LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-TR-477658

NIF Projects Controls and Information Systems Software Quality Assurance Plan

B. Fishler

March 31, 2011

Disclaimer

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

NIF Projects

Controls and Information Systems

Software Quality Assurance Plan

NIF-5022079-AB

April, 2010

ECMS CM RELEASE 20100505 13:51 5022079 AB (pdf format)

Action	Name	Signature
Prepared by:	Paul Van Arsdall, NIF Programs SQA Point of Contact	Approved in ECMS
Reviewed by:	Mike Shaw, System Performance Modeling Manager	Approved in ECMS
Reviewed by:	Bob Reed, Industrial and Safety Controls Manager	Approved in ECMS
Reviewed by:	Tim Frazier, CIS Information Technology Manager	Approved in ECMS
Reviewed by:	Darwin Dobson, CIS Business Applications Manager	Approved in ECMS
Reviewed by:	Ric Beeler, CIS Shot Data Systems Manager	Approved in ECMS
Reviewed by:	Allan Casey, CIS Shot Data Systems Manager	Approved in ECMS
Reviewed by:	Pete Ludwigsen, CIS Integrated Controls Manager	Approved in ECMS
Reviewed by:	Dan Koning, CIS Software Configuration Manager	Approved in ECMS
Reviewed by:	Barry Fishler, CIS Quality Controls Manager	Approved in ECMS
Reviewed by:	Bob Carey, CIS Lead Software Architect	Approved in ECMS
Concurred by:	Larry Lagin, APM Controls and Information Systems	Approved in ECMS
Concurred by:	Don Prigel, NIF & PS QA Manager	Approved in ECMS
Approved by:	Ralph Patterson, NIF Projects Manager	Approved in ECMS

Table of Contents

NIF Projects Controls and Information Systems

Software Quality Assurance Plan

1 Purpose and Scope

Quality achievement for the National Ignition Facility (NIF) and the National Ignition Campaign (NIC) is the responsibility of the NIF Projects line organization as described in the *NIF and Photon Science Directorate Quality Assurance Plan* (*NIF QA Plan*). This Software Quality Assurance Plan (*SQAP*) is subordinate to the *NIF QA Plan* and establishes quality assurance (QA) activities for the software subsystems within Controls and Information Systems (CIS). This *SQAP* implements an activity level software quality assurance plan for NIF Projects as required by the LLNL *Institutional Software Quality Assurance Program* (*ISQAP*). Planned QA activities help achieve, assess, and maintain appropriate quality of software developed and/or acquired for control systems, shot data systems, laser performance modeling systems, business applications, industrial control and safety systems, and information technology systems.

The objective of this *SQAP* is to ensure that appropriate controls are developed and implemented for management planning, work execution, and quality assessment of the CIS organization's software activities. The CIS line organization places special QA emphasis on rigorous configuration control, change management, testing, and issue tracking to help achieve its quality goals.

1.1 Graded Approach

This *SQAP* employs a graded approach that applies quality assurance and quality control (QC) measures according to the consequence and likelihood of failures. The grading process helps assure QA and QC measures mitigate software risks in balance with overall risks to technical performance, cost, and schedule. Software covered by this *SQAP* will be evaluated to assess risk and consequences of failure. The level of risk for a software product is generally assessed by determining consequences of software failure based on a failure modes and effects analysis (FMEA) prepared with appropriate participation of management, subject matter experts and, if needed, safety basis experts. See the *ISQAP* for guidance. Results should be recorded in the *Software Quality Assurance Risk Grading Tool* spreadsheet.

This *SQAP* is generally arranged for Risk Level 3 quality assurance requirements, as defined in the *ISQAP*. Consequently, many of the practices described may be tailored more or less as to the degree of implementation rigor (except where noted herein or required by other NIF procedures). Accordingly, a Risk Level 2 software product may be accommodated by increasing the level of rigor. See the *Software Quality Assurance Risk Grading Tool Template* and the *ISQAP* for more information on software Risk Levels and tailoring recommended software practices.

Note: use of the term "shall" in this *SQAP* indicates a mandatory QA requirement.

In accordance with the *ISQAP* and DOE O 414.1c, additional quality control rigor shall be applied to **Safety Software (SSW)** and **Safety-Related Software (SRSW)**. **Safety Software** is software that performs radiological safety function, analysis, design, or administration. **Safety-Related Software** is software that performs non-radiological worker safety functions,

analysis, design, or administration. **Safety Software** and **Safety-Related Software** will receive appropriate additional SQA rigor. Examples of non-worker-safety software that will receive greater rigor include control system software and software guiding laser operation that implement functional or administrative protections against significant equipment damage. See the *ISQAP* for specific guidance on the correct classification of **Safety Software**.

> **Note:** Risk assessment and grading of **Safety Software** shall be recorded using the Project Risk Grading web-based tool located on the Institutional Software Quality Assurance web site.

1.2 Software Product Identification

CIS software includes a variety of independent and/or interacting subsystem products that together comprise NIF software for the following software subsystems:

- Business Applications (BA)
- Integrated Controls (IC) / Integrated Computer Control System (ICCS)
- Industrial and Safety Controls (ISC)
- Information Technology (IT)
- Lab Systems (LS)
- System Performance Modeling (SPM)
- Shot Data Systems (SDS)

Table 1 lists a summary of the major software products covered by this *SQAP*. Products are generally organized into subsystems based on different software and/or hardware architectures that have been chosen to optimally meet system requirements. Development, operation and maintenance activities are supported by appropriate computer environments, software development languages and tools, run-time libraries and services, software configuration management tools, code testing and debugging tools, and issue tracking tools (Table 2).

1.3 Software Activities

CIS software is comprised of custom in-house developed software and software supplied by vendors or open source. Both software sources and their associated activities are covered by this *SQAP*.

> **Note:** Software developed for, or used by, NIF Projects may have special designations calling for increased QA rigor. The special designations are 1) RQ-Levels for system requirements, 2) NIF Risk Levels and 3) Configuration Management Rigor-Levels. See sections 1.4 and 1.5.

1.3.1 Developed Software

Developed software is software engineered from source code by the CIS organization. Software development lifecycle activities and management practices include the following tasks:

- Requirements analysis,
- Design,
- Coding,
- Configuration management,

- Developer testing and integration,
- Installation,
- Verification and validation,
- Training,
- Software operation,
- Error reporting and maintenance.

Table 1. Summary of CIS Software Products and Safety Designations

Software Product	Software Subsystem	CM & Safety Designations
B298 Radioactivity Allowance Tracking System	BA	SSW
Radiological Inventory Management System	BA	SSW
Enterprise Configuration Management System	BA	
Glovia Enterprise Resource Planning System	BA	
NIF IT Web	BA	
NIF Planning System (NPS)	BA	
ProIntralink	BA	COTS
Rack Database System	BA	
Requirements Management System (RMS)	BA	
Sentinel Real-time Dosimetry Tracking	BA	COTS
Survey Information Management System (SIMS)	BA	SRSW
SMART Maintenance Planning System	BA	
Instrument Based Controller	IC	
Integrated Computer Control System	IC	
Access Control System	ISC	
Amplifier Cooling Control System	ISC	
Argon Control System	ISC	SRSW
Chamber Vacuum Control System	ISC	
Facility Environmental Monitor	ISC	
Final Optics Assembly Cooling System	ISC	
HVAC Direct Digital Control	ISC	
Machine Protection System	ISC	
NIF Site Safety Interlock System (SIS)	ISC	SSW
PABTS Vacuum Control System	ISC	
PEPC Vacuum Control System	ISC	
Spatial Filter Vacuum Control System	ISC	
Support Lab Safety Interlock Systems	ISC	SRSW
Tempered Water Control System	ISC	
Transport and Handling System	ISC	SRSW
Hazardous Materials Management System / Tritium Processing System	ISC	SSW
Asset DB	IT	
Optics Mitigation Control Systems	LS	
Laser Performance Operations Model	SPM	
Virtual Beam Line Model	SPM	
Wavefront	SPM	
Campaign Management Tool	SDS	
LoCoS	SDS	
ODAD/PDAD/CDAD Data Warehouse	SDS	

Optics Damage Inspection	SDS	
Quicklooks	SDS	
Recycle Now	SDS	
Shot Analysis and Data Visualization (SADV/SAVI))	SDS	
Shot Report	SDS	
Shot Planning Tool (SPLAT)	SDS	
Side Illuminated Damage Inspection	SDS/IC	SRSW

1.3.2 Off-the-Shelf Software

Off-the-shelf (OTS) software is any software used in NIF that is supplied by commercial (COTS) vendors, or by other organizations within LLNL, or externally. OTS software is generally installed, configured and used without additional source code development. OTS software includes acquired tools or components that directly support end users or participate in the software lifecycle: e.g., code development, testing, configuration management, issue tracking, computer services and run-time environments. OTS software lifecycle and management practices typically include the following tasks:

- Determination of software applicability and vendor selection,
- Configuration management,
- Installation,
- Training,
- Error reporting and maintenance.

1.4 NIF Requirements-Quality (RQ) Levels and NIF Risk Levels

In addition to the Risk Level definitions given in the *ISQAP* and the CM Rigor Levels given for Configured Systems, NIF has assigned Requirements-Quality Levels (ref. *NIF Project Control Procedure, Establishment of RQ-Levels*) to each system requirement that is managed within the Requirements Management System (RMS). The RQ-Levels are designated from level 1 to 3, where level 1 is the highest risk level assigned to the highest consequence of failure.

Most software requirements have been assigned to the lowest level (RQ-Level 3), which will be treated with a medium level of quality assurance with a graded level of verification activity. Designation of RQ-Level 3 affords managers flexibility to administer varying degrees of QA rigor to the affected software products in order to achieve adequate quality within constraints for software development cost and schedule.

Certain requirements have potentially more serious failure consequences that are characterized by concerns for worker safety, severe equipment damage, schedule impact or environmental impact. These requirements are assigned higher RQ levels (i.e., RQ-Level 1 or 2). RQ-level 1 and 2 requirements shall be treated with an additional degree of quality control and quality assurance rigor, notably including 100% verification of critical requirements.

NIF Procedure 1.6 *Using NIF Risk Levels to Apply a Graded Approach* describes risk grading applied to key attributes of systems, structures and components (SSC) that includes

software requirements. NIF Risk Levels will replace RQ-levels for new systems and are generally equivalent to RQ-levels.

1.5 Configured Systems and Rigor-Levels

The *NIF Configuration Management Plan* (CMP) describes the technical and administrative process for controlling the NIF configuration. This *SQAP* augments the CMP with specific requirements and practices applicable to software artifacts and processes such as source code development and change management.

In addition, the NIF CMP designates a number of elements as Configuration Items (CI) that are critical to the facility safety basis, public and worker safety, the environment, programmatic impact, or security that shall receive additional review and QA rigor. Configured items are organized into Configured Systems (CS) that are managed by Configured System Managers (CSM) to assure that the designs, documentation, and as-built configuration are consistent and receive appropriate review. Configured Items have each been assigned a Configuration Management Rigor-Level (CMRL) in accordance with the management. See *Configured Systems and System Manager List* and the Configured Items List associated with each Configured System.

All factors should be considered along with risk grading criteria given in the *ISQAP* to determine appropriate SQA rigor. The highest risk grade shall determine the minimum recommended software practices and level of management oversight.

2 Reference Documents

2.1 Applicable Standards

In accordance with the ISQAP requirements, a Software Safety Plan based on the *Std 1228 IEEE Standard for Software Safety Plans* shall be prepared for all **Safety Software**. The following standards provide guidance in the planning and implementation of CIS software quality assurance.

- IEEE 729, IEEE Standard Glossary of Software Engineering Terminology
- IEEE 730, IEEE Standard for Software Quality Assurance Plans
- IEEE 828, IEEE Standard for Software Configuration Management Plans
- IEEE 829, IEEE Standard for Software Test Documentation
- IEEE 830, IEEE Guide to Software Requirements Specifications
- IEEE 1008, IEEE Standard for Software Unit Testing
- IEEE 1012, IEEE Standard for Software Verification and Validation
- IEEE 1016, IEEE Recommended Practice for Software Design Descriptions
- IEEE 1028, IEEE Standard for Software Reviews and Audits
- IEEE 1058, IEEE Standards for Software Project Management Plans
- IEEE 1228, IEEE Standard for Software Safety Plans

2.2 LLNL/NIF References, Plans and Procedures

The following references, plans and procedures support this *SQAP*:

2.2.1 LLNL References

- Design and Implementation of Safety Interlock Systems, Section 12.1 Engineering Design Safety Standards

- Institutional Software Quality Assurance Program, LLNL-AM-406580

2.2.2 NIF References, Plans and Procedures

- Action Management, NIF Procedure 1.8, NIF-5023066
- Baseline Change Control, NIF Procedure 1.7, NIF-5018651
- Documentation and Records Control, NIF Procedure 4.1, NIF-5018881
- Control of CM Documents, NIF Procedure 6.4, NIF-5018532
- Engineering Design Reviews, NIF Procedure 5.1, NIF-5018587
- Generation of Cost Account Plans, NIF Procedure 1.9, NIF-0004348
- Information Systems Configuration Management Plan, NIF-5029810
- LPOM Change Control, NIF Procedure 5.27, NIF-5018883
- NIF and Photon Science Principle Directorate Integrated Safety Management System, NIF-0054160
- NIF & PS Directorate Self Assessment Plan, NIF-5025029
- NIF Directorate ES&H Self-Assessment Plan, NIF-0112692
- NIF Information Systems Account Authorization Policy and Procedures, NIF-5029809
- NIF Information Systems Backup Maintenance Policy and Procedures, NIF-5029807
- NIF and Photon Science Directorate Informations Systems Disaster Recovery Plan, NIF-5030408
- NIF Information Systems Incident Response Procedure, NIF-5029868
- NIF Information Systems Configuration Management Policy and Procedures, NIF-5029811
- NIF Information Systems System Maintenance Policy and Procedures, NIF-5029806
- NIF Information Systems Scan and Patch Maintenance Policy and Procedures, NIF-5029809
- NIF Maintenance Plan, NIF-5018526
- NIF Operations Management Plan, NIF-5020544
- NIF and Photon Science Directorate Quality Assurance Plan, NIF-5021183
- NIF Programs Safety Protocols and Requirements Manual Section 3.11 – Safety Interlock System, NIF-0112739
- NIF Safety Basis CM Training, NIF-0114120
- NIF Training Plan, NIF-5018705
- NIF Work Permits, NIF Procedure 5.8, NIF-5018626
- Management Review, NIF Procedure 5.11, NIF-5026065
- Preparation and Revision of System Design Requirements, NIF Procedure 6.1, NIF-5018885
- Problem Reporting and Disposition of Nonconforming Material, NIF Procedure 3.2, NIF-5018653
- Procurement of Configuration Items, NIF Procedure 7.9, NIF-5019091
- Procurement Planning, Scheduling, Review & Approval, NIF Procedure 7.4, NIF-5018887
- Project Monthly Status Report, NIF Procedure 1.4, NIF-0072597

- Project Performance Management Using Earned Value, NIF Procedure 1.12, NIF-0072079
- Preparation and Standard Content for Commissioning Test Plans and Procedures, NIF Procedure 8.3, NIF-5018666
- Safety and Performance Review Board, Management Prestart Reviews, and Working Group Reviews, NIF Procedure 9.3, NIF-5018667
- Safety Basis Change Control Process, NIF Procedure 1.20, NIF-5018625
- Statement of Work Preparation, NIF Procedure 7.6, NIF-5018888
- Supplier Qualification, NIF Procedure 7.1, NIF-0001014
- Tier 2 Safety Basis Document for the Building 581-582 Complex, NIF-5019666
- Using NIF Risk Levels to Apply a Graded Approach, NIF Procedure 1.6, NIF-5018878
- Unified System Hierarchy, NIF-5021086
- Work Authorization Review, NIF Procedure 5.19, NIF-5018658

2.2.3 NIF Configured System Management

- Configured Systems and System Manager List, NIF-5018525
- NIF Programs Directorate Configuration Management Plan, NIF-0015684
- NIF Project and Facility Configuration Management Plan, NIF-5018949
- NIF Operations Qualification Card Training Package: NIF System Manager for Configured Systems – CM and Safety Basis, NIF-5018874

2.2.4 CIS Documents, Plans and Procedures

- CIS Software Development Processes, Guides and Tools, NIF Wiki at https://nif-wiki.llnl.gov/
- CIS Software Tester Qual Card, NIF-5030392
- CIS Software Configuration Management Process and Procedures, online at https://nif-wiki.llnl.gov/display/scm/NIF+Software+Configuration+Management
- CIS Software Defect and Change Request Procedure, online at https://nif-wiki.llnl.gov/display/jira and tool
- CIS Software Defect and Change Request Tool, online at http://jira.llnl.gov/secure/Dashboard.jspa
- Hazardous Materials Management System Software Installation Procedure, NIF-5030452
- Software Design Review Process, online at: https://nif-wiki.llnl.gov/display/sccb/Design+Review+Procedure+and+Templates
- Software Code Review Process, online at: https://nif-wiki.llnl.gov/display/sccb/Code+Review+Procedure
- ICCS Software Architecture Description, online at: https://nif-wiki.llnl.gov/display/iccs/1+Architecture+Overview
- ICCS B581 Production Environment Database Change Request, NIF-5022489
- ICCS B581 Production Environment Software Change Request, NIF-5022490
- ICCS B581 Production Release Installation Procedure, NIF-5014353
- ICCS Software Release WAP Checklist Template, NIF-0115034
- Industrial Control System Sofware Development Qual Card, NIF-5030453
- IT Software Release WAP Checklist Template, NIF-0115033

- NIF CIS Radiological Inventory Management System – RIMS Software Requirements Specification, NIF-5029982
- NIF Industrial Controls Common Requirements and Style Guide, NIF-5005780
- NIF Safety Interlock System Requirements Specifications, NIF-0000525
- NIF Software Configuration Management Plan for Industrial Controls, NIF-50005781
- NIF Project ICCS Software Metrics Plan*, NIF-5013882
- NIF CIS Radiological Inventory Management System – RIMS FMEA and SQA Grading, NIF-5030376
- NIF CIS Radiological Inventory Management System – RIMS Software Safety Plan, NIF-5030394
- NIF CIS Radiological Inventory Management System Software Installation Procedure, NIF-5030386
- NIF CIS Radiological Inventory Management System – RIMS Test Procedure, NIF-5030377
- Safety Interlock System Software Installation Procedure, NIF-5030451
- Safety Interlock System Online Test Procedure, NIF-0088456
- Shot Systems Software and Infrastructure Change Release Process, NIF-5021824
- SIS Software Risk Grading Tool, NIF-0115048
- Software Change Request Process, online at: https://nif-wiki.llnl.gov/display/sccb/SCCB+Membership+Duties
- Software Deskcheck Procedure, online at: https://nif-wiki.llnl.gov/pages/viewpage.action?pageId=10060476
- Software Quality Assurance Risk Grading Tool Template, NIF-0115007
- Software Safety Plan for the NIF Safety Interlock System, NIF-5022632
- Software Safety Plan for Hazardous Materials Maintenance System, NIF-5030343
- SQA Risk Grading and Recommended Work Activities for HMMS, NIF-5030344
- System Requirements Specification HMMA Glovebox Monitoring and Control System, NIF-5029433
- System Requirements Specification Tritium Area and Gamma Monitoring Systems, NIF-5025942
- System Requirements Specification Stack Monitoring System, NIF-5025941
- System Requirements Specification Tritium Processing System, NIF-5026031
- Tritium Area Monitoring and Gamma Area Monitoring System Operational & Performance Qualification CTP, NIF-5030273
- Tritium Processing System Operational & Performance Qualification CTP, NIF-5029832

***Note:** Document is generally relevant to all CIS software products.

3 Management

3.1 Organization

Management positions within the NIF Projects organization are shown in Figure 3 including additional management elements affecting software quality. Roles and responsibilities are described in section 3.3.

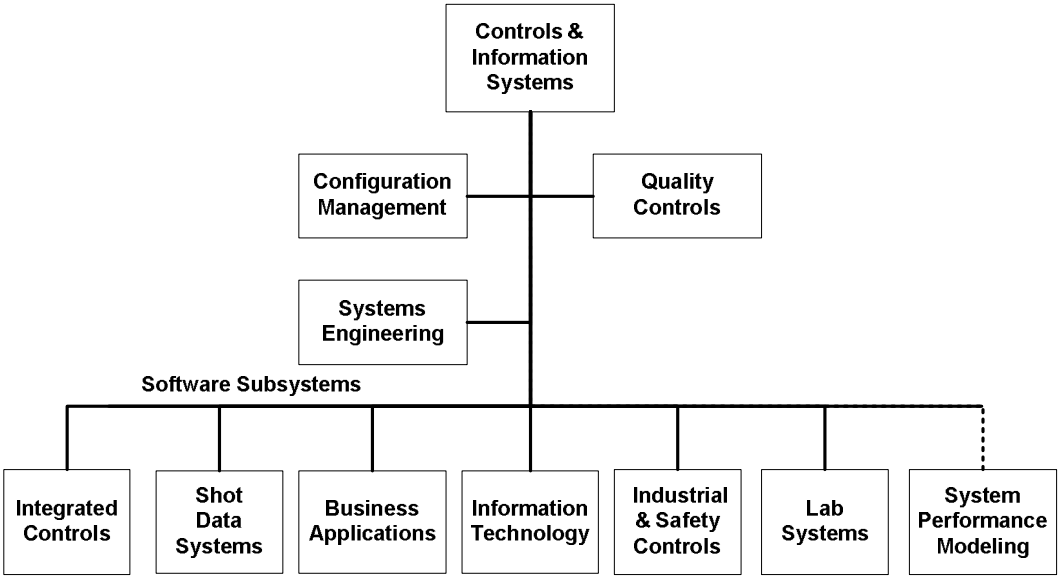

Figure 1. NIF Projects Software Subsystems Organization

By issuing this *SQAP*, NIF Projects management establishes the following:

- Responsibility and authority for developing and implementing a quality program for NIF software,
- The quality program meets all applicable codes, standards, and regulations,
- Effective Quality Assurance (QA) activities for software are to be implemented by CIS, its subcontractors, and suppliers,
- Issues adversely affecting quality are to be identified, and
- Corrective action is taken when necessary.

This *SQAP* will be reviewed and revised as necessary during the lifecycle of the software.

3.2 Tasks

Note: Tasks discussed in this section shall have additional QA rigor applied for higher levels of risk, including regression and off-normal test cases, as appropriate. See the *ISQAP* for guidance on the level of rigor.

3.2.1 Methodology

During the development part of the software lifecycle, CIS software engineering incorporates a risk-mitigation strategy of incrementally developing, integrating, operating, evaluating, and improving software. This strategy includes (where applicable) a sequence of prototypes demonstrated in integration labs followed by commissioning on-line to gain experience, resolve integrated system issues, and derive additional software requirements.

The repetitive development process delivers incrementally increasing levels of functionality sufficient to support the user's schedule while also incorporating tasks to address software risk. Each major software release incorporates a sequence of tasks to develop, test, and deploy the software. Figure 2 shows the full software development and quality control

process that should be used for complex releases. Each incremental release is planned for a work duration that ranges from weeks to months, depending on the complexity, and amount of coding, integration and testing required. The process is repeated as needed, with QA activities performed appropriate to the graded risk.

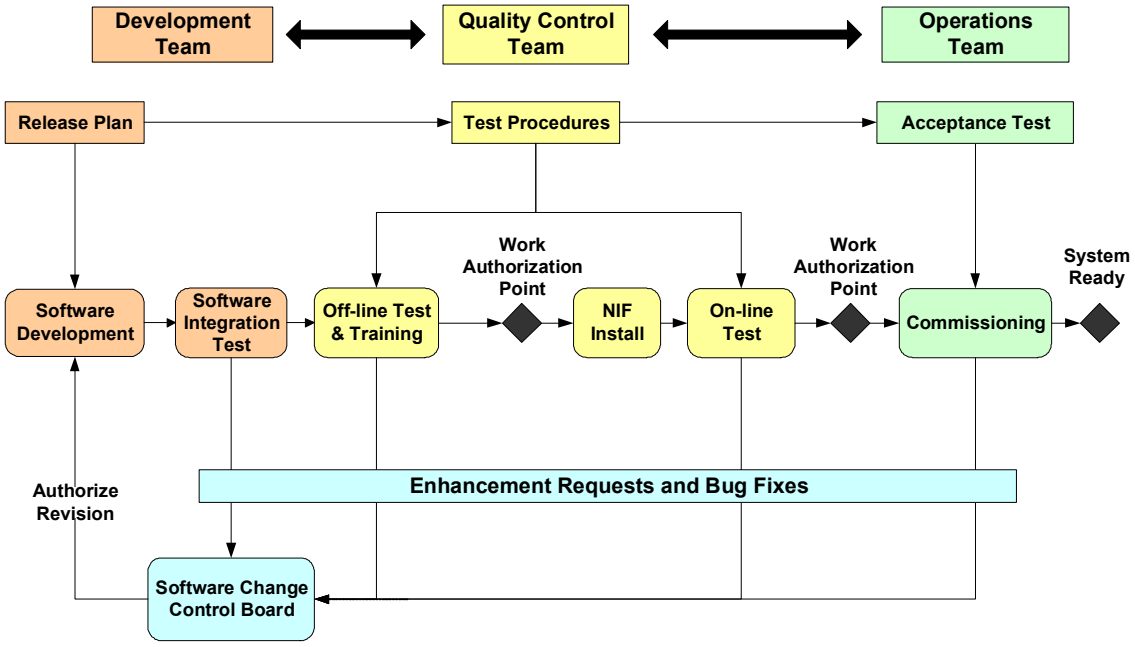

Figure 2. Software Development, Test, and Delivery Process

3.2.2 Software Planning and Development

All software development (e.g., new functionality or bug fixes) of CIS Software Products shall be managed through a formal Software Change Request (SCR) process using a defect tracking tool (e.g. JIRA) that is managed by a Software Change Control Board (SCCB), a Level-5 NIF change control board. Software subsystems having isolated impacts may have distinct subsystem SCCBs; however integrated products affecting multiple subsystems shall be concurred by an integrated SCCB and all applicable Integrated Product Teams (IPT).

Software subsystem managers will develop and maintain jobs, resources, and schedules in the NIF Planning System (NPS). Project tasks and milestones, system requirements, and pending SCRs will be evaluated to develop a *Software Release Plan* (SRP) comprised of one or more major software releases. The SRP should be reviewed with appropriate personnel including any other affected managers and Responsible Individuals (RI), Configured System Managers (CSM) and appropriate IPTs. The subsystem SCCB shall approve SCRs assigned to the SRP to be coded.

The SCCB may at any time direct the software team to prepare interim software releases needed to resolve outstanding issues with the deployed software (e.g., to repair critical software defects or implement important enhancements).

Developers working under the guidance of the cognizant Lead Engineer (LE) and Lead Architect will perform work in the development environment to design software that satisfies

documented requirements. Design should be documented within the design review process. Design reviews for major new software products or significant product upgrades should be conducted (ref. *Software Design Review Procedure*).

Software will be coded and unit tested by developers. New code should be desk checked by another developer prior to integration with other components (ref. *Software Desk Check Procedure)*.

3.2.3 Software Product Integration

After coding is substantially complete, developers will integrate software components together and perform integrated tests of the combined software in the associated integration environment. The purpose of integration is to find and resolve defects and interface issues, typically by running simulated system operations.

The Software Integration Manager should design the integration test. At a minimum, the integration test should verify that delivered SCRs work as intended (to the extent practical offline). The integration test should include selected regression tests to evaluate whether errors have been inadvertently introduced into previously delivered functionality. Critical requirements are normally considered for regression testing. Software defects found during integration testing are entered into the JIRA issue tracking system. CIS managers will review test results, Project priorities and quality requirements to determine which defects must be fixed and re-verified prior to exiting the product integration phase.

For software products that are planned to undergo independent verification, the integrated software release should be demonstrated to the Quality Control (QC) team as part of the hand-off to begin the offline test.

Upon completion of the integration test, the Software Configuration Management team will build and verify the formal software release package and deliver it to the test environment. Content of the release and included SCRs will be documented. The software release should be formally change-managed from the delivery point onward.

3.2.4 Offline Test

Where practical and indicated by assessed risk level, offline tests should be performed by the QC team. The primary objectives of these tests are to verify compliance of software with documented requirements and to obtain early design validation feedback. In many cases it is also important to provide offline training to operators.

To the extent practical, hardware prototypes and/or first article hardware items will be available in the testbed to permit testing. Where prototypes and first article hardware are not available, simulated hardware and/or device emulation will be used. The QC team is responsible for maintaining the essential integrity and/or calibration of equipment and software emulation used for formal testing.

Tests should be developed to systematically and rigorously exercise the delivered software using documented procedures to verify functionality, interfaces, and performance. In addition to testing newly delivered functionality, selected regression tests should be performed to verify that previously delivered functionality still operates as expected.

All test issues should be documented and appropriate SCRs generated. If the software under test are to be deployed to operate equipment in NIF or in hardware integration labs, any

urgent defects identified should be corrected and re-tested before offline testing is considered complete; any deviations from this policy must be agreed by the SCCB, affected RIs and affected CSMs.

A test plan should be prepared that addresses requirements to be verified, hardware, facilities and personnel needed to support the test, and identifies any safety hazards and plans for their mitigation. Tests are designed to verify the ability of the software to execute planned operational scenarios, as well as verify the expected response to selected off-normal scenarios involving system faults and/or human errors. The test plan should be presented at a Test Design Review (TDR) attended by project personnel with a vested interest in the test (i.e., appropriate software and hardware developers, facility owners, quality assurance, and safety representatives). Following the TDR, a software test procedure should be written that describes the detailed instructions for configuring, performing, and de-configuring from the test. The procedure should contain the test acceptance criteria and provide for recording test result.

3.2.5 Work Authorization Plans (WAP) and Work Permits

Work authorization plans are required to control significant transitions of software from development/test to an online operations environment (ref. *Work Authorization Review Procedure, NIF Procedure 5.19*).

Significant software releases are those that include major changes to the software and/or present risks that any software defect may have adverse consequences to personnel safety, the environment, security, equipment protection, or performance of other critical functions.

For significant software releases, a WAP checklist shall be prepared to assure that all applicable preparations and quality control measures have been completed, documented and approved. The WAP incorporates review of documents such as the software release plan, the offline test report, applicable failure modes and effects analyses (FMEA), and the online software test plan and procedure. Items on the WAP checklist must be delivered before the Authorizing Individual (AI) will approve software installation to start online testing. Approval to proceed with installation and online testing is documented on a NIF Work Permit (ref. *NIF Project Site Work Permits, NIF Procedure 5.8*).

Many software changes are sufficiently minor that a full WAP is not required; these changes can be approved through the work permit process. Examples of work-permit-only authorizations are software patches covered under an existing WAP or database parameter changes made in support of commissioning or maintenance.

3.2.6 Installation

Installation and online testing of a software release is scheduled under an approved work permit that is consistent with the overall plan for facility utilization. The Software Configuration Management (SCM) team will assure that the release can be reverted to the previous version if necessary. The SCM team installs the software and verifies the release content and location.

Where applicable, the Database Management Team will install and verify modifications to the database structure as well as populating and/or modifying table data.

The installation of OTS software should be verified separately, unless it is part of a larger release package that is verified during an integrated test.

3.2.7 Online Test

3.2.7.1 ICCS Online Tests

The online test shall verify that the software release functions correctly for its intended purpose after it is installed in the facility. An online test plan shall be prepared that

- Lists requirements and SCRs to be verified,
- Includes the software test procedure and schedule,
- Identifies hardware, facility, personnel resources needed to support the test, and
- Identifies equipment hazards and personnel safety hazards along with plans for their mitigation.

The test plan is briefed to management and test participants at a Test Readiness Review (TRR).

A major difference between offline and online testing is that the Operations team operates the new software with the QC team in a supporting role. All control room shot operations are under the direct control of a Shot Director assisted by the Lead Operator.

Online tests are performed in NIF immediately following successful software installation. The primary objectives of these tests are to verify that:

- No new defects are introduced to previously deployed software,
- The Configuration Database is correctly set up,
- Newly delivered Software Change Requests function correctly,
- Integrated shot controls function correctly for normal operations,
- Off-normal conditions are handled as expected, and
- Operators are familiar with operating the new software.

Following installation, the Duty Engineer starts the software and the preliminary test led by the QC Team is conducted, beginning with regression tests and followed by verification of SCRs as indicated in the test plan. All test issues are documented in LoCoS and associated SCRs are filed in JIRA. Meetings are held between operators, QC personnel, CIS developers / engineers, and RIs during the test to assure that problems are understood and that advancing to more complex phases of the test is safe and appropriate. At any time the test may be canceled, or the test may be suspended to obtain a critical patch needed to fix a serious software defect.

Once the preliminary online test is completed, the test RI, software AI, and Shot Director will review the test results to determine whether to approve continued testing under shot conditions. This may include incorporating suitable workarounds (within the established rules of engagement) to deal with unresolved software issues. The final test phase incorporates both normal and off-normal test cases.

Additional operator training is typically conducted during the online test to gain familiarity with the integrated operation of the software release in the context of complex procedures such as system shots.

3.2.7.2 Shot Data Systems Online Tests

Online tests shall verify that the software release functions correctly for its intended purpose after it is installed in the facility. An online test plan shall be prepared that

- Lists the functionality to be verified either as requirements and/or SCRs,
- Includes the software test procedure and schedule,
- Identifies personnel resources needed to support the test, and
- Identifies equipment hazards and personnel safety hazards along with plans for their mitigation.

SDS online tests are performed by QC personnel immediately following successful software installation. The primary objectives of these tests are to verify that:

- No new defects are introduced to previously deployed software,
- Newly delivered Software Change Requests function correctly, and
- Off-normal conditions are handled as expected.

All test issues are documented in LoCoS and associated SCRs are filed in JIRA. Meetings are held with QC personnel, CIS developers / engineers, and RIs during the test to assure that problems and workarounds, if appropriate to support operations, are understood and disseminated. At any time the test may be canceled, or the test may be suspended to obtain a critical patch needed to fix a serious software defect.

3.2.7.3 Business Applications Online Tests

The online test for Business Applications shall verify that the software release functions correctly for its intended purpose after it is installed in the production environment. An online test plan shall be prepared that:

- Lists the functionality to be verified either as requirements and/or SCRs.
- Includes the software test procedure and schedule.
- Identifies personnel resources needed to support the test.
- Identifies equipment hazards and personnel safety hazards along with plans for their mitigation.

Business Application online tests are performed by QC personnel and/or end users immediately following successful software installation. The primary objectives of these tests are to verify that:

- No new defects are introduced to previously deployed software.
- Newly delivered Software Change Requests function correctly.
- Off-normal conditions are handled as expected.

All test issues are documented in JIRA. Meetings are held with QC personnel, CIS developers / engineers, and RIs during the test to assure that problems and workarounds, if appropriate to support operations, are understood and disseminated. At any time the test may be canceled, or the test may be suspended to obtain a critical patch needed to fix a serious software defect.

3.2.8 Operations and Maintenance

CIS will assure the quality of the delivered software, configure software for use, install databases and scripts, provide technical support, participate in online tests, diagnose system problems, develop performance and reliability metrics, and maintain software change

control. Database management for the online control system is performed by the Database Management Team. Data entry will be performed by qualified staff.

> **Note:** Software and data deployed to NIF facilities shall not be changed without formal operations management approval.

The ICCS SM uses the *ICCS B581 Production Environment Database Change Request* and *ICCS B581 Production Environment Software Change Request* change control forms under the work permit process to coordinate changes and obtain approvals.

3.2.9 Software Change Control

Software source code and constituent libraries and databases shall be change-controlled using appropriate tools for source code and data management.

Each software subsystem may utilize tools optimized for the particular subsystem environment. The software configuration management plan (SCMP) establishes methodology to control the software development process, provide visibility into software status, and protect the integrity of the software product over its entire lifecycle. Refer to the applicable subsystem SCMP and associated procedures.

> **Note:** The *NIF wiki* https://nif-wiki.llnl.gov documents software configuration management policies and procedures that are used to manage computer software for the Integrated Computer Control System, Shot Data Systems and Business Applications. SCRs that are levied against the software code base are reviewed by the SCCB in order to evaluate the requests and recommend a technical course of action. The SCCB also assigns SCRs to specific software releases.
>
> **Note:** The *NIF Software Configuration Management Plan for Industrial Controls* documents change practices for the Industrial and Safety Controls software.
>
> **Note:** The *NIF Project Control Procedure 5.27, LPOM Change Control* documents change practices for the Shot Performance Modeling systems software.

3.3 Roles and Responsibilities

Roles and responsibilities for the CIS organization are briefly discussed below.

3.3.1 Associate Project Manager for CIS Software

The APM is responsible for overall leadership of CIS software including personnel management and establishing standards and expectations for design, coding, unit testing, and integration. The CIS APM also assures adequate support is provided for tools, offline facilities, independent testing, installation, commissioning, operation, and maintenance activities.

3.3.2 Software Subsystem Managers

Software Subsystem Managers are responsible for the overall design, development, and maintenance of software in their respective subsystem architectures.

3.3.2.1 Integrated Controls Manager

The Integrated Controls software subsystem manager is responsible for the NIF control system comprised of front-end processor, instrument-based-controller, supervisory, and server software.

3.3.2.2 Shot Data Systems Manager

The Shot Data Systems software subsystem manager is responsible for the shot planning, laser configuration, optics refurbishment planning, work control, problem reporting, and target diagnostics data analysis software.

3.3.2.3 System Performance Modeling Manager

The System Performance Modeling software subsystem manager is responsible for physics modeling software for the laser system and laser operation.

3.3.2.4 Information Technology Manager

The Information Technology software subsystem manager is responsible for infrastructure software, including network services, storage services, virtualization services, cyber security services, monitoring tools and operating systems.

3.3.2.5 Industrial and Safety Controls Manager

The Industrial and Safety Controls software subsystem manager is responsible for the Safety Interlock System, Industrial Control Systems, Facility Monitoring, Hazardous Materials Management System, Transport and Handling Controls, the Heating, Ventilation and Air Conditioning (HVAC) system, and the Access Control System.

3.3.3 Database Manager

The Database Management manager is responsible for database architecture, schema design, data initialization, and maintenance for all software subsystems (where used).

3.3.4 Quality Controls Manager

The Quality Controls (QC) manager is responsible for verifying that all CIS products satisfy functional, performance, and interface requirements. The QC group operates and maintains the Integration and Test Facility, which supports both development and formal offline testing activities. The QC team will perform a combination of offline and online tests to characterize the software and evaluate functional, performance, and off-normal behavior before the software is used in NIF operations. Final control system validation during commissioning is principally the responsibility of the Commissioning and Operations organization with support from the QC team.

3.3.5 Software Configuration Manager

The Software Configuration Manager is responsible for developing/maintaining the software configuration management process and tools, assisting Change Control Boards, generating software deployments from sources, assuring the completeness of each software deployment, installing and validating the composition of releases in test and operational environments, developing tools for SCM, and controlling changes to the configuration-controlled software resources.

3.3.6 Lead Engineers

Lead Engineers (LE) are responsible for managing the software development for specific portions of software subsystems, including requirements analysis, design, coding, unit testing, integration, and supporting formal testing and operation/maintenance of the software. Lead Engineers will participate in appropriate Integrated Product Teams (IPT), and other

teams as appropriate, to determine appropriate requirements, plans, and tests. The LE has principal responsibility for resolving technical issues, assigning product SCR tasks to developers, and monitoring progress.

3.3.7 Lead Architects

The Lead Architect (or where applicable the Subsystem Lead Architect) is responsible for assuring the uniformity and integrity of the overall software architecture, including the frameworks, databases, and tools used to build the integrated software system. The Lead Architect will perform these functions during design efforts, reviews, tests, change control, and troubleshooting. The Lead Architect will oversee preparation and maintenance of the Software Architecture Description that guides software development.

3.3.8 Software Integration Managers

The Software Integration Managers (SIM) are responsible for planning and leading the effort to assemble the various software applications into an integrated system capable of conducting coordinated laser shots and target experiments. The principal activity is software integration and testing, which is performed by members of the development team with support from the verification team. The SIM will develop a software integration plan for major software releases. The goal is to deliver a working, formally released software build to the offline testing phase. However, the management team may elect to accept known defects and use the software when the potential consequences are deemed acceptable by affected RIs and operations managers.

3.3.9 CIS Operations Manager

The CIS Operations Manager (OM) is responsible for planning and leading the effort to maintain operations of computer hardware and software applications that to support commissioning, maintenance and conducting coordinated laser shots and target experiments. The OM will assure customer interactions and feedback, organize troubleshooting, develop and maintain support staff in the NIF control room, and support operations of information technology infrastructure that is critical to proper function of the software maintained under this *SQAP*.

3.3.10 Subsystem Managers

Subsystem Managers (SSM) are responsible for the configuration, operation, reliability, and maintenance of a NIF subsystem such as ICCS or SIS. SSMs serve as technical customer for the software products and report to the NIF Operations Manager.

3.3.11 Configured System Managers

Configured System Managers (CSM) are knowledgeable and responsible for performing tasks related to maintaining the critical (safety or other) functionality of their system. Tasks include approving all change requests, concurring on all work permits, understanding and reviewing failure modes, and maintaining the system configuration. This position requires successful completion of the *NIF Operations Qualification Card Training Package NIF System Manager for Configured Systems – CM and Safety Basis*.

3.3.12 Duty Engineer

The Duty Engineer (DE) is responsible for operating the ICCS control system software from the NIF control room. The DE is trained and qualified to monitor, start and stop the control system supervisors, servers, and front-end processors. The DE is responsible for supporting control system users and other operators, logging issues in LoCoS and making authorized changes to control system configuration data in files and the online database. The DE reports to the CIS Operations Manager.

4 Documentation

Note: Where applicable for **Safety Software**, a Software Safety Plan shall list additional documentation requirements.

Note: Software documentation artifacts that are components of Configured Systems shall be maintained in ECMS as noted in the applicable Configured Item list.

4.1 Software Requirements Description

The NIF Project manages key attributes, including software requirements, in the NIF Requirements Management System database and/or configuration-controlled ECMS documents. Key attributes are formally managed by the NIF Level 4 Change Control Board.

Software requirement specifications are derivatives flowed down from RMS requirements that more fully specify functional, interface, performance, and other specifications for each product. RQ-Level 1 or 2 software requirements shall be managed and traceable to either RMS requirements or ECMS key attributes through the design, implementation, and test cases.

Lower-level software requirement specifications should be documented for designated software products. Software requirements should address functionality, external interfaces, performance, attributes, and design constraints.

4.2 Software Architecture Description

The Software Architecture Description guides how the software subsystem is to be constructed from common components, services, and patterns in the context of the computer network architecture and how it will collaborate with other software subsystems. The goal is to reduce complexity and cost, while increasing reliability and maintainability. The software architecture commonly includes a software framework of distributed software classes and centralized servers incorporated in a hierarchical architecture that provides a common way to implement the software product's many parts.

Note: The *ICCS Software Architecture Description* discusses the ICCS frameworks and guides how NIF control system front-end processors, supervisors, GUIs, and automated shot controls are to be built from the common framework.

Note: The *NIF Industrial Controls Common Requirements and Style Guide* discusses how programmable controller-based software systems are to be built.

4.3 Software Design

Software Designs (SD) specify how software products will be structured to satisfy the requirements. Software products are typically one or more processes that work in concert

with other distributed processes and servers to achieve an overall integrated functionality. The SD should include appropriate class diagrams, sequence diagrams, concurrency diagrams, public client interfaces, database components, error handling, and a discussion of key design decisions. The SD should demonstrate conformance to the software subsystem architecture and that the design meets performance requirements.

4.4 Software Release Plan

The Software Release Plan (SRP) describes the work required for coding, unit testing, and integrating one or more major incremental software releases. The SRP lists products, tasks, SCRs, and personnel assignments needed to implement and deliver software for a subset of new (or modified) functionality to meet software requirements. The SRP can be shown in the format of a Microsoft Project schedule, or an equivalent list of tasks and milestones. SCR release assignments should be maintained in a change management tool (e.g. JIRA).

4.5 Software Test Procedure

The Software Test Procedure (STP) describes detailed instructions for configuring, performing, and de-configuring from the test. The procedure should implement the test plan by specifying functional, regression, and off-normal tests along with test acceptance criteria and provisions for recording test results.

4.6 Software Configuration Management Plan

The *NIF Project and Facility Configuration Management Plan* describes methods of controlling the NIF configuration including artifacts for the CIS organization. The NIF CMP describes the relations among the Project's baselines, design controls, physical audits and assessments, and the engineering change management process.

Due to the specialized nature of software development, compliers and machine-readable software artifacts, ancillary software configuration management plans (SCMP) and procedures shall be developed to manage source code, libraries, databases, and data files. Software sources will be configuration managed using commercial or open-source management tools appropriate for the specific development environments, target architectures, product scale, complexity, and risk level.

> **Note:** The online NIF wiki resource *https://nif-wiki.llnl.gov* covers software subsystems for Integrated Controls, Shot Systems, Business Applications, Information Systems and Lab Systems. Changes to software shall be controlled by the SCCB using Software Change Requests that are managed throughout the software lifecycle using the JIRA software change management system. The NIF Wiki is approved by the Software Engineering Manager.
>
> **Note:** Software configuration management of the SIS and other industrial controls PLC-based software uses specialized tools that are covered under the *NIF Software Configuration Management Plan for Industrial Controls.* This plan is approved by the ISC Manager.
>
> **Note:** Software configuration management of the Laser Performance Operations Model (LPOM) is covered under the *NIF Project Control Procedure 5.27, LPOM Change Control*. This plan is approved by the SPM Manager.

4.7 User Documentation

User documentation should be prepared that describes the software operation, required operator inputs, program-generated outputs, and the activities necessary for the successful configuration, start-up, and shutdown of the system. When the operator interface employs a graphic interface for interaction, online help should be provided to convey significant instructions to the operator. Off-normal situations, error messages and corrective actions should be identified.

> **Note:** Where applicable for **Safety Software**, the associated Software Safety Plan may require specific user documentation and/or training.

5 Standards, Practices, Conventions, and Metrics

CIS software engineering is guided by a set of standards, practices, and metrics. These are generally derived from IEEE standards and commonly accepted industry standards.

5.1 Standards and Practices

Software design and coding will be performed using approved methodology, languages and tools. The Lead Architect may grant exceptions with concurrence of the SCCB.

> **Note:** Additional guidance and useful templates are available online at the *NIF Wiki https://nif-wiki.llnl.gov*.

5.1.1 Source Code Comments

Each time a software source is modified, a revision comment should be added to the header documentation indicating the nature, date, and author of the revision. The revision author is expected to update source code comments as necessary to maintain the accuracy of the comments in the context of any code revisions.

Software developers are expected to add comments that clearly indicate the purpose, usage, and operation of the software module. In addition, every source-code file (that is not auto-generated) should contain header documentation that includes the following identifying information:

- Name of the software component,
- Creation date,
- Name of the original author,
- Copyright information (originating organization, date of initial copyright),
- Name of modifying author,
- Date of latest modification,
- Software purpose description, and
- Block of revision history entries.

5.1.2 Integrated Computer Control System

Software for the Integrated Computer Control System will conform to the architecture, services and protocols provided by the ICCS Software Frameworks (see *ICCS Software Architecture Description*). Software designs will be expressed in Unified Modeling Language (UML) methodology.

Front-end processor and supervisor software will generally be programmed in the Java or Ada languages. Graphical user interfaces, database access, test scaffolding, stand-alone tools, and target diagnostic embedded controllers will be programmed in the Java language. Distributed interfaces will be coded in CORBA Interface Description Language except when other protocols (e.g., TCP/IP) are required and approved. Image and signal processing algorithms may be coded using higher-level tools such as IDL (Interactive Data Language), Matlab or with standardized libraries and tools.

> **Note:** languages, tools, development environments, libraries, target architectures, etc., must be approved by the Lead Architect and concurred by the SCCB.

5.1.3 Industrial and Safety Controls

Software for Industrial and Safety Controls shall utilize Allen-Bradley Programmable Logic Controllers (PLC) and MS Windows-based Rockwell Automation software tools. See *NIF Industrial Controls Common Requirements and Style Guide* and *NIF Safety Interlock System Requirements Specifications.*

5.1.4 Shot Data Systems

SDS Software will conform to the architecture, services and protocols defined by the Lead Architect and the SCCB. Software designs should be expressed in Unified Modeling Language (UML) methodology.

Software (such as but not limited to Graphical user interfaces, database access, test scaffolding, stand-alone tools, process scheduling) will generally be written in the Java language or in XML data grams. Image and signal processing algorithms may be coded using higher-level tools such as IDL (Interactive Data Language), Matlab or with standardized libraries and tools.

> **Note:** languages, tools, development environments, libraries, target architectures, etc., must be approved by the Lead Architect and concurred by the SCCB.

5.1.5 Business Applications

Business Applications Software should conform to the architecture, services and protocols defined by the Lead Architect and the SCCB. Software designs should be expressed using Unified Modeling Language (UML) best practices.

Software (such as but not limited to graphical user interfaces, database access, test scaffolding, stand-alone tools, process scheduling) should generally be written in the Java language or in XML data grams. Software written to extend or enhance specific COTS applications, e.g. Glovia, should be written in the language appropriate for the application.

> **Note:** Languages, tools, development environments, libraries, target architectures, etc. shall be approved by the Lead Architect and concurred by the SCCB.

5.2 Metrics

Metrics will be developed to help measure the quality and reliability of CIS software products and to help promote productivity and efficiency improvements in the processes by which software is developed, reviewed, tested, and operated. Generally, the JIRA tool, software configuration management tools, automated test tools, and various code analysis tools will be the primary means for collecting metrics data on software development, testing,

and operations. The LoCoS problem reporting tool will be the primary means of collecting reliability data from the field during NIF operations.

Guidance for developing and interpreting measures and metrics are given in the *Software Quality Metrics Plan.* Metrics collected and analyzed include:

- Source code inventory by product, language, and layer,
- Effort required to perform reviews and inspections, and
- Number, kinds, and density of defects identified and process stage where identified.

6 Software Reviews and Assessments

CIS will use reviews, inspections, and assessments to help assure quality goals are met, processes are followed, and products meet requirements and customer expectations. These practices are intended to augment testing, which is the primary quality control activity.

6.1 Design Reviews

Reviews will be organized, conducted, reported, and tracked (e.g. *Design Review Procedure*). Reviews will be held during the software design process to:

- Evaluate the technical adequacy of software designs to meet functional and performance requirements,
- Satisfy cost and schedule constraints,
- Assure applicable standards and architecture are followed,
- Verify software and hardware interfaces to other subsystems,
- Assure review and management of applicable Configured Items,
- Review key class structures,
- Evaluate unit test plans, and
- Examine sequence diagrams, concurrency diagrams, or other supporting information.

6.2 Technical Reviews

CIS will use inspection techniques to implement technical reviews of code. Code inspections will be applied to selected portions of the CIS software product. The degree of quality assurance rigor deemed necessary by the management team will determine whether a desk check inspection or more thorough code review will be performed. CIS software and reliability metrics will be used to guide the management team in the selection of code for technical reviews.

6.2.1 Desk Checks

Desk checks are an inspection technique for confirming that code changes are appropriate, complete, and documented before further software integration is performed. A person that did not develop the code will perform the desk check. Desk checks will be identified, conducted, reported and corrective actions tracked (e.g. see *Software Desk Check Procedure*).

6.2.2 Code Reviews

Certain code will be selected for in-depth review by a review team according to risk factors such as complexity, RQ-Level 1 or 2 requirements, Configured System Item status,

likelihood of latent defects anticipated from historical data, or other concerns. Code reviews will be organized, conducted, reported and corrective actions tracked.

6.3 Other Review and Audits

6.3.1 Management Self Assessments

The CIS organization will participate in periodic self-assessments conducted at the request of Project management. When so determined, internal assessments are conducted using *Management Self-Assessments, NIF Procedure 9.2*.

As deemed necessary, CIS managers will also perform assessments against the execution of the tasks and practices identified in this *SQAP*. These reviews may be focused to examine particular products or processes, and/or the activities of specific workgroups. In general, the management personnel having direct responsibility for the system will carry out these reviews. Corrective actions will be tracked according to Procedure 9.2.

6.3.2 Independent Assessments

The LLNL Quality Assurance Office is responsible for conducting independent assessments. The Directorate Assurance Manager / Directorate Operations is responsible for supporting the conduct of independent assessments and responding to actions when required based on the assessment results. The Directorate Assurance Manager / Directorate Operations is responsible for retaining the results of independent assessments and assuring the required actions are entered into the Issues Tracking System (ITS). CIS line managers are responsible for developing and implementing corrective actions as required and assuring the assigned management actions are closed. See the *NIF QA Plan*.

7 Software Verification

Software shall be tested for defects and proper operation before being released for operations. Testing will include verifying OTS software, either used to produce the software or included to perform functions in the release, works as intended. For low risk software this activity can be performed by the same team that developed, or installed the software. Higher risk software should be independently verified prior to online use.

Formal QC tests should be conducted after developer and integration testing is complete. A review of requirements and features is obtained from the release plan, product integration results, developer notes, and software demonstrations.

For releases delivering substantial new functionality, a test design is created and presented at a test design review (TDR) in accordance with the *Test Design Review Procedure*. The TDR addresses requirements to be verified; the approach that will be used to verify these requirements (including identification of normal and off-normal test cases); the test schedule; facility, equipment and personnel resources required; and identifies any personnel or equipment safety considerations, along with plans for mitigation.

Following the TDR, meeting minutes are published that document agreements reached and any action items assigned. Following the TDR, a software test procedure (STP) is written describing the steps to conduct the test. Software test procedures are reviewed and approved by the Software Manager, hardware subsystem responsible individual (RI), and the QC Manager.

Following handover of a release to the QC team, the first phase of offline testing verifies each individual SCR (that can be functionally tested offline). Functional verification involves either executing existing test cases or creating new ones to exercise new features and fixes. Next, regression tests are performed using checklists of key features, with the goal of confirming that these features have not "broken" since the last deployment. The development team should verify SCRs that are unobservable by testing with inspection techniques. The QC team also verifies database modifications by inspection.

For software involving laser shots, the final phase of offline testing involves executing one or more shot test cases. These tests demonstrate integrated operation of the software of CIS, the campaign manager tool, the laser performance operations model, data archival/analysis software, and industrial control system integration. Typically, several shot test cases are run with emphasis placed on executing test cases similar or identical to those that will be employed in the facility during planned user experiments.

Test issues identified during formal tests will be documented and tracked. Software defects are documented in the JIRA tool for disposition by the SCCB. During test cycles, meetings are held periodically to review and prioritize open test issues. The Product Integration Manager participates in these meetings and coordinates developer resources needed to resolve high priority software defects. Toward the end of offline testing, end-users are brought in to help validate that the release will meet the needs of operations and to receive initial training on new features. Upon completion of offline testing, a Test Readiness Review (TRR) is held to review the test results, discuss open issues, and present plans for online testing. Approval to proceed to online tests is managed through the WAP and work permit process.

Online testing begins with regression testing of manual controls and verification of SCRs that can only be verified in the production facility. The online test culminates with one or more automated shots that demonstrate successful integrated operations. Upon the conclusion of testing, the QC team will summarize the test results and any deficiencies found along with a recommendation of whether or not to retain the new software release for operational use. The test AI and Operations management must concur with the recommendation. In the event the new release is determined not ready for use, the SCM and Database Support teams are responsible for reverting to the last approved release.

The QC team will issue and archive test reports that summarize the test results. All software issues shall be entered into JIRA. QC management should review test issues identified during the test, with the goal of identifying opportunities for improving work processes and quality.

8 Problem Reporting and Corrective Action

Each person in the NIF organization is authorized to identify conditions adverse to quality and has the responsibility to report them to appropriate management. Deficient conditions will be investigated and tracked to closure by the appropriate manager and/or RI.

Generally, the LoCoS Problem Log is used by online facility personnel to generate problem reports in the field. The LoCoS tool includes capability to notify responsible and affected individuals by email. RIs will review these problem logs in a timely manner in order to determine and initiate investigative tasks that could lead to corresponding Software Changes (SCRs) that are documented and managed using the JIRA tool.

Additionally, NIF Subsystem Managers and Configured System Managers will review periodic reports from LoCoS to determine potential defects. Deficiencies identified by tests, inspections, or reviews are also reported and documented using the JIRA tool.

JIRA maintains a database of software problems to be addressed along with the facility affected, when problems occurred in the software lifecycle, the personnel assigned, and status of corrective actions. SCR data should be periodically analyzed for possible trends and used to identify opportunities for quality improvement. This data should be published in a quality metrics report, which should include management assessment and planned corrective actions.

CIS personnel should use the NIF Wiki for internal team communications by posting technical questions and answers directed at improving quality and efficiency.

9 Tool Support and Approval

The following table lists primary tools approved for developing, testing, and operating CIS software. The SCCB manages tool approval.

Table 2. CIS Software Tools

Software Tool	Intended Use	Applicability	Limitations
Accurev	Configuration Management	Source code CM tool	Preferred
Asset Centre	Configuration Management	Source code CM tool	Industrial and Safety Controls
Clear CASE	Configuration Management	Test case CM tool	Not for general source code
CVS	Configuration Management	Optional source code CM tool	
Glovia Development Control	Configuration Management	Source code CM tool	Specific to Glovia code
Maven	Configuration Management	Java build tool	
Perforce	Configuration Management	Source code CM tool	System Performance Modeling, ECMS
Oracle	Database	Database management tool	Preferred
AJAX	Development	Java GUI tool	
Apex/Summit Ada IDE	Development	Ada development	Not for new systems
Artisan Software Modeling	Development	UML design tool	
Atlassian JIRA	Development	Issue / Defect Tracking	Preferred
Business Process Execution Language (BPEL)	Development	Process control	
Cruise Control	Development	Build engine	
Eclipse Java IDE	Development	Java development	
GNAT Pro Ada	Development	Ada development	Preferred

HTML	Development	Web page development	
Icefaces	Development	JSF Java framework	
ImageJ	Development	Image analysis tool	
Interactive Data Language	Development	Analysis tool	
Java SDK	Development	Java development	
jfreechart	Development	Java charting tool	
JIDE	Development	Java GUI tool	
JSP	Development	Java server page tool	
jviews	Development	Java GUI tool	
Labview	Development	Lab Systems	
Matlab	Development	Analysis tool	
Microsoft Visual Studio	Development	C code bindings	
Perl	Development	Scripting tool	
Roundup	Development	Issue / defect tracking	System Performance Modeling
RS View	Development	Supervisory and GUI controls	Industrial and Safety Controls
RSLogix 5000	Development	Allen-Bradley PLCs	Industrial and Safety Controls
Struts	Development	JSP web applications framework	
Web Logic	Development	Java EE platform	
XML	Development	Configuration files, distribution patterns	
JacORB	Distribution	Java CORBA tool	
ORBExpress	Distribution	Ada CORBA tool	
Apache	Operating Environment	Web server	
Linux	Operating Environment	All subsystems	
Microsoft Windows XP	Operating Environment	All subsystems	
Oracle Virtual Machine (OVM)	Operating Environment	All subsystems	
Solaris UNIX	Operating Environment	All subsystems	
VxWorks	Operating Environment	ICCS front end processors	
IBM Rational Functional Tester	Test	Test automation tool	

10 Media Control

CIS software releases are generally delivered over the network using protocols established by the NIF computer security plan. In cases where network delivery is not available, other means such as USB memory stick, CD-R or DVD-R discs may be used.

NIF Information Systems (IS) is responsible for maintaining all computer and network infrastructure, including commercial operating systems, fileservers, software development, configuration management, test and development tools. The appropriate Change Control Board will authorize all system configuration modifications, including upgrades to operating systems, networks, and database products to ensure quality and compatibility is maintained across the product line.

The CIS Information Systems Operations group shall develop, verify, and maintain periodic file system backups including a disaster recovery plan for CIS software assets. Backups of all software source files shall be made periodically and provisions shall be made to maintain copies of the backup in a second location. The backup and disaster recovery plan shall be documented, managed in ECMS, and approved by the NIF Information Systems Manager.

11 Supplier Control

Software that is provided to CIS by other organizations will utilize a software quality assurance and quality control program appropriate for the risk, complexity, and consequence of failure of the software being delivered. COTS software includes operating systems, computer-aided software engineering tools, analysis tools, databases, distribution tools, user interface builders and environments, Interactive Development Environments (IDE), test aids, code analyzers, and compilers. CIS configuration management and quality controls will include verifying COTS software is configured correctly and performs as expected.

CIS management should assure that appropriate maintenance programs are established for all critical COTS software. Maintenance programs should assure technical support, defect reports, corrective actions and updates.

Generally, COTS or open-source software will be acquired from vendors without specifically requiring demonstration of the vendor's quality control program provided the vendor has established a reasonable quality track record in the marketplace.

COTS **Safety Software** and **Safety-Related Software** require the vendor have a quality control program, certification, and/or substantial demonstrated quality track record that is appropriate for the intended purpose and assessed risk level.

Whenever an upgraded version of COTS software is incorporated into software engineering processes or software products, regression tests commensurate with the risk level will be completed to ensure that no new defects have been introduced.

12 Records Collection, Maintenance, and Retention

A system of managing records and documents that allows retrieving pertinent Project files has been established by the Project in *NIF Procedure 4.1, Document and Records Control*. These procedures apply to controlling the documentation for all work on NIF Projects, including that of the CIS organization. All requirements documented within the SDR and SSDR shall be managed in RMS.

> **Note:** Selected software artifacts designated as Configured Items (ref. *Configured Systems and System Manager List*) shall be maintained as prescribed in the *NIF Project and Facility Configuration Management Plan*.

CIS-developed software (e.g., source code, libraries, scripts, databases, and executables) are managed within the applicable subsystem software configuration management system or other database, as appropriate.

Key design documents, procedures, test reports, etc. shall be archived in the Enterprise Configuration Management System (ECMS). In-process working documents (e.g., viewgraphs, plans, design notes, and drafts of requirements, design descriptions, etc.) may be maintained on the NIF Server.

13 Training

CIS personnel who manage, perform, or verify work affecting quality must receive the appropriate orientation and training for doing their assigned work. Line organization managers and supervisors, in accordance with guidelines outlined in the *NIF & Photon Science Training Plan* and applicable Qual Cards will determine the degree and amount of training required.

Additional technical training for CIS personnel will be identified and accomplished as needed. Commercial vendors will be selected and utilized to the extent possible for conducting training in standardized or commercial tools such as programming languages, databases, object-oriented design, and real-time development systems. Specialized training in the application of software frameworks, software configuration, software operation, and testing methodology will be accomplished through a combination of required reading, lectures, web-based training, and mentoring provided by experienced team members. CIS personnel should also reference the NIF Wiki for additional information and training materials.

> **Note:** Where applicable, a Software Safety Plan shall designate additional training requirements.

14 Risk Management

The Systems Engineering organization manages overall risks to the Project. The CIS organization will participate in assessing and mitigating risk as part of the Project-wide Systems Engineering effort. Specific areas of risk in the software that may result in significant failure modes within other software, laser and target area equipment or operations will be identified and addressed by techniques including analysis, reviews, demonstrations, and formal testing.

Risks to the software will also be evaluated strategically within the context of planning incremental software releases. The incremental release cycle affords significant mitigation benefits by delivering portions of the software much earlier in the overall schedule and allowing time to characterize and address issues.

The management team will incorporate prototypes and/or pilots as necessary to achieve effective risk mitigation. As part of this planning process, the CIS management team will evaluate risks and schedule appropriate software development and/or testing necessary to identify and resolve important issues. The management team may also assign special projects to address specific areas of risk that are planned and executed independent of the primary software development cycle.

The management team will periodically assess the state of the software and computer industry to identify trends, obsolescence issues and potential for improvements. Areas that will be considered include vendors, open source, programming languages, protocols, operating systems, development and testing tools, design methodology, as well as computer hardware architectures. Alternative technologies will be introduced as appropriate to reduce overall risk or maximize benefits.

> **Note**: Where applicable, a Software Safety Plan shall designate additional risk management activities.

15 Acronyms and Terms

Acronyms and terms used in this *SQAP* are shown in Table 3.

Table 3. Acronyms and Terms

Acronym / Term	Expansion
AI	Authorizing Individual
API	Application Programming Interface
BA	Business Applications
CI	Configured Item
CMRL	Configuration Management Rigor Level
CORBA	Common Object Request Broker Architecture (middleware)
COTS	Commercial off-the-shelf
CS	Configured System
CSM	Configured System Manager
ECMS	Enterprise Configuration Management System
FMEA	Failure Modes and Effects Analysis
Graded Approach	The process by which the level of analysis, documentation, and actions used to comply with a requirement is commensurate with the relative importance to safety, magnitude of any hazard, lifecycle stage of the facility, programmatic mission, particular characteristics of the facility, likelihood of failure occurrence, and any other relevant factor.
GUI	Graphical User Interface
HMMS	Hazardous Materials Management System
IC / ICCS	Integrated Controls / Integrated Computer Control System
ICCS Framework	A set of re-usable, integrated software services for implementing the NIF distributed computer control system and the coding patterns necessary to build collaborating software applications using those services within a hierarchical architecture of supervisory software and front-end processors.
IDE	Interactive Development Environment
IDL	Interactive Data Language
IDL	Interface Description Language (part of CORBA standard)
IPT	Integrated Product Team
ISC	Industrial and Safety Controls
IT	Information Technology
ITF	Integration and Test Facility

ITS	Issues Tracking System
LE	Lead Engineer
LoCoS	Location Component State (web-based tool for NIF work management and problem reporting)
LPOM	Laser Performance Operations Model
LRU	Line Replaceable Unit
LS	Lab Systems
NIF	National Ignition Facility
NIF Risk Level	Assessed risk level for key attributes of Systems, Structures and Components
PLC	Programmable Logic Controller
QA	Quality Assurance
QC	Quality Control
Regression Testing	Selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with specified requirements
RI	Responsible Individual
RIMS	Radiological Information Management System
RQ-Level	Requirements-Quality Levels for RMS Requirements
SAD	Software Architecture Description
SSM	Subsystem Manager
SSW	Safety Software: Software that performs nuclear or radiological safety function, analysis, design, or administration (ref. DOE O 414.1c)
SRSW	Safety-Related Software: Software that performs (non-radiological) worker safety function, analysis, design, or administration
SCCB	Software Change Control Board
SCM	Software Configuration Management
SCMP	Software Configuration Management Plan
SCMS	Software Configuration Management System
SD	Software Design
S/SDR	System/Subsystem Design Requirements
SDS	Shot Data Systems
SIS	Safety Interlock System
SPM	System Performance Modeling
SQA	Software Quality Assurance
SQAP	Software Quality Assurance Plan
SRP	Software Release Plan

STP	Software Test Procedure
TCP/IP	Transport Control Protocol / Internet Protocol
TDR	Test Design Review
TRR	Test Readiness Review
UML	Unified Modeling Language
WAP	Work Authorization Plan

16 SQAP Change Procedure and History

This *SQAP* will be updated as necessary to incorporate changing management requirements, to refocus practices for evolving phases of the Projects, or to make improvements. Modifications will be prepared by CIS management or by the technical staff as directed, with appropriate review and approvals as shown in the approval section of this document. This *SQAP* shall be configuration managed in ECMS.

Revision No.	Effective Date	Brief Description
NIF-0000288OA	Oct. 1996	Initial issue of document
OB	Feb. 1998	Rewrite to address early production phase of project
OC	Oct. 2004	Major rewrite for current mid-phase / organization of the Project and conformance to requirements of the LLNL Institutional Software Quality Assurance Plan
OD	Oct. 2005	SQAP for the NIF Integrated Computer Control System
NIF-5022079AA	Sep. 2008	Major update supersedes ICCS SQAP NIF-00000288. Expands organization and software products to include application domains in control systems, systems engineering, information technology, shot systems, safety controls and industrial controls. Update incorporates revised ISQAP (Aug 2008) and revised NIF QA PLAN (Apr 2008). Added QA requirements for NIF Configured Systems.
NIF-5022079AB	Apr. 2010	Update for revised Controls and Information Systems organization; corrections and additions to supporting documentation references; change independent assessment responsibility to the institutional QA office; and additions to the software inventory