ELECTRIC POWER RESEARCH INSTITUTE

Handbook for Evaluating Critical Digital Equipment and Systems

WARNING:
Please read the Export Control Agreement on the back cover.

Technical Report

Effective December 6, 2006, this report has been made publicly available in accordance with Section 734.3(b)(3) and published in accordance with Section 734.7 of the U.S. Export Administration Regulations. As a result of this publication, this report is subject to only copyright protection and does not require any license agreement from EPRI. This notice supersedes the export control restrictions and any proprietary licensed material notices embedded in the document prior to publication.

Handbook for Evaluating Critical Digital Equipment and Systems

1011710

Final Report, November 2005

EPRI Project Manager
R. Torok

ELECTRIC POWER RESEARCH INSTITUTE
3420 Hillview Avenue, Palo Alto, California 94304-1395 • PO Box 10412, Palo Alto, California 94303-0813 • USA
800.313.3774 • 650.855.2121 • askepri@epri.com • www.epri.com

DISCLAIMER OF WARRANTIES AND LIMITATION OF LIABILITIES

THIS DOCUMENT WAS PREPARED BY THE ORGANIZATION(S) NAMED BELOW AS AN ACCOUNT OF WORK SPONSORED OR COSPONSORED BY THE ELECTRIC POWER RESEARCH INSTITUTE, INC. (EPRI). NEITHER EPRI, ANY MEMBER OF EPRI, ANY COSPONSOR, THE ORGANIZATION(S) BELOW, NOR ANY PERSON ACTING ON BEHALF OF ANY OF THEM:

(A) MAKES ANY WARRANTY OR REPRESENTATION WHATSOEVER, EXPRESS OR IMPLIED, (I) WITH RESPECT TO THE USE OF ANY INFORMATION, APPARATUS, METHOD, PROCESS, OR SIMILAR ITEM DISCLOSED IN THIS DOCUMENT, INCLUDING MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, OR (II) THAT SUCH USE DOES NOT INFRINGE ON OR INTERFERE WITH PRIVATELY OWNED RIGHTS, INCLUDING ANY PARTY'S INTELLECTUAL PROPERTY, OR (III) THAT THIS DOCUMENT IS SUITABLE TO ANY PARTICULAR USER'S CIRCUMSTANCE; OR

(B) ASSUMES RESPONSIBILITY FOR ANY DAMAGES OR OTHER LIABILITY WHATSOEVER (INCLUDING ANY CONSEQUENTIAL DAMAGES, EVEN IF EPRI OR ANY EPRI REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES) RESULTING FROM YOUR SELECTION OR USE OF THIS DOCUMENT OR ANY INFORMATION, APPARATUS, METHOD, PROCESS, OR SIMILAR ITEM DISCLOSED IN THIS DOCUMENT.

ORGANIZATION(S) THAT PREPARED THIS DOCUMENT

EPRI

MPR Associates, Inc.

NOTE

For further information about EPRI, call the EPRI Customer Assistance Center at 800.313.3774 or e-mail askepri@epri.com.

Electric Power Research Institute and EPRI are registered service marks of the Electric Power Research Institute, Inc.

Copyright © 2005 Electric Power Research Institute, Inc. All rights reserved.

CITATIONS

This report was prepared by

EPRI
3412 Hillview Avenue
Palo Alto, CA 94304

Principal Investigator
N. Thuy

MPR Associates, Inc.
320 King Street
Alexandria, VA 22314

Principal Investigator
D. Herrell

This report describes research sponsored by the Electric Power Research Institute (EPRI).

The report is a corporate document that should be cited in the literature in the following manner:

Handbook for Evaluating Critical Digital Equipment and Systems. EPRI, Palo Alto, CA: 2005. 1011710.

PRODUCT DESCRIPTION

Power plants are increasingly upgrading their instrumentation and control (I&C) systems, replacing aging and obsolete analog equipment with software-based digital systems. For applications that are critical to safety or plant operability, utilities need to have high confidence that these new digital systems are sufficiently dependable that they will not degrade safety or plant reliability. This handbook suggests and describes in detail a "critical digital review" (CDR) technique that examines the design of the digital system in concert with a review of the vendor's testing, operating history, and development and quality assurance practices to make the needed assessment.

Results & Findings

This handbook provides practical guidance on evaluating digital systems to provide assurance that they will function as needed and expected in critical applications. The guidance is useful in protecting against both safety and economic concerns and may be applied in a graded approach that tailors the effort consistent with the complexity of the system and application and the importance to safety and plant economics. CDR evaluations often uncover potential failure modes and undesired behaviors that are easily remedied once discovered but which can have significant consequences, including unnecessary and costly plant shutdowns if they surprise plant operators.

Challenges & Objectives

Utility design engineers need a cost effective, systematic approach to determine whether digital systems going into their plants are of sufficient quality for their intended applications. The equipment often appears simple, but this can be deceptive. Digital systems are usually far more complex than their analog counterparts, and it is usually impossible to verify adequate quality and performance solely through testing. Digital systems can exhibit subtle and unexpected behaviors and failure modes, which can result in more severe and difficult-to-predict consequences compared to the predecessor analog systems. The most difficult problems to address are caused not by aging and wear-out of hardware, but by design errors and undesirable behaviors resident in the systems from the beginning. An effective method to evaluate digital systems and find such potential problems should assess the process actually used by the vendor in developing the system and its software. It must also be able to look "inside the box" to understand the designed-in failure modes and behaviors. Such evaluations need digital expertise often not available to utility engineers, and they can be costly and open-ended.

Applications, Values & Use

As nuclear plants address problems associated with aging and obsolescence of their I&C equipment, with the expectation of license extension and decades of continued operation, more

and more digital equipment will be installed, much of it in applications that are critical to safety or plant economics. At the same time, economic forces will increase pressure to ensure the highest possible plant reliability and availability. As a result, detailed evaluations to provide assurance of high I&C system quality and dependability, once performed only for safety-related applications, will become the norm for all critical plant applications. Utilities, their suppliers, and contractors will all need to become proficient in supporting such evaluations.

EPRI Perspective

Digital technology offers significant potential for improving the safety, reliability, and performance of nuclear power plants. However, many digital upgrades have involved undesired learning curve events with significant economic losses. Most often, problems arise because digital systems are treated as "black boxes," with inadequate understanding leading to unexpected and unwanted behaviors. To avoid such problems, utility engineers should have sufficient knowledge of the quality and inner workings of their systems to anticipate the types of failures and abnormal conditions that could arise. The need for careful evaluations extends well beyond the domain of nuclear safety systems; front line control systems have far more potential to cause significant economic losses than standby safety systems. They tend to be much more complex, and they have far more opportunities to initiate undesired events should they malfunction.

This project is part of a multi-year EPRI initiative to help utilities design, implement, and license digital I&C upgrades in nuclear power plants. This handbook will help its users avoid problems that nuclear plants and other industries have experienced with digital equipment. This guidance is particularly significant in that it reflects the most up to date experience from actual evaluations of real digital systems. EPRI anticipates that this handbook on evaluating critical digital systems will prove to be a valuable tool in helping nuclear plants apply digital technology successfully.

Approach

The objective of this handbook is to enable the utility engineer to take advantage of the CDR techniques, either by overseeing outside consultants or by assembling a competent team and leading it through the process. The handbook was developed from earlier EPRI guidance that focused on evaluating commercial digital equipment for safety applications. The current guidance incorporates lessons learned in applying the original approach, resulting in a more systematic, comprehensive, and cost-effective way to approach the problem.

Keywords

Instrumentation and control
I&C modernization
Digital upgrade
Critical digital review

CONTENTS

LIST OF FIGURES

1
INTRODUCTION

1.1 Critical Systems

A large, complex industrial installation like a nuclear power plant (NPP) operates based on the correct and timely operation of a number of technical functions. The failure of some of these functions has the potential to affect plant *performance* and/or *safety* adversely and significantly. Engineers refer to such functions as *critical functions*. The systems that implement critical functions are referred to as *critical systems*.

For decades, engineers have applied processes in pursuit of assurance that the critical systems will perform the critical functions correctly, and in particular that:

- The critical system and its main components are precisely identified (*identification*).
- The characteristics of the system are clearly, thoroughly and unambiguously described (*functional characterization*).
- These characteristics are adequate for the real needs to be satisfied by, and for the real operational conditions of, the system (*functional adequacy*).
- In these operational conditions, the actual system will conform to its stated characteristics (*digital reliability*).
- Abnormal situations, which may be due to causes external or internal to the system, will result in acceptable system behavior (*robustness* and *safety*).
- Tests can reveal any faults created during manufacturing, installation or operation, with reasonable chance of success and with reasonable effort (*testability*).
- The above properties can be maintained throughout the system's operational lifetime, in particular when it is modified (*maintainability*).

1.2 Digital Systems

Engineers are usually prepared to specify hardware requirements, and to inspect, review, and test *hardware* components, subsystems, and systems. With microprocessor and computer-based systems (*digital systems*) however, there is an added *software* dimension, where established hardware-oriented processes fall short. It is not possible to adequately specify, inspect, review, or test a digital system with the same approaches and techniques used for pumps, valves, piping, and bolts, or even with the same techniques used for analog electronic components and systems.

Unfortunately, there are no uniformly practiced design methods to provide easy assurance that a digital system performs as expected. While there are multitudes of standards, procedures, and methods for developing software and digital systems, none can completely guarantee safe and/or reliable digital design for all types of systems and functionalities. Furthermore, there is wide variation of opinion among commercial suppliers about the cost/benefit justification of formal digital quality assurance methods.

The same lack of uniformly practiced methodology exists for software and digital systems testing as well. It is generally not feasible to perform exhaustive testing, subjecting software modules to every possible situation that may occur. Even if such comprehensive testing were possible for normal circumstances, physical events can produce random memory errors, which essentially modify the program in an unpredictable way.

An excellent source of further reading on the nature of risk with digital technology is the book *Safeware* by Leveson (see Reference [1]).

1.3 Existing EPRI Guidelines Regarding Digital Systems

EPRI Guideline TR-102348 (see Reference [3]) describes how a combination of methods and techniques can be used to provide reasonable assurance for digital I&C upgrades. EPRI Guideline TR-1002835 (see Reference [10]) provides recommendations for diversity and defense-in-depth (D3) evaluation for digital upgrades, and introduces the notion of *defensive measure*.

EPRI Guideline TR-107330 (see Reference [6]) describes how commercially available instrumentation and control (I&C) platforms can be pre-qualified for application to safety systems in NPPs, and EPRI Guideline TR-1001045 (see Reference [8]) provides guidance for proper implementation of these pre-qualified platforms in plant-specific applications.

EPRI Guideline TR-106439 (see Reference [5]) and its supplement EPRI TR-107339 (see Reference [7]) describe how a combination of methods and techniques can provide adequate assurance when using commercial-off-the-shelf (COTS) digital devices for safety applications in NPPs, and introduce the notion of *critical digital review* (CDR). EPRI TR-1001452 (see Reference [9]) lists and evaluates lessons learned from pilot applications of EPRI TR-106439 and EPRI TR-107339.

1.4 Critical Digital Reviews

What steps can a nuclear utility engineer take to gain assurance that a given critical digital system or platform has the necessary properties and will function as expected? Nothing can be done to gain complete (100% guaranteed defect-free) assurance. However, there are methods that the engineer can use to provide an adequate level of assurance. Typically, reaching an adequate level of assurance requires a combination of testing, review of operating history, knowledge of the vendor's system development and quality assurance practices, and a critical examination of the design of the digital system.

One method for assessing a critical digital system is a focused technical review, called a *critical digital review* (CDR), which investigates and documents the potential for unacceptable behavior to occur in service, due to deficiencies in the digital system specification, design, configuration, operation, maintenance, or documentation, or due to misapplication. EPRI TR-106439 (see Reference [5]) and TR-107339 (see Reference [7]), first documented this notion of CDR, and applied it to commercial grade dedication of safety-related digital equipment. This document updates and extends the recommendations of TR-106439 and TR-107339 based on the practical experience gained since the publication of these technical reports.

This document also broadens the recommended applicability of CDRs beyond dedication of commercial grade equipment. A CDR can be helpful in developing assurance of high dependability in other situations where it is warranted due to safety or operability concerns. Thus, the use of this handbook is recommended in other situations, based on consideration of the relevant risk factors. Such applications could include digital equipment important for plant performance, and safety-related digital equipment produced under a 10 CFR 50 Appendix B Nuclear Quality Assurance program (see Reference [14]). This wider scope is consistent with the original intent of the engineers who first developed and practiced these techniques.

Important Note: In this document, the term "system" is used to designate a digital system, usually the digital system being reviewed, which could be a generic I&C platform, an I&C device or a fully developed I&C system for a specific application.

1.5 Objectives and Organization of this Handbook

The purpose of this handbook is to provide practical guidance to nuclear utility engineers on how to conduct generic and application-specific CDRs. After reading this document, the engineer should know what to expect from a CDR and how to organize and prepare for conducting a CDR, whether it is to be performed separately or in conjunction with a vendor survey or audit.

This document is divided into five sections. This section provides a background from which to view the Critical Digital Review (i.e., it answers the "why" question). The second section contains a list of documents that are referenced in this handbook, and defines the main terms, expressions, abbreviations and acronyms. The third provides an overview of the CDR approach and assumptions. The fourth section provides guidance for actually performing the review (i.e., it answers the "how" question). The fifth section discusses the post-review activities.

Throughout this handbook, stand-alone figures and text boxes accompany the main body of the text to provide specific examples, sample questions, and clarifications. There is no "cookbook" for conducting CDRs. At least in part, a CDR is a discovery process, in which information is uncovered, and the answers to one set of questions may lead to pursuing a new line of questions or review topics. The stand-alone figures and boxes provide examples and anecdotal material to convey an understanding of the various paths a CDR might take.

Important note: While the lists of questions, criteria, etc. that are given can be very helpful, they are not all-inclusive and should not be applied simply as check-lists, ignoring the possibility that some answers may give rise to additional lines of questioning that don't appear in this handbook.

2 DEFINITIONS AND TERMINOLOGY

2.1 References

1. Safeware: System Safety and Computers, Nancy G. Leveson, Addison Wesley, 1995 (ISBN 0-201-11972-2).
2. *Guideline for the Utilization of Commercial Grade Items in Nuclear Safety Related Applications (NCIG-07)*. June 1988. EPRI NP-5652.
3. *Guideline on Licensing Digital Upgrades*. December 1993. EPRI TR-102348.
4. *Supplemental Guidance for the Application of EPRI NP-5652 on the Utilization of Commercial Grade Items*. March 1994. EPRI TR-102260.
5. *Guideline on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications*, October 1996. EPRI TR-106439.
6. *Generic Requirements Specification for Qualifying a Commercially Available PLC for Safety Related Applications in Nuclear Power Plants*. December 1996. EPRI TR-107330.
7. *Evaluating Commercial Digital Equipment for High-Integrity Applications: A Supplement to EPRI Report TR-106439*. December 1997. EPRI TR-107339.
8. *Guideline on the Use of Pre-Qualified Digital Platforms for Safety and Non-Safety Applications in Nuclear Power Plants*. December 2000. EPRI TR-1001045.
9. *Generic Qualification of Commercial Grade Digital Devices: Lessons Learned from Initial Pilots*. September 2001. EPRI TR-1001452.
10. *Guidelines for Performing Defense-In-Depth and Diversity Assessments for Digital Upgrades: Appling Risk Informed and Deterministic Methods*. December 2004. EPRI TR-1002835.
11. IEEE Standard 1012, “Software Verification and Validation Plans”, 1986.
12. IEEE Standard 610.12, “Glossary of Software Engineering Terminology”, 1990.
13. IEEE Standard 1228, “Software Safety Plan”, 1994.
14. Code of Federal Regulations Title 10, Part 50, Appendix B, “Quality Assurance Criteria for Nuclear Power Plants and Fuel Reprocessing Plants”.
15. Code of Federal Regulations Title 10, Part 21, “Reporting of Defects and Noncompliance”, 1995.
16. NUREG/CR-6294, “Design Factors for Safety-Critical Software”, 1994.

2.2 Definitions

Application-Specific CDR. A CDR focused on a finalized and operational system, and aiming at providing confidence that this system is appropriate to the particular critical function(s) that need to be performed, and to its particular operational conditions.

Availability. The characteristic of a system expressed by the fraction of time during which it is functioning acceptably.

Critical Digital Review (CDR). A focused technical review applied to a specific critical digital system to investigate the potential for unforeseen events and unacceptable behaviors, and to recommend mitigation strategies.

Common Cause Failure (CCF). Failure of equipment or systems that occur as a consequence of the same cause. The term is usually used with reference to redundant equipment or systems, or to uses of identical equipment in multiple systems. Common cause failures can occur due to design, operational, environmental or human factor initiators. Common cause failures in redundant systems compromise safety if the failures are concurrent failures, that is, failures which occur over a time interval during which it is not possible that the failures would be corrected (see Reference [10]).

Critical Attributes. Those important design, material, and performance characteristics of a system that, once verified, will provide reasonable assurance that the system will perform its intended critical functions. For commercial grade equipment used in safety-related applications, the critical attributes are the critical characteristics as defined by 10 CFR 21-1995 (see Reference [15]). The critical attributes addressed in this handbook are only those that pertain to digital issues.

Critical Failure. A failure that could have an impact on safety, or could cause large financial or social loss.

Critical Function. A function whose failure could have an impact on safety, or could cause large financial or social loss.

Critical System. A system whose failure could have an impact on safety, or could cause large financial or social loss. (Adapted from IEEE 610.12.1990, see Reference [12]).

Defensive Measures. A set of features designed in a digital system to avoid, eliminate or tolerate digital faults, cope with unanticipated conditions, and minimize the likelihood of critical failures.

Dependability. As used in this document, a broad concept incorporating various characteristics of digital systems, including reliability, safety, availability, maintainability, and others. (Adapted from NUREG/CR-6294, see Reference [16]).

Digital Fault. A fault in the design or software of a digital system.

Digital Failure. A failure resulting from the activation of a digital fault.

Digital System. A system based on digital technology.

Failure. Termination of the ability of a functional unit to perform a required function.

Fault. A defect that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function when subjected to a particular set of normal or abnormal operating conditions.

Fault Tree. A deductive logic diagram that depicts how a particular undesired event can occur as a logical combination of other undesired events.

Functional Adequacy. As used in this document, the capability of the stated characteristics of a system to satisfactorily address the services and performance expected of this system, considering the constraints resulting from the system's operational context.

Functional Characterization. As used in this document, the property of a system being described to the degree of detail and accuracy necessary for correct verification and validation, configuration, operation, maintenance and modification.

Functional Specification. A document that specifies the functions that a system or component must perform. (IEEE 610.12.1990, see Reference [12]).

Generic CDR. A CDR focusing on a specific platform, and aiming at providing confidence that the platform is appropriate for a particular range of critical functions and/or applications.

Human-System Interface (HSI). That part of a plant system through which personnel interact to perform their functions and tasks.

Identification. As used in this document, the property of a system being precisely and unambiguously identified, together with its main hardware and software components.

Platform, I&C Platform. Set of hardware and software components that may work co-operatively in one or more defined architectures (configurations). [IEC 61513].

Reliability. The characteristic of an item expressed by the probability that it will perform a required mission under stated conditions for a stated mission time.

Digital Reliability. The part of reliability related to digital failures.

Robustness. As used in this document, the ability of a system to function correctly or to provide graceful degradation in the presence of abnormal conditions, including malicious aggression.

Safety. As used in this document, the ability of a system not to cause unacceptable risk or harm.

System. A collection of components organized to accomplish a specific function or set of functions. (IEEE 610.12.1990, see Reference [12]). As used in this document, unless specifically stated otherwise, a digital system (usually, the digital system being reviewed).

Testability. The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. (IEEE 610.12.1990, see Reference [12]).

Validation. The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements. (IEEE 610.12.1990, see Reference [12]).

Verifiability. The degree to which a system or component facilitates verification.

Verification. The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of the phase. (IEEE 610.12.1990, see Reference [12]).

Walk-Through. A static analysis technique in which a designer or programmer leads members of the development team and other interested parties through a segment of documentation or code, and the participants ask question and make comments about possible errors, violations of development standards, and other problems. (IEEE 610.12.1990, see Reference [12]).

2.3 Abbreviations and Acronyms

A/D	Analog to Digital
ASIC	Application Specific Integrated Circuit
CCF	Common Cause Failure
CDR	Critical Digital Review
CFR	Code of Federal Regulations
CGD	Commercial Grade Dedication
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
D3	Defense-in-Depth and Diversity
D/A	Digital to Analog
DCS	Digital Control System
EPRI	Electric Power Research Institute
EEPROM	Electrically Erasable Programmable Read Only Memory
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
EPROM	Electrically Programmable Read Only Memory

FMEA	Failure Modes and Effects Analysis
FPGA	Field Programmable Gate Array
FTA	Fault Tree Analysis
HART	Highway Addressable Remote Transducer
HSI	Human-System Interface
I&C	Instrumentation and Control
I/O	Input/Output
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
NRC	Nuclear Regulatory Commission
O&M	Operation and Maintenance
PLC	Programmable Logic Controller
PROM	Programmable Read Only memory
RAM	Random Access Memory
RFI	Radio Frequency Interference
ROM	Read Only Memory
TR	Technical Report
QA	Quality Assurance
V&V	Verification and Validation

3 CRITICAL DIGITAL REVIEWS - OVERVIEW

3.1 What is a Critical Digital Review?

Simply stated, a Critical Digital Review (CDR) is a focused technical review applied to a specific critical digital system to investigate the potential for *unforeseen events* and unacceptable behaviors, and to recommend mitigation strategies. Unforeseen events are not limited to the actual operation of the physical (digital) system. For example, missing guidance on configuring systems has led to unforeseen events. Figure 3-1 lists a few questions that a CDR should attempt to answer to evaluate the potential for unforeseen events, and Figure 3-2 provides additional examples of unforeseen events.

By “unforeseen event,” we mean an event or situation that is a surprise to the project team. The event may be associated with unforeseen behavior of the system, or with an unforeseen confusing or difficult situation. It may be a surprise to the vendor as well as the user. However, sometimes the designers know about a potential behavior, but they do not recognize it as significant or potentially undesirable for a particular user application. Listed below are some of the questions a CDR attempts to answer to find potential unforeseen events.

Are the utility and supplier on the same page about project requirements and expectations, and about foreseeable behavior of the system? One could say that this is precisely the CDR objective -- to get digital system designers, programmers, and project team members on the same page to talk about system behavior as it relates to this application. It takes an astute technical perspective to facilitate this process.

Are there design problems (questionable decisions) that open the door to unforeseen failure modes? In some cases, potential for behavior unforeseen by the project team is discovered in tradeoff decisions the designers made about the digital system. The decisions may have been acceptable for some applications, but questionable for the project of interest.

Are there failure modes whose behavior may lead to inaction (or unfortunate actions) by people? Digital designers and programmers often do not go far enough in considering the impact of design issues and decisions. That is, they may not take each failure mode and consider how operators are likely to interpret and react to the system behavior.

Are there conceptual problems (oversights) in the designers’ functional specification, design or testing plans? Conceptual oversights in the functional specification, design or test plans can lead to behavior which is unforeseen by the designers.

Are there weaknesses in the supplier processes that give concern about the level of discipline and teamwork in the design and programming effort? If questioning during a CDR reveals genuine teamwork and uncovers no conceptual problems, then concern by the CDR team about conceptual problems is largely put to rest. On the other hand, if there was clearly no teamwork in developing the concepts and design, this is cause for concern even if no specific conceptual problems were found.

Figure 3-1
Potential for Unforeseen Events

Are there weaknesses in the supplier processes that increase concern about digital faults and system support? A thorough review of the software code is beyond the scope of a CDR. However, through evidence of suitable architectural design, appropriate *defensive measures*, discipline and teamwork in the software development and support processes the CDR team can gain confidence that digital faults will be few after testing, that they will decrease over time with no surprises in future releases or replacements, and that postulated residual faults are unlikely to cause critical failures.

Is there sufficient documentation for efficient and correct component replacements? When a system or component fails and must be replaced, there may be confusion about the correct settings of important jumpers, switches, parameters, application programs, and other critical configuration items. The CDR looks for documentation sufficient to avoid this type of confusion. Such operations are usually performed by the utility technicians, who initially may be unfamiliar with digital issues, and even the engineers who installed the equipment originally may have forgotten all the required configuration items. It may therefore be necessary for the project team to provide simple and unambiguous procedures and directives to configure the system. Thus, the vendor's documentation can be assessed mainly from a project team standpoint.

Are known problems communicated to the vendor staff responsible for maintaining the system? Sometimes unforeseen events arise because the people responsible for maintaining the system have not been informed of known problems.

Figure 3-1
Potential for Unforeseen Events (Continued)

Here are some examples of potential unforeseen events. Some of these were discovered the hard way, through unforeseen significant events. Others were discovered during reviews prior to any operational event occurring. A few are taken from similar systems found in the petrochemical industry.

In a radiation monitor, raw data was "live," but calculated values were "frozen." *Design problem:* The watchdog timer was not designed to assert a failure whenever a critical task does not run. This can happen if high-priority tasks execute more often than usual, or if high-priority tasks require more time than usual to execute. Not executing tasks caused calculated variables to remain at their last calculated values.

In a digital feedwater system: Redundant computers exercise control, with each loop going through a commercial single-loop controller. One computer is active and the other tracks it, with a switching system used to route the active computer through the controller. *Obscure behavior*: A failure (run-time error) in the single-loop controller caused feedwater valves to drift (sample and hold circuits no longer refreshed), with no way for operators to assume manual control.

In the same digital feedwater system: *Conceptual problem:* The inactive computer tracks the actual valve demand signal from the single-loop controller. When a computer failover occurs, the control resumes using a feedwater demand setpoint calculated to match the last measured valve signal output. Because this signal came from a computer in a failed condition, the setpoint could be bogus, and could cause a large disturbance or trip in the feedwater system.

In another digital feedwater system: *Conceptual problem:* Two controllers share information across a communication network. When the configuration is changed in a controller, information being passed to the other controller may be marked as bad for one scan. When this occurs, the digital feedwater system unexpectedly transfers from automatic to manual control, which annunciates in the control room.

Single-loop controller: *Obscure behavior:* In most failure conditions, the entire display will blink or display a clear message. However, for some failures it is possible for the display to simply freeze. During normal operation, there is a single pixel in the display which is made to blink. Operators should be trained to check for the blinking pixel if in doubt about whether the system is operating.

Figure 3-2
Unforeseen Event Examples

System with remote/local units: *Conceptual problem:* A system has remote and local units for display and keypad input. One unit is at the point of service, and the other in the control room. The control room unit is designed to reflect the mode and operation of the other unit. Rapid keypad input caused the two units to lose synchronization and function in different modes.

System with redundant processors: *Obscure behavior:* Redundant processors experienced simultaneous failures because of a software error in communications code common to both processors.

Replacement of switches: *Conceptual Problem:* An old system, with six make-before-break rotary switches feeding individual solid state devices, is replaced by a single rotary switch with six "enable" switches feeding a PLC. Potential unforeseen behavior arises because, in the new arrangement, there are several new input sequences feeding the scanned PLC logic. If the switch passes through a position in less than one scan interval, the PLC may not pick up that transient state.

Figure 3-2
Unforeseen Event Examples (Continued)

3.2 Why Perform a CDR?

Digital technology, while providing a vast array of benefits, can also exhibit obscure behavior and unique failure modes. In particular, contrary to *hardware faults* (which usually appear at random times, e.g., due to aging), *digital faults* (i.e., flaws in the design of the system or in its software) are present in a system right from the beginning and affect system behavior deterministically. Each time the system experiences the same sequence of operational conditions, it will faithfully repeat the same behavior. A sequence that activates a given digital fault to create a *digital failure* will trigger this failure systematically.

Because of the increased complexity (even for digital upgrades intended to provide exactly the same external functionality as analog predecessors), digital faults and digital failures are difficult to preclude. Thus, a good question to ask is: "Can a failure in this system cause or enable significant consequences?" If the answer is "no," then an audit or survey of vendor practices may be sufficient. If the answer is "yes," a CDR can help identify unforeseen events, possible unacceptable behaviors, and potential mitigation strategies. Figures 3-3, 3-4 and 3-5 and Examples 3-1 – 3-4 provide additional insights on issues that can arise during CDRs.

Performing a CDR early in the design process can also:

- Clarify both scope and requirements for utilities and vendors.
- Identify options and/or alternatives prior to detailed design.
- Identify non-suitable equipment and configurations.
- Identify potential risks.
- Identify potential mitigation strategies.
- Promote reasonable vendor/customer expectations.

Each of these benefits has the potential to reduce overall project, and lifecycle, cost.

Conceptual problems typically relate to the more challenging aspects of digital system design. These problems are often overlooked by application engineers unfamiliar with the subtleties of digital technology.

Continuous vs. discrete sampling: The old analog equipment operates based on continuous evaluation of the input values. The replacement digital equipment samples the inputs at discrete intervals (periodic scans), and creates discrete binary representations of the input that reflect the sensitivity and accuracy criteria designed into the digital system and its A/D converters. If the digital system does not sample at the right frequency or with appropriate resolution, accuracy and filtering, it will "see" transients differently from its analog counterpart and may behave very differently from the original device.

Analog vs. digital requirements: The requirements for the analog system are often poorly documented. To duplicate the original behavior, the function of the analog equipment must be understood completely, and some determination made as to whether its design decisions were made from expediency, a desire to avoid complexity, or to meet the needs of the plant application. Requirements for a digital system should reflect digital behaviors that may not have existed for the analog predecessor, e.g., scan interval requirements, behavior during momentary loss of power and boot up, and "must not do" requirements.

Support activities: Operators, engineers, and technicians are familiar with operation, surveillance, calibration, troubleshooting, and maintenance of the older analog equipment. Different concerns, requirements, processes, and training are needed to successfully operate, maintain, and repair the replacement digital equipment. These differences must be incorporated into the system specification, design, training, procedures, and other support activities.

Finite state concepts and synchronization: Digital systems typically employ scanning (analog inputs, switches, keypads, analog and discrete outputs, and such are polled and sampled), serial communication (parameters are sent from one unit to another using a series of discrete bits or values), shared memory, and modules where each has various modes and states. Designing the system so that all parts are consistent and in sync is a very complex task.

Component-wise emulation of solid-state devices: This is discussed in *Example 3-1: Solid State Replacement by PLC Emulation*, *Example 3-2: Conceptual Problem with Transient States,* and *Example 3-3: Conceptual Problem - Exhaustive Testing.*

Effects of dynamics (system and process): *see Example 3-4: Design Problem - Gas Sample System.*

Watchdog timers: This area brings a different type of conceptual problem. It is not a technically difficult subject, but users often do not recognize that implementing a watchdog timer involves several important design decisions that affect the degree of protection that the watchdog actually provides. See additional information in Figures 3-4 and 3-5.

High speed networks: Modules on a network typically utilize common communication service routines. This creates the potential for common-cause failures. High-speed networks are capable of overloading processors unless provisions are made to protect against this. In addition, response times for distributed functions need to consider the complete system architecture and potential variability of response times.

Common cause failure: More generally, the issue of common cause failure should be addressed, considering the deterministic behavior of software, the higher degree of concentration of processing (a digital electronic component can support many functions, and a failure could degrade or disable all these functions), data communications, etc.

Complexity *(functional complexity, design complexity, operational complexity):* Complexity can have many undesirable consequences, such as higher likelihood of design errors, increased V&V difficulty, and increased likelihood of unforeseen situations.

Figure 3-3
Conceptual Problems

***Is there a watchdog timer*?** The answer is usually yes. Without a watchdog component, time-critical tasks may degrade or cease to function entirely with no positive indication that this has happened.

Is the watchdog timer a separate device or part of the CPU? Many computer chips have a built-in watchdog timer circuit, which gives only limited protection against processor "hang-up" short of a chip failure. If the chip itself fails completely, then functionality of both the CPU and the watchdog timer may be lost.

Does timeout cause the equivalent of a power-up reset? In many cases, the design philosophy of the watchdog timer is that it should force an "automatic reboot" to restart the system if it hangs up for any reason. Under this design philosophy a watchdog timeout may have exactly the same effect as rebooting the system. This may or may not be acceptable for the planned application.

Does the watchdog timeout event cause a latched indicator or contact closure? If the event causes a power-up reset, there may be an external indicator or contact closure that is latched to show that a reset has occurred.

What circuits respond to the watchdog timeout signal? In more sophisticated systems, the watchdog timeout signal may be used to directly control the behavior of subsystems. It may, for example, cause output signals to go to a pre-determined state independent of the CPU. Circuits may be designed to force operator displays into a blinking mode on watchdog timeout.

Figure 3-4
Watchdog Timers

A *watchdog timer* protects against failure of the CPU to perform a task, or a set of tasks within a specified amount of time. It has a clocked counter which counts down from an initial value at a fixed rate, and it has a digital input by which the CPU can send a pulse to reset the counter to its initial value. In the event that the timer counts down to zero before the reset pulse occurs, a *time-out signal or event* is generated.

The mere presence of a watchdog circuit does not mean that all critical tasks are protected against time-performance degradation. The "task coverage" of watchdog protection is determined by the organization of tasks and the manner in which timer reset functionality is implemented within those tasks. In design reviews, it is not rare to find that the watchdog timer fails to protect critical tasks against time - degradation. In some cases, one finds that the watchdog protects only against complete failure of the CPU.

To determine "task coverage" of the watchdog timer the review team must understand the specific task organization of the application, the use of interrupts, and the details of where watchdog reset occurs within the software. Typical task organization and the relevant watchdog issues are discussed below.

One common task organization: Background/Foreground
In simple real-time applications, software is often separated into just two sections, one of which is a small streamlined task triggered by a clock-driven interrupt so that it executes at regular time intervals, providing the "heartbeat" of the application. At every tick of the clock, this *foreground* portion performs input-output housekeeping, and increments a counter used by the other software section for timing.

At each clock tick, foreground interrupts the other section (*background*) which executes from some starting point to an end point where it branches back to the beginning. Sometimes the frequency of background execution is limited with an idle loop at the end which continually checks the contents of a service counter (incremented by the foreground portion). When the counter equals or exceeds a specified value, the background section exits the idle loop, resets the counter, and branches back to the start.

Figure 3-5
Watchdog Task Coverage

A watchdog may protect against only foreground degradation, against only background degradation, or against degradation in either section. If the watchdog is reset unconditionally in the foreground section, then there is no protection against background degradation. Even if background enters a tight infinite loop, foreground executes every clock tick and resets the watchdog.

One common way to protect both foreground and background is to put the reset portion in foreground, but with logic requiring the setting of an “I’m alive” flag by background since the previous watchdog reset.

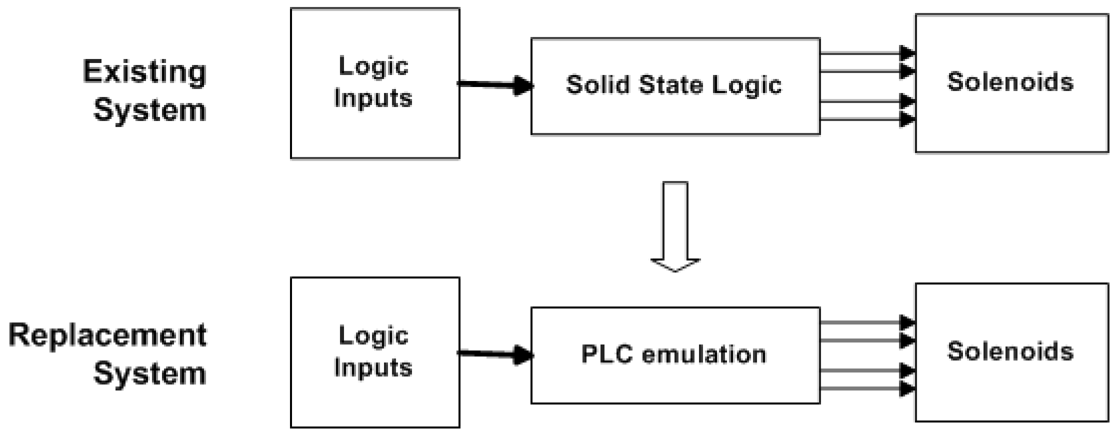
Another common organization: Executive with scheduled, prioritized tasks
In more complex real-time applications, software is organized into concurrent tasks which are marked for execution using service counters, and receive execution time based on assigned priority. There is still a “heartbeat” section (called the *executive* or *kernel*) driven by a clock interrupt, whose main job is to maintain timing counters for each task, to mark tasks for execution, and to pass execution to the highest priority task marked for execution.

In software organized in this way, it is possible for tasks to be “locked out” by higher priority tasks, and if reset functionality is placed in higher priority tasks, no watchdog timeout will signal the event.

Figure 3-5
Watchdog Task Coverage (Continued)

Example 3-1: Solid State Logic Replacement by PLC Emulation

Solid state logic is to be replaced by a system using a PLC as illustrated in the diagram below. This is to be a one-for-one replacement in which the PLC will simply replicate the existing logic. The PLC ladder logic is to be derived by mapping components and connections of the solid state device into corresponding ladder logic components.



Correct operation is to be validated by testing output values against an input-output table for all possible combinations of inputs. This is expected to provide an exhaustive test of the system’s input-output behavior.

Also, it is known that with the existing system, certain “invalid” solenoid states cannot occur unless there is a malfunction. As an additional test for PLC health, a section of ladder logic will be inserted to test output values at the end of each scan cycle. On detection of an “invalid” output state, the PLC will assert a failure condition.

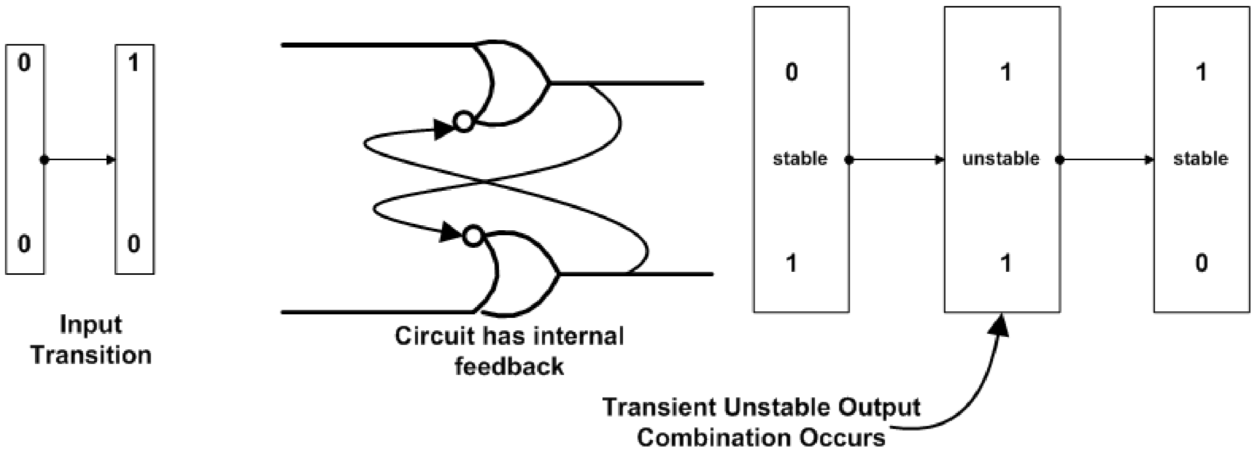
As discussed in subsequent examples, ***there are three conceptual problems with this plan****. First, the possibility of transient “invalid” states is not considered. Second, in a fault-tolerant scheme with multiple PLC units, this has the potential for enabling common-cause failures. Third, the PLC and the application program have memory, and operation cannot be exhaustively tested simply by applying known combinations of inputs.*

Example 3-2: Conceptual Problem with Transient States

If a logic circuit contains feedback around internal components and designers fail to consider the differences between relay logic and the PLC emulation of relay logic, the circuit is unlikely to work correctly.

In a logic circuit with feedback, certain output combinations may exist for brief transition periods in relay logic, but never appear as *stable* output values in relay logic. The transition state outputs may not even propagate to the circuit or solenoid outputs. However, in PLC emulation of relay logic, the transition state outputs are likely to occur for one or more scans, while the improperly constructed logic is solved enough times to achieve a stable output.

To see this, consider a component-level emulation of a flip-flop implemented with OR and NOT gates using feedback. This is just an illustration -- a flip-flop is an individual component within PLC programming and there is no generic issue with the utilization of flip-flops. However, PLC applications can be constructed which do have more complex versions of the problem illustrated below. The diagram below shows an output transition from one stable state to another forced by a nonzero input applied to the first input.


0
0
1
0
Input Transition
Circuit has internal feedback
0
stable
1
1
unstable
1
1
stable
0
Transient Unstable Output Combination Occurs

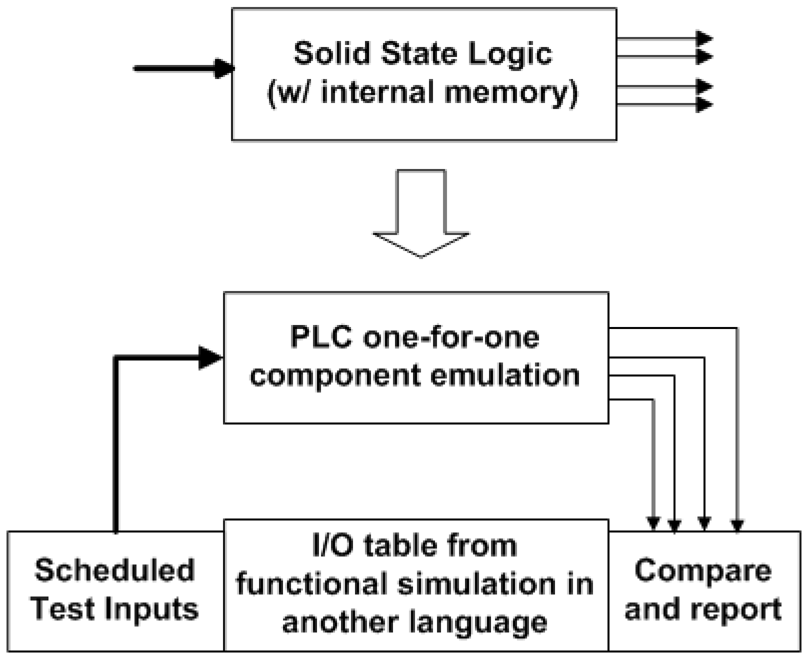
In a PLC emulation of this circuit, feedback is achieved by passing output states forward to be used as inputs to the OR gates in the subsequent scan cycle. For the input transition shown, a PLC will require two scan cycles to reach the stable output, with the transient unstable output occurring at the end of the scan cycle where the input transition is detected. Now referring back to the case of Example 3-1, invalid solenoid output combinations can occur at the device outputs during input state transitions. These unstable output states may exist for such a brief time that the solenoids do not respond. Depending on this behavior is not good programming practice and should be avoided. If a scheme is implemented to detect “invalid” output states, it may react to the transition states and trigger actions that are not required, which may create more issues than the detection scheme is supposed to resolve.

There are two general conclusions from this, suggesting the need for careful analysis in the design of PLC one-for-one component emulation replacements for solid-state circuits containing internal feedback around components:

- For some input transitions, PLC emulation may require multiple scan cycles to converge to the stable final output value, where the number of cycles depends on the application design. If feedback is complex, correct operation may even require that input values be latched in the PLC until transients finish.
- It may not be correct to treat the occurrence of an unstable, transient output combination as an error. In fact, this itself can cause more severe failures to occur.

Example 3-3: Conceptual Problem - Exhaustive Testing

In formulating test plans for a PLC emulation of a solid-state circuit with internal flip-flops (or other forms of memory), a conceptual problem can occur with respect to the assumption of exhaustive testing. This can also occur in traditionally coded implementations.


Solid State Logic (w/ internal memory)
PLC one-for-one component emulation
Scheduled Test Inputs
I/O table from functional simulation in another language
Compare and report

In a logic circuit that utilizes flip-flops (or other memory devices), the relationship between the output values and the input values is not static; that is, the output values depend not only on the current input values, but on previous input values as well.

If the PLC logic has memory, a test sequence that simply cycles through all possible input combinations will not test every state the logic can assume. An exhaustive sequence requires that every reachable set of internal flip-flop values should be combined with every possible set of input values. A test procedure would call for the application of a set of sequences, where each sequence has two parts: the first part of the sequence drives the internal flip-flops to one achievable combination, and the second part sequences through one of the possible input combinations.

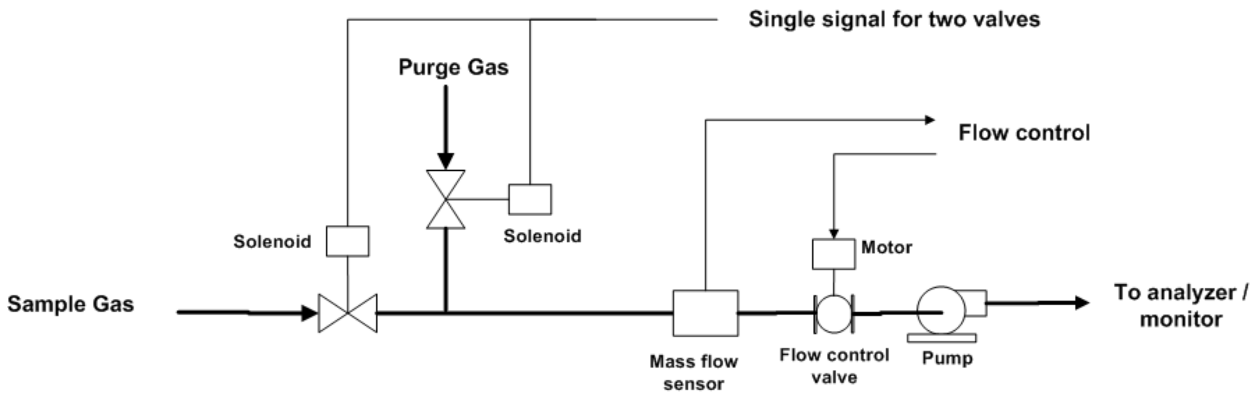
For all but very simple logic devices, it is extremely challenging to design a feasible test procedure which can be shown to achieve exhaustive testing. The number of tests and elements in the sequence increases very rapidly with the number of memory elements and the number of memory elements.

Example 3-4: Design Problem – Gas Sample System

One driving force for technical questioning during a CDR is the question of whether the team can identify credible silent failure scenarios, where system functionality may be lost with no subsequent indication in the control room or at points of routine surveillance.

To explore this question, the CDR team should gain an understanding of the non-digital components of the application, and the interaction of those parts with the digital portion.

The diagram below shows the skeleton of a digitally controlled gas and particulate sampling system which might be found in a nuclear or petrochemical application. It will be used to illustrate two failure modes.



Failure of the purge valve to the open position

Consider this credible silent failure scenario: Should the purge valve stick in the open position, an unknown mixture of purge air and sampled gas will flow through the sampling system. There will be no indication in the control room or at the sampling skid of this failure. If purge gas is similar to “normal” sample gas, then all systems will appear to be functioning normally with the purge gas valve stuck open, while the system fails to provide indication of the sampled gas.

While this scenario does not involve a failure of the digital portion, it does expose a design problem with the digital application. An additional contact closure on the purge valve, sampled as a discrete input, could be used to confirm complete closure of the purge valve.

Failure of the mass flow sensor mid-scale

The software checks sample flow against pre-set, configured limits. If the mass flow exceeds these limits, then the system asserts a failure. A failure of the mass flow sensor to an extreme high or extreme low value will cause such an assertion to occur.

Should the mass flow sensor “freeze” within the pre-set limits, however, this system will not assert a failure. Unless the flow is exactly as expected, the flow controller will drive the valve either fully closed or fully open. If the valve goes fully closed, there will be no sample flow through the detector. Since the failed flow meter is the only flow detector, there is no indication that flow has stopped. The pump may subsequently fail, with no direct failure indication. Before the pump fails, the gas sample may no longer be representative of the real conditions.

3.3 Relationship between a CDR and an Audit or Survey?

A CDR by itself does *not* constitute an audit or a survey. A CDR is *not* a substitute for a commercial grade survey or a nuclear quality assurance audit. There is, however, some overlap in activities, objectives, and results. Utilities may find that the CDR, or the results of the CDR, may be used as input to, in conjunction with, or as an integral part of an audit or survey. Combining these activities may be the most efficient and cost-effective approach for both the utility and the vendor.

3.4 Limits of a CDR

To maximize efficiency and effectiveness, a CDR should be confined to digital issues, avoiding such hardware-related issues as aging, electromagnetic compatibility, or ambient conditions, except as faults and failures resulting from those conditions affect the system's digital design and software. It should also focus on a specific, well-identified system, preferably in the framework of defined operational conditions, including operation and maintenance, configuration, and the application profile.

3.5 CDR Assumptions

There are nine underlying assumptions of the CDR process described in this document. These can provide guidance in considering the merits and difficulties of a CDR for a particular project, and in making adequate preparations so that the review is successful and cost-effective:

1. A CDR is not a self-contained activity, as follow-up is required.
2. A CDR is not a substitute for high quality.
3. A digital system is comprised of a generic platform and an application.
4. Almost all digital systems have design faults, but some may not be important.
5. Hardware and digital failures will occur in sub-systems.
6. Defensive design measures are valuable.
7. The "real system" consists of the digital system, vendor processes, and the project perspective
8. Development processes rarely function as described on paper.
9. CDR and Commercial Grade Survey activities are synergistic processes.

3.5.1 A CDR is not a Self-Contained Activity

The written report delivered after the review is *not* the final goal of the CDR. The CDR report usually generates recommended action items. These actions might include:

- Digital hardware/software modifications.
- Minor modifications to the engineering package.
- Utility planned test activities.
- Modifications or enhancements to the licensing documents for the system installation.
- Modifications to the Plant Simulator.
- Revisions to training.

Modification of operating procedures may also be indicated as mitigation strategies separate and apart from the system. The project team should resolve any concerns, questions, and recommendations included in the CDR report. For commercial equipment to be used in nuclear safety related applications, the results of the CDR should be incorporated into the dedication package if the CDR is credited as verifying some of the critical characteristics.

This assumption brings up another question, which can help in thinking about the potential benefits of a CDR: "What is the realistic scope of actions open to the team at this point in the project?" The point of greatest potential benefit is at the completion of detailed software design, before implementation has begun.

3.5.2 A CDR is not a Substitute for High Quality

A CDR is not a means of injecting quality where there is none, or where quality does not meet expectations. When performing a CDR on a poor quality system, the normal conclusion is to declare this system unfit for critical applications, or to recommend remediation actions. The implementation of such actions is not part of the CDR, and a new CDR could be necessary to evaluate the effectiveness of the actions implemented.

3.5.3 A Digital System is Comprised of a Platform and an Application

There are usually two basic layers to a digital system: the *platform* layer, and the *application* layer. Distinguishing between these two layers is important in considering the merits of a CDR, and in preparing to conduct one.

A platform is a portion of the digital system used as a generic base for developing applications, including the proposed project. The application layer is that portion of the digital system where the platform is adjusted, configured, and/or programmed to implement the requirements of a specific project.

A platform may be minimal, as in the case of most "smart" devices (sensors or actuator controls), where the vendor has designed specialized functionality around a microprocessor, with the hardware and software developed in-house using development and testing tools for the specific processor. In this case, the platform layer consists of the microprocessor architecture (hardware and software) together with the development tools, and the application layer consists of a limited number of parameter values.

On the other hand, a platform may be much more complicated, with a complex array of hardware and software components, as in the case of programmable platforms like programmable logic controller (PLC) or digital control system (DCS). Such platforms can be customized to meet the requirements of the project through a combination of:

- Hardware component selection, assembly and configuration.
- Software module selection and configuration (connection, option selections, parameter values).
- Programming (possibly in a language unique to the platform).
- Tuning.

The vendor may have developed a generic platform as a base for a family of applications, as is the case with some radiation monitoring systems. Or, it may be a commercial product (such as a PLC, DCS, smart instrument or smart actuator) purchased by the vendor for use in this application.

For a smart device, the application consists solely of the configuration, which adapts the platform to the proposed application. For a PLC or DCS, the application is likely much more complex, with one or more programs written or revised specifically to implement the plant requirements.

For a CDR to be effective, both the platform and the application should be subjected to technical questioning. A good application design residing on a poor platform will likely yield a poorly performing, unreliable system. A poor application design residing on a good platform will almost certainly yield a poorly performing, unreliable system.

3.5.4 Digital Systems can have Design Faults, but Some may not be Important

For all practical purposes, it is impossible to demonstrate formally that a digital system or a significant piece of software has no residual faults. This assumption is not meant to discourage the use of digital systems, but to simply acknowledge that in spite of our current best efforts in design techniques, peer review, testing, fault avoidance and fault removal, one cannot assume perfect software and digital design.

Fortunately, as explained in EPRI Guideline TR-1002835, it usually is a long way from a digital fault to a critical failure:

- A fault is a static defect, and unless activated, it does not cause harm.
- Only when particular operational conditions activate the fault can it cause the system to deviate from the intended behavior.
- Some deviations have insignificant consequences on the overall system behavior and do not prevent the system from meeting its specification. For example, the effects of the activation of the fault can be overridden and corrected by subsequent steps in the normal function of the system.
- Not all failures are critical, as many failures do not have the potential to affect plant safety or performance seriously.

- Critical systems are often redundant, being composed of several (usually identical) independent channels. In order to cause a system failure (as opposed to a single channel failure), the activating condition must affect several channels concurrently.
- When nuclear safety is concerned, critical plant systems, including their digital I&C, are often organized in multiple, independent, and diverse layers of defense (defense-in-depth). A *digital common cause failure* (CCF) of all layers of defense would require the same digital fault to be present and concurrently activated in each layer.

3.5.5 Defensive Design Measures are Valuable

The potential for critical digital failure or CCF can often be evaluated by assessing the designed in measures (hereafter called *defensive measures*) taken to minimize the likelihood of residual digital faults and activating conditions that could lead to critical failures or CCF.

EPRI Guideline TR-1002835 (see Reference [10]) introduces and discusses defensive measures extensively. In particular, this guideline suggests that:

- The different types of possible digital faults should be systematically identified, including specification faults.
- Measures should be taken to minimize or preclude the occurrence of each type of fault.
- The different *influence factors* that could affect the behavior of the digital system should be systematically identified. These factors may be internal or external to the system.
- The influence factors that could activate postulated faults and lead to critical failures should be eliminated or restricted to a point where testing and other forms of verification can provide a sufficient level of confidence.

3.5.6 Hardware and Digital Failures can Occur

Hardware failures will occur. Digital failures cannot be precluded. Digital systems that are not designed to cope with such failures are unlikely to guarantee an acceptable behavior when failures occur. Thus, a normal practice of designers (and CDR reviewers) of critical digital systems and software should be to ask: "What can go wrong? What reasonable measures can be taken to prevent that from becoming a catastrophe?"

3.5.7 The CDR also Covers the Project Perspective

The object of a CDR is more than the digital system itself: it also includes vendor processes and the *project perspective*. Term *project perspective* is used to mean the scope, formal requirements, documented operational conditions, and informal assumptions of the project. Informal assumptions are often unstated and simply assumed by project members. While they may hold for engineers familiar with the plant system(s) in which the equipment will be applied, a vendor's engineers — who may be quite competent in their respective areas — may not understand the subtleties of a specific customer application.

Suppose, for example, that high-consequence events may be enabled or caused by an unusual failure in the digital system, and that the CDR does not rule out the possibility of this type of failure. The project team can recognize that the productive focus for mitigation is likely to be the project perspective. Perhaps the level of automation expected is higher than absolutely necessary. Maybe simple mitigation can be found in a manual bypass of the complex automation.

Figure 3-6 provides a list of questions that help clarify the project perspective.

The first task of the CDR team is to understand the project scope, formal requirements, and informal expectations. This begins before the actual on-site reviews by reading specification documents and system descriptions, and by talking to project team members.

The CDR team should attempt to uncover underlying expectations, unstated assumptions, and any criteria that the project team will use to measure success. Here is a limited set of questions that are intended to help uncover this information:

Is this a Class 1E application, an application that challenges safety systems, or a system whose failure has large economic or non-nuclear safety impacts? In these applications, there is clearly a high degree of concern about loss of functionality due to single component failure, and due to common-cause failure of redundant or similar components due to behavior of a replicated software component. In critical digital reviews, the search for single-component failure modes is not confined to software, but includes the entire system.

What functions are served by this system, and how do they fit in the overall scheme of things?

What was the initial driving motivation for the planned system change or upgrade?

What improvements over the current situation are expected?

What are the worst conceivable consequences of (possibly silent) failure of one or more of these functions?

What human interfaces are in the control room? It is primarily through these that the operator sees and understands system behavior.

What human interfaces are outside the control room? This is a slower window to system behavior.

What process equipment does the system directly control or actuate: pumps, valves, motors?

More generally, are all the entities (other systems and equipment, or staff) that interact with the system identified and adequately characterized? Also, what are the different modes of operation (normal and abnormal) of these entities? What are or what should be the effects of these modes on the system?

What process variables does the system sense or measure?

What physical constraints (cables, cabinets, console space, etc.) were important in the design?

What portions of the old system are reused in the new system?

Are there informal project expectations that are assumed but not stated?

What expectations are held by the maintenance department?

Does the maintenance department have qualified technicians for this or similar equipment?

Are there expectations for self-diagnosis by the system?

Figure 3-6
Project Perspective

3.5.8 Development Processes Rarely Function as Described

During the CDR, paper descriptions of system and software development processes and procedures will be reviewed. However, a set of verification and validation (V&V) documents that meets recommended guidance (as stated in the IEEE Standard for Software Verification and Validation Plans, IEEE Std 1012-1986 (see Reference [11]), for example) can be produced independent from any actual development process. Reading impressive-looking documents may not give the utility project team a clear or accurate picture of the actual discipline, organization, and teamwork, used in developing the software of interest to the project. Systems that were produced several years ago may have been designed and developed under totally different procedures.

During the CDR exercise, the review team will work closely with marketing personnel, designers, programmers, and project managers. This process allows the CDR team to gain a qualitative perspective of the discipline, organization, and teamwork in the digital design, development, and support processes. This perspective is vital to determining the "value" of documentation provided.

While "retrospective" validation is better than no validation, it does not impact the real-time development process. Just "fixing" problems found in a retrospective validation is not generally adequate. If errors are corrected, the changes should be performed in a very rigorous manner, as this may sometimes introduce new (and thus unknown) problems.

3.5.9 CDR and Commercial Grade Survey Activities are Synergistic Processes

The original process for *Commercial Grade Dedication* (CGD), defined in EPRI NP-5652, selected a set of characteristics that defined the safety requirements and physical characteristics of the equipment. From those characteristics, which EPRI NP-5652 called *critical characteristics*, a dedicator or qualifier can define the processes needed to dedicate the equipment.

For physical hardware, the dedicator or qualifier can evaluate strength, composition, hardness, brittleness, and other physical characteristics. The dedicator or qualifier can also measure size, weight, power consumption, and other physical properties and completely describe the object to be dedicated.

For digital design and software, the process must be augmented with additional characteristics that are more difficult to measure. EPRI Guideline TR-106439 provides a set of example critical characteristics for digital equipment (see Section 4.2 of that report). Defining the critical characteristics is one of the more difficult things we have to do in the commercial grade dedication process. These characteristics are the key to the tests, evaluations, and reviews we will perform during the process. Pick the wrong characteristics, and the dedicator or qualifier may not do enough work to create a defensible dedication, or the dedicator or qualifier may have to do too much work, making the equipment prohibitively expensive.

The CDR evaluates the digital design and the software against the critical characteristics. It targets any new issues that are uncovered that deal with dependability and integrity. The CDR process does very little with physical or environmental characteristics, which the commercial grade acceptance processes address.

3.6 CDR Prerequisite Information

Normally, prerequisite input information useful for a CDR has been generated by the end of the conceptual design stage. The CDR Team should have the following information from the project prior to the actual on-site review(s):

- Project scope.
- Design objectives.
- Design basis.
- Initial risk analysis.

The CDR team should consider the "unstated" or "assumed" project perspective relative to each of these areas. Many systems that meet the "documented contractual requirements" have failed in practice when they failed to meet the "intention."

Wherever possible, the CDR team should request and review any available vendor literature. Marketing material can often provide excellent overviews, and may illuminate the user expectations. Other customers should be contacted, and any issues noted for the review. Good preparation can minimize on-site time requirements.

The CDR team should carry any drawings, specifications, or descriptions that define the context in which the new system will be situated.

3.7 CDR Team

The success of a CDR relies heavily on the team assembled. The team should have the technical expertise in digital systems required to penetrate the technical design of the system under review and capture the part of its operational context that could affect its digital behavior, and enough experience to exercise reasonable judgment in determining the depth of the review that is needed.

Penetrating to the core technical architecture and capturing the operational context require an *astute technical perspective*. In short, astute technical perspective means sharp technical expertise tempered by a pragmatic sense of what level of detail is important in the discussion of vendor processes, system architecture, system environment, operational processes, and unforeseen behavior.

Technical expertise in digital hardware, software, and real-time systems is certainly required of the review team. The team should also be able to guide the discussions based on potential consequences to the plant system, which requires detailed knowledge of the behavior of the plant system. One or more team members should be able to postulate reasonable failures of the plant system and assess the impacts of failures and abnormal behaviors of the proposed vendor solution on the plant system.

The CDR team itself will usually comprise a subset of the project team augmented by specialists as required. There are three functional “types” of members required for a successful CDR team:

- Lead Reviewer
- Applications Engineer
- Digital System Engineer

One person may perform more than one of these functions. Also, the titles of these individuals will vary from organization to organization, and may change depending on whether the review is done in conjunction with a formal vendor audit/survey, or as a separate design review. Note that a Certified Lead Auditor, who would normally lead an audit or survey team, may not necessarily have the technical expertise required to fulfill the role of Lead Reviewer defined here for a CDR. Another member of the team may need to serve as Technical Lead for the CDR in that case.

3.7.1 Lead Reviewer

The function of a Lead Reviewer is to maintain a technical overview perspective during the CDR process. In this sense, a good Lead Reviewer is a good facilitator. The Lead Reviewer should be able to:

- Understand technical issues.
- Focus discussions.
- Resolve disputes.
- Identify root causes.
- Facilitate technical discussions.
- Provide associative skills.

Ideally, the Lead Reviewer should be capable of viewing the system objectively, without undue pressure from project schedule or cost. Those with direct accountability will naturally focus on those areas for which they are most accountable.

3.7.2 Application Engineering

CDRs should be performed in the context of the application or applications for which the equipment is intended. Individuals who are knowledgeable about the application(s) are integral to the success of the review.

The individual(s) who provides the function of the Application Engineer should understand the design of the current system (including interfacing systems), the functional requirements for the new system or component, any regulatory or licensing issues, and any design basis issues. Generally, the individuals filling this role will be system engineers or design engineers.

3.7.3 Digital System Engineering

The individual(s) performing this function should have a good understanding of real-time software, operating systems, microprocessor design, structured analysis, and verification and validation.

These individuals should have experience with real-time process systems, understand fundamental plant systems, understand general and "best practices" for systems/software engineering (e.g., design basis, modification process, digital communications), and be familiar with any applicable regulatory issues.

3.8 CDR Process

To optimize effort and to minimize initial design uncertainties regarding the appropriateness of pre-developed platforms, CDRs can be divided into two main types:

- A *generic CDR* focuses on a specific platform, and aims at providing confidence that the platform is appropriate for particular types of critical functions and/or applications. Often, a generic CDR can be performed in conjunction with a commercial grade vendor survey.
- An *application-specific CDR* focuses on a finalized and operational system, and aims at providing confidence that the system is appropriate to the particular critical function(s) that need to be performed, and to its particular operational conditions. It can usually rely on the results of the generic CDRs of the platforms used to implement the system, verifying that the recommendations made by the generic CDRs have been satisfied or that appropriate alternative measures have been taken.

A CDR (generic or application-specific) can be divided into five main stages:

1. Screening.
2. Off-site preparation.
3. On-site reviews.
4. Issuing the CDR report.
5. Post CDR analysis.

In the following, unless explicitly stated, "CDR" designates both generic and application-specific CDRs. The system under investigation is a platform in the case of a generic CDR, or a finalized system in the case of an application-specific CDR.

3.8.1 Screening

Not all systems are suitable for critical applications. The purpose of screening is to apply judgment on readily available information and at early stages of the project, to identify obviously and hopelessly unsuitable systems and thus avoid wasting effort, resources and time.

3.8.2 Off-Site Preparation

Preparation for on-site reviews begins during the conceptual design phase of a project. The various system-related issues and risks that are defined during this stage are input for the CDR team.

The Lead Reviewer should determine the size of the on-site review team, the need for specialists, the duration of the review, and the depth of the review. Parameters that should be considered include application and system criticality, system complexity, technologies used, customer experience, and vendor experience in the specific application.

The CDR team should ask for, and analyze, any available and transmissible documentation that can provide appropriate insight regarding the properties expected of the system. Such documentation may concern the system itself, its development process, and its operational history. It may also concern the vendor, its overall quality record and experience. Lastly, it may concern the project and its technical perspective. The team should try to identify potential issues, and try to resolve or at least clarify them ahead of on-site reviews.

The CDR team should then list the issues to be addressed during on-site reviews, determine the on-site reviews that will be necessary, and determine the approach that will be taken and the questions that will be asked during each on-site review. The list of issues and questions will usually be sent in advance to the vendor so that any necessary information or expertise can be made available for on-site reviews.

Usually, an on-site critical review will be performed at a vendor’s development location. In cases where a vendor produces a platform at one site, and integrates the application at another (or where one vendor uses a platform produced by another), both sites may require visits.

3.8.3 On-Site Reviews

An on-site review for a well-defined project of low criticality, and low complexity may be completed in 1 to 3 days. A critical project with relatively high complexity may require 5 days or more.

An actual on-site critical review follows a five-step process. As the review progresses, the later steps may actually be intermixed to facilitate the review. The five steps include:

1. System Orientation.
2. Process Orientation.
3. Thread Analysis.
4. Risk Analysis.
5. Identification of pending issues.

3.8.3.1 System Orientation

The System Orientation should normally begin during off-site preparation. The objectives of the System Orientation during on-site reviews are:

- Introducing the CDR Team and its function to the vendor.
- Gaining an overview of the documentation that is available for examination and of the system architecture, including hardware architecture, software architecture, and process flow through the system.
- Discussing the results of the off-site System Orientation preparation activities, and selecting topics for focused technical discussion.
- Identifying and scheduling vendor resources for remaining steps.

The System Orientation step will set the expectations for the remainder of the review. Many vendors are accustomed to audits where the customer focuses on the Quality Assurance (QA) process, often to the exclusion of the product. It is good to stress before the visit that the team will need access to design and programming experts who are knowledgeable of the specific product.

The vendor should also be informed that parts of the review will look at procedures, and other parts will entail technical question and answer sessions. Providing the vendor with a list of areas that the CDR will explore may help the vendor determine who should be available and present for the CDR, give some idea of the technical depth of the review. However, the vendor should not be given a detailed agenda for which they should prepare. The CDR team should have the flexibility to follow emergent paths and concerns without being constrained by rigid schedules.

The CDR should start with an introduction of team members. The team members should describe their area of expertise, and their experience. This introduction provides the vendor with a point of reference. The Lead Reviewer should provide an overview of the CDR, and the objective of the review. The vendor should then be asked to introduce their team in a similar fashion. CDR team members should note the members of the vendor team who represent their area of interest.

The vendor is asked to provide a system overview, showing the basic components and the functional flow of information and control. This overview allows the vendor to use presentation materials they may already have, and allows the review to start in a non-threatening atmosphere. The CDR team should note areas of interest during the overview, but leave detailed questioning for the architecture descriptions.

3.8.3.2 Process Orientation

Like System Orientation, Process Orientation should normally begin during off-site preparation. The objectives of the Process Orientation during on-site reviews are:

- To gain an overview of the documentation that is available for examination.
- To discuss the results of the off-site Process Orientation preparation activities, and to select topics for focused discussion.
- To identify and schedule vendor resources for remaining steps.

The process orientation examines the vendor's policies, procedures, and standards used in the development, documentation, testing, and maintenance of the product. It also examines the changes in policies, procedures and practices during the life of the product. Record keeping, failure investigation, and customer complaint handling are included in the review of the maintenance phase of the life cycle.

The process portion of the orientation typically does not include a critical review of the Quality Assurance organization, but may do so in cases where the CDR is integrated into a formal audit or survey.

3.8.3.3 Thread Analyses

Thread analyses follow specific issues through a vendor's documentation, testing, and implementation. Examples of thread analysis include: 1) tracing signals from field inputs through data acquisition hardware, software, outputs and display, and 2) tracing a customer complaint from the initial call, to the failure investigation, to the design change, to the document update, and to configuration control. In the first example, the thread analysis follows a purely technical path, while in the second, it follows a work process, evaluating technical aspects at discrete locations. Typically, the issues examined are determined during the preparation phase, and a minimum of two threads are analyzed.

The thread analyses require penetration to the technical core by interacting with vendor design, programming, and quality staff. The CDR team strives to form a clear picture of the technical core and its relation to other aspects of the project.

This is the crucial step in the review, where the CDR team should be prepared to work together to keep discussions on track. Team members with technical expertise should dig deep enough into internal mechanisms to get a clear understanding of potential (possibly silent) failure modes. On the other hand, team members should also guard against wasting time on technical detail that is not relevant to the project issues.

3.8.3.4 Risk Analysis

The final phase of the on-site CDR is the risk analysis. To optimize the coverage, both a qualitative fault tree and a qualitative failure modes analysis are performed. The qualitative fault tree will normally include predefined faults identified with input from the project team, and additional faults that the CDR team may add. The qualitative failure modes analysis postulates: failures of hardware modules, unintended software behavior, human errors, and field device failures. The failure modes are analyzed for unacceptable system behavior.

3.8.3.5 Identification of Pending Issues

Issues that have been identified but not resolved during the on-site review need to be listed, and their significance needs to be evaluated. A resolution approach needs to be defined (preferably in agreement with the vendor) for each pending issue deemed important.

3.8.4 Issuing the CDR Report

The conclusions of the CDR team should be formalized in a CDR report. This report should in particular:

- Clearly state the scope and limits of the CDR.
- Summarize the findings and conclusion of the CDR team.
- Describe the bases justifying these findings and conclusion.
- List the pending issues that had not been resolved to full satisfaction at the time of the editing of the report.
- Provide the project team with recommendations regarding the pending issues, the post-CDR analysis, and the use of the system.

3.8.5 Post-CDR Analysis

Once the CDR report is complete, it should be turned over to the project team for final disposition. The project team should revisit the issues on a regular basis to ensure the identified issues are resolved, and to capture any new issues that may adversely affect the project. For some projects, the CDR team may be asked to review designs as they evolve.

3.9 Properties to be Evaluated by a CDR

As mentioned in the Introduction, the first objective of a CDR is to determine the potential of the system under investigation to produce unforeseen events and unacceptable behaviors. To this end, it should examine a number of essential properties:

- **Identification**: Are the critical system and its main components precisely and uniquely identified?
- **Functional Characterization**: Are the characteristics of the critical system clearly, thoroughly and unambiguously described?
- **Functional Adequacy** and **availability**: Are these characteristics appropriate for the real needs and operational conditions of the system? Should faults or failures occur, will nominal service be interrupted, and for how long?
- **Digital reliability:** In these operational conditions, will the actual system conform to its stated characteristics?
- **Robustness** and **safety**: Will abnormal situations, which may be due to causes external or internal to the system, result in acceptable system behavior?
- **Testability**: Can the defaults introduced in the system during manufacturing or operation be revealed with good chance of success and with reasonable effort?
- **Maintainability**: Can the above properties be maintained throughout the system's operational lifetime, in particular when the system is modified?

In evaluating these properties, the CDR team should consider that there are likely to be tradeoffs between the properties, and that judgment will be necessary, especially if commercial-off-the-shelf (COTS) technology is considered.

3.9.1 Identification

Identification provides assurance that:

- The system under investigation and its main hardware and software components are identified unambiguously.
- The system-related documents examined in the course of the CDR are unambiguously and correctly associated with this particular system or its main components.
- Any assumptions that the CDR needs to take into consideration regarding the use and configuration of the system or its main components are stated clearly.
- Any field information examined in the course of the CDR (e.g., failure reports, use conditions, or operating times) is clearly related to this system or its main components.

The basis to assess identification usually includes hardware and software architecture documentation, version and configuration management, processes for tracking operating history.

3.9.2 Functional Characterization

Functional characterization provides assurance that the system characteristics necessary for correct verification and validation, configuration, operation, maintenance and modification are specified clearly, thoroughly, and unambiguously. In particular, these characteristics include functionality, interfaces, configuration capabilities, configuration constraints, performance characteristics, limitations, failure modes, on-site installation, and support for operation and maintenance.

Appropriate functional characterization for complex systems usually requires much more than just commercial documentation, product data sheets and most user manuals, as these “standard” documents do not usually specify many important characteristics, or do not specify them in sufficient detail. Often, focused investigations are necessary to obtain the appropriate level of precision.

3.9.3 Functional Adequacy

Functional adequacy is the capability of the stated characteristics to address the services and performance expected of the system satisfactorily, considering the constraints resulting from the system’s *operational context*. In the case of a generic CDR, the operational context is an “envelope” covering the intended applications. The operational context includes the plant process under control, interactions with connected systems and equipment, interactions with personnel, on-site installation and configuration, commissioning, and user interfaces for operation and maintenance. The description of the operational context should address the different operational modes of each entity interacting with the system, as some of these modes and combinations of modes may require specific system behaviors.

Functional adequacy also considers avoiding features that would create unnecessary and excessive complexity in the design and in the operation of the system. However, the CDR team should consider that removing features from an existing system may introduce new problems, and may generate issues with long-term support from the vendor. It is also possible that other desirable features are only available on the more complex system, which leaves the CDR team to decide whether to accept more complexity or lose desired features.

To assess functional adequacy, the CDR team usually relate the characteristics of the system on the one hand, and the project perspective and the application requirements on the other hand.

3.9.4 Digital Reliability and Availability

Reliability is the capability of the system to provide the expected services for a specified time under stated conditions. Digital reliability is the part of reliability relating to digital failures.

Unfortunately, for realistic software and digital designs, it is usually impossible to demonstrate perfect digital reliability. Thus, one important objective of a CDR is to gain reasonable assurance of high digital reliability, focusing mainly on characteristics like defensive measures (including testability and verifiability of design and software), development and V&V processes, operating history (in the case of pre-developed systems or platforms already in use for other applications), complementary tests, and measures constraining the operational conditions.

Another important objective of a CDR is to gain reasonable assurance of availability, i.e., of the ability of the system to maintain nominal service even during maintenance, and to return to nominal service rapidly should a failure occur. Simple digital systems such as smart instrumentation will usually not offer the internal redundancies that would allow repair during operation, but could be used in overall architectures that tolerate their unavailability. More complex systems like control room systems may need much more time to initialize, restore/repair data that is dynamically modified during operation, and resynchronize with the other systems and equipment with which they are connected.

3.9.5 Robustness and Safety

Robustness is the ability of the system to provide the expected services and performance even in abnormal conditions, or to behave in a pre-defined, acceptable manner (graceful degradation) if the system can no longer maintain the expected services and performance. Abnormal conditions may be caused by failures or faults internal or external to the system (fault tolerance), or by conditions external to the item, including voluntary aggression (cyber security).

Robustness is not an absolute objective. An appropriate balance should be found between robustness and increased complexity, which could jeopardize system dependability. In practice, it is preferable to specify the robustness requirements explicitly.

Safety (in the context of this document) is the part of robustness that provides assurance that the failure modes most adverse to plant safety will not occur. It entails an evaluation of the vendor's design methods to ensure that the vendor has minimized the hazards in the system. Ideally, these hazards result in known failure mechanisms, which can be analyzed and incorporated in the design plans and licensing documents for the replacement system.

Many of the approaches and methods applied to systems important to safety are applicable as well to critical systems that are not important to safety. However, systems important to nuclear safety do have specificities that are discussed in Figure 3-7.

3.9.6 Testability

Testability is the ability of the system to be tested and/or monitored so that any defaults appearing in the system during manufacturing or operation can be revealed with good chance of success and with reasonable effort.

This property is particularly significant for systems operating *on demand*, i.e., for systems that remain in a stand-by mode most of the time, and whose behavior changes significantly in demand situations. Indeed, for such systems, the risk is that faults that appear during operation but are not activated in the stand-by mode remain unrevealed and accumulate during the long stand-by periods, and cause the system to fail when a demand situation occurs.

3.9.7 Maintainability of these Properties Throughout the Life of the System

Since most critical systems have a long operational lifetime, it is necessary to consider that in the course of this lifetime:

- The services and performance expected of the system, and its operational context, may change, which could affect functional adequacy, digital reliability, robustness and safety.
- The system or its configuration may be modified, which could affect identification, digital reliability, robustness and safety.
- The vendor is likely to produce new hardware designs to deal with component obsolescence, resulting in new software versions.
- The vendor is likely to revise the software to correct identified issues and provide enhancements.

The assessment of the maintainability of a system can be based on an evaluation of:

- The technical documentation of the system and its configuration.
- The modification procedures applicable to the system and its configuration.
- The procedures for tracking changes in the operational context.

There is little useful guidance from the U.S. NRC defining their expectations for an acceptable system safety process. The IEEE standard associated with software safety cannot be applied easily, as there are many aspects of nuclear systems design that are already part of an Appendix B compliant process. Certainly, process elements that already exist should not be duplicated by just adopting a process in complete compliance with IEEE Standard 1228. Instead, the vendor should look at the evaluations, analyses, and reviews that this IEEE Standard provides, and determine which of these evaluations, analyses, and reviews are missing from the existing nuclear programs.

It is also important to note that system safety supports nuclear safety concerns. However, system safety concerns are not bounded by nuclear safety concerns. Nuclear safety concerns are important, and guide design and development activities. However, there are aspects of safe system designs that do not follow directly from nuclear safety concerns. For example, it would be difficult to argue that the safety impacts from using a multi-tasking operating system can be directly attributed to any single nuclear safety requirement. Indirectly, the rod cladding might be better protected by not using a multi-tasking operating system in a safety-related reactor protection system, but there is no direct link between challenging the rod cladding and the multi-tasking operating system.

For a preliminary hazards analysis, nuclear safety concerns are certainly one of the elements to consider. The CDR team, now working as a system safety analyst, should add consideration of the elements of diversity, or where else have similar pieces of equipment been used in the plant, and what would happen if all of them failed simultaneously or close in time. The CDR team should also add consideration of the elements of defense-in-depth, especially for safety systems, which considers other means, using non-safety related equipment, to place and keep the plant in a safe shutdown condition.

Dr. Nancy Leveson noted in a private electronic mail:

> *"The founders of System Safety back in the 1950s were trying to say something very different. C. O. Miller, one of those people, wrote that one cannot understand the difference between safety and reliability without understanding the difference between a hazard and a failure. Reliability engineering does not differentiate. Reliability tries to prevent failures. System Safety tries to prevent hazards. Hazards are not necessarily the result of failures and failures do not necessarily lead to hazards."*

She also stated in a separate electronic mail:

> *"A system safety approach would be to take the system hazard analysis, translate it into safety requirements constraints on the component(s), and then ensure that the component(s) satisfies those safety requirements and constraints directly."*

In some circumstances, a system that is more available can be less safe than a system that fails to a fail-safe state. It is possible to increase reliability and decrease safety. Dr. Leveson noted in another electronic mail:

> *"As an example, a very reliable software component to ensure aspects of aircraft separation (such as a traffic alert and collision avoidance system) may be designed in a way that it induces pilot error in carrying out the aircraft maneuvers necessary to actually prevent [a collision]. In that case, the safety-function component is very reliable but not very safe."*

Figure 3-7
Specificities of Digital Systems Important to Nuclear Safety

4
CRITICAL DIGITAL REVIEWS – PRACTICAL HANDBOOK

4.1 CDR Outline

The practical guideline suggested in this chapter is organized into ten main topics. These topics are presented in the most likely chronological order. They are listed in Figure 4-1, together with an indication of the stages of the CDR process where the corresponding activities are the most likely to be performed.

Topic	CDR Process Stage
Identification of critical attributes	Screening, Off-Site Preparation, On-Site Reviews
Functional review	Screening, Off-Site Preparation
System orientation	Screening, Off-Site Preparation
Process orientation	Screening, Off-Site Preparation
Thread analyses	On-Site Reviews
Staff capabilities and attitude	On-Site Reviews
Risk analysis	Off-Site Preparation, On-Site Reviews
Operating history review	Off-Site Preparation, On-Site Reviews
Resolution of pending issues	Issuing the CDR report
CDR report	Issuing the CDR report

Figure 4-1
Main CDR Topics

4.2 Identification of Critical Attributes

The essential properties addressed in Section 3.9 are high level properties. Their assessment is usually based on the evaluation of sub-properties that are easier to analyze and evaluate. These sub-properties are called *critical attributes*. This section presents a brief overview of critical attributes of interest for a CDR. For commercial grade equipment to be used in safety-related systems, the critical attributes will include the *critical characteristics* (as defined by 10 CFR 21, see Reference [15]) that pertain to digital issues. The reader may find additional information on critical characteristics in EPRI reports NP-5652 Guidelines NP-5652 (see Reference [2]), TR-102260 (see Reference [4]), TR-106439 (see Reference [5]), and TR-107330 (see Reference [6]).

4.2.1 Physical Critical Attributes

Although many physical critical attributes are out of the scope of CDRs, the CDR team should consider the following types of critical attributes:

- The vendor’s model identification approach and the vendor’s processes for identifying new revisions to hardware and software. The CDR team should verify that the vendor changes the model identification when anything in the hardware or software changes.
- Power and power quality requirements, as they might affect the behavior of the system.
- Analog and discrete inputs and outputs that the (digital) system provides. The critical characteristics could include the analog-to-digital and digital-to-analog conversions, filtering, and any associated capabilities or restrictions that are appropriate. This should include consideration of the input impedance when power is on and when power is off.
- The electronic or mechanical constraints placed on the system’s output by the system’s design.
- Some systems provide data communications with other equipment. For these systems, the CDR team should consider interfaces and separation between redundant channels, critical to non-critical isolation, detection of failed links and failed equipment, and a large number of other issues associated with data communications.
- Many systems provide a human-system interface. For these systems, the CDR team should consider the human factors associated with the system, and make appropriate decisions about the interface. The authors of this handbook note that human factors should consider not only the control room operator, but also the maintainer and the technician performing surveillance testing.

4.2.2 Performance Critical Attributes

To address system performance concerns, the CDR team should consider the following types of critical attributes:

- How the system processes each input.
- The specific functions and algorithms used in the system.
- The data presented on the human-system interface, and how the user interacts with the system.
- The on-line and off-line test and diagnostic capabilities of the system. Diagnostics are necessary in most systems, but the evaluation should consider the impacts of those diagnostics on operations.
- The methods that the Operations, Maintenance, and Engineering staff will use to determine when the (digital) system or its plant system are failed, and when they are operable. These should not include statements like “it will be intuitively obvious to the Operator when this system has failed,” since it is not obvious when some systems fail.
- Any features that are provided for surveillance testing or calibration.

- The response time, response time variability, and throughput for the system.
- Issues associated with various system attributes including accuracy, variable ranges, rounding, numerical errors, hysteresis, and stability.

4.2.3 Environmental Critical Attributes

To address environmental concerns, the CDR team should start with the following types of critical attributes and considerations:

- The range of temperatures at which the system functions, the range of temperatures expected in the plant, and any effects on system operation from changes in temperature.
- The relative humidity range to which the system will be exposed and any effects on system operation from changes in humidity.
- The radiation exposure for the system, and any life limits or requirements for large radiation doses for the system. Most current digital electronics tested recently survives more than 1000 Rads of integrated gamma dose to the silicon before significant system performance degradation occurs.
- The system's response to seismic activity.
- Susceptibility to and emission of electromagnetic interference.
- Testing the system for susceptibility to electrostatic discharge (ESD) events, or, as many electrical engineers are recommending, restricting the end user and require that appropriate ESD protection methods be employed when transporting, installing, storing or working on the system.
- Evaluate the heat released by the system and any needs for ventilation, cooling, or heating.

4.2.4 Dependability or Integrity Critical Attributes

The critical attributes described in the previous sections are measurable. The critical attributes described below are not; they relate to dependability or integrity concerns. Here, the CDR team is concerned with evaluating safe, reliable, available, maintainable, and inspectable (or testable) operation. Since these concerns are very difficult to quantify, the CDR usually resorts to qualitative evaluations (good/bad, acceptable/unacceptable, etc.).

Figure 4-2 provides general guidance on the types of characteristics to be used for evaluating dependability and integrity. These attributes are based on many aspects of good engineering processes and practices. Attributes can include design and development processes; use of unsophisticated methods in early system development; the vendor's evaluations; commercial quality assurance practices; and defensive measures designed into the system, including fault avoidance and removal, well-identified and limited *influence factors* (i.e., factors that can influence the digital behavior of the system), fault tolerance, and diagnostic capabilities. Considerations should be included for the vendor's configuration control capabilities; change control practices; and problem tracking, resolution, and reporting practices. Much use can be made of documented and appropriate operating history as compensation for missing process and design elements.

Critical Attributes	Acceptance Criteria
Built-in quality through structured and controlled quality assurance processes for: design, testing, manufacturing, and error tracking	QA Program complying with recognized standards and addressing key elements of organization, design control and verification, document control, and record keeping. QA Program certifications based on external audits. Evidence that the QA Program was applied in the development of the system.
Built-in quality through application of a suitable digital system/software development process	A digital system/software development process is followed which includes: • A software life cycle development model consistent with industry standards • Documented design requirements • Requirements traceability • Documented V&V plans • Validation tests and documented reports Evidence that the development process was applied to the system.
Fault and failure management and diagnostics	Continuous built-in self-testing is provided that will detect memory failures, internal communication failures, power supply failures, etc. Alarms or other indications of failure are provided. Potential failure modes, particularly those that would affect safety functions, have been identified and adequately addressed.
Configuration control and traceability of software, firmware, and errors	A formal configuration control program is in place which includes: • Documented procedures • Baseline maintenance • Change management • Firmware control during manufacture and servicing • Control of development tools • Error reporting
Problem reporting	Vendor has established error reporting procedures that provide comprehensive tracking of problems. Vendor has demonstrated error resolution process.
Reliability and Dependability	Documented systems operating history shows system stability, reliability, and freedom from critical software errors.

Figure 4-2
Examples of Digital Equipment Dependability Critical Attributes

The following groups of critical attributes are a reasonable place to start, including:

- Response to abnormal conditions and events, including elements such as hardware failures, operating system faults, incorrect inputs, unusual plant conditions, unusual states of connected systems and equipment, unusual human actions, corruption of internal memory, and single event upsets.
- Use of the failure analysis results during the design and development of the system.
- An evaluation of the state the outputs achieve on failure, which could be fail-safe to a specified state, or fail-as-is. The authors of this handbook note that fail-as-is is rarely an acceptable state for the human-system interface, as it is difficult for an operator to recognize the failure immediately. We also note that some element of the human-system interface should be animated by the main microprocessor or microcontroller, and not by the video interface controller for the display, since the video interface controller may continue to operate after the primary microcontroller stops updating variables, which allows the human-system interface to appear as if it is working, when it is not.
- Evaluate the inherent, built-in quality in the vendor's design, including the use of design and development processes, defensive measures, documentation, safety analyses, peer review, test, and project management to enhance quality and reduce risk.
- Evaluate the quality of the vendor's manufacturing operation, or, if out-sourced, how the vendor manages their out-sourced manufacturer.
- Evaluate fault and failure management in the system. The CDR team should also evaluate the likely plant response to such faults and failures.
- Evaluate the human-system interface as a source of failures for the various classes of users, including operators, engineers, maintainers, and surveillance testers.
- Evaluate the vendor's configuration management and change control capabilities.
- Evaluate the equipment's support for configuration management and change control when installed at the utility.
- Evaluate the vendor's problem reporting, tracking, root cause evaluation, resolution, and problem reporting mechanisms. Either these work well, or there should be some compensation. For safety applications, the compensation should support the dedicator's 10 CFR Part 21 error reporting mechanism (see Reference [15]).

When the list of critical characteristics is complete and captured in a peer reviewed document, the CDR team can then decide how each critical attribute will be verified and validated to ensure that the equipment meets the requirements. The list should also be annotated for those aspects that can be verified and validated once, as opposed to those that have to be validated for each component or batch of components. In sorting these lists, the CDR team should consider that many of the physical tests are destructive, and that the sample cannot be supplied as safety related equipment.

During development of the EPRI document that introduced the notion of CDR (EPRI Guideline TR-106439, see Reference [5], and TR-107339, see Reference [7]), the CDR process developers asked their NRC Engineering Inspector, John Calvert (then at NRC headquarters), to summarize the questions he used as an Engineering Inspector. These are the key issues he used when evaluating software based systems.

Seven Easy Questions

1. Is there a design specification(s) detailing architecture? Inputs? Processing? Timing? Outputs?
2. What standards or procedures were used for software design? V&V? Test?
3. What is the top level structure of the software and how is it documented?
4. a) How is the application translated into machine executable instructions? How is assurance gained to confirm that the translation was correct (i.e., performs to specification)?

   b) How is assurance gained to confirm the correct application is actually installed in the machine?

   c) If an external configuration device is used (e.g., personal computer or PC):

   i) Where is the application translated into machine executable instructions? The configuration device? The system?

   ii) What is actually transferred from the configuration device to the system? Source code? Executable image?
5. What indications are available to determine when the hardware is not working? The software is not working?
6. How many problems and/or changes have occurred since the system was installed?
7. Have transient effects such as power spikes, induced signals in inputs/outputs, and radio waves (EMI/RFI or EMC) been considered in the design and test of the equipment?

Three Hard Questions

1. What was the process that was used to minimize the probability of incorrect translation of the system basis to hardware/software requirements.
2. a) What are the possible:

   i) Hardware errors

   ii) Software errors

   iii) System interface errors

   iv) Human-System Interface errors

   that would prevent the system from performing correctly?

   b) Are these errors detected?

   c) Are these errors processed or indicated so that action can be taken?
3. a) What are the possible undesired events that would influence the system:

   i) Hardware

   ii) Software

   iii) System Interfaces

   iv) Human-System Interface

   in such a way as to cause incorrect and/or unsafe operation?

   b) Are there defenses against these hazards?

   c) Are the defenses effective?

Figure 4-3
Questions to Consider when Performing a CDR

4.3 Functional Review

The end goal of the Functional Review is to determine if the system being examined is correctly identified and characterized, and if it is adequate for the intended application(s).

4.3.1 System/Model/Version Identification

The CDR team should evaluate the rigor of the vendor's model/version identification method and process. This evaluation should consider the following topics:

- Unambiguousness of versions identification. Ideally, any modifications in components that might affect the functioning and behavior of the digital system (either by design changes, component changes, or manufacturing process changes) should result in a different system version and a different identification.
- Differences between system versions. This might be of particular interest in particular for the Operating History Review.

4.3.2 Functional Characterization

The CDR team should evaluate the quality of the system description and characterization. This evaluation should consider attributes including:

- Functionality for plant system operation, possibly depending on plant state.
- Performance under various possible influencing conditions (e.g., system configuration; operational modes, including degraded modes; excessive communication message traffic).
- Interfaces to other systems and equipment, in normal and abnormal conditions.
- Human-system interfaces, considering not only the system operator, but also the maintainer and the technicians performing system configuration or system testing.
- System testing.
- Customization, including how the system is configured or programmed and the kinds of parameters involved in the configuration or programming.
- Installation, operation, and maintenance constraints.
- Any applicable limitations.
- Any new support infrastructure required at the plant, including training and the Simulator.
- System failure modes and failure management, including the state of each input and output when powered on and powered off.
- Cyber security and the protection of the system and its configuration from malicious or inadvertent modifications.
- The potential risks from undocumented features.
- The potential risks for vendor future support of this system.
- The potential risks of and compensatory measures for system obsolescence.

4.3.3 Functional Adequacy

The evaluation of the adequacy of the system to the intended application(s) is a summary activity based on engineering judgment. The CDR team needs to understand the place of the system being evaluated in the plant, as well as the way the system will interface with other systems and equipment, and the way the system and its associated plant system will be operated and maintained.

The CDR team should consider whether the system being evaluated is intended for a specific application or for generic application in nuclear power plants. For an application-specific CDR, there are identified operational conditions, failure mechanisms and requirements associated with the plant system. For a generic CDR with no specific associated plant system, these are more difficult to determine and have to reflect all issues presented by the intended applications.

The CDR team should also determine how the plant system requirements, normal and failure modes, and operation and maintenance (O&M) could affect the digital system and its design and failure modes, including the way the system is configured, used and maintained by operators, technicians, and engineers. In the case of a safety critical system, the CDR team should also determine the nuclear plant safety requirements for this system. Among the issues are things like fail-safe operation to a defined state, or fail-as-is. For a generic CDR, the CDR team needs to consider the plant applications where this system is intended to be used, and generate a set of generic failure modes for the review.

The CDR team needs to consider the types and capabilities of each of the inputs and outputs against the plant interface requirements. The CDR team needs to define the state of these inputs and outputs in various failure conditions.

The CDR team should consider any data communications requirements that exist for the system, and determine how the utility might have to change the existing plant equipments to interface with the system. As an example, consider a smart transmitter, with Highway Addressable Remote Transducer (HART®) communication signal added on to the existing 4-20 mA current loops, interfaced with traditional process control and indication equipment. The utility engineering staff would need to evaluate all devices on the loop with the smart transmitter, since the HART® communication signal perturbs the 4-20 mA signal.

The CDR team needs to evaluate the design to determine if it is likely to be as reliable, available, maintainable, and inspectable (or testable) as needed for the plant application.

The CDR team should consider the system's preventive maintenance requirements, life limited parts, and plant requirements for surveillance testing, and calibration, to determine if changes are required in the plant's programs to accommodate this new system. The CDR team should also evaluate needs for technician, engineer, and operator training. The CDR team should consider how this equipment will affect the plant Simulator, and whether new or changed operator training is required.

For purposes of the review, the CDR team should compare the system capabilities against the plant needs. Complexity is not necessarily something to avoid, but digital systems with capabilities that far exceed the needs of the plant system may result in less reliability or safety than desired. However, simpler systems may not provide the diagnostics and self-test capabilities needed for failure detection. Judging whether the system is appropriate for the plant needs, and neither too complex nor too simplistic, is part of the CDR process.

4.4 System Orientation

The end goal of System Orientation is to determine if, in the operational context determined by the project, the overall design of the system being examined provides appropriate guarantee of digital reliability, robustness, safety, testability and maintainability. To this end, the CDR team needs to understand the hardware and software architecture of the system being evaluated, as well as the way the system will interface with other systems and equipment.

4.4.1 Hardware Architecture

Describing the hardware architecture will usually entail the use of system drawings showing the nature, identity and role of the main electronic components, such as microprocessors or microcontrollers, memory (including size and type), application specific components (ASICs or FPGAs), interface and communication circuits, watchdogs, clocks, and any other complex electronic circuits. The CDR team should familiarize themselves with the vendor drawings. If formal drawings are not offered, the CDR team should request them. The availability, the quality, and the formality of drawings should be noted.

This is also an opportunity to begin the technical penetration of the vendor design. The CDR team should critically examine the architecture, identify the components that are key to the correct and robust functioning of the system or that could be at the origin of unanticipated situations (e.g., external interrupts), and begin to consider possible failure scenarios. The CDR team should also examine the configurability of the hardware architecture, i.e., the possibility of defining simpler architectures that confer a higher degree of confidence. The team should avoid detailed discussions of specific failure scenarios at this point, but should note issues that may need to be pursued during the failure investigation. Figure 4-4 provides a list of suggested questions. These questions should *not be used as a checklist*, and are provided solely to provoke thought by the CDR team.

Detailed questioning at this point will also ensure that the vendor has assembled the proper level of technical staff to answer these questions. When questions cannot be answered by the personnel available, the vendor should be requested to take the question as an action item, and a list of action items should be maintained. These should be resolved before the on-site review is complete, or appropriate commitments made to resolve the issues on a defined schedule.

Platform/Application
PLC platform? Distributed control platform? Other module-configuration platform? Some vendors create their own platform to use as a base for a variety of applications. Some suppliers of radiation monitoring systems, for example, take this approach.

Hardware Complexity
System complexity directly impacts the level of the review effort. What make/model is the main processor? They range from simple to very complex. Does the design use redundant components for fault tolerance?

Is functionality distributed over remote and local units? Do multiple processors share memory?

Do boards other than the main CPU board have processors? If so, what make/models are used? If there are multiple CPU's, how do they communicate - over networks, serial data links, or through shared memory?

Does the hardware use Application Specific Integrated Circuit (ASIC) or Field Programmable Gate Array (FPGA) technologies? If so, what are the functions? Who developed the logic in the device? Using what process? How are these components integrated with the rest of the system? Do the designers understand all the boundary conditions, and were the boundary conditions tested?

Which boards and circuits are required, and which are optional? Which boards are to be used in this project?

Memory
How much memory is associated with each processor? Does that exceed the addressable space of the processor (this could be a sign that complex dynamic program overlays are used)? What types of memory (RAM, ROM, EPROM, EEPROM) are associated with the processor?

Watchdog Timers
Is there a watchdog timer? If so, is the watchdog timer a separate device or part of the CPU? What circuits respond to the watchdog timeout signal and how do they respond? Is complete system operation identified by the watchdog? Does timeout cause the equivalent of a power-up reset?

External Interfaces
What are the visible components and/or indications in the control room? What points and support are provided for surveillance? What transducers does the hardware manipulate for control? What sensors are used to generate analog/digital inputs? What signals are generated as system outputs? Are there external clocks? Are there external interrupts? In particular, are there plant process related interrupts? What measures are taken to provide electrical isolation with non-critical, non-qualified equipment, or between redundancies?

Processor/Auxiliary Board(s)
Is the memory error correcting or error detecting? Do other boards have processors? How do they communicate? Are there separate I/O processors? Do they control trouble/alert/alarm contacts? Does the design use redundant components for fault tolerance? How is the overall complexity of the system affected by such redundancy?

What is the modular organization of the boards and circuits? What external switches and contacts generate inputs to the software? Is the hardware designed for program loading as a manufacturing step? Is the system designed to save parameters and data on power dips and/or power failure? Which boards and circuits are required, and which are optional?

Figure 4-4
Hardware Architecture Questions

Analog to Digital (A/D) and Digital to Analog (D/A) Converters
Are there individual A/D and or D/A converters for each input and output, or does one A/D or D/A serve multiple channels? How many channels are there? If one converter serves multiple channels, how many does it serve? Are sample and hold circuits used? Does the system support A/D or D/A converter functional or calibration checks?

EMC
Has the design minimized potential concerns with compatibility with the electromagnetic environment? Has the system been tested against EMI/RFI standards? If a digital system is not adequately protected against interference of this type, unforeseen and unpredictable behavior is possible.

Availability
What are the availability requirements for this system? These will be determined primarily by the design of the plant systems, including both I&C and electromechanical equipment. Is the system designed to be maintained during operation? (For example, with an appropriate redundant architecture.)

System Level Requirements
Is the proposed hardware is likely to meet system level requirements? This should include environmental concerns.

Figure 4-4
Hardware Architecture Questions (Continued)

4.4.2 Software Architecture

Describing software architecture is generally more problematic for vendors than describing hardware architecture. The software architecture is typically not available as drawings in documentation. The digital specialist on the CDR team should ask the developers to sketch the major software components on a white board if no documentation exists, or if the documentation is not sufficient for a critical examination.

An impromptu sketch will often reveal more about the software architecture than a ten page write up. The CDR team should capture any sketches or white board drawings. Where possible, the software architecture should be left on the board for reference during the remainder of the review.

Figure 4-5 provides a list of suggested questions. These questions should *not be used simply as a checklist,* and are provided primarily to provoke thought by the CDR team.

4.4.3 Data Flow

The CDR team should understand and analyze the flow of information to and from external interfaces, and through hardware and software. The team should consider such issues as initialization; default values; channel latency; power on, off, and rapid power cycling behavior; network disconnect and reconnect behavior; and the handling of unanticipated or undefined data.

Platform/Application
Commercial real-time operating system? Embedded real-time kernel/executive? Is program development in Assembly, C, C++, Basic, or Fortran required as part of this project?

Are custom “blocks” or “modules” required as part of the development in this project? It is common for distributed computer control vendors to offer programmable modules, which can be programmed in a simple language similar to Basic. These can be programmed and then “connected” to conventional blocks.

Software Lineage
What does the family tree look like for this architecture, and where does this variation fit? Can the vendor give a clear depiction of the architecture, separate from the details? Can a clear picture be presented on the use of global variables? Can core software for the base architecture be “configured” for a wide variety of situations?

Language, Compilers, Tools
What language, or languages, is the source code written in? What tools were used? Are tools and compilers supported by the vendor (e.g., is the latest version being used, or does the vendor use only a trusted version)?

Task Structures
How is the software partitioned? How does data flow through the system?
Is there any critical timing? If so, how is the timing ensured? What happens if the timing is off?
Do portions of the software behave as a state machine? Are there state diagrams?
Are distributed state-machine portions required to stay in sync? What methods are used?
Are there continuous control loops executed by the software?
To what extent were object oriented and/or structured programming methods used?
What real-time engineering calculations are required in the application?
Are value changes in global variables localized in a clear manner?
Are there re-entrant tasks? If so, what considerations have been given to stack overflow?

Operating System
Does the use of an operating system? If so, which one?
Is it a vendor’s operating system or a commercial operating system? If a commercial operating system is used, what is the depth of understanding of the operating system and the interfaces between the vendor’s code and the operating system? Did the vendor’s staff review the commercial operating system source? Does it have the source files for the operating system? Has it determined which pieces of the operating system are needed for this system?
Are there any restrictions on that operating system from the application or the hazards analysis?
Does the operating system use stacks (if the answer is no, then the chances are they don’t understand the operating system)? If so, what analysis or provisions have been made to ensure that stack overflow does not occur?
Is the system multi-tasking? Does the system use time-slicing? Priority scheduling?

Synchronization
What synchronization mechanisms and schemes are used in the case of multitasking or distributed processing? What are the mechanisms preventing incorrect access to, and use of, shared resources?

Memory Management
Can a memory cell contain different types of data or is it statically allocated to a specific piece of information?
Is memory allocated dynamically? If so, what prevents the system from running short on memory? What prevents memory from being incorrectly accessed and used? What mechanisms are used to verify on-line that memory is not corrupted?

Figure 4-5
Software Architecture Questions

Availability
How long will it take for the software to return to nominal functioning after a failure and the system is repaired or restarted? In particular, how long will it take to load and initialize? How long will it take to restore data and configuration? How long will it take to fully resynchronize the system with the other systems and equipment it is connected to?

Watchdog Timers
Does the system use of watchdog timers? If so, are they implemented in hardware or software? If software, what happens if a processor quits? If hardware, where is the timer reset in the software? Is the watchdog reset in more than one location? Does the watchdog all software execution, or just verify starting software execution? Can a watchdog failure be detected externally? Is the watchdog system testable?

Use of Interrupts
Are interrupts used? Are the interrupts maskable? If so, are they ever masked?

Communication Protocols
Does the system use or communicate with distributed processors? Is the communication protocol token passing? If so, what happens when a token is dropped? Are distributed devices polled? If so, what happens if they do not respond? Is the protocol Ethernet? If so, what is the limitation on the number of distributed devices? Are broadcast messages allowed? If so, what would happen during a broadcast storm (i.e., a processor gets stuck in a loop sending out broadcast messages)? Are devices interrupted to service communication requests from other processors?

Defensive Measures
Had there been an attempt to systematically identify the possible types of digital faults? What measures have been taken to avoid or eliminate the different types of faults?
Had there been an attempt to systematically identify the factors that could influence the digital functioning of the system? What measures have been taken to provide reasonable assurance that these factors will not activate a postulated digital fault?
What measures have been taken to reduce the significance of the failure modes that cannot be precluded by the previous measures?

Software Complexity
What is the size of the software program (in lines of code and in bytes)? What measures have been taken to facilitate testability, verifiability and modifiability of the software program?

Figure 4-5
Software Architecture Questions (Continued)

4.4.4 Diagnostics

The CDR team should understand and analyze the internal failures that could affect the system. Some of the topics to be considered are:

- Software and internal processing functions.
- Multi-tasking operation.
- Inputs and outputs.
- Communication links, both internal and external.
- Watchdog timers in both hardware and software.

- Power supply, including rapid, cyclic changes in the power supplied.
- Backup batteries.
- Calibration.

The CDR team should identify the effect on the state of the system outputs and on the ability of the system to operate given the failure. The reviewers should also identify the following failure categories:

- Any single points of failure.
- Those failures that will be detected by the on-line diagnostics.
- Failures that can only be detected by surveillance testing.

The CDR team should understand and analyze how internal failures are detected and how the user is informed. Possible means are diagnostics, self-test, manual test, and calibration capabilities of the system. While diagnostics increase the complexity of software, and possibly of the hardware as well, having the system find its own faults and failures does decrease the undetected or silent failure. Reasonable diagnostics and self-test capabilities can find certain faults and failures in the system rapidly, but rarely provide 100% coverage for all faults and failures. The CDR team should ask how the vendor determined coverage, and how the vendor verified and validated the correct function of the diagnostics and self-tests. The CDR team should establish the broad classes of faults and failures that are self-detected, and make recommendations about additional manual testing. It is likely that the equipment will provide greater self-diagnostic coverage, with more fault and failure detection than the existing equipment at the utility. The CDR team should evaluate any analog inputs and outputs to determine if calibration is required, and if the system checks the analog inputs and outputs for accuracy.

4.4.5 Configuration Security

The tools that the manufacturer and/or utility use to configure the system can have a significant effect on its critical function and should be fully described and analyzed. Configuration, as used here, describes general customization as well as calibration, surveillance, and troubleshooting. The CDR team should evaluate the ability of the configuration mechanisms, if misused or malfunctioning, to improperly configure the system.

A separate issue the CDR team should consider is configuration security, and any protection supplied to prevent unintended or unauthorized configuration changes. The team should evaluate the configuration procedures, and clearly recommend what training and procedures would be helpful to minimize human errors. If no procedures exist, the CDR team should determine whether the interface is sufficiently complex or used so infrequently that they should recommend generating procedures designed to minimize risk of personnel error.

For some systems, actually using the protection features results in more risk than the protection removes. As an example, the protection jumpers on some field mounted smart instruments require the technician to go to the instrument, open the case and remove the jumper prior to configuration. Another trip is needed to open the case and replace the jumper after testing is complete, followed by an additional verification that the instrument is still working after the jumper is re-installed.

Since cyber security is a very important topic and since the state-of-the-art is changing so rapidly, this report provides no guidance for reviewing for cyber security concerns. The CDR team should consider the guidance available for cyber security at the time of the review.

4.4.6 Known Faults and Failures

The CDR team should evaluate any known hardware and software faults and failures, assessing also how the vendor resolved or addressed each. The CDR team should evaluate if these faults affect the system functionality, and if the vendor has addressed these faults and failures in the user documentation. The CDR team should look for trends, forming a pattern of failures that identifies system weaknesses, either in the system or in vendor programs and practices.

The CDR team should also look at the history of the system, and analyze the faults history and the trends in failure rates. The CDR team should also assess how the vendor reacted to these faults (e.g., what measures did the vendor take to detect similar existing faults or existing faults that resulted from the same cause, and to prevent similar faults from occurring in the future?).

The CDR team should also determine whether the vendor keeps track of the faults and failures of the components used in the system, when these faults and failures are revealed in other systems or by other users of these components.

4.4.7 Common Cause Failure

The CDR team should evaluate the design features and processes the vendor used to reduce the risk of common cause failure. The CDR team should recognize that the vendor can do little to evaluate common cause failure in the plant. However, the vendor can provide a high quality system with appropriate designed-in defensive measures, which will be less likely to fail in this manner. EPRI Guideline TR-1002835 (see Reference [10]) provides extensive information and recommendations regarding defense against CCF.

4.4.8 Development Tools

Software tools used in the development, testing, or manufacturing of the system architecture and of the software or firmware components can affect the system's integrity. Therefore, the CDR team should consider all development tools, including compilers, linkers, third-party components, configuration management, source control, and test equipment, that were used during development of all hardware and software components. The providers of each tool should also be evaluated, as appropriate.

4.4.9 Human Errors

The human-system interface is often a major source of hazards. Some sources claim that poorly designed human-systems interfaces are the most prevalent failure source in nuclear power plants. From experience, it is reasonable to state that operators, technicians, and engineers are set up to fail by poor designs, unreadable or incorrect documentation, and incomplete software designs.

The CDR team should expect that the vendor will have considered the human-system interface during system design. The CDR team should not expect most non-nuclear vendors to understand or use human factors engineering principles. The CDR ream needs to evaluate the human-system interface briefly during the CDR, using at least some of the human factors engineering principles, to determine if the interface is acceptable, and at least moderately usable. The CDR team would then document their recommendations for corrective, or compensatory, actions at the vendor and the utility.

In particular, the CDR team should identify any human error that can contribute to the failure of the system. First, the team should identify the actions that can lead to human errors during operation. Some potential for introduction of human error exists during the following operations of the system:

- **Initial Configuration:** For example, the vendor may configure the system for interface with a specific set of input and output devices, and with specific application logic. Therefore, the CDR team should determine how the vendor controls and verifies the initial configuration. The CDR team should also verify if the end user could modify this configuration.
- **Status Monitoring:** The CDR team should look into whether a personal computer or laptop can be connected to the system to monitor the system status. The CDR team should consider how this capability (i.e., monitoring) is done. For example, does it require a specific password to access the system? Can the operator inadvertently change the system configuration in this mode? What effects in the system are likely to occur while status monitoring?
- **Configuration Editing:** The CDR team should evaluate how the configuration of the system can be modified, any protection provided to protect from faulty parameters, and any protection to reduce the possibility of unintended change. Several examples of configuration protections include the following: A password may be required to enter the configuration mode and make changes; Modifying, saving, and restoring the configuration of the controller may require a password; Configuration change safeguards may be included, such as parameter bounding prevents assigning values that would make the controller operate inaccurately; Changes to the configuration may be tracked and system-level support provided to remove modifications.
- **Manual Operation:** The CDR team should determine if the system has displays or panels that allow the users to set the system to manual. In addition, the CDR team should determine the capabilities of the system in manual mode. For example, can a user control each output individually or inadvertently leave an output in a forced manual state?

- **Setpoint and Calibration Modification:** The CDR team should determine if the system allows a user to modify setpoint values and calibration. The CDR team should also determine if the setpoint and calibration values can only be set within a specific range, prescribed during the initial controller configuration by the vendor. Then the CDR team should determine how these ranges (e.g., operational limits) were established. Finally, the CDR team should determine if appropriate administrative controls of activities performed during and after setpoint modifications and calibration will reduce the possibility of human errors, and if procedures should exist to specify the practices used by technicians during these activities.

4.4.10 Results

Having established a base understanding of the system, the CDR team is now in a better position to determine which areas are most critical. The CDR team should note any issues that have surfaced during the orientation, and should identify the most critical characteristics. These issues and characteristics should be the object of more focused thread analyses, usually during on-site reviews.

4.5 Process Orientation

4.5.1 Purpose

The objective of the Process Orientation is to evaluate the vendor's Quality Assurance program, processes, procedures, and practices for development, testing, complaint handling, failure reporting, and modification mechanisms. It is not a formal audit, but could be combined with one. The intent is to assess the quality of the system based on the process used to develop it. This provides the CDR team with a context in which to view the system. Based on this context, the CDR team can focus their attention to spend less time in areas where a vendor shows strength, and more time in areas of greater concern.

A vendor's processes and procedures are likely to change over time. The current processes and procedures may not reflect the process and procedures that were used during the system development. A vendor's current practice may reveal maturity, innovation, or lack of understanding. The vendor's poor documented practices may also not reflect the good engineering practices implemented by the vendor staff, because they know that good practices are the only way to generate good systems.

If a company is found to have no procedures, or inadequate procedures, the justification to use the system will rest solely upon the technical basis, with support from the operating history. The utility that buys a system from a vendor such as this should be prepared to compensate for the vendor's process deficiencies. This may entail the purchase of source code and development tools, or assisting the vendor in establishing proper controls.

However, a complete lack of standards and/or procedures, even for a commercial vendor, is a clear warning sign. A utility should proceed with the CDR only if: 1) there is a clear understanding that only this vendor can meet the utility needs and requirements, 2) the utility is willing to perform and document in-depth design reviews, code reviews, and testing, and 3) the utility is willing to maintain the complete system design basis.

Figure 4-6 provides other general considerations regarding the vendor's processes.

Usually, the vendor's investment in a good process directly relates to their capabilities and the quality of their systems. A vendor that does not invest in documentation and review usually tries to compensate by testing. The software industry accepts that quality cannot be tested into software. Testing can only confirm that quality has been built into the system. Quality has to be built in, using in good design practices and V&V.

That is not to say that retrofitting a software quality assurance process to an existing system has no value. However, "after the fact" documentation will not remove errors made early in the development process. Activities like code review have been demonstrated to be a method of improving the quality of an existing system. The authors of this handbook have retrofitted design documentation, reviews, and testing to equipment several times, and have always found design errors that the vendor corrected after the review. This process is just not as cost effective as doing the work correctly from the start.

The CDR team should expect to see some form of discrepancy reporting, tracking, and resolution process in place during design and development, as well as after the system has been shipped. During design and development, the vendor should use the discrepancy reporting process as a part of the peer review and testing processes, to ensure that the vendor retains and resolves comments appropriately. During design and development, the discrepancy reporting process is a key part of verification and validation. Depending on the vendor's process requirements, verification and validation may require all discrepancies identified in a particular phase to be resolved before entering the next development phase. Other vendor life cycles may hold discrepancies that affect work done in earlier phases, and recycle through the entire life cycle to resolve the discrepancies. No matter what the vendor process is, the CDR team expects to see all discrepancies resolved in an acceptable manner.

After the system is shipped, the CDR team expects to see a discrepancy resolution process that collects and retains all problems reported, performs root cause evaluations of significant problems, and tracks resolution of the problem to closure. For safety critical systems, this process will interface with the Part 21 reporting mechanism for the U.S. NRC and the nuclear utilities.

On several occasions, the authors of this handbook have evaluated vendors where an acceptable process existed, but where it was not applied to the development of the selected system. Those vendors implemented, or third parties implemented, an after-the-fact verification and validation process. In some cases, the significant digital faults were resolved, and compensatory measures were defined for the remainder. In another case, the key design feature for their evolutionary system was abandoned as unworkable after their verification and validation efforts.

In other cases, the authors of this handbook found no documented engineering procedures, but the individual engineers implemented good practices. For one of these, the quality of the overall system was superior to systems from an Appendix B vendor with a reasonably good set of procedures. For this vendor, the quality of the system depended totally on a single engineer. Their system was used in a critical, but non-safety, application in a nuclear power plant and no problems had occurred when the CDR was performed. The utility chose not to retrofit verification and validation to that system for this application. The utility certainly would not use that system in safety related service without retrofitted design documents, verification, and validation.

Figure 4-6
General Process Considerations

At other times, the authors of this handbook have found systems with long development histories that are well respected in the industry, widely used, and obviously highly reliable and safe. However, the early design and development activities were primitive, and did not meet the current vendor's, reviewer's, or regulatory expectations. However, there are various methods of compensating for the missing design documents. In some cases, the existing staff had to read through the code, documenting as they went, to support redevelopment to replace some obsolete integrated circuits. In other cases, the vendor was redeveloping the system and wanted to reuse the existing algorithms in the new system, thus requiring the staff to read through the code, document the existing design, and then redesign for the new system. In another, less successful case, the vendor had studied just the small part of the code that they wanted to modify. Later, they found themselves back in the code, resolving a timing race that had been a fixed once before.

It is possible to develop compensatory actions for many design or process issues. However, when the total compensatory actions required become onerous, the CDR team should stop the process, and abandon the attempt to use this system. If there are too many things wrong with the system, process, or vendor, it is likely less expensive and a lower risk to choose a different system, or even a different vendor.

It is very difficult to compensate for a vendor discrepancy resolution process with significant issues. As an example, one CDR revealed that the vendor did not record customer failure reports as problems until they had decided on a resolution. Further investigation revealed that this was driven by a corporate policy dictate that problems must be resolved within a few weeks. Unsolved problems could linger indefinitely without notification to customers.

Figure 4-6
General Process Considerations (Continued)

4.5.2 Method

The CDR team should request copies of any applicable procedures and/or standards prior to on-site visits. Reviewing procedures prior to a visit can save all parties valuable time.

While reviewing procedures, the CDR team should look for indications that may provide insight into the process. Items to note include:

- Approval dates.
- Revision numbers.
- Authors.
- Reviewers/approvers.

Approval dates should be compared against the system development dates. Procedures that were approved after a system was developed may simply indicate the codification of prior informal practices, may indicate a lack of prior control, or may indicate process improvement.

Revision numbers may indicate that a process has been refined through time, or indicate recent attempts to meet ISO standards.

The authors of various procedures may indicate how the organization works. Do the technical experts write the procedures they will live by, or does a QA organization dictate rules and regulations?

The reviewer and approval signatures, or lack of signatures, on a document can also reveal the level at which procedures are reviewed. Do the reviewers report to the author organizationally? Do the reviewers/approvers have an equivalent, or better, level of expertise as the author? Do the author, reviewers, and approver work for the same organization?

While no single item can be used to gauge an entire system, the total aggregate of items may. Again, this review is not an audit. While this process may provide a good starting point for a formal audit of procedures and practices, the CDR team's focus is the digital system.

Figure 4-7 provides a list of suggested areas for review. This should *not be used as a checklist,* but should be used as a starting point, to facilitate generation of questions appropriate to your review.

Corporate Culture and Organization
Company history and background.
Organizational charts show reporting chains. If a separate Quality Assurance organization exists, do they report to higher levels of management where they are not unduly influenced by the pressures of project schedules and budget?
The organizational charts should also be reviewed to determine the size and depth of the technical organization. Does the organization rely on single individuals for key functions (e.g., programming)? If so, can the organization perform qualified peer reviews?

Document Control and Distribution
If the organization has formal standards and procedures, how are they controlled? How does the technical staff access the documents? How is the staff notified when procedures are modified or created? Who is responsible for ensuring that all manuals are maintained up-to-date?

Development History
When did the vendor develop the technology for the system and its main components? How did the system and its main components evolve?

Development Process
Does the organization have a defined development process? Are deliverables from each project phase clearly defined? Are requirements reviewed and approved? Is there a formal verification and validation program, specifying activities to be performed, methods and tools, acceptance criteria, independent V&V? Are the V&V activities appropriately documented? In which measure are they automated? Are design review meetings held? Are there code walk-throughs? How are comments and action items captured and tracked? Is there a formal risk assessment? Does failure analysis continue throughout the design, development, coding, peer review, and testing process? Is there a testing methodology that ensures both documented module and system level testing? Does the testing methodology require testing all boundary conditions? What standards are applied?

Configuration Control
How does the organization control hardware and software revisions? How is the software stored/archived? Are copies retained in off-site storage? Does the version and configuration control environment guarantee that all the baselined items of a given version of the system can be retrieved? What kind of media does the vendor use? How are design documents controlled? Can the source code for embedded PROMs (where applicable) be found? Are development tools such as compilers archived with the software programs? Are all versions of released software maintained? How are revisions identified for software? Can modifications to hardware or software be made without issuing new customer discernible model or revision labels? Does the vendor have configuration and documentation control for changes and modifications? How do they communicate changes or modifications to the client? Who has access to the software? Who can make modifications to the software? What process does the vendor follow for this task?

Figure 4-7
Process Orientation Questions

Acceptance Testing
Are there defined acceptance criteria? Inspection procedures? Test procedures? Software tools for acceptance and testing? What types of tests are performed (e.g., burn-in test)? Are there procedures to address discrepancies or failures?

Certification to recognized standards
Has the system or part of it been certified to recognized quality, dependability or safety standards? If so, to which level of the standard (if the standard defines several levels)? Does the certification apply to this very version of the system? What is the certification body? What are the limits and assumptions of the certification? Is it possible to audit the certification procedures and detailed results?

Manufacturing Process
What are the manufacturing methods and practices? Is manufacturing work contracted out? How does the vendor maintain or ensure quality in the received products? What are the quality procedures to ensure the quality goals are met? Does the vendor quality control staff perform audits and inspections?

Customer Service and Complaints
Does the vendor have a customer service policy? Are knowledgeable staff available during off hours? Are all calls documented and tracked? Is there a clear process for identifying system errors? How are customer returns handled? Are hardware boards upgraded automatically? Can customers request specific revision levels be maintained when having repairs made, or when ordering spare parts?

Error Tracking and Reporting
Is there a clear procedure for handling error reports? What are the technical means to record error reports? What type of information is collected in an error report (system identification, system configuration, operational context, failure conditions, symptoms, ...)? Is there a customer notification process? If so, who is contacted and how? Are open error reports available for customer review? How is an error report tracked and closed?

Failure Investigation
Do error reports result in failure investigations? Are failure reports reviewed with management? Are design errors merely fixed, or do failure investigations look for root causes? Are prior systems, or revisions to existing systems checked for similar problems? Are failures/errors tracked and trended for “big picture” analysis?

Modification process
Does the vendor have clearly defined processes and procedures for deciding, defining, developing, verifying, documenting and tracing system modifications (including in manufacturing)? Does the vendor have a clearly defined policy for informing customers regarding system modifications and for maintaining and servicing older versions? What measures are taken to ensure that properties essential to critical systems are maintained or improved? Is documentation systematically kept up-to-date? Are customers systematically informed of modification or modification plans? What plans and proven record does the vendor have for staying abreast of component obsolescence?

Figure 4-7
Process Orientation Questions (Continued)

4.5.3 Results

The CDR team should note any issues that have surfaced during this orientation. These issues will be further examined through Thread Analysis, usually during on-site reviews.

4.6 Thread Analyses

4.6.1 Purpose

Thread analyses are usually performed during on-site reviews and serves four main purposes:

- To verify the information and processes examined during System and Process Orientation.
- To further analyze the issues raised by System and Process Orientation.
- To reveal “informal” practices and “tribal knowledge.”
- To provides a detailed, in-depth sample of the system’s integrity.

4.6.2 Method

A thread analysis is a detailed, systematic tracing of specific functions through a vendor’s system and processes. Thread analyses are analogous to a statistical sampling.

Performing a detailed review of an entire system’s software, which could easily exceed 100,000 lines of source code, is completely impractical. The same could be said for tracing all the requirements through design documents, drawings, and test procedures. By carefully choosing “threads to pull,” we can gain a reasonable level of confidence, or concern, in the system.

4.6.2.1 Selecting Threads

Selecting threads to pull is the first step. While certain critical threads (e.g., safety shutdown) can be determined prior to the CDR based on the application, the CDR team should utilize the experience they have gained up to this point in the review to select other threads to pull.

At this point, individual team members may have found specific items of concern, or interest. If the vendor has adequate resources — and a lack of resources may or may not be of concern — each team member may choose to pull his or her own thread. The CDR team should, however, understand what each thread will be, and who has adequate technical knowledge to follow the thread.

For small systems, where the criticality of the system is relatively low, two threads may be adequate. Most systems, however, should be subject to no less than three threads, and possibly more depending on criticality and complexity. The chosen threads should, at a minimum, contain one process and one technical thread (see Figure 4-8).

Where a system is distributed, the CDR team may want to consider pulling threads for each distributed unit, and one or two that will follow the integrated system.

Process Thread
To perform a process thread, find a system error or failure that led to a revision of either hardware or software. If possible, find a fault that was initially reported by a customer.

First, trace the reporting mechanism. Ask to see the initial call report, or complaint/error form. Does the vendor form provide critical data such as date reported, party reporting event, system affected, description of the problem, and some unique identifier?

From the initial report, trace the complaint to the modification process. Is there any indication of root cause analysis, or does the staff merely "fix bugs?" Are errors communicated to existing customers? Are previous releases or products that share similar design features reviewed for common problems?

Are all of the requirements and design documents reviewed and revised to ensure any changes are adequately reflected? Are changes to software documented in the source listings?

Is regression testing performed? Who determines what testing is adequate? Are changes tested with older versions to ensure compatibility where both old and new designs may co-exist?

How are changes communicated to the production process? How is the final change package approved, and by whom?

Technical Thread
A technical thread will follow a system function from input to output. A typical thread would start with a signal from the field device. The purpose of this thread is to evaluate the core functions of the system.

First, the bounds of the system are defined for both normal and failed conditions. Failed conditions may include high, low, cycling, or shorted.

Trace the signal into the data acquisition hardware. Questions to ask include what type of analog-to-digital converter is being used, and is the converter dedicated to a single input, card, or system? How frequently is the signal converted? Where is the value stored? Is the signal filtered? If so, how? Is there any protection from field surges?

Check design documents for accuracy and completeness.

How is the output signal generated? Does the system verify output signals to the field? If so, how?

Check test documents for completeness. Did their testing cover this signal path? Under what conditions?

Application Thread
Choose a critical function from the application for which this system is being considered. For example, shutdown of some pump. From here, the thread is pulled in the same fashion as a technical thread.

Figure 4-8
Thread Analysis

4.6.2.2 Items for Consideration

The basic areas that should be considered during Thread analyses include:

- The physical system.
- Software.
- Documentation.
- Vendor processes.

The physical system includes all of the various hardware components that comprise the operational system, plus any auxiliary components (e.g., handheld programming device, printers and display devices).

The software includes the software residing in the operational system, and any software used to build, maintain, or otherwise develop the system (e.g., compilers, assemblers, operating systems, and automated test programs).

The documentation includes requirements documents, design documents, drawings, manuals, test procedures, error reports, failure analysis documentation, customer complaint records, and any other documentation that was created with the system that supports the system's use, or that documents the system's behavior.

The vendor processes include all processes reviewed during the Process Orientation, plus any processes that are internal to the development, support, or maintenance functions that are revealed during the analysis.

4.6.2.3 How to Pull the Thread

As the King said to the White Rabbit in Lewis Carroll's *Alice's Adventures in Wonderland*, "Begin at the beginning and go on till you come to the end: then stop."

The starting point for pulling a thread begins at one end of the thread, or just before some aspect of the thread that the CDR team wishes to evaluate. One example of a thread end might be the conceptual or requirements document. The other end of the thread may be the actual source code for a module, or an A/D converter. The beginning and end will differ depending on the vendor, the findings up to this point in the review, and the areas of most concern. Another example might be correcting a specific design error, which might start with the error report and cycle back into the code, and then to the review, test, and release portion of the life cycle.

The CDR team should determine their starting points without regard for what the vendor has or has not defined in their process overview. While this may seem counter-intuitive (e.g., why start with the requirements document if they have said they do not have one), the point in the thread review is to re-construct the development process and discover what actually exists. Lab notebooks and other informal, handwritten documentation may provide some assurance of design, review, and testing activities.

4.6.2.4 What to Look for

Vendors may have good procedures that are not followed, excellent procedures that are followed, or even informal, non-codified processes that exhibit technical excellence.

Similarly, vendors that follow good procedures that are followed do not necessarily produce high quality, technically sound systems.

Emphasis should be placed on technical merit, not dotted “i’s” or crossed “t’s.” However, meticulous documentation may provide an indication of technical excellence. Sloppy or missing documentation may reflect questionable practice.

If starting, say, from the requirements end of a thread for alarm checking, the first question to pose is simply: “Can you show us where the requirements for alarm checking are first stated?”

Commercial vendors may not subscribe to waterfall development methodologies, and may not have a neatly defined “System Requirements Document.” However, the vendor should be capable of producing some artifact that describes the alarm checking requirements.

If a vendor produces a user reference manual, which vendors usually produce after development is complete, the CDR team should continue to pursue the questioning. It is possible the vendor writes the user’s manual first, and bases the design on that manual. Still, the developers can be asked how they knew to write software for an alarm checking function. Often, this line of questioning will reveal the internal development practices.

From requirements, the questioning should lead to design documents, peer review documentation, test documents, code and detailed hardware layouts. At each stage, the reviewers should look for both formal and informal practices.

At the software module level, the reviewer should note items such as headers, comments within the code, readability of the code, and use of global commons or data hiding techniques. If numerous individuals have written code, the CDR team should ensure that they examine several portions of code to determine if there is consistency in the coding practices. If there are written coding standards, the code should be compared against the standard.

Hardware design should go through a similar design review. Separation of tracing on boards, the use of jumpers, polarized capacitors (which may work in the reversed direction for short testing periods), hand versus machine stuffed boards, and ESD handling should all be considered.

Process Threads will follow a similar route, but the threads will extend into customer notification procedures, error reporting, change control, configuration control, and regression testing. The same documents that are reviewed for the technical threads should also be reviewed to determine if they have been updated. Software coding and hardware drawings should be examined to determine if changes have been clearly identified.

At each stage reviewers should look for evidence of technical reviews, noting their independence, formality (e.g., are there action items?), and follow-through. Lack of technical reviews should raise concerns.

4.6.3 Results

Thread analyses can answer the following questions:

- Does the staff follow vendor procedures?
- Is documentation planned, or an afterthought?

- Does the staff perform technical reviews on a regular, planned basis?
- Is there a well maintained design basis?
- Does the vendor manage design change adequately?
- Does the vendor notify customers of errors?
- Is the software well structured and understood?
- Is the staff knowledgeable?
- Does the vendor need to formally document their existing informal practices?
- Does the vendor use good practices?

While negative answers to some of these questions may not eliminate the need or desire to use a specific system, they may lead to new requirements for the vendor and/or project team. If a vendor shows poor configuration control, the utility may need to negotiate the rights to design documentation. If a vendor has not performed technical reviews, a project team may want the CDR team to increase the scope of the review, and may require more extensive testing.

The analysis can also deepen the CDR team's knowledge of the internal mechanisms of the system. This knowledge is critical in the final phase of the CDR, the risk analysis.

4.7 Staff Capabilities and Attitude

Since the final quality and integrity of a digital system depends mostly on the quality and integrity of the staff that designed, implemented and tested it, the CDR team should consider the vendor's technical staff. A good staff trying to do good work can produce a high quality system in spite of bad procedures: their attitude and beliefs about the software quality assurance process, as well as their attitude toward safety, sets the resulting system quality and integrity. If the staff is not competent or motivated to do good work, good procedures by themselves cannot create a high quality system.

However, the CDR team does not formally evaluate the vendor's staff. Rather, the CDR team documents their review of the staff through the proofs of the quality and capabilities of the system; review of the design, development, review, and test work products; and discussions with the vendor staff. This is one of the most subjective evaluations. In the U.S., large changes in staff are possible with little or no advance warning. The CDR team would prefer to discover that the vendor does not depend on always retaining key staff members. Reliance on a single person's memory and knowledge of a system is dangerous, and may result in issues with long term maintenance of that system. The authors of this report have seen several cases where the vendor had to abandon acceptable quality systems after key staff members left.

If the technical staff believes that design paper is just something they have to do to fulfill a contract, then the likely result is a system that is not as safe or high quality as desired. If the technical staff believes that the design paper is there to support them in design, development, and long-term maintenance of a system, that software design and development is a team endeavor, and that safety and integrity are design goals, then the system is much more likely to meet or exceed the CDR team expectations and requirements.

Similarly, the CDR team should evaluate the technical staff's attitude towards change control and configuration management. If the vendor does not control change, and if the staff has a cavalier attitude towards change control, it is likely that the staff has and will continue to introduce uncontrolled changes into the system. This attitude normally is prevalent where the vendor does not produce or maintain design documentation and does not practice peer review. This means that the utility is more likely to have to purchase a sufficient number of replacement parts along with the installed system, since it is unlikely that the same, or even similar, configurations can be generated later. Failure to purchase sufficient spares means the utility may have to re-perform a CDR later, along with any the required qualification and testing activities, to ensure that later system is acceptable.

The staff will likely not make any single statement that allows the CDR team to understand their feelings and core beliefs about software quality assurance processes. It is up to the CDR team to gain this understanding through observation, through the tone of discussions with the vendor's staff, and through reading and study of their design, verification, and validation documentation.

Sometimes, the vendor's staff will state directly that safety is a key, essential, known requirement. Such discussions can include statements that their first training of a new engineer includes direct statements about safety and the safety impact of their equipment. In one case, the vendor's staff challenged their new engineers to continue to think about "having their family live beside a plant using their equipment, and, if they are not comfortable with the design, to challenge and change it until they would be comfortable." This is the attitude the CDR team wishes to find in the vendor design and development staff associated with high criticality systems.

The CDR team also needs to evaluate the qualifications, capabilities, and competence of the design, development, peer review, and test staff. The CDR team cannot just ask directly about these issues, but can guide discussion into areas where the reviewer can form their own opinion about the vendor's staff capabilities. The internal question for the team seems to be, "Would I hire this person to fulfill this function, with equipment of this safety significance?" If the CDR team does not feel comfortable with the vendor staff, then either the CDR team needs to have more discussions that lead to answers that reassure the team, or the team may recommend that this vendor and their equipment not be used in the proposed application. As the CDR team evaluates system performance through the operating history and through the verification and validation processes, the CDR team also evaluates the vendor staff.

Straightforward, forthright communication is essential. When a utility evaluates a vendor, it should be evaluating the advantages and disadvantages of a long term relationship with that vendor. Should the utility come across a problem with the system, they will likely need to call the vendor, communicate an issue or problem they have with the vendor, and get an honest, immediate answer. After all, when a utility calls, they usually have a big problem and need immediate resolution so the plant can continue to operate, or restart expeditiously. If the CDR team thinks they are going to have difficulty getting honest, forthright answers from the vendor, the utility should seriously consider other vendors, even if the other vendors' systems are technically inferior.

The CDR team needs to evaluate the vendor's staff honesty, openness, and integrity, as well as their capability to communicate information. If the vendor hides issues and problems with their system or if the vendor is not open about plans for the system's future, the CDR team should seriously reconsider plans for involvement with this vendor. Nuclear power plants buy equipment for long term use. Suppliers establish long term relationships with their commercial equipment vendors. If the vendor will not work now in an open, honest, frank manner while they are highly motivated by their attempt to sell equipment, one should not expect them to change after the vendor has sold the equipment and the motivation level drops.

4.8 Risk Analysis

The Risk Analysis can be the most important facet of the CDR. This activity will use the knowledge gained in the previous sections to help answer the questions: "Is the use of this digital system a reasonable solution? Are there risks of undesirable behavior?"

This phase combines both a qualitative *fault tree analysis* (FTA), and a qualitative *failure modes and effects analysis* (FMEA). The term "qualitative" is used deliberately to avoid any confusion with FTA or FMEA approaches that seek to quantify failure potential and reliability. No method for predicting digital failure rates or digital faults has been established for the diverse range of digital systems that utilities are likely to use.

Experience has shown that no single method of risk analysis can provide full coverage of all risks. Studies have shown, however, that combining a top-down method with a bottom-up method provides better coverage than either method alone. While the use of a combined method increases coverage, no method or combination of methods can guarantee full coverage of all potential risks.

Figure 4-9 provides a list of suggested areas for investigation. These should *not be used as a checklist,* but to facilitate generation of appropriate questions.

4.8.1 Prerequisites

The Risk Analysis requires in-depth knowledge of software, hardware, interfaces, operation, and application. The vendor should be asked to have knowledgeable staff present during these sessions. The CDR team, having examined the various system documents, should specify the documents that need to be available during the sessions. These documents may include both formal and informal documents.

System Inputs: Consider each analog input freezing mid-scale: What is the resulting behavior? Consider each analog input failing low or failing high: What is the behavior?

Software: For each section of code, suppose an infinite loop were inserted at different points: Are there software sections where inserting an infinite loop would not cause watchdog timeout?

Error Handling: How are real-time error events handled? Are CRC memory checks made at startup or during code execution? Are there any error checks that occur in real time?

System Behavior During Failure: Can the system stop functioning without indicating it has stopped? If the operators do know the system has failed, can they take remedial actions in time to prevent a serious accident, a challenge to a safety system, or lost revenue? Can the system fail in such a way that false information may mislead the operators? Can a failure prevent the operators from taking manual control? For each hardware subsystem, classify control room/surveillance behavior under different failure scenarios.

Redundancy: In redundant failover schemes, are transients and/or race conditions a factor? Can both processors become master at the same time? Can both processors become backup at the same time?

Watchdog Timer: Where is the watchdog refreshed in the code? What does the watchdog timer do when it is triggered? Does the system reboot without indication to the operator? Can essential parts of the software stop running without being detected and clearly indicated to the operator?

Multitasking: If a multitasking system is used, will failure (e.g., aborts, infinite loops) of every task be detectable?

Diagnostics: Are diagnostics run on system boot-up? Are diagnostics run in real time? What tests are run during diagnostics? Can diagnostics interfere with real-time processing?

Undocumented Software: Does the vendor leave undocumented code in the system? Is there dead code in the system?

Figure 4-9
Risk Analysis Questions

System architecture documents, which are critical for the review, are most useful in the form of drawings and/or diagrams. The ideal set of documentation for software architecture would include information regarding the following:

- Software partitioning.
- Data flow.
- Control flow.
- Critical timing and throughput.
- Hardware interfaces.
- Operating system documentation.

Hardware architecture documentation should include information and drawings for the actual boards in the system, and defined interfaces to and from all controlled devices.

4.8.2 Qualitative Fault Tree Analysis

4.8.2.1 Principle

The FTA works from undesired states or hazards, and then works back to determine the potential cause(s) of the undesired states. The FTA is documented in a fault tree diagram where the primary event is at the top of each tree. Events and/or conditions that must be present to cause this primary event are drawn under the primary event with lines connecting the two. Where needed, "and" gates should be shown; "or" gates are implicit. Each new event is then deconstructed in a similar fashion until basic events have been uncovered.

What constitutes a "basic" event is an arbitrary decision. The basic events should provide the CDR and Project teams with enough information on which to base design decisions. For example, "software failure" would not provide adequate information, while "infinite loop in calculation module" could. For the infinite loop example, a detailed examination of the code in the calculation module could determine if there is the possibility for infinite loops.

4.8.2.2 Hazard Identification

Defining the undesired or hazardous states is the first step in the FTA. The question the CDR team should be asking is: "What are the worst things this system could do?" The project team should have identified many of the system level hazards prior to the CDR, but the CDR team should use their recent knowledge of the system to postulate additional hazards.

For most systems, there will be numerous "worst things." System failure, while an obvious "worst thing," is only a beginning. What if the system or a component fails without indication? What if the system ceases to calculate values properly? What if the system fails in such a way as to give the appearance of working (e.g., lights remain lit and there is no visible indication of failure)?

4.8.2.3 Fault Tree Construction

For each "worst thing," construct a tree starting at the top with the undesired state, and show what states or events would have to occur to cause this event.

Each subsequent event or state is decomposed until initial events are found which cannot be decomposed. (As noted earlier, this is often an arbitrary decision.)

4.8.3 Qualitative Failure Modes and Effects Analysis

4.8.3.1 Principle

The qualitative FMEA postulates failures at a module level — both hardware and software — and determines the effects at both local and system levels. For each postulated module failure, the CDR team should determine and document:

- The postulated failure.
- The effects of the postulated failure.
- Possible causes of the failure.
- Possible mitigating actions/design features.

For each module, there may be several potential failures to consider.

4.8.3.2 Assembly Drawings and Specifications

The use of system architecture drawings for the digital, electrical, and mechanical sub-systems is required to perform the analysis. Software architecture drawings or sketches from the system orientation should be available.

4.8.3.3 Postulate Failures of Modules

Using the drawings, identify modules. A module can be a complete computer, a board in a computer, or a component on a computer board. Module sizing should be chosen according to the perceived system threats.

If a computer is being used merely to record data, then the computer may be viewed as a "module" during the initial pass. However, if a computer board is relied upon for an emergency shutdown, then the board itself would be a more appropriate module choice. At the greatest detail, individual components on the board may be appropriate.

Software modules should also be chosen in the same fashion. For some applications, a task composed of numerous subroutines may be the appropriate module size, but for others, the subroutine itself may be the appropriate size.

Each module is assumed to fail. For the initial pass, do not consider the likelihood of failure. (A high consequence, low probability failure led to the Bhopal disaster.)

Software, which is really a set of "instructions," cannot by definition "fail," but it may introduce "faults." Many software attributed failures are often the result of compromised or faulty memory locations. Bad indexing, altered program counters (bit flipping), and a host of other hardware-related failures can cause tasks to abort. While these failures may initiate in hardware, the effect on the software is the interesting issue.

The modules chosen are somewhat arbitrary in terms of defining what makes up a module. In general, when a chosen module can be the cause of a serious failure, the decomposition of the module into smaller modules may be warranted. For example, if the undetected failure of an input/output (I/O) module can cause human harm, then the I/O module may require further decomposition to understand the failure modes internal to the module in order to determine the likelihood of the failure occurring.

4.8.3.4 Identify Failure(s)

For each module, identify the failure. For some modules, there may be multiple failure modes. For instance, a software module may simply abort, or the module may become trapped in an infinite loop. A pump may also fail in several ways, each having a different effect.

4.8.3.5 Determine the Effects of Failure

Once the modules are chosen and the failure modes defined, the effect of each module failure is traced through the system. The effect of this failure is then documented.

Careful attention should be paid to the human factor in each failure. If the operator of the system has clear and unambiguous indication of the failure, the effect may be minimal compared with the effects from a silent, undetected failure.

4.8.3.6 Identify Probable Causes

For each failure, identify potential causes of the failure. For instance, processor reboots or random memory failures may be the result of a power surge.

4.8.3.7 Identify Mitigating Actions and/or Design Decisions/Changes

For each failure, determine what actions could be taken to either prevent or mitigate the consequence of the failure.

4.9 Operating History Review

If the vendor has shipped the system widely to industrial users, the CDR team can use the industrial experience with the system to further refine and support the CDR evaluation. Use at other nuclear facilities can also provide supporting evidence. However, if the system is not in wide use, or if it has only been shipping for a short time, operating history may not be useful.

The objective of the Operating History Review is to evaluate a system with significant field experience, based on actual usage in applications similar to those expected to be encountered in the intended nuclear power plant application(s).

The review of operating history should be focused on those areas where the history data is needed to support verification of the critical characteristics. The review should also focus on obtaining information regarding any failure modes and history of failures. In the evaluation, the amount of operating experience may be compared to the requirements for availability and reliability of the system in its intended application to determine whether the experience data is sufficient to indicate that the probability of undetected failures is acceptable.

The relevance of the operating history should be established to ensure that factors such as configuration of the system, the environment, and the way in which it is operated are sufficiently similar to the intended application(s) so that the experience data is valid.

Operating history data may be gained from a number of different sources. The applicability of the data can differ greatly depending on the source that is used. The various sources of operating history data may include the vendors, users who have gained experience with the system, and industry user groups and industry reports, if such exist.

The CDR team normally asks the vendor for at least six to eight different user names and telephone numbers. If the system has only been available for a few years, ten to fifteen different users might be necessary to increase the total operating hours. One should not normally expect to get data from all users. The authors of this handbook suggest that the CDR team should attempt to interview those users that have applied a significant number of the system in critical applications, preferably for several years, as these users tend to remember problems with such applications. If an industrial user indicates that they would not install the system in such areas, the authors of this handbook recommend that nuclear power users follow the industrial user's lead. The CDR team should try to document the reasons that the users provide for such decisions. These are especially critical to discuss with the equipment vendor.

The CDR team should document the telephone conversations with the users, and include them in the quality records and in the reference list for the CDR report. The CDR team should discuss with the vendor any failures that the users did not understand, or that are potentially interesting, during the CDR.

The specific questions that should be asked when soliciting operating history data will depend on the following factors:

- The source of operating history being queried,
- The type of equipment and the application in which it will be used,
- The critical characteristics of the equipment, particularly those that are most in question and/or most critical to the dedication based on the failure analysis, and
- The extent to which the operating history is needed to complete the verification of the critical characteristics.

Section 5.6.3 of EPRI TR-107339 provides a sample list of questions that can be used as a starting point for compiling the operating history questionnaire for the CDR. In addition, Figure 4-10 provides a sample list of questions.

Contact Information:

- Person Contacted
- Company
- Telephone Number

Application Information:

- Number Units Installed
- Number Units Purchased
- Average Length of Service
- Model Number(s)
- In what application(s) are the devices installed? Are they used in Emergency Shutdown applications?
- Are these devices critical to operations (e.g., used in Safety Instrumented System, Emergency Shutdown System, or in an economically sensitive function)?
- Are valve response speed and/or accuracy critical? If so, what response speed and/or accuracy are required?
- Installation Environment (heat, humidity, wind, dust, vibration, shock, corrosive atmosphere, EMI, RFI, electrical transients, radiation, etc.)

Device Configuration and Function:

- How do you control (or take data from, or perform control actions using) this device?
- How do you normally configure the device – using the local vendor-supplied interface, software running on a PC, or another vendor tool?
- How do you calibrate the device – using the local vendor-supplied interface, software running on a PC, or another vendor tool? What is the calibration frequency?
- Have you ever improperly configured the device with any of these tools?
- Do you use the setup and calibration protection? If yes, have you ever had any problems using/removing protection?
- Do you use the auto calibrate capabilities? If yes, how well does it work?
- Do you use user adjusted or expert tuning? Does it work well for you?
- What valve characteristic do you use (linear, equal percentage, quick opening, or custom)? Does it work well for you?
- Do you use the set point filter time? Does it work well for you?
- Describe diagnostic capabilities used. Do you use the local vendor-supplied interface, software running on a PC, or another vendor tool for diagnostics, which includes the list of program features? How do you incorporate diagnostics into maintenance program? Are you satisfied?

Device Specific Options:

- Do you use the remote mount position sensor module?
- Do you use standard pneumatic relay or something else?
- What style of actuator is used with the device used with (spring and diaphragm, piston double-acting without spring, piston single-acting with spring, piston double-acting with spring)?

Figure 4-10
Operating History Questionnaire for a Digital Valve Controller

Failures:

- Number failed units
- Reasons for failure (if known)
- Have you ever experienced fault/failures due to EMI?
- Did the device fail to the desired position (valve open/closed/hold current position)?
- Effects/consequences of failure
- Corrective actions (e.g. replace unit, reset system, shut down, etc.)
- Device failure rates/replacement/failure cause tracked?
- Were you satisfied with the vendor's response to your failed device?

General:

- Are there any unfavorable features or unexpected characteristics of the device about which you would warn prospective users?
- In retrospect, would you purchase the device again for the same or other applications? Explain why or why not:

Figure 4-10
Operating History Questionnaire for a Digital Valve Controller (Continued)

4.10 Resolution of Pending Issues

There are always application limitations on any system, and almost no vendor complies with all the expectations. Therefore, the CDR team needs to consider the system reviewed, document the set of appropriate restrictions that occur to the team, and consider what the vendor and/or the utility can do to compensate for weaknesses and gaps between the system and the expectations. An example of a compensatory action that might be recommended in a CDR report would be a requirement imposed by the utility on itself to write nuclear plant procedures for configuration changes within the system.

The CDR team has to synthesize what was learned about the vendor throughout the discussions and document reviews, to determine what gaps exist between expectations and what the vendor has done. Gaps are not differences in document names or organization, or moving actions between phases in a life cycle. Gaps are actions or documents that do not exist in any form, and can therefore not be credited.

The CDR team should separate compensatory actions into individual groups of actions for the vendor, the dedicator, and the end user. Any compensatory actions performed by the vendor or the dedicator should be documented as resolving this issue, reviewed and approved as the same level of quality document as the CDR, and the documentation of these actions kept with or referenced by the CDR.

For the vendor, the gap analysis might show that the vendor did not address faults and failures well during the initial design, and that operational history shows a series of system failures. If assessment activities continue, the vendor likely would need to strengthen or perform failure analyses for this system. From experience, root cause analysis for the operational failures would be useful as a starting point, but would likely not resolve all issues with the system. Activities for the vendor are not common. However, just about all commercial vendors will correct design defects uncovered during these reviews. Most vendors enter these reviews hesitantly, convinced that the CDR team will insist that the vendor retrofit major process changes or expensive documentation. Such activities are rare, but possible.

For the end user, the CDR team should document those issues and application cautions that the end user would need to consider in using this system in the application(s) considered in this CDR. The CDR team should document any issues that are noted in the design, development, and maintenance of the equipment before, during, and after installation in the end user's nuclear power plant. For example, with valve positioners, some CDRs have noted that the control system will require re-tuning, since the automatic tuning in the valve positioner would improve the performance of the valve, and thus change the system response. If the user was not aware of this, then the user could de-tune the valve positioner and lose performance enhancements, or the user could do nothing and adversely affect plant stability. The choice needs to be identified to the end user, with strong recommendations as to the preferred, recommended approach to resolving the problem, which would be to re-tune the process control system and accept better plant performance.

4.11 CDR Report

The CDR report has several audiences, and several purposes. The report documents the review of the system, and provides technical detail for use in installing and maintaining the equipment. Thus, the utility engineers responsible for the equipment are a primary audience. The CDR report is also useful for the engineers responsible for designing, implementing, and testing the system and its interface to the plant, including any licensing analyses required for the modification process. The system vendor is also an audience for this report. The authors of this report have concluded that the CDR can serve as a "report card" for the vendors, showing how well the vendor complies with the nuclear industry's expectations, and where improvements are appropriate.

The outline suggested in Figure 4-11 has evolved with each CDR performed. The authors of this report expect the outline and data expected in CDR reports will continue to evolve, with changing expectations and improved software processes.

1 Introduction and Background
  - 1.1 Purpose
  - 1.2 Overall Project Goals
  - 1.3 Equipment Covered by This CDR
  - 1.4 Purpose of the Equipment
  - 1.5 System Architecture Overview

2 Summary, Cautions, and Recommendations for Use
  - 2.1 Summary
  - 2.2 Conclusions from the Review
  - 2.3 Application Cautions
  - 2.4 Recommendations for Use
    - 2.4.1 Qualification and Maintenance of Qualification
    - 2.4.2 Installation and Maintenance

3 Functional Review
  - 3.1 System Identification
  - 3.2 System Characterization
  - 3.3 Functional adequacy
  - 3.4 Differences Between the Current Installed System and Proposed System Implementations (in the case of an upgrade)

4 Architecture Review
  - 4.1 System Design
    - 4.1.1 Hardware, Module by Module
    - 4.1.2 Software, Module by Module
    - 4.1.3 External Interfaces
    - 4.1.4 Data Flow
    - 4.1.5 Diagnostics
    - 4.1.6 Configuration Security
  - 4.2 Development Tools

5 Process Review
  - 5.1 Corporate Culture
  - 5.2 Development History
  - 5.3 Hardware Design and Development Processes
  - 5.4 Software Design and Development Processes
  - 5.5 Configuration Issues
  - 5.6 Acceptance Testing
  - 5.7 Manufacturing Processes
  - 5.8 In-Process Tests, Burn-In, and Final Acceptance Test
  - 5.9 Security
  - 5.10 Reporting Failures

Figure 4-11
Suggested CDR Report Outline

6 System Failure Analysis

6.1 Known Hardware Faults and Failures

6.2 Known Software Faults and Failures

6.3 Failure Detection

6.4 Self-Test and Diagnostics

6.5 Common Mode Failure

6.6 Hardware Risks Evaluation

6.7 Software Risks Evaluation

6.8 Human Errors

7 Operating History Survey

7.1 Introduction

7.2 Summary of Key Responses

7.3 Foundation for Using the Operating History Data

7.4 Operating History and Failure Data

7.5 Other Relevant Operating History Information

7.6 Conclusion from Operating History Data

8 Acronyms and Definitions

9 References

Figure 4-11
Suggested CDR Report Outline (Continued)

5
POST CDR ANALYSIS

As stated earlier, a Critical Digital Review is assumed to be part of an overall project. The CDR team will return from the review with objective evidence that describes the vendor, the vendor's practices, the digital system(s), and the potential issues that may arise from the use of the system. The CDR team may also return with suggestions, comments, and recommendations that affect the conceptual design of the project.

The CDR team should meet after the review to discuss any overall impressions, capture any open issues left with the vendor, and determine how to present their analysis to the project team.

The CDR should be well documented. All concerns, recommendations, and identified risks should be incorporated into whatever mechanisms the project team uses to track open action items.

The project team should review the results of the CDR with the entire CDR team to ensure the correct transfer of issues and recommendations. The project team should then take ownership of the report, and ensure resolution of all items.

The *resolution* of the risks *does not mean elimination* of all risks. In some cases, the risk cannot be avoided without unwarranted expense. But mitigation of the failure by, for example, operator training, may reduce the consequences to an acceptable level.

WARNING: This Document contains information classified under U.S. Export Control regulations as restricted from export outside the United States. You are under an obligation to ensure that you have a legal right to obtain access to this information and to ensure that you obtain an export license prior to any re-export of this information. Special restrictions apply to access by anyone that is not a United States citizen or a permanent United States resident. For further information regarding your obligations, please see the information contained below in the section titled "Export Control Restrictions."

Export Control Restrictions

Access to and use of EPRI Intellectual Property is granted with the specific understanding and requirement that responsibility for ensuring full compliance with all applicable U.S. and foreign export laws and regulations is being undertaken by you and your company. This includes an obligation to ensure that any individual receiving access hereunder who is not a U.S. citizen or permanent U.S. resident is permitted access under applicable U.S. and foreign export laws and regulations. In the event you are uncertain whether you or your company may lawfully obtain access to this EPRI Intellectual Property, you acknowledge that it is your obligation to consult with your company's legal counsel to determine whether this access is lawful. Although EPRI may make available on a case-by-case basis an informal assessment of the applicable U.S. export classification for specific EPRI Intellectual Property, you and your company acknowledge that this assessment is solely for informational purposes and not for reliance purposes. You and your company acknowledge that it is still the obligation of you and your company to make your own assessment of the applicable U.S. export classification and ensure compliance accordingly. You and your company understand and acknowledge your obligations to make a prompt report to EPRI and the appropriate authorities regarding any access to or use of EPRI Intellectual Property hereunder that may be in violation of applicable U.S. or foreign export laws or regulations.

The Electric Power Research Institute (EPRI)

The Electric Power Research Institute (EPRI), with major locations in Palo Alto, California, and Charlotte, North Carolina, was established in 1973 as an independent, nonprofit center for public interest energy and environmental research. EPRI brings together members, participants, the Institute's scientists and engineers, and other leading experts to work collaboratively on solutions to the challenges of electric power. These solutions span nearly every area of electricity generation, delivery, and use, including health, safety, and environment. EPRI's members represent over 90% of the electricity generated in the United States. International participation represents nearly 15% of EPRI's total research, development, and demonstration program.

Together...Shaping the Future of Electricity

Program:
Nuclear Power

© 2005 Electric Power Research Institute (EPRI), Inc. All rights reserved. Electric Power Research Institute and EPRI are registered service marks of the Electric Power Research Institute, Inc.

Printed on recycled paper in the United States of America

1011710

ELECTRIC POWER RESEARCH INSTITUTE
3420 Hillview Avenue, Palo Alto, California 94304-1395 • PO Box 10412, Palo Alto, California 94303-0813 USA
800.313.3774 • 650.855.2121 • askepri@epri.com • www.epri.com