LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-CONF-204216

Software Quality Assurance for Nuclear Safety Systems

D. R. Sparkman, R. Lagdon

May 18, 2004

International System Safety Conference
Providence, RI, United States
August 2, 2004 through August 6, 2004

UCRL-CONF-204216

Software Quality Assurance for Nuclear Safety Systems

Debra Sparkman, Lawrence Livermore National Laboratory
Center for Application Development and Software Engineering

Richard Lagdon, US Department of Energy
Office of Environment, Safety and Health

Keywords: software quality assurance, nuclear safety, safety software

Abstract

The US Department of Energy has undertaken an initiative to improve the quality of software used to design and operate their nuclear facilities across the United States. One aspect of this initiative is to revise or create new directives and guides associated with quality practices for the safety software in its nuclear facilities. Safety software includes the safety structures, systems, and components software and firmware, support software and design and analysis software used to ensure the safety of the facility.

DOE nuclear facilities are unique when compared to commercial nuclear or other industrial activities in terms of the types and quantities of hazards that must be controlled to protect workers, public and the environment. Because of these differences, DOE must develop an approach to software quality assurance that ensures appropriate risk mitigation by developing a framework of requirements that accomplishes the following goals:

- Ensures the software processes developed to address nuclear safety in design, operation, construction and maintenance of its facilities are safe
- Considers the larger system that uses the software and its impacts
- Ensures that the software failures do not create unsafe conditions

Software designers for nuclear systems and processes must reduce risks in software applications by incorporating processes that recognize, detect, and mitigate software failure in safety related systems. It must also ensure that fail safe modes and component testing are incorporated into software design. For nuclear facilities, the consideration of risk is not necessarily sufficient to ensure safety. Systematic evaluation, independent verification and system safety analysis must be considered for software design, implementation, and operation.

The software industry primarily uses risk analysis to determine the appropriate level of rigor applied to software practices. This risk-based approach distinguishes safety-critical software and applies the highest level of rigor for those systems. DOE has further defined a risk approach to nuclear safety system software consistent with the analyses required for operation of nuclear facilities. This requires the grading of software in terms of safety class and safety significant structures, systems and components (SSCs). Safety-class SSCs are related to public safety where as safety-significant SSCs are identified for specific aspects of defense-in-depth and worker safety.

Industry standards do not directly categorize nuclear safety software and DOE sites are not consistent in their approach to nuclear safety software quality assurance. DOE is establishing a more detailed graded approach for software associated with safety class and safety significant systems. This paper presents the process and results that DOE utilized to develop a detailed classification scheme for nuclear safety software.

Introduction

The intent of this document is to provide a baseline for establishing the framework for the application of software quality assurance to nuclear facilities within the DOE complex. DOE is using this baseline information to assist in the development of safety software quality assurance requirements and guidance for nuclear facility applications. The new requirements are written as performance/outcome oriented statements and supported by a guide providing

more detail on how to apply the requirements. This document intends to lay a foundation for common understanding, and present concepts and ideas on:

- The types of safety software in common use
- The application of safety software to DOE activities and facilities
- Classification and grading
- The activities, processes and practices applied to safety software to assure quality

This information is based upon accepted industry practices in software engineering and software quality engineering, industry standards, and preliminary industry examples. Information from federal agencies (e.g., NASA, DoD, DOE, FAA), standards organizations (e.g., ISO, ASME, IEEE), and DOE facilities has been reviewed and folded into the concepts and approaches in this document.

This document contains well-known concepts and information. The purpose is to establish basic understanding and common terminology derived from DOE programs, national standards, and DOE requirements to provide a basis for developing SQA requirements. This document discusses these concepts and lays the foundation for the approaches that are presented. Tables identifying the types of software to be graded are included and provide the framework for identifying the appropriate standards in the guidance documents for approaches to SQA.

Nuclear Safety Requirements

Nuclear safety requirements are defined in 10 CFR 830, “Nuclear Safety Management.” This rule governs the conduct of operations that may affect the safety of DOE nuclear facilities. In the past, DOE generally required software quality assurance, however, it is now seeking to define those requirements exclusively for the nuclear facilities it operates to improve its safety posture and minimize the risks of operation.

DOE nuclear facilities are categorized in terms of safety according to the nuclear material contained in the facility. Hazard category 1 facilities represent significant offsite and onsite risks to the public and workers, respectively. Hazard category 2 nuclear facilities represent significant onsite risks to the workers and may represent an offsite risk. Hazard category 3 nuclear facilities represent only localized risks to the facility. Risks are qualitatively or quantitatively derived. In terms of nuclear safety analysis, the hazard category defines the degree to which assurances of safety are required.

Nuclear safety system components are procured and installed with a high degree of quality that is specified throughout the design, construction, testing, and operation. Once installed, nuclear systems are periodically tested to ensure that the system continues to meet its safety functions. Periodic testing may consist of entire system tests or individual component tests. Frequency of testing is based on operational history and failure data and the relative importance to safety of the component or system.

Software utilized in nuclear facilities, regardless of the application, should be treated with the same quality assurance pedigree as the hardware. For nuclear systems, software is one part of the entire system. Failure of the software should be considered in conjunction with the system analysis that is conducted to ensure that the operation remains within safety limits.

Safety analyses for nuclear safety software in DOE facilities should include:

- What makes the system, including the software, safe.
- How the software system interfaces with the overall system and understand its impacts.
- Identify single failure points and their impact on system operation.

Software utilized in nuclear facilities that can affect its safety should follow accepted industry quality assurance practices for existing nuclear facilities. The approach applied for software needs to address independent verification of analysis to ensure that installed systems can perform their safety functions. This includes application of the

unreviewed safety question process utilized by nuclear facilities to ensure that changes, tests, or experiments to the facility, including software, do not decrease the facilities safety margin.

System Classification and Software Grading Levels

There are three basic classifications for systems that are derived from DOE nuclear safety management rule 10 CFR 830: safety class, safety significant and other systems. Grading levels are used to label the software that monitors and controls these systems, that provide safety management or administrative controls, and that are used in analysis or design decisions.

The number of software grading levels and the mapping between the software and its associated system classification still need to be finalized. Initial thoughts are to have two software grading levels (A and B, for example). One organization within DOE has five levels (A-E) to classify all software including non-safety related software; others use two or four levels, depending on the applications, appropriate for their facilities.

Software directly associated with a safety class or safety significant system should be at the highest software grading level. It is difficult to distinguish significant differences in QA practices for software associated with safety class systems from that of software associated with safety significant systems. Sample descriptions for two software classification levels are:

Level A: This grading level includes high safety consequence software applications that meet one or more of the following criteria:

1. Software applications where failure could have an adverse effect on nuclear safety systems (i.e., Safety Class or Safety Significant SSCs), toxic material, or chemical hazard protection systems that are credited in the facility safety analysis for protecting against or limiting exposure to the general public and workers below regulatory or evaluation guidelines.

2. Software applications where failure could result in non-conservative safety analysis, misclassification of SSCs, or inappropriate safety related decisions.

Level B: This grading level includes low safety consequence software applications that meet one or more of the following criteria:

1. Software applications whose failure would cause a reduction in the degree of safety or defense-in-depth.

2. Software application that supports safety management decisions regarding a facility or system operating activity (e.g., software whose failure would not impact performance of a safety function, but could result in: missed surveillances; confusion regarding system status; or noncompliance with nuclear safety regulatory laws, environmental permits or regulations, and/or commitments to compliance).

Software Applications

There are five basic applications for the safety software in DOE facilities/activities: Instrumentation and Control (I&C) process monitoring and control applications, networking and interface applications, safety management and administrative control applications, safety analysis applications, and design and analysis applications.

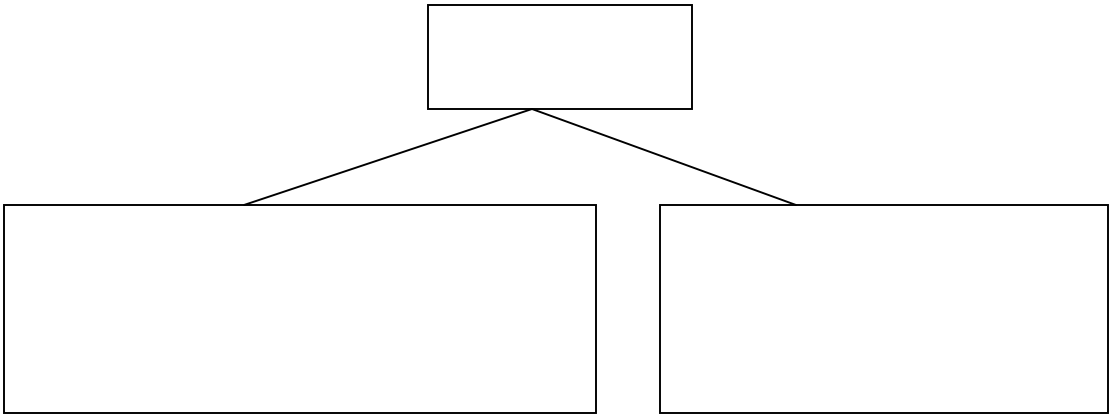

Figure 1. Safety Software Applications

Instrument and Control Process Monitoring and Control Applications: These applications are the software and firmware to control and monitor equipment and components such as valves and switches, including all programmable logic controllers within the safety-class and safety significant structures, systems, and components.

Network and Interface Applications: Software in these applications includes the software and firmware within the network components that interface with structures, systems and components, or other components that perform safety functions.

Safety Management and Administrative Control Applications: Software in these applications includes database applications used in the safety management and administrative controls associated with structures, systems and components, or within the facility. Examples include software used for inventory and material tracking and drum or container hazard assessment calculations.

Safety Analysis Applications: Software in these applications is used for consequence analysis of potential accidents and the evaluation of design basis events.

Design and Analysis Applications: Software in these applications is used for the design and analysis of the structures, systems and components, or the facility. Examples include structural, electrical, mechanical, ventilation, criticality safety, and fire protection design and analysis applications.

Sources and Types of Software

Five types of software are commonly used in DOE applications: custom, configurable, commercial-off-the-shelf, utility calculations, and commercial design and analysis tools. Table 1 maps the five software types to the five applications for the software. The software is primarily from three sources: DOE or their contractors, procured as a service or package, or DOE's Safety Software Central Registry.

The DOE Safety Software Central Registry is a virtual repository of software applications used in the design and analysis of DOE's nuclear facilities. Currently this repository contains six software application codes to calculate and analyze fire source term, leakpath factor, chemical release/dispersion and consequence, and radiological dispersion and consequence. The quality assurance attributes for these codes have been assessed and determined to have an acceptable pedigree for DOE's nuclear facilities.

Type	Application				
	Instrument & Control Process Monitoring & Control	Network and Interface	Safety Mgmt & Admin Control	Safety Analysis	Design and Analysis
Custom	X	X	X	X	X
Configurable	X				
COTS	X	X	X	X	X
Utility Calculation			X		X
Commercial Design & Analysis Tools				X	X

Table 1. Software Types by Applications

Custom: Custom developed software is built specifically for a DOE application. It may be developed by DOE, one of its contractors, or contracted with a third party software company through procurement. This software may be used at more than one facility or DOE site. Examples of this type of software could include material inventory and tracking database applications, accident consequence applications, and control system applications. Some software codes in DOE's Safety Software Central Registry may be custom developed.

Configurable: Configurable software is a commercially available software or firmware that allows the user to modify the structure and function of the software in a limited way to suit the user's needs. An example of this is the software associated with programmable logic controllers.

Commercial-off-the-shelf (COTS): (COTS software is generally supplied through basic procurements. The COTS software includes the operating systems, database management systems, compilers, software development tools, as well as commercial calculational software and spreadsheet tools (e.g., Mathsoft's MathCad and Microsoft's Excel, respectively).

Utility calculation: Utility calculation software typically uses COTS spreadsheet applications as a foundation and user developed algorithms or data structures to create simple software products. The utility calculation software within the scope of this document is used frequently to perform calculations associated with the design of an SSC. Utility software that has a high frequency of use may require the same software quality practices as custom software.

Commercial design and analysis tools: Design and analysis software can be proprietary or available for purchase. Proprietary software is typically custom developed software generally not available to the public, but used by the owner as part of a service. An example would be where DOE contracts for design services. The design service uses their independently developed software (without DOE involvement or support). DOE then receives a completed design. Purchased software is one that is available publicly and is generally procured directly from a vendor. Procurement contracts can be enhanced to require details of the verification and validation performed on the software product. DOE or its contractor in performing design and analysis activities then uses this software. Examples include ANSYS and ABACUS.

Approaches to Grading Software Quality Practices

From the customer's perspective, knowing the quality of the product being used provides a level of confidence in that product. The quality can be specified and built into the product during the software development activities (from concept to retirement). Confidence in the end product is determined by the verification and validation practices performed throughout the software development lifecycle. Unfortunately a customer cannot always control building quality into a product or participate in verification and validation activities performed during its development.

Therefore, confidence in the software must be determined by post-development assessment activities including validating that the software performs its intended function(s). Basically, was the right system built?

The customer can assess the quality of the product or process through verifying that the vendor itself or a 3rd party certifies that the software quality requirements were met. Or the customer may take a more active role in assessing the quality of vendor's development program by visiting the customer site to perform their own assessment. The customer can also just accept the quality of the software based upon key characteristics of the software such as a wide use across multiple applications.

These approaches are performed after the software is built. They cannot increase the quality of the software but only assess the quality to gain a confidence that the software performs correctly and produces the correct results. Customers or oversight organizations will typically use these approaches.

The various software quality assurance and assessment approaches that can be used for each type of software are discussed below. Many of the approaches complement each other and one or two can be mutually exclusive. Table 2 maps the various types of software to the software quality assurance and assessment approaches. The table identifies all approaches that could apply. Not all approaches would be appropriate to be performed in every case. However, multiple approaches may be appropriate in some cases. This table is an example and is intended only for use in understanding the issues that must be considered before developing any DOE requirements.

Approach	Type				
	Custom	Configurable	COTS	Utility Calculation	Commercial Design & Analysis Tool
Build quality into the product	X				
Perform model/algorithm validation	X		X		X
Perform hand calculations				X	
Perform acceptance testing	X	X	X	X	X
Perform assessment of vendor	X				X
Assess vendor certification			X		X
3rd party vendor certification	X	X	X		X
Accept the quality based upon key characteristics			X		X

Table 2. Approaches for Software Types

Building Quality into the Product: This approach applies only to custom software built by DOE, one of its contractors or software vendor contracted for a custom product. It can only be applied during the building of the software. It would not be appropriate for software purchased from others such as commercial-off-the-shelf or Commercial Design and Analysis Tools software used by contractors for the design of a nuclear facility. This approach would not be applicable for software versions already accepted into the Central Registry. This approach can be used with any of the other approaches. However, if this approach is used, the other approaches may not need to be applied at the same level of rigor as without implementing this approach.

To build quality into a safety software product, software engineering and software quality engineering practices are used in a graded approach. These practices are based upon recognized industry standards and guidelines. Some of these standards may include IEEE software engineering standard set, ASME NQA-1, ANS 10.4, DoD 882D, and other government organization standards such as those of NASA and the FAA.

Depending upon the software's impact on safety, the various recommended software engineering and quality engineering practices, including safety analysis for the software and all levels of software testing, are performed to

the specified level of rigor for that grade. These practices are performed throughout the software life cycle from concept to retirement. One such approach is a five level approach (A-E) and the software engineering practices associated with these levels. The DOE approach that identifies the grading levels and the amount of rigor applied to implement the software engineering and software quality engineering practices are being detailed and finalized.

Perform Model and Algorithm Validation: This approach applies to custom, commercial-off-the-shelf, and Commercial Design and Analysis Tools software. A similar but less complex approach, hand calculations, can be used for spreadsheet calculations. This approach is not applicable for configurable software. This approach can be used with any of the other approaches. The drawback is the availability of a similar software product for comparison. Design and analysis codes may provide the widest selection of similar codes that can be used for model and algorithm comparisons.

The confidence level of the software can be determined through performing comparisons with the results of similar software products. This is common when moving from one version of a software product to a newer one or from one generation of a software product to its more advanced off-spring with increased capabilities. It is also very common with accident analysis software to verify that the results from similar products produce similar outcomes.

Perform Hand Calculations: This approach applies only to utility calculation software. Although this approach is not appropriate to validate the complete custom software product, hand calculations might be appropriate to validate the algorithms and formulas within the components of the custom software. Other than the above-mentioned use for custom software, this approach generally is not used with any other approach. Its real benefit (which can be significant) applies to utility calculation software.

Many of the utility based calculations that are labeled as software, use functions and macros in commercial-off-the-shelf products such as Microsoft's Excel and Mathsoft's MathCAD to assist in decisions related to safety. These calculations are generally simple mathematical algorithms or formulas that can be validated using hand calculations. For safety critical systems, formal mathematical proofs are often used to validate the correctness of the calculations performed.

Perform Acceptance Testing or Qualification Testing: This approach can apply to all software types. The approach is commonly used by customers to understand the quality of the software. The approach can be used by itself or to compliment any of the other approaches. It can be one of the most costly in terms of manpower effort if exhaustive testing is performed.

Operational testing may be part (and most likely should be) of the acceptance testing. But nothing precludes acceptance testing being done in a non-operating environment. Acceptance testing would include functional testing, performance testing, security testing, stress testing, and load testing.

Generally the only testing level that is performed here is the system level testing. Where possible when a new version of a software product is obtained, the site should perform predetermined and ad-hoc test cases and procedures to validate that the system meets the requirements and does not perform any unintended functions. If the system is operational, only positive testing may be possible. Users' guides and operational profiles are instrumental in identifying and detailing the positive test cases and procedures. Failure mode analysis can be used for defining negative test cases and procedures.

However, in many instances negative or off-normal testing is not possible. Performing negative tests in an operational or nearly operational system has the potential to cause unexpected defects that place the system in an unsafe state or cause harm to equipment. Because of the potential for a hazard being exposed, negative testing works best for new systems that have not been placed into operations.

Perform Assessment of Vendor: The assessment of the supplier is probably the most rigorous and complete of the assessing quality approaches. It is basically an assessment of how the quality was built in by the vendor. While acceptance testing focuses on the quality of the software product, vendor assessment focuses on the quality of the software development process. It can be more than a document review. Interviews with key vendor staff supplement

the documentation produced from the development process. Observation or witness testing during the development testing activities can be key elements in this type of assessment.

This approach can apply to custom and commercial design and analysis tools software.

This approach is best performed through document reviews, interviews of vendor staff, and witness testing. However, any one of these can be done separately but most likely producing less confidence in the quality of the software than if two or more aspects were performed. This approach would not be applicable if the Building Quality into the Product, Assess Vendor Certification, Third Party Vendor Certification, or Accepting the Quality Based Upon Key Characteristics approaches are performed. However it can be used in conjunction with the Perform Model and Algorithm Validation and the Perform Acceptance Testing or Qualification Testing approaches.

Assess Vendor Certification: This approach requires vendor procurement contracts to include the quality standards and quality requirements the vendor must meet and for the vendor to certify they have met those requirements. The review of the vendor certification prior to use of the software would be all that is needed.

This is a simple assessment approach that applies to custom, configurable, commercial-off-the-shelf, and commercial design and analysis tools software.

Most likely this approach is mutually exclusive from the more active Perform Assessment of Vendor and Third Party Vendor Certification approaches. This approach can complement the Perform Model and Algorithm Validation, Perform Acceptance Testing or Qualification Testing, and the Accept the Quality Based Upon Key Characteristics approaches.

Third Party Vendor Certification: This approach allows for an independent party (neither the user nor the vendor) to assess the vendor against national standards criteria. These third party certifications can be for the quality management systems (QMS) such as ISO 9001, the software process such as SEI CMM, or product safety certification such as UL. The user/purchaser may use the certification to confirm that the vendor has applied the subject standards. The user/purchaser may specify certification by a third party in procurement documents when valid for this application of the software.

This approach may apply to custom, configurable, commercial-off-the-shelf, and commercial design and analysis tools software.

If this approach is used, Building Quality into the Product, Third Party Vendor Certification and Assess Vendor Certification approaches are not needed. When this Third Party Vendor Certification is used, Perform Model and Algorithm Validation, Perform Acceptance Testing or Qualification Testing, and the Accept the Quality Based Upon Key Characteristics approaches can be used to supplement this approach.

Accept the Quality Based Upon Key Characteristics: Just accepting the quality may seem like giving up on quality but there are certain circumstances where applying this approach is appropriate. Basically key characteristics for the vendor and/or product define the quality of the software such as wide spread application or a large customer base.

Software that is widely used (i.e., key characteristic) has a large user base that tends to promote proper correction of major defects and sharing of information to work around existing defects that have significant consequences. Thus the quality of the software can be indirectly derived or assumed to be at a certain acceptable level. Types of software in this area include the proprietary safety design codes and calculational software such as Excel.

This approach applies to commercial-off-the-shelf and commercial design and analysis tools software. It is not applicable for custom software. This approach is the foundation for accepting the quality levels for many software products and tools used in the development and operation of software systems. It provides the confidence level when using compilers, operating systems, code generators, and other software development tools.

Software Quality and Assessment Practices

Each of the approaches described include specific practices that can be performed to achieve the desired level of quality. The rigor of implementing these practices should be based upon the grade (level) of the software as referred to in the System Classification and Software Grading Levels section of this document. Table 3 is a sample template for mapping the practices to the software grade. This table or one similar is completed for each of the software types (i.e., custom, configurable, COTS, utility calculations, and commercial design and analysis tools). Level n is used in the event that software that is important to safety, which does not fit the level A or B criteria, is to be considered in the development of the DOE Order and guides.

The grouping and list of practices are derived from industry recognized standards and concepts. These include ASME NQA-1, IEEE 12207, NASA 8719.13B, ASQ SQE BOK, SEI CMM, and USAF Software Technology Support Center.

	Level A	Level B	Level n
Cross Life Cycle Practices			
Software Safety Analysis			
Software Risk Analysis			
Software Quality Assurance			
Software Reviews and Audits			
Software Project Planning			
Software Configuration Management			
Problem Reporting and Corrective Actions			
Measurements and Metrics			
Procurements & Vendor Management			
Vendor Assessments			
Vendor Certifications			
3rd Party Certifications			
Accept Based Upon Key Characteristics			
Training			
Life Cycle Practices			
Software Concept			
Software Requirements Analysis and Management			
Software Design			
Software Implementation			
Software Testing			
Developer Testing			
Acceptance Testing			
Model/Algo Validation			
Hand Calculations			
Software Product Build and Product Release			
Software Product Installation & Verification			
Software Operations			
Software Maintenance Activities			
Software Retirement			

Table 3. Sample Template of Practices

Conclusion

DOE is committed to operating high quality safety systems in its nuclear facilities. DOE strives to utilize appropriate industry standards for safety software quality and assessment practices rather than developing DOE specific practices. DOE's directive process will result in the generation of a Guide that will incorporate the template from Table 3 with recommended practices for each type of software application and each defined level. The DOE directives, an Order and a Guide, are to be completed by early 2005. The approaches described in this paper will be used to detail and finalize these directives.

References

1. Alain Abran and James W. Moore, Executive Editors. Guide to the Software Engineering Body of Knowledge. Trial Version. Los Alamitos, CA: IEEE Computer Society, May 2001.
2. ASME NQA-1-2000, Quality Assurance Requirements for Nuclear Facility Applications.
3. DoD 882D-2000, Department of Defense Standard Practice for System Safety, February 10, 2000.
4. DOE-STD-3009-94, Change Notice No. 2, DOE Standard, Preparation Guide for U.S. Department of Energy Nonreactor Nuclear Facility Documented Safety Analyses, April 2002.
5. DOE/RW AP-S1.1Q Revision 5 ICN 1, DOE Office of Civilian Radioactive Waste Management, QA Software Management.
6. DOE/RW-0333P, DOE Office of Civilian Radioactive Waste Management, Quality Assurance Requirements and Description.
7. IEEE/EIA 12207-1997, Standard for Information Technology, Software life cycle processes – Implementation considerations, April 1998.
8. NASA GB 8719.13B (Draft), NASA Software Safety Guidebook (Draft), February 2002.
9. NASA Std 8719.13B (Draft), Software Safety Standard, NASA Technical Standard (Draft), May 22, 2003.
10. WSRC 20-1 Rev 8, Westinghouse Savannah River Company, Software Quality Assurance, October 16, 2003.

Biography

Richard Lagdon, Director, Quality Assurance Programs, Office of Environment Safety and Health, US Department of Energy, 1000 Independence Avenue SW, Washington, DC 20585-0270, USA, telephone – (301) 903-4218, facsimile – (301) 903-4210, e-mail – chip.lagdon@eh.doe.gov. Richard Lagdon has extensive experience in nuclear facility operation, safety and engineering.

Debra Sparkman, Software Quality Engineer, Center for Application Development and Software Engineering, Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550, USA, telephone – (301) 903-6888, facsimile (301) 903-4210, e-mail – sparkman1@llnl.gov. Debra Sparkman is an ASQ certified software quality engineer, has extensive experience in scientific research, control systems, and database applications development and high integrity systems standards.

This work was performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48.