes, etc. The requirements should state what checks the system will make on the authority of the individual making the parameter changes, and how it will log and report such changes.

   Importance to safety of above guideline: MEDIUM

2. *Access control:* Requirements should state what security boundaries are present in the system, how the software will control access at the boundaries, and how accesses will be stored. The requirements should indicate how access will be maintained in the event of changes and upgrades. Failure Description No. 10 demonstrates that requirements in this area are sometimes overlooked, even in an application as critical as remote banking.

Importance to safety of above guideline: MEDIUM

3. *Auditing:* Requirements should state how the software will report security accesses and potential security violations.

   Importance to safety of above guideline: MEDIUM

2.8 Guidelines Related to Timing

Importance to safety ranking: HIGH (all guidelines in this section)

Timing-related software requirements include performance, response time, and capacity. Such requirements are essential for ensuring correctness and stability of responses to reactor excursions, ESFAS control, and auxiliary system control. Plant technical specifications usually impose a hard deadline (that has to be met in every instance) on the operation of the RTS, ESFAS, and auxiliary control systems. Often, constraints are imposed by the underlying hardware (including sensors, existing I&C networks, A/D converters, and actuators) may impose hard or soft deadlines[5] dictated by stability requirements for closed loop controls. The SRP Appendix 7.1-B, Section 1, requires in part the identification of the performance requirements—including system response times, system accuracies, ranges, and rates of change of sensed variables—that must be maintained until completion of the protective action. The applicant/licensee's analysis, including the applicable portion provided in Chapter 15 of the SAR, should confirm that the system performance requirements are adequate to ensure completion of protective actions. The software requirements must conform to these constraints.

1. *Response times for the integrated system*: Response time requirements must account for the entire signal path. This can include (EPRI 1996, p. 4-2):

   - The time effects of analog or digital input filtering
   - A/D conversion time
   - Data interface-to-processor transfer time (I/O module to processor in the case of PLCs[6])
   - Two scans of an application program (to account for the case that a transient begins immediately after the input value is processed)
   - Processor-to-data interface transfer time
   - D/A conversion time (if applicable)

[5] Hard deadlines must be met on every iteration; soft deadlines requirements refer to response times that must be met on the average

[6] Most safety grade systems use real time systems with a fixed iteration rate, often called a *scan time* for PLCs. A description of PLC operation, including input, output, and processing iteration rates, is contained in NUREG/CR-6463, Appendix A.

- Maximum time to perform any failure detection and recovery (that may have occurred in the course of processing; see also recovery time in Section 2.4 above)

The last bullet also includes response time of communication channels under noisy conditions that may require (a) error correction by means of an error detection and correction code, or (b) retransmission in care of uncorrectable errors. For parameter display, response time requirements must be allocated to signal acquisition, processing, and display updating, and the sum of these allocations must be consistent with (i.e., less than or equal to) the latency imposed by the design basis or other factors. For control, the response time allocation must consider the system acquisition, processing, output, and actuator response times.

2. *Mode of operation:* Timing requirements (response time, sampling rate, throughput) should be set on the basis of the most stringent mode of operation ("worst case") of the appropriate system. In many cases, the worst case mode of operation will be during a transient event or an accident, rather than under steady state conditions. Most safety system processes make use of static scheduling, where each major process is allocated to a given fraction of the total cycle. Dynamic scheduling is usually more efficient but poses problems in proving that a given process will meet its response time requirements.

3. *Sampling rate:* Sampling rate requirements should be based on the most constraining of the following factors:

   ! *Response time requirements*: Requirements for the time interval between samples (reciprocal of the sampling rate) should be set such that the system can sense the worst case transient event associated with that sensor and can provide the required action (display or actuator signal) within the timing constraint imposed by the design basis.

   ! *Noise and aliasing*: The sampling rate requirements should reflect not only what is required to capture the process dynamics, but also the noise making up the total signal entering the digital I&C system. The noise information is necessary to prevent its aliasing (NUREG-1709, USNRC, 2000, p. D.2). The means by which the spectrum of the process dynamics for a given signal is captured is not within the scope of this document (it may include analytical calculations and measurements of worst case transients in a high fidelity simulator with a spectrum analyzer), nor is the choice of anti-aliasing filters. However, the requirements should reflect the appropriate sampling rate derived from these considerations. The sampling rates must also take into consideration external noise introduced by EMI/RFI. In some cases, the sampling rate must be significantly higher than required in the previous paragraph so that signals in areas other than the process signal bandwidth (i.e., noise) can be subtracted out. This is particularly important if EMI/RFI signal strength is on the same order as the process signal itself.

- *Closed loop control system stability*: Closed loop sampling rate requirements must be consistent with the constraints imposed by the closed loop control system. Most class 1E safety systems terminate unsafe conditions and induce mitigation actions, and therefore there is not much use for continuous Proportional Integral Derivative (PID) feedback control. There is, however, binary feedback from on/off devices such as valve positions (Prekshott, 1993, p. 57). Moreover, systems incorporating PID feedback control may be important to safety or might initiate unsafe events. Sampling requirements related to system stability should be evaluated for these systems as well. Methods such as the phase/gain margin or the signal rise time (Phillips and Nagel, 1995; Franklin, et al., 1994) can be used to determine the minimum sampling rate. Examples of guidelines are that the sampling rate must be six times higher than the closed-loop bandwidth or four times the minimum rise time (NUREG-1709, USNRC, 2000, p. 23).

- *Periodicity of operation:* Functions in the overall system may depend on data being updated at specific intervals.

The techniques to gather the data and perform the calculations are beyond the scope of this requirements document. An example approach can be found in Appendix E of NUREG-1709, (USNRC, 2000). Of the greatest significance to reviewers are that:

- The appropriate design and analytical bases for the sampling rate determination exist.

- The validity of the assumptions under which the data were gathered and analyses performed has been assessed.

- The resultant requirements properly reflect both the worst case and limiting effects arising from the analyses.

Importance to safety of above guideline: MEDIUM

4. *Degraded and failed system states*: Capacity and timing requirements must be developed not only for normal operational states but also for partially degraded computer and I&C system states. Failure Description No. 4 demonstrates that when requirements do not properly address capacity issues, a long duration outage can result.

2.9 Human-Computer Interaction

Importance to safety ranking: HIGH (all guidelines in this section)

The importance of complete information for operator decisions has been widely recognized both within the NRC and in other industrial automation applications (Jaffe, 1989). The contributions of improper displays in causing or exacerbating accidents has been documented by many reports, articles, and books (see, for example, Kharanbanda, 1988). Extensive standards developed by the NRC (NUREG-0700, USNRC, 1996b) address the general issue of functionality and are not within the scope of this document. This section lists guidelines that should be considered when user interface requirements are implemented in software.

1. *Event Notification and Alarm Display*: Event notification and alarm display requirements should define (Jaffe, 1991; Leveson, 1995, p. 367):

    a. The events to be queued and displayed
    b. The number of event categories and alarm classes
    c. The order in which events will be displayed
    d. The means by which the operator is notified of high priority events (e. g., blinking)
    e. The means by which alarms and notifications can be acknowledged
    f. How updates are to be shown for both displayed and queued events

    Failure Description No. 24, relating to early A 320 problems, shows that lack of prioritization in alarm displays aggravated the crew difficulties in dealing with anomalous situations.

2. *Responses to events:* The requirements should map each message, display, or alarm to operating procedures. Allowance for human decision and reaction times should be stated. The subway automatic door incidents described in Failure Description Nos. 28 and 29 illustrate the importance of establishing proper requirements for responses to events. In these cases, the requirements had to balance avoiding nuisance alarms and maintaining safety. Failure Description No. 24 demonstrates the problems with requirements that do not properly specify warning conditions, resulting in spurious warnings that, in turn, create additional hazards (an aircraft with a landing gear warning).

3. *Layout*: The requirements should address the layout of controls and displays, and the relative location of each display and instrument (i.e., grouped with..., above..., etc.) should be described.

4. *Manual controls and overrides:* As noted above, manual initiation of protective functions is required under SRP Appendix 7.1-B. Software requirements should identify what controls and displays are available to the operator for each manual and override state. Failure Description No. 5 demonstrates the importance of adequate requirements (at the system level and/or the software level) for manual controls and overrides. The lack of such overrides caused an operating theater to be dark despite the presence of a backup generator system.

5. *Cancels and Aborts:* The system should show a complete display of status when the operator attempts to abort a sequence (see Space Shuttle Abort Failure, Leveson, 1995, p. 371).

6. *Termination*: Requirements should specify how the operator may terminate a safety function and what impact this will have on the rest of the plant.

7. *Feedback:* The system should provide full and complete feedback on status during the operation of a command sequence.

8. *User Characteristics:* Requirements should identify essential characteristics of the users of the software such as reactor operators, maintenance personnel, or plant managers. NUREG/CR-6101 (Lawrence, 1993) recommends that for each class of user the requirements specify:

    - Educational level
    - Experience with nuclear reactors
    - Experience with real-time digital control systems
    - General technical proficiency (code reading, system diagnostics)

9. *Interlocks and Permissives:* The requirements should identify to the operator what interlocks and permissives are in place and how this will impact operation of the safety function. Failure Description No. 6 shows that an automobile engine overspeed system caused the engine to stop when the clutch was disengaged. The proper response was to slow down the engine (which did happen when the clutch was engaged). Although the consequences in this case were relatively minor, a similar error in a safety function could have more severe consequences.

10. *Reversibility:* The requirements should specify which, if any, commands are reversible, as well as the system response when the operator intends to reverse an action.

11. *Avoiding harm to operator:* Requirements in the alarm and display system should ensure that the operator is not exposed to excessive light, noise, shock, physical vibration, or other disturbances. Failure Description No. 13 describes an incident in which an air traffic controller was temporarily incapacitated by electrostatic discharge routed through her headset.

12. *Maintenance outage indications:* Requirements to inform the operator of systems that are non-operational due to maintenance actions.

13. *Personnel availability:* Senior personnel (operators and maintenance) are frequently "on call" outside of their shift hours and are presumed to be available when anomalous conditions arise. Requirements should state how the availability of "on call" personnel is to be monitored and how substitute personnel are identified when the designate "on call" person is out of the area.

3. Methodology for Conducting a Review

The reviewer of a software requirements document will be confronted with at least the following questions:

1. Does the scope of the document include all generic topics that have been found essential for generating a software design from requirements?

2. Does the content of the document provide sufficient and unambiguous detail to assure that the software will meet the requirements for the specific application? For NRC reviews this particularly includes compliance with safety provisions.

3. Can the stated requirements be verified?

The first two questions are covered together in Section 3.1 because they can be covered by the same review methodology. The last question is addressed in Section 3.2. Rejection criteria are covered in Section 3.3

3.1 Review for Scope and Contents

The requirements will not usually be furnished in a format or order that corresponds to that in which the guidelines have been presented in the previous chapter. The reviewers will therefore have to generate a matrix that relates the topics of the requirements document presented by the licensee to the topics of the guidelines. A hypothetical example of such a matrix is shown below. The list of topics in the Software Requirements Specification (SRS) is taken from Section 5 of IEEE Std 830-1993 (This standard is referenced in BTP HICS-14). The column headings are the major guideline subjects discussed in the preceding chapter.

Table 3-1. Association of SRS Topics and Guideline Subjects

Topics in the Software Requirements Specification	NUREG-0800	Precision	Completeness	Reliability	Robustness	Maintainability	Security	Response Time	HCI
5.2 Overall Description									
Product Perspective									
- System Interfaces			X		X				
- User Interfaces									X
- Hardware Interfaces		X		X		X	X		X
- Communications Interfaces					X		X		
- Memory Constraints								X	
- Site Adaptation Requirements	X		X						
Product Functions (overview)	X		X						
User Characteristics									X
Constraints									
- Regulatory Requirements	X			X	X	X	X		
- Interfaces to Other Applications			X			X			
- Redundant Operation	X			X	X	X			
- Audit Functions	X						X		
- Reliability Requirements	X			X	X				
- Criticality of the Application	X				X				
Assumptions & Dependencies						X			
5.3 Specific Requirements	See Table 3–2								
5.4 Supporting Information									
Safety and Hazards Analyses	X				X				
Physical Security Requirements	X						X		

The following listing of specific requirements is to be generated for each operating mode, such as reactor start-up, steady-state, reactor shut-down, reactor maintenance, and software test. The topics (rows) of Table 3–2 are based on Template A.2 of IEEE Std. 830. Where the same topic appears in both tables, it is understood that the listing in Table 3–2 refers only to deviations from the general requirements discussed in Table 3–1.

Table 3-2. Association of Specific Requirements and Guideline Subjects

Subjects in the A.2 Template of IEEE Std. 830	NUREG-0800	Precision	Completeness	Reliability	Robustness	Maintainability	Security	Response	HCI
Interfaces									
- User Interfaces			X						X
- Hardware Interfaces		X		X		X	X		
- Software Interfaces				X	X			X	
- Communication Interfaces					X			X	
Software Functions	All following row titles are to be repeated for each function								
- Validity Checks on Inputs					X			X	
- Sequence of Operations			X						
- Response to Anomalies					X				
- Effect of Parameters			X			X		X	
-Output/Input Relationships		X	X						
Performance Requirements									
- Total and Simultaneous Users							X	X	
- Volume of Data								X	
- Response Time Requirements			X					X	
Database Requirements									
- Data Types and Hierarchy			X						
- Frequency of Use								X	
- Integrity and Accuracy		X			X				
Constraints									
- Reliability and Availability					X				
- Security							X		
- Maintainability						X			
- Portability						X			

A practical review plan can be generated by going down each column and noting the subjects marked with an "X." The list of these marked subjects in each column constitutes the review plan for the topic indicated in the column heading. As an example, the construction of a review plan for the SRP Appendix 7.1-B Criteria is shown in Table 3–3.

Table 3-3. SRS Topic for Review vs. SRP Appendix 7.1-B Criteria

SRS Topic	Examples of Specific Review Topics*
Site Adaptation	Consistency with the body of the SRS and with the design basis
Product Function (overview)	Traceability to system requirements
Regulatory Requirements	Correctness, verifiability, and unambiguity of requirements
Redundant Operation	Completeness with respect to design basis
Audit Functions	Verifiability of access
Criticality of the Application	Prioritization of requirements
Safety and Hazards Analysis	Completeness with respect to design basis
Physical Security Requirements	Completeness (as above) and traceability to system requirements

*The complete Review Plan will typically contain multiple entries for each SRS topic

The checklists shown in Appendix B were constructed by the methodology shown in this example.

3.2 Review for Verifiability

Appendix 7.1-B of the SRP requires that:

[t]he information provided for the design basis items should be stated or provided in such a way as to facilitate the establishment of verification criteria and the performance of analyses and reviews of the various safety systems.

As a specific implementation of this provision, BTP HICS-14 defines *verifiability* for computer-based I&C systems as requiring:

… that it be possible to construct a specific analysis, review, or test to determine whether each requirement has been met.

Thus, the material provided for review should indicate how each of the provisions of the SRS is to be verified. The adequacy of the verification measures can be evaluated by asking the following questions:

1. Has this verification measure previously been used for verification of a comparable requirement? If yes, has it been found adequate? If no, are there fall-back provisions in case it is found to be inadequate?

2. Is the proposed measure suitable for determining compliance with the requirement at an early phase in the software development cycle?

Because problems found during verification late in the development process are much more difficult to correct, and because late correction is more likely to affect other functions, preference should be given to verification means that are suitable for early development phases, such as formal methods and structured reviews. Use of these verification methods does not eliminate the need for testing (late phase verification) but it reduces the probability that problems will be found in tests.

3. Is the verification method objective and repeatable?

   Peer reviews may not always be objective, and some forms of random testing are not repeatable. Where these verification methods are proposed, they should be supplemented by methods that are objective and repeatable—possibly they can be performed at a later development phase.

4. Will there be visibility as the details of the verification activities are developed?

   It is usually not possible to supply full details on the verification activities at the time the requirements are generated. A statement that formal methods will be used may later be augmented by the specification of the language and proof engine that will be employed. Test plans prepared during the requirements phase need to be fleshed out with test specifications, test procedures, and test schedules. The milestones at which such additional detail will be available should be identified as part of the requirements review.

3.3 Rejection Criteria

Rejection criteria can, in most cases, be developed directly from the checklists in Appendix B. As an example, for the SRS topic "Site Adaptation," the requirements documents should be rejected if the site adaptation provisions are not consistent with the design basis. In the review for some column headings, particularly robustness requirements, rejection criteria may have to be developed at a more detailed level. Two examples are presented. For the SRS topic "Regulatory Requirements" in Table 3–1, the decision to reject under the robustness heading may be based on lack of the following criteria:

(a) Reference to applicable regulations and standards (e. g., IEEE Std-603)
(b) Definition of the scope of the applicability of the standard (e. g., from sensed variable to actuator)
(c) Verification methods for determining compliance

Similarly, for SRS topic "Redundant Operation" in Table 3–1, requirements for avoidance of common mode failures can be rejected if they do not separately address the following:

(a) Source code design
(b) Tool utilization (including compilers and linkers)

(c) Verification provisions, in this case with emphasis on test case generation
(d) Code maintenance

In addition to the rejection criteria related to safety concerns discussed above, the requirements documentation may have format deficiencies that preclude the application of the checklists in Appendix B or of the guidelines as a whole. IEEE Std-830 permits much latitude in the presentation of the SRS topics, including ordering by operating mode, by user class, by object controlled, by stimulus, by feature, etc. But the material to be presented in an SRS is described in sufficient detail (in Section 6 of the standard) to support a meaningful review by the guidelines of this report. It follows that it might be a reason for rejection if material required in Section 6 of the standard is not present in the documentation furnished for the review.

References

ANSI/IEEE Std 100-1977, *IEEE Standard Dictionary of Electrical and Electronics Terms*, Frank Jay, Editor-in-Chief, New York: Institute of Electrical and Electronic Engineers, 1977.

Beltracchi, L., "NRC Research Activities," *Proc. Digital Systems Reliability and Nuclear Safety Workshop*, NUREG/CP-0136, conducted by the NRC in conjunction with NIST, 1994.

Bouissou, M., F. Martin, and A. Ourghanlian, "Assessment of a Safety-Critical System Including Software: A Bayesian Belief Network for Evidence Sources," *Proc. Reliability and Maintainability Symposium*, Washington, D.C., 1999.

EPRI Working Group on Commercially Available Programmable Logic Controllers for Safety Related Applications, *Generic Requirements Specification for Qualifying a Commercially Available PLC for Safety Related Applications in Nuclear Power Plants*, Electric Power Research Institute, EPRI-TR-107330, December, 1996.

Fabio, D., and L. Kolle, "Digital Bus Installations are Safer and Cheaper," *Intech*, November 2000, p. 55.

Franklin, G., J. Powell, and M. Workman, *Digital Control of Dynamic Systems*, 2nd ed., Reading, MA: Addison Wesley, 1994.

Hecht, H., A. Tai, and K.S. Tso, "Class 1E Digital System Studies," NUREG/CR-6113, Washington, D.C.: U.S. Nuclear Regulatory Commission, October 1993.

Hecht, H., M. Hecht, and G. Dinsmore, NUREG/CR-6293, "Verification and Validation Guidelines for High Integrity Systems," Washington, D.C.: U.S. Nuclear Regulatory Commission, March 1995.

Hecht, H., M. Hecht, S. Graff, et. al., "Review Guidelines on Software Languages for Use in Nuclear Power Plant Systems", NUREG/CR-6463, Washington, D.C.: U.S. Nuclear Regulatory Commission, June, 1996.

Hecht, M., D. Tang, and H. Hecht, "Quantitative Reliability and Availability Assessment for Critical Systems Including Software," *Proc. of the 12th Annual Conference on Computer Assurance*, June 16-20, Gaitherburg, Maryland, 1997.

IEEE Std. 100-1977, *Glossary of Electrical and Electronic Engineering Terms*, New York: Institute of Electrical and Electronic Engineers, 1977.

IEEE Std 279-1971, *Criteria for Protection Systems for Nuclear Power Generating Stations*, New York: Institute of Electrical and Electronic Engineers, 1971.

IEEE Std 603-1991, *IEEE Standard Criteria for Safety Systems for Nuclear Power Generating Stations*, New York: Institute of Electrical and Electronic Engineers, 1991.

IEEE Std 754-1985, *IEEE Standard for Binary Floating Point Arithmetic,* New York: Institute of Electrical and Electronic Engineers, 1985.

IEEE Std 7-4.3.2-1993. *IEEE Standard for Digital Computers in Safety Systems of Nuclear Power Generating Stations,* New York: Institute of Electrical and Electronic Engineers, 1993.

IEEE Std 830-1993, *IEEE Recommended Practice for Software Requirements Specifications*, New York: Institute of Electrical and Electronic Engineers, 1993.

Jaffe, M. and N. Leveson, "Completeness, Robustness, and Safety in Real-Time Software Requirements Specifications," *Proceedings of the 11th International Conference on Software Engineering,* May 15-18, 1989, Pittsburgh, Pennsylvania. IEEE Computer Society/ACM Press, ISBN 0-8186-1941-4, 1989.

Jaffe, M., N. Leveson, B. Melhart, and M. Heimdahl, "Software Requirements Analysis for Real-Time Process-Control Systems," *IEEE Transactions on Software Engineering*, March 1991.

Johnson, G., D. Scrader, and R. Yamamoto, *Review Templates for Computer-Based Reactor Protection Systems*, UCRL-ID-139344, 1998. Also available as NUREG/CR-6680.

Kharanbanda, O.P. and E.A. Stallworthy, *Safety in the Chemical Industry: Lessons from Major Disasters,* Columbia, Maryland: G.Putnam Publishing, 1988.

Lawrence, J.D., *Software Reliability and Safety in Nuclear Reactor Protection Systems*, Nuclear Regulatory Commission, NUREG/CR-6101, November 1993.

Leffingwell, D., and D. Widrig, *Managing Software Requirements*, Addison Wesley Longman, 1999.

Leveson, N., *Safeware: System Safety and Computers,* Reading, Massachusetts: Addison-Wesley, 1995.

R. Lutz and R. Woodhouse, "Failure Modes and Effects Analysis," *Encyclopedia of Electrical and Electronics Engineering*, J. Webster, ed.: John Wiley and Sons Publishers, Vol. 7, 1999.

Neumann, P., *Risks to the Public*, electronic newsletter available at http://www.csl.sri.com/~risko/risks.txt, and http://catless.ncl.ac.uk/Risks, 2001.

Phillips, C. L. and H. T. Nagel, *Digital Control Systems, Analysis and Design,* 3rd Edition, Englewood Cliffs, New Jersey: Prentice Hall, 1995.

Prekshott, L., *Data Communications*, Nuclear Regulatory Commission, NUREG/CR-6082, May 1993.

SIAM News, Vol. 29. Number 8, 1996, Society for Industrial and Applied Mathematics, available from http://www.siam.org/siamnews/general/ariane.htm, October 1996.

Smidts, C., "Software Engineering Measures for Predicting Software Reliability in Safety Critical Systems," NUREG/CR-0019, October, 2000.

Tang. D. and M. Hecht, "Evaluation of Software Dependability Based on Stability Test Data," *Proc. 25th Int. Symp. Fault-Tolerant Computing*, Pasadena, California, pp. 434-443, June 1995.

U.S. General Accounting Office, *Patriot Missile Defense: Software Problem Led to System Failure at Dhahran, Saudi Arabia*, GAO/IMTEC-92-26, 1992.

USNRC, "Criteria for Digital Computers in Safety Systems of Nuclear Power Plant," Regulatory Guide 1.152. Rev. 1. Washington, D.C.: Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission, January 1996a.

USNRC, "Human-System Interface Design Review Guideline," NUREG-0700, Vols. 1 and 2., Office of Nuclear Regulatory Research, June, 1996b.

USNRC, Branch Technical Position HICB-14, "Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems," Rev. 4, June 1997a.

USNRC, "Software Requirements Specifications for Digital Computer Software Used in Safety Systems of Nuclear Power Plants," Regulatory Guide 1.172., Washington, D.C.: U.S. Nuclear Regulatory Commission, 1997b.

USNRC, "Standard Review Plan," NUREG-0800 (formerly NUREG-75/087), Appendix 7, Washington, D.C.: U.S. Nuclear Regulatory Commission, June 1997c.

USNRC, "An Approach for Using Probabilistic Risk Assessment in Risk-Informed Decisions on Plant-Specific Changes to the Licensing Basis," Regulatory Guide 1.174, Washington, D.C.: U.S. Nuclear Regulatory Commission, July 1998.

USNRC, "Selection of Sample Rate and Computer Wordlength in Digital Instrumentation and Control Systems," NUREG-1709, Washington, D.C.: U.S. Nuclear Regulatory Commission, July 2000.

Appendix A—Glossary

Critical characteristics are those properties or attributes that are essential for performance of an equipment's safety function (IEEE Std 934, "Requirements for Replacement Parts for Class 1E Equipment in Nuclear Power Generating Stations"). A similar definition is provided in EPRI
NP-5652, "Guideline for the Utilization of Commercial Grade Items in Nuclear Safety Related Applications," in relation to commercial dedication. (NUREG-0800)

Design output includes documents, such as drawings and specifications, that define technical requirements of structures, systems, and components (ASME Std NQA-1, "Quality Assurance Requirements for Nuclear Facility Applications"). For software, design outputs are the products of the development process that describe the end product that will be installed in the plant. The design outputs of a software development process include software requirements specifications, software design specifications, hardware and software architecture, code listings, system build documents, installation configuration tables, operations manuals, maintenance manuals, and training manuals. (NUREG-0800)

Dynamic non-linearity, or DNL, reflects the maximum deviation in going from one step to the other and may be quite significant where measurements are being done on the difference between values (e.g., flux level changes).

Design process comprises technical and management processes that commence with identification of design input and lead to and include the issuance of design output documents (ASME Std NQA-1).

Design requirement is a requirement that specifies or constrains the design of a system or system component (IEEE Std 610.12, "IEEE Standard Glossary of Software Engineering Terminology").

Deterministic refers to a property of a computer or communication system such that the time delay between stimulus and response has a guaranteed maximum and minimum (NUREG-0800).

Embedded software or *firmware* is software that is stored in read-only memory and built into a computer dedicated to a pre-defined task. Normally, embedded software cannot be modified by the computer that contains it, nor will power failure erase it; some computers may contain embedded software stored in electrically erasable programmable read-only memory (EEPROM), but changing this memory typically requires a special sequence of actions by maintenance personnel (NUREG-0800).

Firmware—see *Embedded software*

Function is a specific purpose of an entity or its characteristic action (IEEE Std 610.12, "IEEE Standard Glossary of Software Engineering Terminology").

Functional characteristic is a trait or property of a design output that implements a functional requirement, a portion of a functional requirement, or a combination of functional requirements. BTP

HICB-14 identifies specific functional requirements considered in software reviews (NUREG-0800).

Functional requirement is a requirement that specifies a function that a system or system component must be capable of performing (IEEE Std 610.12).

Gain error describes the deviation between the full-scale actual change in the input signal and the output of the A/D converter.

Hardware critical characteristics are those properties or attributes of computer, peripheral, or communication hardware that are essential for performance of the connected equipment's safety function. This includes meeting specifications that are required to execute the software intended to run on the hardware, as well as attributes of reliability, testability, or predictability upon which the staff's safety findings are based. (NUREG-0800)

Linearity error is the maximum deviation of the A/D converter from the ideal to the actual across the range of the A/D converter.

Maintainability refers to attributes related to both online, in-service testing and diagnostics and to the means by which the source code reduces the likelihood that faults will be introduced during changes made after delivery.

Missing code is a deviation of the A/D conversion in which one or more increments of the input signal are not reflected in the A/D output.

Non-monotonicity is the deviation of the direction (sign) of A/D conversion relative to a change in the input signal (i.e., decreasing when the input signal is increasing or increasing when the signal is decreasing).

Offset error is the consistent shift (higher or lower) between all values of the A/D converter and the input value.

Predeveloped software (PDS) is software that already exists, is available as a commercial or proprietary product, and is being considered for use in a computer-based function (IEC Std 880, "Software for Computers in the Safety Systems of Nuclear Power Stations," Supplement 1 draft). Commercial, off-the-shelf (COTS) software is a subset of PDS.

Reliability is the probability that the (hardware or software) component will not cause a failure of a system for a specified time under specified operating conditions. The probability is a function of the inputs to and use of the system, as well as a function of the existence of defects (faults) in the component. The inputs to the system determine whether existing defects (faults), if any, are encountered. (ANSI-AIAA R-013 1982)

Robustness refers to attributes related to the specification of behavior of the software in the presence of unexpected; incorrect; anomalous and improper (1) input, (2) hardware behavior, or (3) software.

Response time requirement refers to the time constraints for operation of a function.

Setpoints are the levels of monitored parameters that define the onset of unsafe conditions.

Software critical characteristics are those properties or attributes of a software or firmware product that are essential for performance of the related equipment's safety function. This includes functional requirements that are allocated to the software product, as well as attributes of robustness, testability, or dependability upon which the staff's safety findings are based. (NUREG-0800)

Appendix B—Checklists for Conducting a Review of Requirements

The following checklist is based on the topics and their ordering for a Software Requirements Specification (SRS) in accordance with IEEE Std 830. Although this standard is listed in BTP HICS-14, there is no assurance that a given requirements document will adhere to it. Also, the standard allows for several alternative ordering of the subjects. Where the standard is not followed, or where the ordering differs from that assumed here, the reviewer will have to use judgment in identifying equivalents to the topics used in this checklist.

The checklists are presented in the order in which guideline subjects are listed in Chapter 2 of the Software Requirements Guidelines.

Table B–1. Checklist for SRP Appendix 7.1-B Criteria

From	SRS Topics	Examples
Table 3–1	Site Adaptation Requirements	Consistency with design basis
	Product Function (Overview)	Traceability to system requirements
	Regulatory Requirements	Correctness and verifiability of SRS statements
	Redundant Operation	Completeness with respect to design basis
	Audit Functions	Verification of access
	Criticality of the Application	Prioritization of requirements in the SRS
	Safety and Hazards Analysis	Completeness with respect to design basis
	Physical Security Requirements	Traceability to system requirements

Table B–2. Checklist for Precision

From	SRS Topics	Examples
Table 3–1	Hardware Interfaces	Memory and bus word size consistent with precision
Table 3–2	Hardware Interfaces	For specific function—see above
	Output/Input Relationships	Explicit statement of precision requirements
	Database Integrity and Accuracy	Must be consistent with above

Table B–3. Checklist for Functionality

From	SRS Topics	Examples
Table 3–1	System Interfaces	Complete definition of the run-time environment
	Site Adaptation Requirements	
	Product Function (Overview)	Traceability to system requirements
	Interfaces to Other Applications	Initialization, synchronization at start-up
Table 3-2	User Interfaces	Indications and controls for degraded states
	Sequence of operations	Recovery from degraded states
	Output/Input Relationships	Completeness of process variables list
	Response Time Requirements	Completeness of parameter descriptions
	Data Types and Hierarchy	Functional completeness of software requirements

Table B–4. Checklist for Reliability

From	SRS Topics	Examples
Table 3–1	Hardware Interfaces Regulatory Requirements Redundant Operation Reliability Requirements	Failure detection capabilities
		Avoidance of single point failure mechanisms
		Avoidance of correlated failures
		Specification of qualitative and quantitative requirements
Table 3–2	Hardware Interfaces Software Interfaces	Failure modes for specific operations
		Failure detection and recovery

Table B–5. Checklist for Robustness

From	SRS Topics	Examples
Table 3–1	System Interfaces	Independence of redundant sensors, actuators and communication channels
	Communication Interfaces	
	Regulatory Requirements	Independence of monitors from the monitored function
	Redundant operation	Independence of internal processing for redundant processes
	Reliability Requirements	Specification of minimum acceptable data set
	Criticality of the Application	Identification of uninterruptible operation
Table 3–2	Software Interfaces	Independence applied to specific functions
	Communication interfaces	Toleration of interruptions and noise
	Validity Checks on Inputs	Input processing requirements
	Response to Anomalies	Operations requirements
	Integrity and Accuracy	Limits on step size in outputs
	Reliability and Availability	Consistency with requirements

Table B–6. Checklist for Maintainability

From	SRS Topics	Examples
Table 3–1	Hardware Interfaces	Status monitoring and diagnostics; reporting of configuration data
	Regulatory Requirements	
	Redundant Operation	Minimum required equipment list
	Assumptions and Dependencies	Restoration capabilities and times
Table 3–2	Hardware Interfaces	Status monitoring and diagnostics for specific operations
	Effect of Parameters	Upgrade and modification support
	Maintainability and Portability	Consistency with system requirements

Table B–7. Checklist for Security

From	SRS Topics	Examples
Table 3–1	Hardware Interfaces	Physical security
	Communication Interfaces	Unauthorized access and service interruptions
	Regulatory Requirements	Consistency with system requirements
	Audit Functions	Access and operations logs
Table 3–2	Hardware Interfaces	Prevention of unauthorized parameter changes
	Total and Simultaneous Users	Access control
	Security Constraints	Consistency with system requirements

Table B–8. Checklist for Timing

From	SRS Topics	Examples
Table 3–1	Memory Constraints	Memory access time
Table 3–2	Software Interfaces	Consistency with response time requirements, particularly with regard to error checking and roll-back.
	Communication interfaces	
	Validity Checks on Inputs	Effect of missed checks on response time must be evaluated
	Effect of Parameters	Consider worst case conditions
	Performance Requirements*	Must be separately evaluated for each mode of operation
	Database Frequency of Use	Consistency with response requirements

* All subheadings under this heading must be evaluated; see Table 3–2.

Table B–9. Checklist for Human-Computer Interactions

From	SRS Topics	Examples
Table 3–1	Hardware Interfaces	Software actions consistent with hardware layout
	User Interfaces	Event notification; feedback on operator inputs
	User Characteristics	Notification to operator of high priority events
Table 3–2	User Interfaces	Event notification for specific tasks

Appendix C—Importance to Safety Rankings

The following table summarizes the "Importance to Safety" rank for each major guideline. Only two rank levels are used: "H" for guidelines of high importance, and "M" for guidelines of medium importance. The rationale for this ranking for each guideline is also shown. Which guideline received which ranking can be explained as follows: Where a requirements error or omission could directly lead to a catastrophic failure the "H" level was assigned; where the error or omission could contribute to a catastrophic failure or could directly cause a non-catastrophic failure the "M" level was assigned. A rank of H/M denotes predominantly H importance with individual instances of M rank, while M/H denotes a predominantly M level with an individual instance of an H rank.

Table C–1. Importance of Safety Rankings

Guideline Number and Title		Rank	Rationale
2.1	SRP Criteria	H	Criteria recognized as being essential to safety.
2.2	Precision	H	Incorrect specification of precision in digital systems can cause unexpected and serious effects.
2.3	Specificity and Completeness	H	Ambiguous or incomplete requirements are cited as causes for many failures documented here.
2.4	Reliability	H/M	Violation of functional requirements, such as independence of channels, can cause catastrophic failures; quantitative requirements (failure rates) are ranked M.
2.5	Robustness	H	Toleration of disturbed environments is essential for accomplishing objectives of safety systems.
2.6	Maintainability	M/H	Lack of upgrade support is ranked as H because it has led to reported failures; others are ranked as M because they do not by themselves cause catastrophic failures.
2.7	Security	M	Violation of requirements does not by itself cause catastrophic failures.
2.8	Response Time	H	This is essential for prevention of catastrophic failures.
2.9	Human-Computer Interface	H	Missing, ambiguous, or delayed information to the operator can cause catastrophic failures.

Appendix D—Linking of Guidelines to Standards

The following table lists standards and related documents that are referred to in the guidelines. Most standards cover all of the major requirements topics identified in Chapter 2. The difference between them is the scope of applicability, ranging from the entire nuclear power plant to digital portions of the plant protection system. The listing in Table D–1 reflects direct contributions to the guidelines developed in Chapter 2.

Table D–1. Contributions of Standards to the Guidelines

Guideline Number and Title		Applicable Standards and Related Documents
2.1	NUREG-0800 Criteria	IEEE 279, IEEE 603, IEEE 7-4.3.2
2.2	Precision	IEEE 279, IEEE 754, NUREG 1709, NUREG/CR-6101
2.3	Specificity and Completeness	IEEE 279, BTP HICS-4, 5, 6, and 12
2.4	Reliability	IEEE 100, IEEE 279, IEEE 603, IEEE 7-4.3.2
2.5	Robustness	MIL-STD-1629, NUREG-0800, NUREG-1709
2.6	Maintainability	
2.7	Security	NUREG-0800
2.8	Response Time	IEEE 279, NUREG 1709
2.9	Human-Computer Interface	NUREG-0700

Appendix E—Reviewers

Dr. Peter Popov is a research fellow at the Center for Software Reliability (CSR) at City University in London, England. Over the past three years, Dr. Popov has been investigating the impact of diversity on software fault tolerance for critical systems. Dr. Popov, together with Alexander Romanovsky (see below), has been funded by a research program sponsored through the U.K. nuclear regulatory authority to investigate the application of software diversity in nuclear safety systems. This work is part of the Diversity in Safety Critical Software (DISCS) project, a research entity established jointly by the City University (London) and the University of Newcastle-upon-Tyne, both institutions with more than 25 years of research in software reliability assessment and software fault tolerance.

Dr. Alexander Romanovksy is a research associate in the department of computer science at University of Newcastle-upon-Tyne. Dr. Romanovksy's research interest include exception handling, rollback, and diversity, all of which are key elements in the setting of requirements for safety critical systems. Dr. Romanovksy has published more than 100 articles through conferences, journals, and book chapters, and holds a patent for a software recovery technique (assigned to Lucent Technologies). He works in collaboration with Dr. Popov researching the application of the diverse off-the-shelf software described above.

Mr. Reuven Greenberg is employed by the licensing division in the Israel Atomic Energy Commission (IAEC). He is also a private consultant in system and software safety. Prior to joining IAEC, he served as an operational researcher in the Israeli Air Force and was Faculty of Engineering at Judah and Samaria College. Mr. Greenberg is a member of the IEEE Software Safety Planning Group of the IEEE and the IEC TC 56 committee on dependability. His areas of interest include system and software safety, accidents research, development and analysis of component-based systems, etc.

Dr. Sourav Bhattacharya is an associate professor in the computer science/engineering department of Arizona State University in Tempe, Arizona. His primary research interests are networked and parallel computing, dependable communication, and ATM networks. He was selected for the IEEE Computer Society Distinguished Visitor Program, 1996–1999. He is the editor of the IEEE Computer Society Press Practice Series and has been guest editor for a special issue on "High-Assurance Systems Engineering" in the journal *Communications of the ACM* (CACM), Feb 1997.

Dr. Ann Tai is president of Iatech, based in Los Angeles, California. Dr. Tai has an extensive background in dependable computing and modeling, and she has worked on past NRC projects on design and verification of Class 1E systems and programming guidelines for nuclear power plant safety systems. Her other research work includes integrated performance and reliability assessment and work on reliability and fault tolerance in long duration space missions.