

Software Quality Assurance Plan (SQAP)

ROSE Software Project

CASC/Computing

Version 1.3.3

6/20/2019

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

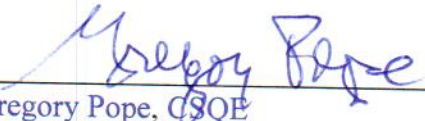
Auspices Statement

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Approvals

Approval signature(s) on this page indicate agreement to support and/or follow this plan for the software development effort.

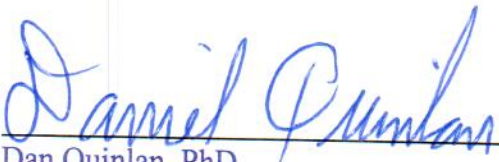
Prepared By:



Gregory Pope, CSQE
Product Realization, SQE

2/5/2019
Date

Reviewed By:



Dan Quinlan, PhD
Principal Researcher, Technical Director

7/9/2019
Date

Revision History

Document Version	Revision Date	Originator(s)	Revision Description
1.0	9/26/2016	G Pope	Initial draft
1.1	10/20/2016	G. Pope	Review edits
1.1	11/11/2016	G. Pope	Added Figure numbering
1.2	1/24/2017	G. Pope	Added ROSE review notes
1.3	7/2/2018	G. Pope	Update Org Chart
1.3.1	10/16/2018	G. Pope	Typos corrected from Ellen Hill review
1.3.2	2/15/2019	G. Pope	Release versioning method, update to org chart.
1.3.3	6/20/2019	G. Pope	Update Org Chart

Table of Contents

Approvals.....	iii
Revision History	iv
1 Purpose.....	1
2 Reference documents	1
3 Management.....	3
3.1 Organization	3
3.2 Tasks.....	4
3.3 Roles and responsibilities.....	7
3.4 Quality assurance estimated resources	7
4 Documentation.....	7
4.1 Purpose	7
4.2 Minimum documentation requirements	7
4.2.1 Software requirements description	8
4.2.2 Software design description.....	8
4.2.3 Verification and validation processes	10
4.2.4 Verification results report and validation results report	13
4.2.5 User documentation	13
4.2.6 Software configuration management activities.....	14
4.3 Other documentation	15
5 Standards, practices, conventions, and metrics.....	16
5.1 Purpose	16
5.2 Content	16
6 Software reviews.....	16
6.1 Purpose	17
6.2 Minimum requirements	17
6.2.1 Software specifications review	17
6.2.2 Design review	18
6.2.3 Managerial reviews	18
6.2.4 Post implementation review.....	19
6.2.5 Verification and validation plan review.....	19
6.2.6 Software configuration management plan review	19
6.2.7 Baseline configuration audit	19
6.2.8 Functional audit	20
6.2.9 Versioning.....	20
7 Test.....	20
8 Problem reporting and corrective action.....	20
9 Tools, techniques and methodologies	21
10 Media control	23
11 Supplier control.....	23
12 Records collection, maintenance, and retention	24
13 Training.....	25
14 Risk management.....	25
15 Glossary	26
16 SQAP change procedure and history	27
17 Software application retirement.....	27

Appendix A.	Implementation Form for ROSE Built Tools.....	28
Appendix B	Implementation Form for ROSE Core	37
Appendix C	Identified Software Effort Development Risks	47
Appendix D	Release Policy for ROSE and ROSE-based Tools.....	49

List of Tables

Table 1, Task Entrance and Exit Criteria.....	5
Table 2, Roles and Responsibilities	7
Table 3, Planned Reviews	17
Table 4, High-Level Tool Information	21
Table 5, Media Types and Control	23
Table 6, Training Requirements	25
Table 7, Acronym/Definitions	26

List of Figures

Figure 1 ROSE Project Organization.....	3
Figure 2 ROSE Development Lifecycle	5
Figure 3 ROSE Framework Architecture	10
Figure 4 Test Matrix Combinations.....	12
Figure 5 ROSE Verification and Validation Process.....	13
Figure 6 ROSE Configuration Management (CM) Process	15

1 Purpose

The purpose of this Software Quality Assurance Plan (SQAP) is to document the software quality assurance (SQA) approach that the application development team will follow. The SQA activities planned in this document meet the SQA requirements of its sponsors, stakeholders, and users. The SQAP and its cited documents provide the framework for implementing the software engineering activities and flowing down SQA requirements.

This plan applies to all software procured, acquired, or developed within the software effort's budget. For acquired software applications and/or components and new development for legacy code, only parts of this SQAP are applicable. It does not address work products produced by end users that utilize any of the effort's software. In addition to the tools listed in Table 4, High-Level Tool Information, this plan applies to the following software products:

- ROSE Software Framework and Special Purpose tools built using the ROSE Framework.

The software effort will follow a [IEEE](#) life-cycle model. See Section 3.2 for additional life-cycle details.

ROSE is an open source compiler infrastructure developed at Lawrence Livermore National Laboratory (LLNL), to build source-to-source program transformation and analysis tools for large-scale C (C89 and C98), C++ (C++98 and C++11), UPC, FORTRAN (77/95/2003), OpenMP, Java, Python and PHP applications. The primary goal of the ROSE project is to optimize applications within the U.S. Department of Energy (DOE), Department of Defense (DoD), other government agencies, industry partners, and academic institutions.

ROSE consists of a library (and set of associated tools) used by computer scientists to apply compiler techniques to source code to improve application performance, developer productivity, deeper understanding of the code, and automate porting to current and future platform architectures. ROSE also contains features to analyze binary code derived from the source languages to provide further analysis, insights, and assurance of the compiled codes integrity.

The ROSE Framework is a Research and Development tool suitable for scientific discovery when used by researchers and optimized for exploration and flexibility. The ROSE framework also supports building of more narrowly focused special purpose tools which are targeted for application developers who demand additional rigor and reliability in conducting their analysis, transformations, and optimizations. This Software Quality Assurance Plan addresses the software development processes for both the research and special purpose tools built using the ROSE frameworks.

2 Reference documents

The following are the applicable governing Federal Regulations and DOE Orders:

- [DOE O 414.1D Admin Chg 1](#), Quality Assurance
- [DOE O 200.1A](#), Information Technology

The IEEE Standard for Software Quality Assurance Plans ([IEEE Std 730™-2002](#)) was used as a guide in the development of this document to flow down the requirements identified in [RID-0118](#), Software Quality Assurance Consensus Standards for Quality Assurance Criteria, using the graded approach contained in [FRM-3104](#), SQA Required Practices for Non-830 Software. No claim to conformance with the standard is made.

The following institutional documents are directly applicable to software:

- [DES-0108](#), Non-830 Institutional Software Quality Assurance Program
- [PRO-0107](#), Software Risk Grading

In addition, the following institutional documents are used as applicable, unless superseded by programs, standards, policies, procedures, or processes required by the organization:

- [DES-0115](#), LLNL Quality Assurance Program
- [ES&H Manual Document 3.1](#), Nonnuclear Safety Basis Program
- [DES-0048](#), LLNL Assessment Program
- [PRO-0050](#), Internal Independent Assessment
- [PRO-0052](#), Management Self-Assessments
- [PRO-0053](#), Performing Management Observations & Inspections
- [ES&H Manual Document 2.2](#), LLNL Institutional-Wide Work Planning and Control Process
- [Personnel Policy and Procedures Manual](#), Section I, Employee Development
- [ES&H Manual Document 40.1](#), LLNL Training Program Manual
- [LTRAIN](#) (Livermore Training Records and Information Network) software application
- [DES-0206](#), Records Management
- [PRO-0207](#), Managing Records
- [LLNS Procurement Standard Practices Manual](#)
- [PRO-0097](#), Acceptance of Procured Items and Services
- [PRO-0098](#), Determining Procurement Quality Levels and Controls
- [PRO-0099](#), Supplier Evaluation
- [PRO-0100](#), Nonconformance of Procured Items and Services
- [SCM IQP 0010](#), Supplier Corrective Action Request Process
- [PRO-0042](#), Issues and Corrective Action Management

Additional documents referenced in this SQAP are:

ROSE User Manual: A Tool for Building Source-to-Source Translators Draft User Manual (version 0.9.7.73) <http://rosecompiler.org/>

3 Management

This section describes the software effort's organizational structure, its tasks, and its roles and responsibilities.

3.1 Organization

The organizational structure that influences and controls the quality of the software is described below. The roles and responsibilities for each item in the organizational description are defined in the table in Section 3.3.

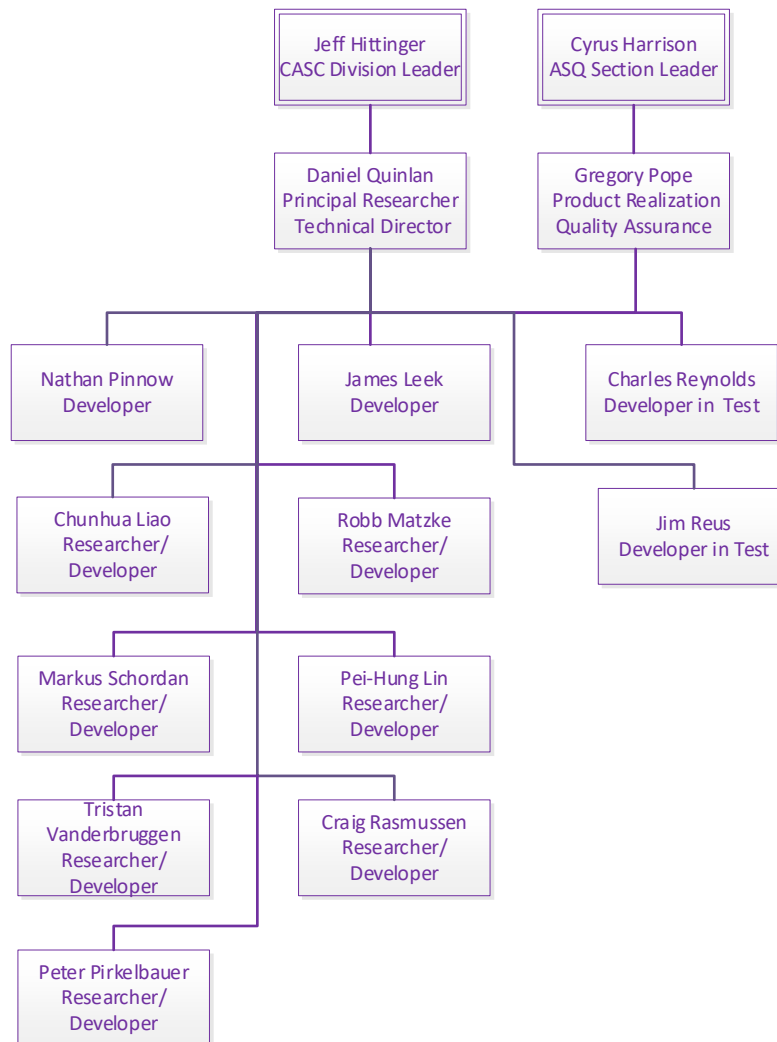


Figure 1 ROSE Project Organization

The ROSE Project Organization Chart is shown above in figure 1. The organization reflects the dual nature of the ROSE project as both a research group and a tool production group. Traditionally

ROSE has been a research project, started over 15 years ago and winning an R&D 100 award in 2009. ROSE research has led to the development of a powerful and flexible framework that can be used by researchers to develop custom transforms and analysis to advance computer science and software engineering. The ROSE framework is also used by the ROSE team to develop specialized tools for application developers, which requires a more disciplined approach. The ROSE organization reflects this by having tandem leadership for each of the two ROSE customer types. Dan Quinlan heads the research activities of the ROSE project as well as supporting ROSE specialized products. Gregory Pope heads the programmatic responsibilities for delivering specialized tools built using the ROSE framework. These specialized tools are subjected to additional levels of rigor specific to the customer's requirements.

Dan reports to the CASC (Center for Applied Scientific Computing) Division, primarily a research organization, and Greg reports to the ASQ (Applications, Simulation, and Quality) division an organization of application developers and software quality engineers. The need for the tandem management approach evolved out of the popularity of ROSE and the diversity of its customers. While the ROSE framework gives researchers almost unlimited flexibility, application programmers demand a more focused set of features that are targeted for development environments where deadlines and additional robustness are required. The ROSE project operates at two different Risk Levels (RLs). The ROSE research framework is a RL4 code targeted for researchers who demand flexibility and is subjected to rigorous automated testing. The specialized tools built using the ROSE framework are RL 3 codes, since their consequence of failure may have delaying impacts on customer software development schedules. These codes are subjected to an additional level of independent functional, documentation verification, and stress testing.

3.2 Tasks

The life-cycle followed on ROSE consists of an Agile/Iterative as shown in the process flow diagram below in figure 2. The stages/points in the process where quality checkpoints are located are highlighted in yellow. These are decision points for continuing to the next phase of the life-cycle.

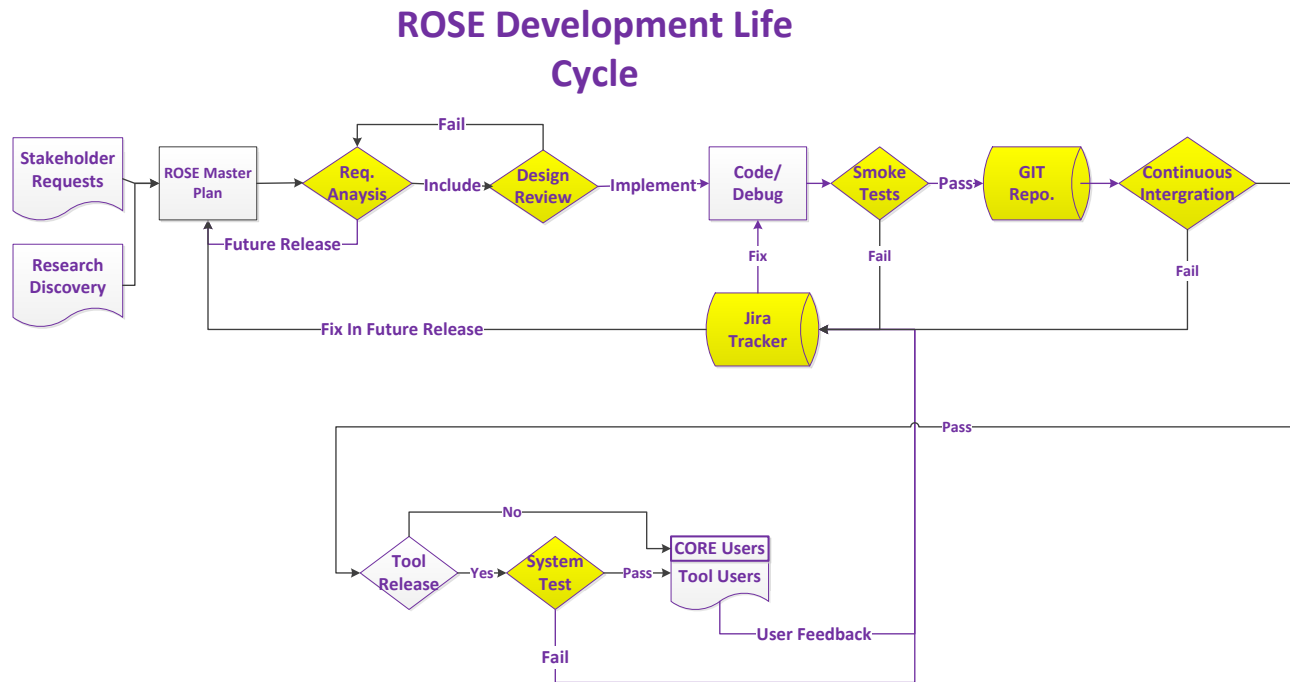


Figure 2 ROSE Development Lifecycle

The tasks to be performed, with entry and exit criteria for each task, are shown in Table 1. The following product and assurance tasks will be performed:

Table 1, Task Entrance and Exit Criteria

Tasks	For Life-cycle Phase Transitions	
	Entry Criteria	Exit Criteria
ROSE Master Plan	Stakeholder Request / Discovery	Candidate for Next Release
Requirements Analysis	Feature is Release Candidate	Feature Understood
Design Review	Feature Implementation Approach	Approved for Implementation
Smoke Tests	Code Ready to Commit	Pass Smoke Tests
GIT Repository	New / Modified code added	Code Under CM Control
Continuous Integration	New Build Rigorously Tested	All Tests Pass
System Tests	New Build is a Tool	All Tests Pass
JIRA Tracker	Failure of a Test or User Concern	Issued Logged and Tracked

The various ROSE projects are planned and tracked using the ROSE master plan shown in the ROSE development lifecycle. The master plan includes a schedule, milestones, deliverables, and assigned staff for each task. The master plan also acts as a clearing house for new ROSE projects, since requests for ROSE resources may come from many different stakeholders. Each request is evaluated for technical content and programmatic impact to other task commitments before being added to the ROSE master plan. This reduces the risk of ROSE resources being over committed. The Principal

Researcher and Product Realization Manager at a minimum will evaluate each new task request. The ROSE master schedule is maintained using the Microsoft Project tool.

Tasks on the ROSE master schedule will be tracked using JIRA task tickets assigned to the ROSE staff. A weekly scrum (a short project task review) is held to evaluate each ROSE staff member's task progress using the Kanban Board feature of JIRA. The Kanban Boards show current tasks, future tasks, and completed tasks for each staff member using the JIRA task tickets. During the weekly scrum, any obstacles to progress or task slippage are immediately identified so corrective actions can be implemented. Corrective actions may include staffing changes, de-scoping, make/buy, reuse, milestone realignment, outsourcing, feature prioritization changes, etc. During the week, any time a JIRA task ticket is updated an email is sent to other staff or stakeholders that may be impacted.

Project milestones on the master schedule are further broken down into feature lists needed to complete each milestone in the requirements analysis step of the life cycle. Requirement reviews may be conducted with the ROSE team members and stakeholders to clarify requirements. The feature lists are further divided into sprints (a three or four week period) containing the features to be added during that sprint, called the sprint backlog. This process is conducted in the design review life cycle step. A feature may be a capability or repair added to the ROSE core framework, or features added to build a tool being built with the ROSE framework. Further design review may also be conducted by assembling the developer team in cases where the feature implementation is not straight forward or has impact on multiple developers.

Next in the code/debug lifecycle step each feature and high level design is converted to C++ code by developers. The new code is then debugged and committed to the GIT repository which triggers a set of smoke tests to execute on the new source code. Smoke tests are a small but representative set of the total test suite that execute in 2 to 3 hours and verify basic functionality. If the smoke tests pass the new code joins the developer branch where a much larger set of regression tests begin execution in the Continuous Integration step of the ROSE life cycle.

If the project is an enhancement to the core ROSE framework passing of the suite of regression tests concludes successfully and passes the next feature in the sprint backlog is added. If the new code is written using the ROSE framework to build a tool for a user, the next feature can be started by developer(s) but when the sprint backlog is completed (or sooner if appropriate) an additional level of independent system testing is conducted by the developer in test and/or the independent tester. The independent testers install the ROSE tool when the current sprint is completed and test it from the user's point of view and assure the supplied documentation and tutorial problems work correctly. Issues discovered in regression testing, independent testing, or after release (see appendix D for release process) are tracked using the JIRA issue tracker.

In general, the ROSE project lifecycle attempts to identify as early as possible unanticipated technical challenges or other obstacles so that corrective action can be taken and stakeholder expectations can be updated weekly. This programmatic approach significantly reduces the risk of missing delivery release dates for tools built using the ROSE framework by the ROSE team.

3.3 Roles and responsibilities

The major responsibilities (including delegated responsibilities) assigned to each role are described in the following table.

Table 2, Roles and Responsibilities

Role	Responsibilities
Principal Researcher	Lead Researcher, Technical Direction
Product Realization SQE	Program Management, Software Quality
Researcher/ Developer	Computer Science Discovery, Code Development
Developer	Requirements Analysis, Design, Code Development
Developer Build, Test, Release	Build and Test Automation, CM, Release Management
Developer in Test	Test Design, Integration Test, System Test, User Support
Independent System Test	Independent System Testing, System Test Automation

3.4 Quality assurance estimated resources

The estimated resources and costs to implement this plan are:

Software quality assurance is institutionalized within the ROSE Organization, so to a large extent each team member is responsible for contributing to the SQA process. Some of the resources that are fully dedicated to SQA would include the Developer in Test and Independent System Tester since they do not write the prime ROSE code. Also, ROSE employs a number of platform resources for supporting continuous integration. The platforms support various distributions of UNIX, Apple MacOS, and Microsoft Windows operating systems. In addition, the ROSE Product Realization Manager is an ASQ certified software quality engineer and co-manages much of the non-research activities in the ROSE team.

4 Documentation

NOTE: Throughout this SQAP, the terms “document”, “documents”, and “documentation”, when used as a noun, may refer to a physical, paper-based or electronic document; artifact repository; database; web page; or other means employed by the software development team to capture the indicated information.

This section identifies document deliverables, as well as documents, that describe quality processes. The level of detail in the documents is commensurate with the complexity and significance of the system, the work environment, and the personnel proficiency and capability (education, training, experience). These documents are controlled documents and require review and approval.

4.1 Purpose

This lists the documents needed for development, verification and validation, use, and maintenance of the software.

4.2 Minimum documentation requirements

To provide assurance that the implementation of the software satisfies the technical requirements, the documentation content in the following subsections is required as a minimum. These documents

may be split into multiple documents or combined and/or included in other documents, as long as the noted content is adequately addressed.

4.2.1 Software requirements description

The software requirements are documented and tracked throughout the software effort's life-cycle, including those for any procured software.

Each requirement (stakeholder desired feature or bug fix) is uniquely identified and defined such that its achievement is capable of being objectively verified and validated. These requirements may be internally and/or externally (e.g., customer) generated. The tracking tool that is used for this purpose is the Confluence Wiki and the JIRA tracking tool. Customer requirements may also take the form of written and controlled documents or meeting notes.

The software requirements may address some or all of the following:

- Basic issues of functionality
- External interfaces (hardware, user, operator, other systems)
- Performance
- Design inputs
- Design constraints imposed on implementation
- Access control
- Security related (i.e., vulnerability protection and cyber security)
- Installation considerations
- Portability, correctness, maintainability, etc.

For detailed implemented and released requirements information, see the ROSE User Guide http://rosecompiler.org/ROSE_UserManual/ROSE-UserManual.pdf . For requirements being implemented in the current development branch see the ROSE Project Confluence for each project's release and sprint back logs.

NAP-24A additional considerations: The ROSE framework has been analyzed by the Klocwork static analyzer and security vulnerabilities fixed. The ROSE open source code is released with a signed hash and the released and signed ROSE download available to the public may not be modified by other than the ROSE project team. If a downloaded ROSE framework is changed by the public locally it will no longer match the signed version.

4.2.2 Software design description

The software design description shows how the software is structured to meet the requirements in the context of the intended operating environment. Parts of the design may be included with the requirements and some may be included in the computer program listings/code. Design elements are identified in such a manner as to enable verification and validation of the requirements.

Design components and subcomponents include items as listed below. Most requirements are translated from English directly to C++ source code by developers; some design information may be captured in code comment fields and code headers. The Doxygen tool is used to produce documentation from the C++ source comments. Requirements causing major impact may be

reviewed by the code team in a design review. Design information is contained in C++ comments and in the ROSE Documentation page at http://rosecompiler.org/?page_id=11 and includes all of some of the following:

- Class definitions
- Methods
- Process flow
- Computational sequences
- Algorithms
- Physical models
- Control flow
- Control logic
- States and state changes
- Data flow
- Data structures
- Inputs and output
- Exception/error handling, reporting, and logging
- Design features that ensure proper use of the software; such as access control features, help text, user interface design features, etc.
- Internal interfaces
- Relationships between the above

ROSE Framework Architecture is shown in figure 3 below:

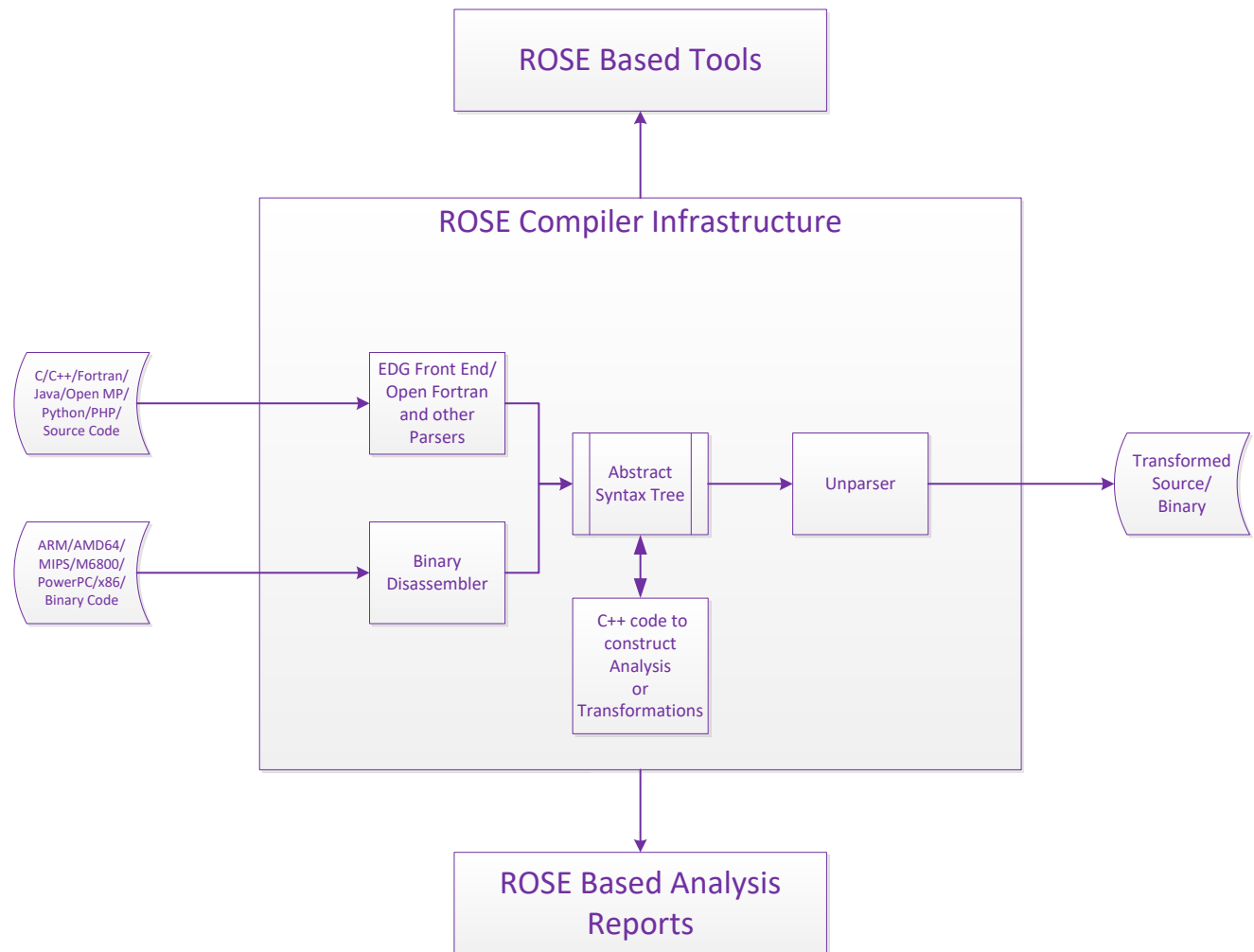


Figure 3 ROSE Framework Architecture

In addition the ROSE project can use tools such as Visustin, Klocwork, or McCabe IQ to create detailed calling diagrams, UML activity diagrams, or flow charts for the as built C++ code. These tools are available to the ROSE project through the Product Realization Manager and SQE.

4.2.3 Verification and validation processes

Verification and validation (V&V) processes are used to determine if developed software products conform to their requirements and whether the software products fulfill their intended use and user expectations. This may include analysis, evaluation, review, inspection, assessment, and testing of the software products and the processes that produced the products. Also, the software testing, validation, and verification processes apply when integrating purchased or customer-supplied software products into the developed product.

The extent of verification and the methods chosen are a function of the complexity of the software, the degree of standardization, and the similarity with previously proved software.

The ROSE V&V processes include the description below of the qualification and acceptance testing that is performed:

- There are multiple levels of testing for ROSE. ROSE uses a continuous integration (CI) method of testing; the testing process starts as soon as changes are committed to the developer's branch.
- Developers are responsible for conducting unit testing on new or modified code. When the developer is confident the new or modified code is ready for the next level of testing the code is committed to the GIT repository, where a set of smoke tests run on the new or modified code using a ROSE baseline on the developer's branch.
- If the smoke tests pass the new or modified code is committed to the main developer branch and continuous integration testing commences.
- Continuous integration testing at a minimum consists of Matrix Testing, Regression Testing, and Standards Testing.
- Orthogonal to the CI testing is the Matrix testing which checks the new code and baseline against numerous Compiler and Boost versions combinations. Additional testing combinations include operating systems and versions. The total number of combinations can reach into the tens of millions, hence matrix testing can take up to two weeks of continuous running. Figure 4 indicates the matrix test names and number of combinations:

Name	Count
assertions	1
boost	14
build	1
compiler	30
debug	1
dlib	9
doxygen	12
dwarf	1
edg	1
java	1
languages	3
magic	1
optimize	1
python	1
qt	1
readline	3
sqlite	1
warnings	1
wt	4
yaml	4
yices	2
Debian 6.0.10	
Debian 8	

RHEL 6.7	
RHEL 6.8	
RHEL 7.2	
RHEL 7.3	
Ubuntu 14.04	
Ubuntu 16.04	

Figure 4 Test Matrix Combinations

- Regression testing consists of running over 50,000 individual tests, many derived from prior errors found on ROSE over many years.
- Standards testing consist of running ROSE against the Plumb Hall compiler industry benchmark for C++ and C codes.
- If the software under test is a tool built from the ROSE framework then an additional level of independent testing is conducted to assure the tool meets the user's acceptance criteria, the documentation agrees with the code, and the installation process works correctly on the platform(s) of interest.
- The objective acceptance criteria for ROSE Framework development is passing of the smoke and CI tests. The objective acceptance criteria for ROSE built tools include the additional layer of user-based tests by independent testers.
- Each automated test case for smoke and CI tests contains an expected correct answer. The expected correct answer has been verified a priori by the developers. The criteria for the independent system testing are verification against the user guide, agreement with tutorial problems, and the stakeholder statement of work.
- Independent assessments of the ROSE project have been performed multiple times in FY 16 and will continue into FY 17 and beyond as part of the robustification effort. Peer reviews, design reviews, and requirement analysis reviews are conducted as part of the sprint planning and as needed.
- The workflow for the ROSE development process is automated using the Jenkins and JIRA tools to assure the lifecycle is followed.

This section of the SQAP describes the software verification and validation processes and tasks to be performed to satisfy the V&V requirements.

NAP-24A additional considerations:

- Test requirements are documented
- Someone other than the developer performs the testing of that developer's software
- In the case of independent testing, the test scripts are designed by someone other than the developer
- Software nonconformance's (anomalies) may occur in the software quality engineering processes or in the product. Software V&V processes (including testing) are used to identify these nonconformance's.

The ROSE Project V&V process is shown in figure 5:

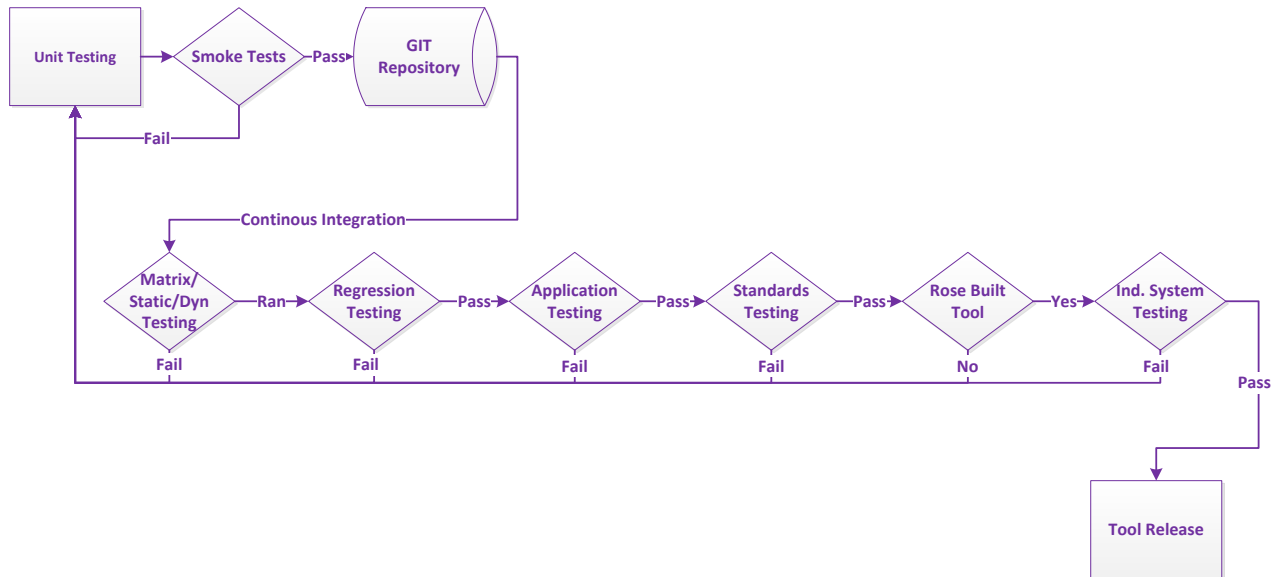


Figure 5 ROSE Verification and Validation Process

4.2.4 Verification results report and validation results report

Verification and validation results reports include a description and summary of the V&V activities conducted per Section 4.2.3, including, but not limited to:

- The results from all reviews (per Section 4.2.3 and Section 6)
- Issue reports in JIRA
- GIT commits, merges, diffs, and build logs
- Matrix Test Report (<https://hoosierfocus.com/matrix>)
- Test Results

The V&V results reports include the identification of the verifiers.

Test reports contain the test number, date of test, time of test, the expected result of running the test, the actual result of running the test, pass/fail.

The V&V results are approved prior to using the software. Using the software is defined as "by users after a formal release of the software". This approval of results signifies that the criteria in Section 4.2.3 have been met. The approval is done by the Principal Researcher prior to a formal tool release which goes to users (details of the release process are contained in the ROSE Confluence Wiki) .

4.2.5 User documentation

User documentation guides the users in installing, operating, managing, and maintaining software products. The following user documents will be provided, as applicable:

- A User Guide and Tutorial problems for the core ROSE framework.

- Tools built by the ROSE team using the ROSE framework will contain additional user documentation and tutorial problems.
- Installation instructions are contained on ROSE Confluence Wiki (http://rosecompiler.org/ROSE_HTML_Reference/group_installation.html)
- Additional information about ROSE, including a “how to” section is also available on the ROSE Confluence Wiki space. (http://rosecompiler.org/?page_id=11)

4.2.6 Software configuration management activities

The software configuration management activities are used to control the software and the development process.

Software configuration processes include:

- The GIT repository establishes the ROSE development branch, test branch, and released branches.
- The GIT repository controls configuration changes (including approving change requests, dispositioning of change requests, controlling/tracing changes throughout the life-cycle, verifying changes, backing out unsuccessful changes, and disposition of denied changes).
- Managing releases
- Disaster recovery: The contents of the ROSE GIT repository are backed up by Livermore Computing.
- Managing associated repositories to ensure reproducibility of configuration items and baselines. This includes backup storage and recovery procedure
- Managing formal documents for review, approval, revision, issuance, and ensuring that current versions are used.

The following lists the types of configuration items (CI) that will be maintained:

- Requirements documentation (existing requirements are contained in the User Guide, requirements for future version are tracked in JIRA).
- Design documentation is contained on the ROSE Confluence Wiki or as comments in the ROSE C++ source code.
- Design drawings are contained in the ROSE Confluence Wiki.
- Interface specifications are contained in the ROSE C++ source code and in the User Guide as descriptions of the ROSE commands.
- Quality process descriptions (This SQAP)
- Standards that were used to create items such as C++ v11
- ROSE source code
- Builds
- Build, execution, test, configuration, and release scripts and other automation mechanisms
- Build data
- Support software such as EDG, Boost, etc.
- Test plans
- Test cases and results

- Unit and coverage tests
- Maintenance and operating manuals/instructions
- Items needed to reinstate a previous baselined position in the life-cycle to once again move forward on the build cycle
- Additional items that are software effort specific CI types such as needed environments (e.g., dev, test, prod)

NAP-24A additional considerations:

- Configuration control is applied to design inputs, design calculations and analyses, design qualification, and design documents
- Testing is performed in known environments where the configuration is controlled

The ROSE CM process is shown below in figure 6:

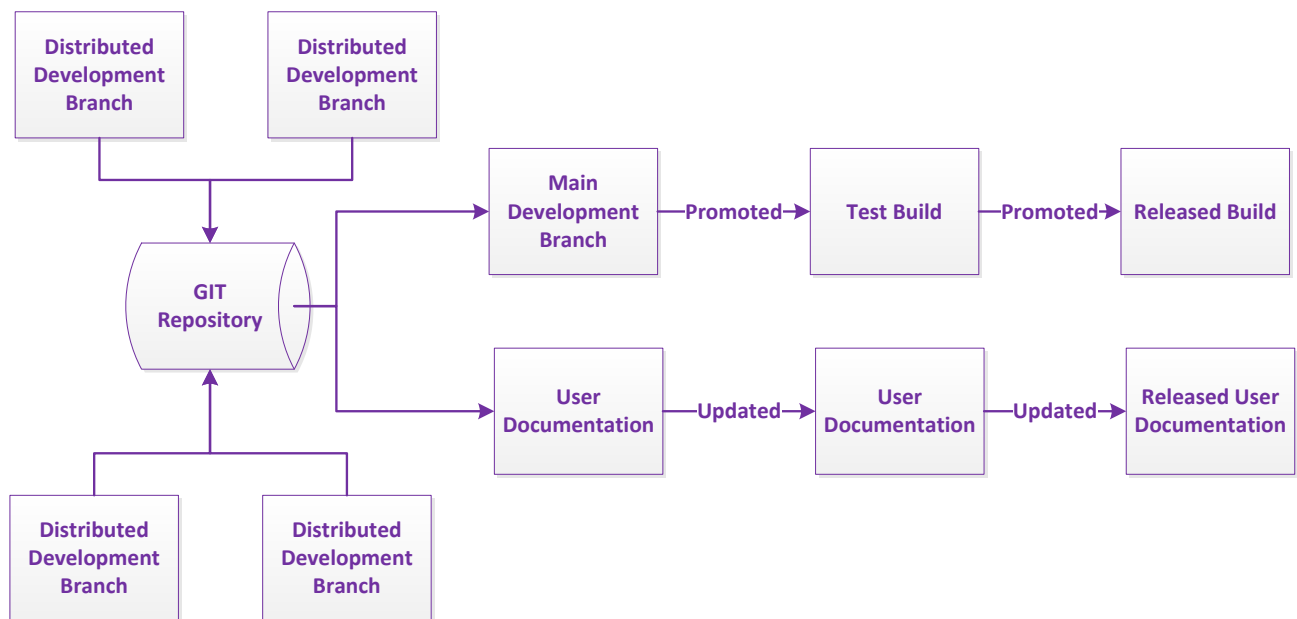


Figure 6 ROSE Configuration Management (CM) Process

4.3 Other documentation

Other documentation produced or referenced includes:

- White papers about ROSE applications
- Software development standards description (C++ standards, Open MP, etc.)
- Software engineering methods/procedures/tools description (GIT and JIRA documentation)
- Software project management plan and master schedule

5 Standards, practices, conventions, and metrics

5.1 Purpose

This section identifies the standards, practices, conventions, and metrics used to guide the development process and provide assurance that the quality processes are being followed and are effective.

5.2 Content

This SQAP is consistent with the standards flow down contained in [RID-0118](#), Software Quality Assurance Consensus Standards for Quality Assurance Criteria, and the SQA practices as contained in [FRM-3104](#), SQA Required Practices for Non-830 Software. In addition, the following software effort specific standards, practices, conventions, and metrics are being used:

- Applicable regulatory requirements are listed in section 2 of this SQAP
- LLNL supplied SQAP template, Doxygen, Confluence Wiki
- Compiler standards for FORTRAN, C++, C, Java, PHP, Open MP, Python
- Coding standards/conventions
- Variable and module naming conventions
- Commentary standards/conventions uses Doxygen
- Programming styles
- Inspection techniques
- Testing standards and practices are in section 4.2.3 of this SQAP
- Quality requirements are listed in this SQAP
- Quality metrics include:
 - Matrix testing results
 - Static analysis reports
 - Code count
 - Comment count
 - Code cyclomatic complexity

The Principal Researcher enforces the adherence to the aforementioned standards.

A latest copy of the ROSE Developer's Guide is separately maintained at:
http://rosecompiler.org/ROSE_DeveloperInstructions.pdf

Quality metrics are collected for testing during the Continuous Integration process. Additional code quality metrics are reported when a static analysis run is completed using the Klocwork tool.

No additional standards, practices, conventions, or metrics, other than those outlined elsewhere in this document, will be followed.

6 Software reviews

The types of review to be conducted during the life-cycle of this software effort include, as applicable (but are not limited to), managerial reviews, technical reviews, inspections, code

walkthroughs, configuration audits, and acquirer-supplier reviews/audits. These reviews typically occur in the Friday afternoon status meeting and are chaired by Justin Too.

Documentation of review comments and their disposition will be retained until they are incorporated into the updated software or associated document(s). Comments not incorporated and their disposition will be retained until the software is approved for use. See Section 12 of this plan for additional records retention requirements.

Review comments, action items, and notes from the status meetings are kept in the ROSE Confluence Wiki.

6.1 Purpose

Software reviews are conducted throughout the software effort's life-cycle to satisfy governing quality assurance and application-level requirements and to provide assurance that the product is ready to advance to the next life-cycle phase.

6.2 Minimum requirements

At a minimum, the following software reviews/audits will be conducted. Complementary reviews/audits may be combined.

Table 3, Planned Reviews

Review	Type of Review	Review Lead	Reviewer(s) and Roles
Software Specifications	Requirements	Principal Researcher	Developers/Users
Detailed Design	Peer Review	Principal Researcher	Developers/Users
Managerial Review(s)	Weekly Status	Realization Manager	Researchers/Developers
Post implementation	Post Mortem	Realization Manager	Developers/Users
SVVP or Section 4.2.3	Test Report	Principal Researcher	Developers/Users
SCMP or Section 4.2.6	Build Features	Principal Researcher	Developers/Users
Baseline configuration audit	Build Log	Principal Researcher	Developers/Users
Functional audit	Build Features	Principal Researcher	Developers/Users

Or: Include this detail in each of the following sections.>

6.2.1 Software specifications review

This review is held to assure the adequacy of the requirements prior to releasing the software for production use.

This review is conducted to assure the adequacy of the applicable requirements for the ROSE software. As shown in the ROSE Lifecycle, this review takes place to analyze the list of stakeholder desired features and to assure that they are adequately understood by the developers. Outcome of the review is to assign features to the next release and developers to work on those features. A second possible outcome of the review is to defer features for future releases or deprecate existing features that are no longer desired. Additional information may be sought from stakeholders if features are

not fully understood during this review. Some features may be straight forward enough that this level of review is not necessary or can be combined with the ROSE weekly staff meeting. .

NAP-24A additional consideration: New features to be added may also be accompanied by one or more new test cases to verify these features.

6.2.2 Design review

This review is held prior to releasing the software to:

- Verify the acceptability of the software design in satisfying the requirements
- Verify the design complies with applicable standards, references, regulations, policies, physical laws, and business rules
- Validate the design sequences of states and state changes using logic and data flows coupled with domain expertise, prototyping results, engineering principles, or other basis
- Assess the appropriateness of design methods and standards used
- Verify design elements are testable to objective acceptance criteria

Those involved in this review include individuals or groups who did not perform the work being reviewed. The design approval occurs when this review is deemed successful.

A Design Review may be called by the Principal Researcher, Realization Manager, or Developers when the implementation of a new feature is sufficiently complex or has multiple possible valid solutions or implementations. The design may be represented with pseudo-code, C++ or C code, UML, or flow charts. This optional review takes place after the requirements are analyzed. Typically, the review can be in a conference room or automated using Jabber.

NAP-24A additional consideration:

- Verify the adequacy of designs prior to final software acceptance, including:
 - Applicable standards are used
 - Performance against standards are adequately verified and validated
 - Qualification methods are adequate for contributing software

6.2.3 Managerial reviews

Managerial reviews are held periodically to assess the execution of all of the actions and the items identified in this SQAP. Management reviews are carried out by, or on behalf of, the management personnel having direct responsibility for the system. The reviews may be formal or informal.

Examples of formal managerial reviews that may be performed include, but are not limited to:

- [PRO-0053](#), Performing Management Observations & Inspections
- [PRO-0052](#), Management Self-Assessments
- [PRO-0050](#), Internal Independent Assessments, are an option for high risk software. These are performed by the MAS SQA office by request.

Formal reviews are held at management's discretion to assess the execution of actions and items identified in this Software Quality Assurance Plan. The results of these reviews and the metrics identified in Section 5.2 of this document form the basis for quality improvement per [DES-0115](#), LLNL Quality Assurance Program, Section 4.2.3.

Managerial reviews are held weekly or more frequently in person or using the Jabber communicator tool. Documentation of items to be discussed is distributed prior to the staff meeting. The Product Realization Manager runs the meeting as the developers discuss progress/obstacles. Anomalies, risks, and show stopper level defects may also be discussed and inserted as tasks or bugs into the JIRA tracker tool.

NAP-24A additional considerations: As required in NAP-24A, Attachment 2, Section 3.1.1 process improvements are captured as tasks in the JIRA tracking tool.

6.2.4 Post implementation review

This review is held at the conclusion of the software effort to assess the development activities and provide recommendations for appropriate actions, such as process improvements. The outcome of this review includes things that went well and should be continued, as well as those that didn't go well with suggestions for improvement.

This review is held after a major release. It is post mortem review to assess what went well and what needs to be improved for the next release. This meeting is chaired by the Product Realization Manager and attended by developers, testers, and optionally users. This review may also be combined with a weekly staff meeting.

6.2.5 Verification and validation plan review

The Verification and Validation plans are contained in section 4.2.3 of this SQAP. The Verification and Validation process is periodically reviewed as part of the weekly staff meeting. The Jenkins tool controls the Continuous Integration of new or modified code. At the time the new or modified code is checked in the GIT repository the test suite begins running thousands of automated tests. A test report is generated recording the pass/fail status of all tests run.

6.2.6 Software configuration management plan review

This review is held to evaluate the adequacy and completeness of the configuration management methods defined in section 4.2.6 of this SQAP and whether they are in accordance with the software effort's risk grading and level of development control. This review is conducted as a topic in the weekly staff meeting and includes confirming that all source code, tests, test reports, User Guides are contained in the GIT repository and version controlled. Other documentation for ROSE is kept on the Confluence Wiki.

6.2.7 Baseline configuration audit

This audit is performed prior to acceptance testing (major release) to verify that the baseline is complete, and that the system is ready for acceptance testing. It is done by examining the build

scripts and build logs to assure all of the required software and supporting libraries and versions are contained in the build.

6.2.8 Functional audit

This audit is held prior to the software delivery to verify that all required features are in the release and have passed on the release test report.

6.2.9 Versioning

Version numbering for the release of ROSE follows an w.x.y.z. format. Therefore, z increments for a release with bug fixes only, y increments when a release contains feature improvements, x increments when the release contains a major new feature or features. Lastly w increments for major public releases.

7 Test

All testing is covered in Section 4.2.3 of this plan and its referenced documents.

Testing is to be completed and reviewed prior to the software being released to users.

8 Problem reporting and corrective action

The automated problem reporting tool JIRA is used by the ROSE team to implement the change control process.

The change control processes include:

- Approving change requests,
- Dispositioning of change requests,
- Controlling/tracing changes through the life-cycle,
- Verifying changes,
- Backing out unsuccessful changes,
- Disposition of denied changes

.

NAP-24A additional considerations:

Discovery and removal of code defects is one of the purposes of the Verification and Validation process described in section 4.2.3 of this SQAP. Minimizing defects delivered to users is one way to improve the quality of the user's experience. Logging the discovered defects into JIRA helps assure they will be repaired and retested. However quality also includes the defect prevention and that the code contains the user desired features. On the prevention side requirement reviews may generate a problem report if the desired feature description is not adequate or conflicts with other features. Requested features themselves can also be entered and tracked by the JIRA tool. Developers are responsible for developing their own unit tests and assuring a peer review is conducted prior to committing their new code to the

development branch. Additional tools are available to conduct static analysis using Klocwork.

The organizational responsibilities for implementing this process are covered in Table 2, Roles and Responsibilities, in Section 3.3 of this SQAP.

NAP-24A additional considerations:

- The qualification of the person evaluating nonconformance and authorized to perform disposition is the Principal Researcher who has been with the code since the beginning of the project and has had close contact with most ROSE code users and is a ROSE code user himself.
- The Principal Researcher has competence in the knowledge domain of the ROSE software, and therefore understands fully the requirements, and has access to pertinent background information if needed.
- The Principal Researcher determines non-compliance of stakeholder requests or implemented features.

If problems or issues are reported which need to be tracked at the institutional level, [PRO-0042](#), Issues and Corrective Action Management, will be followed.

9 Tools, techniques and methodologies

This section identifies the software tools, techniques, and methods used to support SQA processes, including software configuration management and V&V. See Table 4, High-Level Tool Information, for a list.

Table 4, High-Level Tool Information

Tool/Technique/Method Name	Type	Intended use / Conditions for use	Applicability / Usage limitations
Project planning	Tool	MS Project, Atlassian Portfolio	Planning and tracking
Compilers	Tool	Intel, Gnu, Clang	Daily
IDEs	Tool	Eclipse	Daily
Requirements management	Tool	GIT, Confluence Wiki	Daily
Configuration management	Tool	GIT	Daily
Source code control	Tool	GIT	Daily
Issue/bug tracking	Tool	JIRA	Daily
Test automation	Tool	Jenkins	Daily
Dynamic analysis	Tool	Intel Coverage	Periodic Use
Static analysis	Tool	Klocwork	Periodic Use
Lower level system software layers	Tool	Visustin, McCabe IQ	Periodic Use
System utilities	Tool	Boost	
Inspections	Technique	Klocwork	Periodic Use
Agile methods	Method	Kan Ban , Atlassian	Daily

Tool/Technique/Method Name	Type	Intended use / Conditions for use	Applicability / Usage limitations
		Portfolio, Scrum	
Technical literature comparison	Method	Published Journals	Periodic Use

The following is additional information about tool vendors:

- Confluence
- Wiki
- Atlassian
- Contact information, including:
 - 1098 Harrison St, San Francisco, CA 94103
 - <https://www.atlassian.com/why-wiki-collaboration-software>
 - <https://www.atlassian.com/resources/support>
 - (415) 701-1110
- Site License
- Annual renewal
- Software Development Management

- JIRA
- Defect, Task, Requirement Tracker
-
- Atlassian
- Contact information:
 - 1098 Harrison St, San Francisco, CA 94103
 - <https://www.atlassian.com/why-wiki-collaboration-software>
 - <https://www.atlassian.com/resources/support>
 - (415) 701-1110
- Site License
- Annual renewal

- GIT
- Version Control System
- Open Source
- Contact information:
 - <https://github.com/git/git-scm.com>
- Open source

- Eclipse
- Integrated Development Environment (IDE)
- Open source
- Contact information, including:

- <http://www.eclipse.org/downloads/packages/eclipse-ide-java-developers/marsrEmail>
address
- Cisco Jabber
- Unified Communication
- Cisco
- Contact information, including:
 - <http://www.cisco.com/c/en/us/products/index.html>
 - (800) 553-6387
- Site license
- Annual

10 Media control

The software effort deliverables, as well as items provided through acquisition, will use the following media types and be controlled as indicated below in Table 5, Media Types and Control. The primary means for dealing with damage (i.e., corruption) is to back-up the software.

Media is protected by being on a private lab network, requiring OUN, badge smart card, and PIN to access the network. In addition, permissions must be established to access the Confluence Wiki pages for the ROSE pages which contain the download links to the media.

Table 5, Media Types and Control

Media Type	Control Methods
Electronic Application Files	Electronic signing

11 Supplier control

For procured/acquired software, institutional governing requirements and procedures will be followed. The relevant documents are listed below. They contain information to assure the supplier receives adequate and complete requirements and the product is suitable for its intended use (i.e., it meets established requirements).

- [LLNS Procurement Standard Practices Manual](#)
- [PRO-0097](#), Acceptance of Procured Items and Services
- [PRO-0098](#), Determining Procurement Quality Levels and Controls
- [PRO-0099](#), Supplier Evaluation
- [PRO-0100](#), Nonconformance of Procured Items and Services
- [SCM IQP 0010](#), Supplier Corrective Action Request Process

All procured software has been acquired in accordance with the LLNL approved procurement systems.

12 Records collection, maintenance, and retention

Institutional governing documents and procedures, listed below, are followed for managing documents and records. All controlled document versions are managed as records. The governing documents specify how the records are to be collected, stored, safeguarded, and maintained and what timeframe for retention.

- [DES-0115](#), LLNL Quality Assurance Program
- [DES-0206](#), Records Management
- [PRO-0207](#), Managing Records

The following will be retained for at least 10 years from the date of this document:

- 06-00-040 Quality Assurance Program Files
- 01-00-003 Policies, Procedures & Guidance Files (for operations)
- 01-00-013 In-House Code Repository

The following documents will be collected, maintained, and retained in the appropriate software configuration management repository:

- Software Quality Assurance Plan (In GIT)
- Software requirements descriptions (In JIRA and Confluence Wiki)
- Software design description (In Confluence Wiki and source code comments)
- Verification and Validation Plan (In SQAP)
- Verification and validation results report(s) (In JIRA)
- Individual V&V activities results (In JIRA)
- Software Configuration Management Plan (In SQAP)
- User and developer documentation (In Confluence Wiki)

Any procurement related documentation will be managed through the Procurement organization.

In addition, the following will be maintained as quality records:

- Test results
- Anomaly records [in JIRA](#)
- Software change requests [in JIRA](#)

The Principal Researcher determines the methods for reviewing and approving quality records.

NAP-24A additional considerations: Use of software development tools such as JIRA, Jenkins, GIT, and the Confluence Wiki produce records and logs to provide evidence of software development process activities. These records and logs are maintained on the tools for a period of time to be determined by the Principal Researcher or their designee and/or as required to support audits as determined by the Product Realization manager. .

13 Training

All developers, testers, and evaluators are hired with commensurate skills, knowledge, and abilities for the job and are subject to the institutional training requirements as specified in the [Personnel Policy and Procedures Manual](#), Section I, Employee Development and the [ES&H Manual Document 40.1](#), LLNL Training Program Manual. Software effort-specific training requirements are determined by the Software Effort Lead consistent with [DES-0115](#), LLNL Quality Assurance Program, Section 4.2.2. Some training may be accomplished via on-the-job experience. Formal in-house training is recorded in the institutional tool, Livermore Training Records and Information Network ([LTRAIN](#)). Additional training requirements are specified in The ROSE development team members have formal academic education in computer science as well as many years of relevant experience in scientific software. Additional training requirements are determined by the functional management for appropriate levels of security and safety training requirements as documented by LTRAIN. Completion of informal on-the-job training is captured (e.g., email, memo-to-file, training checklist) and approved (e.g., mentor's/manager's signature) for future auditing purposes.

NAP-24A additional consideration: evidence of training is tracked in the LTRAIN tool. Training is continuing to maintain proficiency.

Table 6, Training Requirements.

The ROSE development team members have formal academic education in computer science as well as many years of relevant experience in scientific software. Additional training requirements are determined by the functional management for appropriate levels of security and safety training requirements as documented by LTRAIN. Completion of informal on-the-job training is captured (e.g., email, memo-to-file, training checklist) and approved (e.g., mentor's/manager's signature) for future auditing purposes.

NAP-24A additional consideration: evidence of training is tracked in the LTRAIN tool. Training is continuing to maintain proficiency.

Table 6, Training Requirements

Training	Target Participants
CA0750-W, Software Quality Assurance Awareness	Team leader, developers, testers, and evaluators of the software

14 Risk management

Risks associated with the actual development effort such as resource availability and funding shortfalls, are managed according to the ROSE Master plan.

Risks are identified at the ROSE weekly staff meeting. Risks that cannot be mitigated immediately are added to the JIRA tracker, prioritized, and assigned to an ROSE team member by the Principal Researcher for mitigation implementation.

NAP-24A additional considerations: Software security related risks as well as structural issues are identified using the Klocwork static analyzer. The areas of concern in the code will be repaired and retested to remove important code security vulnerabilities and structural issues.

The application is software risk graded according to [PRO-0107](#), Software Risk Grading. The Process/Development Environment report, which is part of the software risk grading process, gives a high-level evaluation of possible risks during the development life-cycle.

15 Glossary

Table 7, Acronym/Definitions

Acronym/Term	Description
AST	Abstract Syntax Tree
Admin	Administrative
Anomaly	An observation of an off normal or unexpected event/behavior. These may be process and/or product related. Other terms commonly used include: issue, bug, and defect.
CD	Compact Disk
Chg	Change
CI	Continuous Integration
DC	Development Control
DES	Description document
DOE	Department of Energy
DOE O	DOE Order
DVD	Digital Versatile Disk
EDG	Vendor that supplies C,C++ compiler front ends
ES&H	Environment, Safety and Health
FRM	Form
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers, Inc.
LLC	Limited Liability Company
LLNL	Lawrence Livermore National Laboratory
LLNS	Lawrence Livermore National Security, LLC
LTRAIN	Livermore Training Records and Information Network
MAS	Management Assurance System Organization
Non-830 Software	Software that is not Nuclear Safety Software
PRO	Procedure
Rev	Revision
RID	Requirements Interpretation Document
RL	Risk Level
SCM IQP	Supply Chain Management Internal Quality Procedure
SCMP	Software Configuration Management Plan
SQA	Software Quality Assurance
SQAP	Software Quality Assurance Plan
SQL	Structured Query Language
Std	Standard

SVVP	Software Verification and Validation Plan
Url	Uniform Resource Locator
USB	Universal Serial Bus
V&V	Verification and Validation

16 SQAP change procedure and history

This SQAP may evolve over time and/or may be incrementally completed. Some work products may be separate documents or eventually incorporated into this document. Changes to this plan are managed using the change control procedures per Section 4.2.6 ([or within a separate Software Configuration Management Plan](#)). The revision history is maintained at the beginning of this document.

17 Software application retirement

Upon retirement, support for the software product will be terminated, download availability terminated from the ROSE site, and notification posted on the ROSE Confluence Wiki that the ROSE version or tool is no longer to be used.

Appendix A. Implementation Form for ROSE Built Tools

Risk Level 3, Non-830 Software, Major Development Control

Software: [ROSE](#)
Date: [9/26/2016](#)
Author: [Gregory Pope](#)

This form documents the software quality assurance (SQA) activities and implementation plans for Risk Level 3 practices for Non-830 Software when there is major development control. A software risk grading was performed in accordance with LLNL Institutional Software Quality Assurance Program for Non-830 Software¹ requirements implementing DOE O 414.1D Admin Chg 1 and other LLNS contract requirements.

The flow down of SQA requirements to consensus standards is documented in RID-0118-00, *Software Quality Assurance Consensus Standards for Quality Assurance Criteria*. RID-0118 references the following IEEE (Institute of Electrical and Electronics Engineers, Inc.) standards:

IEEE Std 730TM-2002, *IEEE Standard for Quality Assurance Plans*

IEEE Std 828TM-2012, *IEEE Standard for Configuration Management in Systems and Software Engineering*

IEEE Std 1012TM-2012, *IEEE Standard for System and Software Verification and Validation*

Document templates have been created for flowing down the SQA requirements. These templates may be downloaded from the [SWING](#) wiki. The use of the templates is optional and may be used to ensure requirement flow down.

Documents	Templates
Software Quality Assurance Plan (SQAP)	Non-830 Software SQAP-3M

¹ DES-0108-00, *Non-830 Institutional Software Quality Assurance Program* and FRM-3104, *SQA Required Practices for Non-830 Software*

Software Configuration Management Plan (SCMP)	Non-830 Software SCMP-3M
Software Verification and Validation Plan (SVVP)	Non-830 Software SVVP-3M

The table on the following pages details the required plans and products. Each practice has an indicator of the level to which the practice is to be implemented. These levels are:

- X Indicates the practice is required and fully implemented.
- * Indicates the practice is required, but may be partially implemented with written justification.
- O Indicates the practice is recommended, but optional.

Cross-references to the IEEE requirements that are to be used to implement the practice are indicated in “[]” for each practice. For example [SQAP S1, S3 (all)] refers to sections 1 and 3 (with all of its subsections) of IEEE 730-2002.

This Implementation form applies to tools built for customers using the ROSE framework

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 1: Software Project Management and Quality Planning				
Prepare, review, and approve a Software Quality Assurance Plan (SQAP) using the institutional template specific to the risk grading.	O		Software Quality Assurance Plan to be written to accommodate work for others and DOE	Complies
Plan and manage project activities and commitments (including organizational structure/interfaces; roles and responsibilities / authority; schedule; budget; and resources). [SQAP S4.3.1, S4.3.2]	*		ROSE master plan is done using MS Project and includes staffing, tasks, sub tasks, s schedule, budget, resources.	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Identify, define, plan, and track software quality assurance activities, including software development methodology, configuration management, verification and validation, management reviews, and testing. [SQAP S4.4.2.1, 2, 3, 6; S4.5.1, 2; S4.6.1; S4.6.2.8; S4.7; S4.8; S4.9; S4.11]	X		Covered in ROSE Software Quality Assurance Plan	Complies
Determine applicable regulatory requirements. [SQAP S4.5.1, 2]	X		Covered in ROSE Software Quality Assurance Plan section 2	Complies
Identify formal documents. [SQAP S4.4.2.1, 2, 3, 6]	X		Covered in ROSE Software Quality Assurance Plan section 4.2 and 4.3	Complies
Identify records of evidence to be produced. [SQAP S4.12]	X		Covered in ROSE Software Quality Assurance Plan section 12	Complies
Select and utilize additional standards, as needed. [SQAP S4.5.1, S4.5.2; SVVP Table 1c S9.3(1) a) 2), 7)]	*		Covered in ROSE Software Quality Assurance Plan section 5.	Complies
SQA Work Activity 2: Software Risk Management	This work activity is focused on project and software development risks. See SQA Work Activity 7 for safety hazards.			
Identify, assess, track and manage software risks, including development, implementation and usage. [SQAP S4.14]	*		Covered in ROSE Software Quality Assurance Plan, tracked in JIRA Tool	Complies
Complete a Process/Development Environment Risk Report in the Software Risk Grading Tool following PRO-0107 , <i>Software Risk Grading</i> . [PRO-0107]	X		Covered in ROSE Software Quality Assurance Plan and Risk grading tool.	

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 3: Software Configuration Management	Institutional documents are managed via DES-0032 , <i>Management of Institutional Documents</i> , and PRO-0033 , <i>Developing, Reviewing, and Approving Institutional Documents</i> .			
Prepare, review, and approve a Software Configuration Management Plan (SCMP) using the institutional template specific to the risk grading.	O		N/A	N/A
Plan, document, approve, and implement a software configuration management process (including criteria for identification of configuration items, configuration change control, and release version control). [SQAP S4.4.2.6, S4.8; SCMP S9.2.4.1, S10.2.3]	*		Contained in the ROSE SQAP section 4.2.6	Complies
Develop and use processes for document review, approval, revision, and issuance; ensure usage of current applicable documents. [SCMP S9.2.4.1, S10.2.3]	X		Contained in the ROSE SQAP section 6	Complies
Develop and use processes for records review, approval, storage, and retrieval. [SQAP S4.12]	X		Contained in the ROSE SQAP section 12	Complies
Plan and implement processes to identify and configuration control the critical characteristics of the operating environment, if any. [SQAP S4.4.2.6]	*		Contained in the ROSE SQAP section 6	Complies
Implement and maintain SCM repositories. [SQAP S4.10; SCMP S8.2.5.8, S8.2.5.10]	X		Contained in the ROSE SQAP section 6, using GIT	Complies
Plan and implement disaster recovery processes. [SQAP S4.10; SCMP S8.2.5.10]	X		Contained in the ROSE SQAP section 4.2.6 and	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Prevent usage upon retirement, as needed. [SCMP S14.2.8]	O		Contained in the ROSE SQAP section 12	Complies
SQA Work Activity 4: Procurement and Supplier Management	SQA follows the standard Procurement processes. Note that the Risk Level grading is for the purchased/acquired software, which may be different than the Risk Level grading for the associated software development effort.			
Follow institutional procurement processes for evaluating and selecting suppliers. [POAP 1502]	X		Contained in the ROSE SQAP section 11	Complies
Follow institutional procurement processes for supplier corrective actions. [SCM IQP 0010]	X		Contained in the ROSE SQAP section 11	Complies
Follow institutional procurement process for specifying requirements (technical and quality) and acceptance criteria for software (products and tools). [SQAP S4.11; Supply Chain Management Standard Practice 46.1]	X		Contained in the ROSE SQAP section 11	Complies
Qualify procured and otherwise acquired software for use. [PRO-0097, PRO-0100]	X		Contained in the ROSE SQAP section 11	Complies
SQA Work Activity 5: Software Requirements Identification and Management				
Identify (to the extent known), document, review, approve and update requirements. [SQAP S4.4.2.1, S4.6.2.1]	X		Contained in the ROSE SQAP section 4.2.1	Complies
Verify that applicable requirements were incorporated into the design. [SQAP S4.6.2.3; SVVP Table 1c S9.3(1) a) 2)]	X		Contained in the ROSE SQAP section 4.2.1	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Identify, document, and track security requirements (e.g., vulnerability protection and cyber - security) commensurate with the risk from unauthorized access or use. [SQAP S4.4.2.1, S4.14]	O		Contained in the ROSE SQAP section 4.2.1	Complies
SQA Work Activity 6: Software Design and Implementation				
Identify (to the extent known), document, review and approve the software design, including software and system interfaces. <i>(Note: This documentation may be combined with the requirements documentation and/or captured via source code comments.)</i> [SQAP S4.4.2.2, S4.6.2.3; SCMP S12.2.1, S12.2.2; SVVP Table 1c S9.3(1) a) 3), 7)]	X		Contained in the ROSE SQAP section 4.2.2	Complies
Identify and utilize coding standards and/or conventions. [SQAP S4.5.1, S4.5.2, S4.9; SVVP Table 1c S9.3(1) a) 2), 7)]	*		Contained in the ROSE SQAP section 4.2.2	Complies
Include design features to ensure proper use such as access control, help text, user interface design, as needed. [SQAP S4.4.2.2]	*		Contained in the ROSE SQAP section 4.2.2	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 7: Software Safety	This work activity only applies to software applications with potential ES&H consequences.			
Review nonnuclear safety basis documents, if any, for mitigation strategies (i.e., controls) for software components that have potential ES&H consequences if a fault occurs. [ES&H Manual Document 3.1, Nonnuclear Safety Basis Program; SQAP S4.14]	*		N/A	N/A
Perform reliability and usability analyses as needed.	O		N/A	N/A
Include designation for intended usage and users.	O		N/A	N/A
SQA Work Activity 8: Software Verification and Validation	The V&V activities are conducted in connection with all of the work activities.			
Prepare, review, and approve a Software Verification and Validation Plan (SVVP) using the institutional template specific to the risk grading.	O		N/A	N/A
Plan and perform Verification and Validation (V&V) (including, as applicable, unit, integration, system, installation and acceptance testing; ad hoc testing; audits; inspections; and walkthroughs and desk checks). [SQAP S4.4.2.3, S4.4.2.4, S4.6.1, S4.6.2.1 S4.6.2.3, S4.6.2.8, S4.7; SVVP Table 1c S9.6(1) a) 1), 2), 5), 6); Table 1c S9.6(1) b); Table 1c S9.7(2) a) 1), 2), 5); Table 1c S9.7(2) b), c)]	*		Contained in the ROSE SQAP section 4.2.3	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Plan and perform peer reviews (including inspections, walkthroughs and desk checks). [SQAP S4.4.2.4, S4.6.1, S4.6.2.1 S4.6.2.3, S4.6.2.8]	✱		Contained in the ROSE SQAP section 4.2.3	Complies
Test implemented changes for correctness and to ensure no breakage occurred. [SQAP S4.4.2.4, S4.7; SVVP Table 1c S9.6(1) a) 1), 2) 5) 6); Table 1c S9.6(1) b); Table 1c S9.7(2) a) 1), 2), 5); Table 1c S9.7(2) b), c)]	✱		Contained in the ROSE SQAP section 4.2.3	Complies
Ensure design reviewer participation includes adequate independence. [SQAP S4.6.1]	X		Contained in the ROSE SQAP section 4.2.3	Complies (Tools built with ROSE framework use regression tests written by all team members and then independent systems testers)
Verify and validate that the software and design satisfies the requirements using objective acceptance criteria for the components. [SVVP Table 1c S9.3(1) a) 1); Table 1c S9/3(1) f) 2)]	✱		Contained in the ROSE SQAP section 4.2.3	Complies
Approve V&V results prior to using the software. [SQAP S4.3.2]	✱		Contained in the ROSE SQAP section 4.2.3	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 9: Problem Reporting and Corrective Action				
Plan, document, and implement methods for reporting/documenting, evaluating, approving, and correcting software problems, including roles and responsibilities. [SQAP S4.8; SCMP S9.2.5.1, S9.2.5.2; SVVP Table 1c S9.10(2)]	X		Contained in the ROSE SQAP section 8	Complies
Implement process improvement activities (including problem prevention and soliciting and analyzing quality feedback). [SQAP S4.6.2.8, S4.6.2.10, S4.8; SCMP S7.2.2.2, S7.2.2.4; SVVP Table 1a S7.1(6) c), d), e), f); Table 1c S9.10(2)]	X		Contained in the ROSE SQAP section 8	Complies
Create release notes, as needed.	O		Contained in the ROSE SQAP section 8	Complies
SQA Work Activity 10: Training of Personnel in the Design, Development, Use and Evaluation of Software	SQA utilizes the institutional training and qualification processes, which includes LTRAIN . SQA also utilizes the job posting process that specifies qualifications.			
Select and train team members. [LTRAIN ; SQAP S4.13]	*		Contained in the ROSE SQAP section 13	Complies
Development team leaders, developers, testers, and evaluators to complete CA0750-W, Software Quality Assurance Awareness. [SQAP S4.13]	X		Contained in the ROSE SQAP section 13	GAP – Schedule LTRAIN course.
Produce user training materials, as required by the customer. [SQAP S4.4.2.5]	*		Contained in the ROSE SQAP section 13	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Train application users and operations staff in how to properly use the software, as needed. [SQAP S4.3.2]	*		Contained in the ROSE SQAP section 13	Complies
Software developers and evaluators to complete EC4063, Software Safety Engineering Practices. [SQAP S4.13]	O		N/A	N/A
Produce user and installation guidance, as required by the customer. [SQAP S4.4.2.5]	O		Contained in the ROSE SQAP section 4.2.5	Complies

Appendix B Implementation Form for ROSE Core

Risk Level 4, Non-830 Software, Major Development Control

Software: [ROSE](#)
Date: [9/26/2016](#)
Author: [Gregory Pope](#)

This form documents the software quality assurance (SQA) activities and implementation plans for Risk Level 4 practices for Non-830 Software when there is major development control. A software risk grading was performed in accordance with LLNL Institutional

Software Quality Assurance Program for Non-830 Software² requirements implementing DOE O 414.1D Admin Chg 1 and other LLNS contract requirements.

The flow down of SQA requirements to consensus standards is documented in RID-0118-00, *Software Quality Assurance Consensus Standards for Quality Assurance Criteria*. RID-0118 references the following IEEE (Institute of Electrical and Electronics Engineers, Inc.) standards:

IEEE Std 730TM-2002, *IEEE Standard for Quality Assurance Plans*

IEEE Std 828TM-2012, *IEEE Standard for Configuration Management in Systems and Software Engineering*

IEEE Std 1012TM-2012, *IEEE Standard for System and Software Verification and Validation*

Document templates have been created for flowing down the SQA requirements. These templates may be downloaded from the [SWING](#) wiki. The use of the templates is optional and may be used to ensure requirement flow down.

Documents	Templates
Software Quality Assurance Plan (SQAP)	Non-830 Software SQAP-4M
Software Configuration Management Plan (SCMP)	Non-830 Software SCMP-4M
Software Verification and Validation Plan (SVVP)	Non-830 Software SVVP-4M

The table on the following pages details the required plans and products. Each practice has an indicator of the level to which the practice is to be implemented. These levels are:

- X Indicates the practice is required and fully implemented.
- * Indicates the practice is required, but may be partially implemented with written justification.
- O Indicates the practice is recommended, but optional.

Cross-references to the IEEE requirements that are to be used to implement the practice are indicated in “[]” for each practice. For example [SQAP S1, S3 (all)] refers to sections 1 and 3 (with all of its subsections) of IEEE 730-2002.

² DES-0108-00, *Non-830 Institutional Software Quality Assurance Program* and FRM-3104, *SQA Required Practices for Non-830 Software*

This Implementation form applies to new code or improvements made to the ROSE framework.

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 1: Software Project Management and Quality Planning				
Prepare, review, and approve a Software Quality Assurance Plan (SQAP) using the institutional template specific to the risk grading.	O		Software Quality Assurance Plan to be written to accommodate work for others and DOE	Complies
Plan and manage project activities and commitments (including organizational structure/interfaces; roles and responsibilities / authority; schedule; budget; and resources). [SQAP S4.3.1, S4.3.2]	*		ROSE master plan is done using MS Project and includes staffing, tasks, sub tasks, s schedule, budget, resources.	Complies
Identify, define, plan, and track software quality assurance activities, including software development methodology, configuration management, verification and validation, management reviews, and testing. [SQAP S4.4.2.1, 2, 3, 6; S4.5.1, 2; S4.6.1; S4.6.2.8; S4.7; S4.8; S4.9; S4.11]	*		Covered in ROSE Software Quality Assurance Plan	Complies
Determine applicable regulatory requirements. [SQAP S4.5.1, 2]	X		Covered in ROSE Software Quality Assurance Plan section 2	Complies
Identify formal documents. [SQAP S4.4.2.1, 2, 3, 6]	*		Covered in ROSE Software Quality Assurance Plan section 4.2 and 4.3	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Identify records of evidence to be produced. [SQAP S4.12]	*		Covered in ROSE Software Quality Assurance Plan section 12	Complies
Select and utilize additional standards, as needed. [SQAP S4.5.1, S4.5.2; SVVP Table 1c S9.3(1) a) 2), 7)]	*		Covered in ROSE Software Quality Assurance Plan section 5.	Complies
SQA Work Activity 2: Software Risk Management	This work activity is focused on project and software development risks. See SQA Work Activity 7 for safety hazards.			
Identify, assess, track and manage software risks, including development, implementation and usage. [SQAP S4.14]	*		Covered in ROSE Software Quality Assurance Plan, tracked in JIRA Tool	Complies
Complete a Process/Development Environment Risk Report in the Software Risk Grading Tool following PRO-0107 , <i>Software Risk Grading</i> . [PRO-0107]	O		Covered in ROSE Software Quality Assurance Plan and Risk grading tool.	Complies
SQA Work Activity 3: Software Configuration Management	Institutional documents are managed via DES-0032 , <i>Management of Institutional Documents</i> , and PRO-0033 , <i>Developing, Reviewing, and Approving Institutional Documents</i> .			
Prepare, review, and approve a Software Configuration Management Plan (SCMP) using the institutional template specific to the risk grading.	O		N/A	N/A
Plan, document, approve, and implement a software configuration management process (including criteria for identification of configuration items, configuration change control, and release version control). [SQAP S4.4.2.6, S4.8; SCMP S9.2.4.1, S10.2.3]	*		Contained in the ROSE SQAP section 4.2.6	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Develop and use processes for document review, approval, revision, and issuance; ensure usage of current applicable documents. [SCMP S9.2.4.1, S10.2.3]	*		Contained in the ROSE SQAP section 6	Complies
Develop and use processes for records review, approval, storage, and retrieval. [SQAP S4.12]	*		Contained in the ROSE SQAP section 12	Complies
Plan and implement processes to identify and configuration control the critical characteristics of the operating environment, if any. [SQAP S4.4.2.6]	O		Contained in the ROSE SQAP section 6	Complies
Implement and maintain SCM repositories. [SQAP S4.10; SCMP S8.2.5.8, S8.2.5.10]	*		Contained in the ROSE SQAP section 6, using GIT	Complies
Plan and implement disaster recovery processes. [SQAP S4.10; SCMP S8.2.5.10]	*		Contained in the ROSE SQAP section 4.2.6 and	Complies
Prevent usage upon retirement, as needed. [SCMP S14.2.8]	O		Contained in the ROSE SQAP section 12	Complies
SQA Work Activity 4: Procurement and Supplier Management	SQA follows the standard Procurement processes. Note that the Risk Level grading is for the purchased/acquired software, which may be different than the Risk Level grading for the associated software development effort.			
Follow institutional procurement processes for evaluating and selecting suppliers. [POAP 1502]	*		Contained in the ROSE SQAP section 11	Complies
Follow institutional procurement processes for supplier corrective actions. [SCM IQP 0010]	*		Contained in the ROSE SQAP section 11	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Follow institutional procurement process for specifying requirements (technical and quality) and acceptance criteria for software (products and tools). [SQAP S4.11; Supply Chain Management Standard Practice 46.1]	*		Contained in the ROSE SQAP section 11	Complies
Qualify procured and otherwise acquired software for use. [PRO-0097 , PRO-0100]	*		Contained in the ROSE SQAP section 11	Complies
SQA Work Activity 5: Software Requirements Identification and Management				
Identify (to the extent known), document, review, approve and update requirements. [SQAP S4.4.2.1, S4.6.2.1]	*		Contained in the ROSE SQAP section 4.2.1	Complies
Verify that applicable requirements were incorporated into the design. [SQAP S4.6.2.3; SVVP Table 1c S9.3(1) a) 2)]	*		Contained in the ROSE SQAP section 4.2.1	Complies
Identify, document, and track security requirements (e.g., vulnerability protection and cyber - security) commensurate with the risk from unauthorized access or use. [SQAP S4.4.2.1, S4.14]	O		Contained in the ROSE SQAP section 4.2.1	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 6: Software Design and Implementation				
Identify (to the extent known), document, review and approve the software design, including software and system interfaces. (Note: This documentation may be combined with the requirements documentation and/or captured via source code comments.) [SQAP S4.4.2.2, S4.6.2.3; SCMP S12.2.1, S12.2.2; SVVP Table 1c S9.3(1) a) 3), 7)]	*		Contained in the ROSE SQAP section 4.2.2	Complies
Identify and utilize coding standards and/or conventions. [SQAP S4.5.1, S4.5.2, S4.9; SVVP Table 1c S9.3(1) a) 2), 7)]	*		Contained in the ROSE SQAP section 4.2.2	
Include design features to ensure proper use such as access control, help text, user interface design, as needed. [SQAP S4.4.2.2]	*		Contained in the ROSE SQAP section 4.2.2	
SQA Work Activity 7: Software Safety	This work activity only applies to software applications with potential ES&H consequences.			
Review nonnuclear safety basis documents, if any, for mitigation strategies (i.e., controls) for software components that have potential ES&H consequences if a fault occurs. [ES&H Manual Document 3.1, Nonnuclear Safety Basis Program; SQAP S4.14]	*		N/A	N/A
Perform reliability and usability analyses as needed.	O		N/A	N/A

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
SQA Work Activity 8: Software Verification and Validation	The V&V activities are conducted in connection with all of the work activities.			
Prepare, review, and approve a Software Verification and Validation Plan (SVVP) using the institutional template specific to the risk grading.	O		N/A	N/A
Plan, document, and perform testing. [SQAP S4.4.2.3, S4.4.2.4, S4.6.2.8, S4.7; SVVP Table 1c S9.6(1) a) 1), 2), 5), 6); Table 1c S9.6(1) b); Table 1c S9.7(2) a) 1), 2), 5); Table 1c S9.7(2) b), c)]	X		Contained in the ROSE SQAP section 4.2.3	
Plan and perform peer reviews (including inspections, walkthroughs and desk checks). [SQAP S4.4.2.4, S4.6.1, S4.6.2.1 S4.6.2.3, S4.6.2.8]	*		Contained in the ROSE SQAP section 4.2.3	Complies
Test implemented changes for correctness and to ensure no breakage occurred. [SQAP S4.4.2.4, S4.7; SVVP Table 1c S9.6(1) a) 1), 2) 5) 6); Table 1c S9.6(1) b); Table 1c S9.7(2) a) 1), 2), 5); Table 1c S9.7(2) b), c)]	*		Contained in the ROSE SQAP section 4.2.3	Complies
Ensure design reviewer participation includes adequate independence. [SQAP S4.6.1]	*		Contained in the ROSE SQAP section 4.2.3	Complies (ROSE Framework uses regression tests written by all team members)
Verify and validate that the software and design satisfies the requirements using objective acceptance criteria for the components. [SVVP Table 1c S9.3(1) a) 1); Table 1c S9/3(1) f) 2)]	*		Contained in the ROSE SQAP section 4.2.3	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Approve V&V results prior to using the software. [SQAP S4.3.2]	*		Contained in the ROSE SQAP section 4.2.3	Complies
SQA Work Activity 9: Problem Reporting and Corrective Action				
Plan, document, and implement methods for reporting/documenting, evaluating, and correcting software problems. [SQAP S4.8; SCMP S9.2.5.1, S9.2.5.2; SVVP Table 1c S9.10(2)]	*		Contained in the ROSE SQAP section 8	Complies
Implement process improvement activities (including problem prevention and soliciting and analyzing quality feedback). [SQAP S4.6.2.8, S4.6.2.10, S4.8; SCMP S7.2.2.2, S7.2.2.4; SVVP Table 1a S7.1(6) c), d), e), f); Table 1c S9.10(2)]	*		Contained in the ROSE SQAP section 8	Complies
Create release notes, as needed.	O		Contained in the ROSE SQAP section 8	Complies
SQA Work Activity 10: Training of Personnel in the Design, Development, Use and Evaluation of Software	SQA utilizes the institutional training and qualification processes, which includes LTRAIN . SQA also utilizes the job posting process that specifies qualifications.			
Select and train team members. [LTRAIN ; SQAP S4.13]	*		Contained in the ROSE SQAP section 13	Complies
Development team leaders, developers, testers, and evaluators to complete CA0750-W, Software Quality Assurance Awareness. [SQAP S4.13]	X		Contained in the ROSE SQAP section 13	GAP – Schedule LTRAIN course.
Produce user training materials, as required by the customer. [SQAP S4.4.2.5]	*		Contained in the ROSE SQAP section 13	Complies

DOE O 414.1D Work Activity/ Practice	Impl Level	Justification for Non-Applicability or Tailoring	Current/Planned Practice Implementation	Gap Closure/Acceptance and/or Change Plans
Train application users and operations staff in how to properly use the software, as needed. [SQAP S4.3.2]	*		Contained in the ROSE SQAP section 13	Complies
Software developers and evaluators to complete EC4063, Software Safety Engineering Practices. [SQAP S4.13]	O		N/A	N/A
Produce user and installation guidance, as required by the customer. [SQAP S4.4.2.5]	O		Contained in the ROSE SQAP section 4.2.5	Complies

Appendix C Identified Software Effort Development Risks

Description	Status	Magnitude of Impact	Priority	Prob of Occurrence	Risk Responses	Mitigation Actions
One team member becomes indispensable to the team because of knowledge of sections of the code. Modifications of sections of the code can only be done by	Active	1 - Highest	1 - Highest	1 - Highest	Mitigate	Increased responsibility for Marcus and Liao. Training materials or classes from EDG for their portion of the front end code.
Code going directly from researcher/developer to user without any independent system testing. Documentation not verified.	Active	1 - Highest	1 - Highest	1 - Highest	Mitigate	Independent system testers added, funded by ISCP, documentation versions synchronized to versions.
System testing not planned or included in ROSE development model.	Active	1 - Highest	1 - Highest	1 - Highest	Mitigate	ISCP funding covers 1.5 FTE for system testing and documentation verification. All tools built for users form the ROSE framework will require independent system
Deliverable milestones missed and/or capability of tool does not meet customer expectations. Tool too hard to install and/or use.	Active	1 - Highest	1 - Highest	1 - Highest	Mitigate	The customer based ROSE masterplan to be reviewed on a weekly basis detecting problems, identifying obstacles, and setting work arounds into place at the earliest point.
Team or team members attracted to new and exciting sounding code challenges diverting resources and causing longer term project milestone slip right.	Active	1 - Highest	1 - Highest	2 - High	Mitigate	Limit annual time budget for Bluebird requests to a capped amount. Some of these efforts are useful (eat your own dog food) and important , so do not want to
Technical debt has accumulated in ROSE code over the past 15 years making it harder to trouble shoot and maintain.	Active	1 - Highest	1 - Highest	5 - Lowest	Unassigned	Prioritize and repair 600 or so defects in backlog
Do not want to retard research and exploration, want to have more rigorous code tools built from ROSE.	Active	1 - Highest	2 - High	1 - Highest	Mitigate	Dual track risk levels in SQAP. RL 4 for ROSE framework (core) for researchers, RL 3 for tools built with ROSE framework for application developers.
ROSE has historically been funded from various sources without long term continuity. This has made the team leadership conservative about growth and support and	Active	1 - Highest	2 - High	2 - High	Mitigate	Additional LLNL internal funding applied to help fill in the funding GAPS. Expansion of number of tasks for GS using planning, hiring, and contracting to supplement ROSE.
Rose managed as one large project with various add-ons for versus customers.	Active	2 - High	1 - Highest	2 - High	Monitor	ROSE master plan breaks out ROSE work by customer projects.
Versions of EDG, BOOST, Compilers, Operating Systems, Compiler Standards, and language popularity changing impacts ROSE performance.	Active	2 - High	2 - High	2 - High	Mitigate	Automated testing reduces effort necessary to adapt to new condition. Previously relased ROSE tool impacts considered. Updates to new version only done when
ROSE users may not think like compiler specialists	Active	2 - High	2 - High	2 - High	Mitigate	Orient documentation to support application developers that are not compiler experts
Documentation is emense and hard to find specific topics	Active	2 - High	2 - High	2 - High	Mitigate	Create cookbook instructions for tools and a knowledge base forcommon user problems
Users service requests may get lost.	Identified	2 - High	2 - High	2 - High	Mitigate	Deploy service desk tool in JIRA for customers and workflow for assuring request gets serviced.

Platforms used for testing ROSE are aging and need to be upgraded to be more consistent with customer platforms.	Identified	2 - High	3 - Medium	2 - High	Mitigate	Use a portion of the project funds to purchase new platforms.
User support is handled by asking developers for help directly which may distract them from other commitments.	Active	2 - High	3 - Medium	2 - High	Mitigate	System testers will be used as the first level of user support. If system testers can not help the user then they will talk with developers to get additional information to
The ROSE install packages includes ROSE tests which use up a majority of the users time loading and a large amount of platform memory.	Identified	2 - High	3 - Medium	2 - High	Move	Tests will be removed from user install media. The tests can still be obtained but only if the user wants to have them.
The ROSE tool folder contains dozens of different tools collected over the years. Supporting all of these tools is very time consuming.	Identified	2 - High	3 - Medium	2 - High	Avoid	Pare the tool folder down to less than a dozen useful tools. Use the system testers to validate the tools work correctly.
EDG goes out of business	Active	2 - High	3 - Medium	4 - Low	Mitigate	ROSE demonstrated to work with LLVM (CLANG) for C.
ROSE using best SE practices	Active	2 - High	3 - Medium	2 - High	Mitigate	Need to continue to encourage and incentivize ROSE staff to use SE best practices, reviews, and robustification funds
There are 14 x 35 variations of BOOST and Compilers which are currently tested in the matrix testing	Active	3 - Medium	2 - High	2 - High	Avoid	Consider reducing the number of combinations based on market analysis and customer requests.
Open Source Users consume time of senior developers seeking help. This creates unolanned work.	Active	3 - Medium	2 - High	2 - High	Move	Set up help desk, FAQ, and a priority escalation schema to releave senior developers from helping users except in rare cases.
Proposals submitted primarily to secure funding may not agree with what ROSE sources can do in the time frame given.	Active	3 - Medium	2 - High	3 - Medium	Mitigate	Going forward assure proposals also support the master plan and directions, focus on beefing up previous research work.
LC does not support all the platforms that customers require ROSE to run on.	Active	3 - Medium	3 - Medium	2 - High	Move	Out source to cloud vendors.
	Unassigned	5 - Lowest	5 - Lowest	5 - Lowest	Unassigned	
	Unassigned	5 - Lowest	5 - Lowest	5 - Lowest	Unassigned	

Appendix D Release Policy for ROSE and ROSE-based Tools

Version 0.9.0

This document details the release process for all tools to customers that are not release as part of ROSE directly. Tools released in ROSE may follow different standard depending on the target audience. We can find out if they have any additional requirements and discuss these. We expect these tests will be driven by a Jenkins run separate from our current Jenkins testing.

1. All tools will have specific features expected from production level tools. Features common to all tools released under this process:
 - a) Command line help
 - b) Documentation in a README file at the top-level directory
 - c) Smoke tests
 - d) One or more examples to use as regression tests
2. Obtaining the release for independent testing
 - a) All tools will be made available from one or more repositories available locally at LLNL. Some repositories might be end customer specific. ROSE Garden is the current preferred repository for ROSE-based tools (for a repository maintained internally at LLNL on the Blue Network and for tools not distributed within ROSE).
 - b) Testing will be possible directly from the repository and from binary releases of ROSE and/or individual tools.
 - c) The release of tools for independent testing will be made from a single repository (might be customer specific).
 - d) Binary builds of ROSE-based tools will be constructed from source code from a single repository (might be customer specific).
3. What tests are run in independent testing and in what environment
 - a. All tools will be run in a Linux environment, specific customers may require additional environment specifics to support specific tools (threading environments, different Operating Systems (e.g. real-time OS), etc.)
 - b. Follow installation instructions to build the tool using a specified version of ROSE and any dependences.
 - c. Tests that will be run:
 - i. Check for installation directions (README in top level directory)
 - ii. Test building the tool against the specific versions of ROSE and Boost.
 - iii. Each tool should have a set of smoke tests to make sure it is installed properly; these tests should be run (e.g. make smoke).
 - iv. Each tool should have command-line documentation, run tool to verify presence of command line help (e.g. tool-name -help)
 - v. Regression tests will be provided, and testing should include passing these tests (e.g. make check).
 - vi. Verify presence and correctness of document on each tool.

- d. Record passing independent testing, tool version number, ROSE version number, date, and any additional details (OS used, etc.). This should be recorded in a release log that is checked into the top level of the tool directory within the repository.
4. Package tool for customer
- a. Review documentation and prepare and correct any existing documentation.
 - b. Check-in any changes to documentation.
 - c. Package to send tool to customer (tar file is likely sufficient).