TABLE OF CONTENTS

Rev	Date	Description	POC	RM
0	06/23/16	Initial issue as provisional document.	Tobin Oruch, *ES-DO*	Mel Burnett, *CENG-OFF*
1	05/25/17	Made chapter mandatory. GEN forms renumbered. Added template 3046, *Software Risk Register*; DOE-STD-1073, *Configuration Management.* Clarified applicability and added Nuclear Criticality Safety, review of supplier software error reports, modified file numbering scheme, Less-Than-Minor Change definition, other minor changes throughout.	Tobin Oruch, *ES-DO*	Lawrence Goen, *ES-DO*

As with all LANL Engineering Standards, but especially with this new chapter, please contact the chapter POC with comments, issues, etc.

1.0 PURPOSE, APPLICABILITY, AND ORGANIZATION

1.1 Chapter 21 Purpose and Applicability

A. Purpose: Chapter 21 is the software plan for the Facility Conduct of Engineering program (FAC-COE), safety basis, and nuclear criticality safety activities. It provides reasonable assurance that in-scope software will consistently, compliantly, and efficiently satisfy its intended use. It describes the "how, when, and who" for implementing the software quality assurance (SQA) requirements[1] of:

1. DOE Order 414.1D, *Quality Assurance,* (hereafter O 414.1D),
2. ASME NQA-1-2008/NQA-1A-2009, *Quality Assurance Requirements for Nuclear Facility Applications, Part I and Part II* (hereafter NQA-1), and
3. DOE-STD-1073, *Configuration Management.*

B. Applicability:

1. For new and "modified" software, this chapter is required when software is within the scope of:
   a. PD340, *Conduct of Engineering for Facility Work*,
   b. PD110, *Safety Basis*, or
   c. SD130, *Nuclear Criticality Safety Program*

   "Modified" software includes Major Computer Program Changes (see Definitions). Both Major Computer Program Changes (including upgrades from non-safety to safety software) and software considered non-compliant[2] are subject to this chapter.

   Also included is Research and Development (R&D), programmatic and other engineering software that affects: (a) structures, systems, or components (SSCs) described in Chapter 3 of a Documented Safety Analysis (DSA)[3] and/or (b) adjacent SSCs[4].
2. The chapter defines and applies to two major types of software:
   a. System, structure, and component (SSC) software; and,
   b. Non-SSC software (see definitions), including "Simple and Easily Understood" software used in the design of SSCs.[5]

---

1 Requirements are clarified for LANL application in SD330 *LANL Quality Assurance Program*. Chapter 21 differences relative to P1040 *Software Quality Management* (SQM) and P840-1 *Quality Assurance for Procurements* are authorized by QPA-VAR-2016-005.1 (EMRef-75).

2 Compliant means meeting applicable DOE O 414.1D and NQA-1 SQA requirements, thus compliance with LANL P1040 or prior chapter revisions is also acceptable. When in doubt about compliance, assess against relevant requirements; Quality and Performance Assurance (QPA) may be able to assist.

3 Criteria are from PD340. If safety/technical baseline is not affected and the work is R&D engineering work, then the work is conducted to PD370, *Conduct of Engineering for Research and Development [R&D].* If safety/technical baseline is not affected and the work is other than R&D engineering work, then this chapter may be used if use is consistent with the governing quality assurance program. Safety SSCs described in Chapter 3 of the DSA are Safety Class, Safety Significant, or Other Hazard Controls (OHCs).

4 Adjacent SSCs are SSCs that are located adjacent to the safety or OHC SSCs in a nuclear facility such that changes to these SSCs could negatively impact the safety or mission of the activity. (adapted from DOE-STD-1073-2003, *Configuration Management).*

5 See definitions in SOFT-GEN Appendix A, *Chapter 21 Definitions and Acronyms*. Includes safety and non-safety software. Also, see applicability of quality assurance controls summarized in SOFT-GEN Attachment 1, *Chapter 21 Summary Table* and further details in Chapter 21 sections.

C. Exclusions: This chapter does not apply to:

1. Personal productivity and collaboration software that does not provide calculation output.

2. Software covered by existing, compliant software quality management plans/processes.[6] The issuance of this chapter does not necessitate immediate replacement of those controls (grandfathering applies) unless/until a Major Change.

3. Uncomplicated software tools, including productivity aids that are adequately controlled through the design process. *Examples include unit conversion software, spike programs, equipment selection software, diagnostic software, and interpolation calculators.*

D. Implementation:

1. LANL: Once the need to adopt this chapter for a given software is identified, initiate performance feedback in the Performance Feedback and Improvement Tracking System (PFITS) per P322-4. The PFITS action must require the software to be compliant with this chapter by a due date agreed to by both Software Owner Responsible Line Manager (SRLM) and chapter Point of Contact (POC). [7]

   Should this chapter conflict with other LANL Conduct of Engineering direction, contact the Chapter 21 POC for direction.

2. Subcontractors (architect-engineers, consultants, etc.): Chapter 21 provides minimum requirements for software quality assurance (SQA) including both SSC and Non-SSC software.

   a. For Non-SCC software (e.g., safety design or analysis), the chapter includes basic reporting requirements to facilitate LANL's oversight of project-specific determinations, etc. *It may also be used as an example of a comprehensive approach when assessing Subcontractor SQA programs.*

   b. For SSCs with software being specified or designed, and for any software being delivered for LANL ownership and use, Chapter 21 requires the same documentation deliverables that LANL personnel provide[8] (to the extent the information is known in the design and construction phases; when not known, insert TBDs and deliver as "Draft").

      Unless otherwise indicated in subcontract documents, the construction Subcontractor serves as the Software Owner (SO) up to the point of turnover. Upon turnover, the SO responsibility transfers to the LANL system engineer. The FDAR is the SRLM and retains that responsibility throughout the software life cycle however may delegate activities through subcontract to the Subcontractor.

---

[6] However, a software summary as described in SOFT-GEN Section 2.1 is required for new projects, including when compliant SQA programs other than this chapter are used.

[7] Six months is believed appropriate for most software while some may justify longer. Example wording: "For [software application], develop, issue and follow the SQA documents required by Engineering Standards Manual Chapter 21 – Software."

[8] For SSC software: Use of chapter forms/document numbers provides standardization and retrievability of documents that LANL must maintain going forward. As necessary, documentation will be updated or produced by construction subcontractors and verified by LANL in the start-up/commissioning process in conjunction with the system engineer function.

c. Subcontractor quality assurance programs must meet the requirements of this chapter, comply with O 414.1D, and—where required by subcontract—NQA-1, including Part I and Part II, Subparts 2.7 and 2.14.

E. Associated Management levels (MLs)—rather than software type, category, or Software Risk Level (SRL) terms—are used throughout.[9] Where ML-1, ML-2, ML-3 or ML-4 software is written, this is the associated software ML.

See Form 2033 for an ML crosswalk (guidance table) to software types, categories, and software risk levels (SRLs).

F. Several of the concepts discussed above are depicted in the following figure:

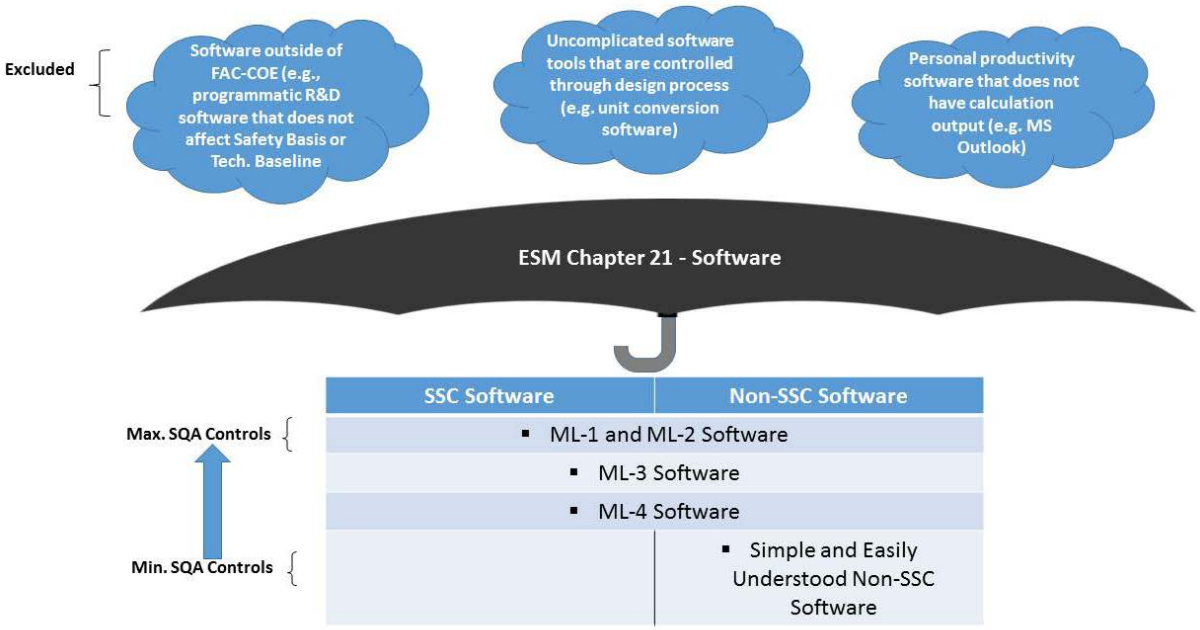


Figure 21.1-1 Chapter 21 Overview

1.2 Chapter 21 Use, Training, and Organization

A. Concept: Chapter 21 implements the SQA life cycle as shown in Figure 21.1-2, SQA Process Life Cycle.

[9] See Appendix A for definition. Promotes a common vernacular for users.

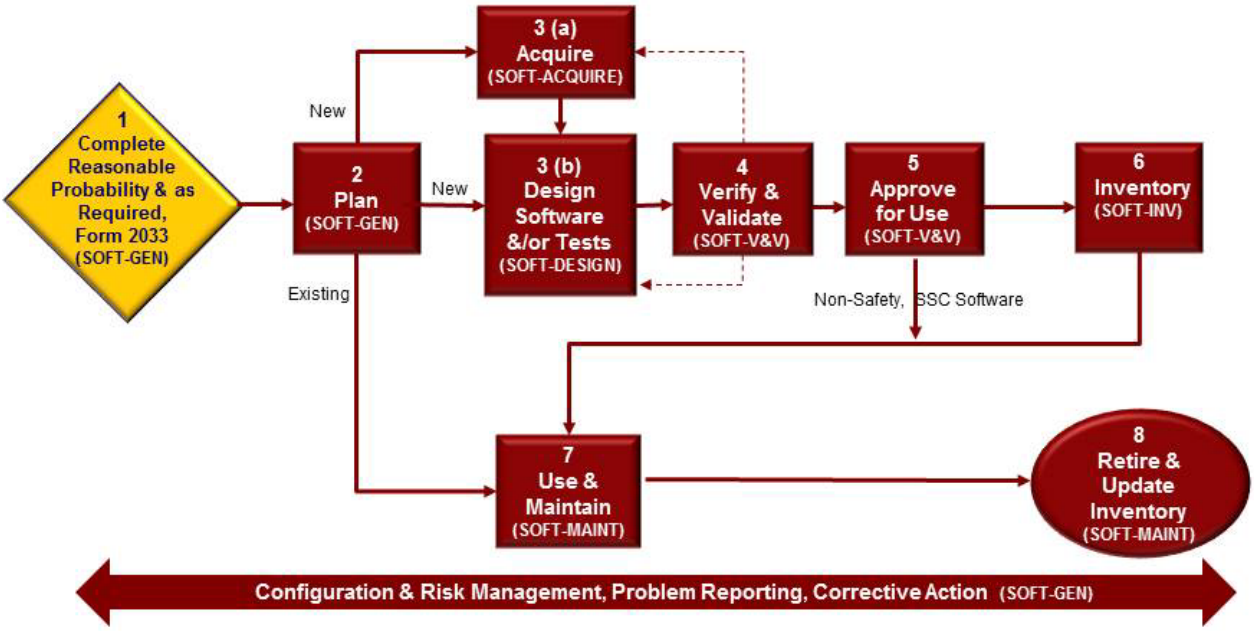


Figure 21.1-2 SQA Process Life Cycle

B. This chapter uses an umbrella concept to implement the SQA life cycle; previously presented Figure 21.1-1 depicts this concept at a high level, and there is more detail in Figure 21.1-3, *Chapter 21 Concept, Sections, and Deliverables*. That is, Chapter 21 serves as the software plan for software within its scope. Implementation of this chapter/plan will produce software-specific SQA implementation documentation (e.g., software baseline, software data sheet) for multiple software items. The software-specific documentation includes information that is unique[10] to each software program. Examples of software-specific information include the software technical requirements, software ownership, and software test plans.

C. Training:

1. Testing: For final SSC software testing in the operating environment that is used to verify the quality of ML-1 and ML-2 SSCs in Hazard Category II or III nuclear facilities, certify test personnel per applicable NQA-1 requirements.

2. LANL: Training requirements will be controlled by ESD Training Program Plan (ESD-PLAN-001, kept on the CoE T&Q webpage).[11] *Guidance: Live training has been developed for classroom training on this chapter.* For NQA-1 test personnel certification and details of maintaining certification, see P330-8, *Inspection and Test* and ESM Chapter 15 – *Commissioning*.

[10] Unique in that it cannot otherwise readily be addressed on a higher level (in this chapter proper). This centralized + distributed model ensures consistency of approach/rigor within user organizations while eliminating the production of standalone SQA plans that repeat boilerplate; it also utilizes a living, stand-alone documentation approach for those documents that are subject to change, easing revision of same without necessitating revision of an entire plan. Also, revision of Chapter 21 does not invalidate the approach or documentation associated with a previous chapter revision unless the revised chapter material specifically states this.

[11] Based on the JTA for Chapter 21 in ESD-SAT-006. For these and additional training requirements for implementing this chapter, use PD781-1, *Conduct of Training* and P343, Facility Engineering Training and Qualification Manual to establish and maintain them.

D. Organization: Chapter 21 is divided into six (6) sections that span the software life cycle: SOFT-GEN, SOFT-ACQUIRE, SOFT-DESIGN, SOFT-V&V, SOFT-INV, and SOFT-MAINT. These sections and a summary of deliverables are depicted in Figure 21.1-3*, Chapter 21 Concept, Sections, and Deliverables*.[12] Note, not all six sections will apply to every software used, and section usage may not be in sequential order. That is:

1. For all software, use SOFT-GEN, SOFT-V&V, SOFT-INV, and SOFT-MAINT.

2. For software that is acquired, also use SOFT-ACQUIRE.

3. For software that is designed (used interchangeable with developed), also use SOFT-DESIGN.

E. Definitions and Acronyms: See SOFT-GEN Appendix A, *Chapter 21 Definitions, Responsibilities, and Acronyms* for key definitions and acronyms. Defined terms are normally capitalized, set in quotes, or both in this chapter.

F. Work Summary: Refer to SOFT-GEN Attachment 1, *Chapter 21 Summary Table*, for a summary of deliverables for this entire chapter. Refer to the table(s) at the beginning of each section for a work summary of that section. These tables address both SSC software and Non-SSC software.

G. Italics: Throughout the ESM, italics are one method used to indicate guidance. Exception: When italics are used to highlight document titles.

[12] The 6 phases encompass all life cycle phases described by other industry documents in a simplified approach.

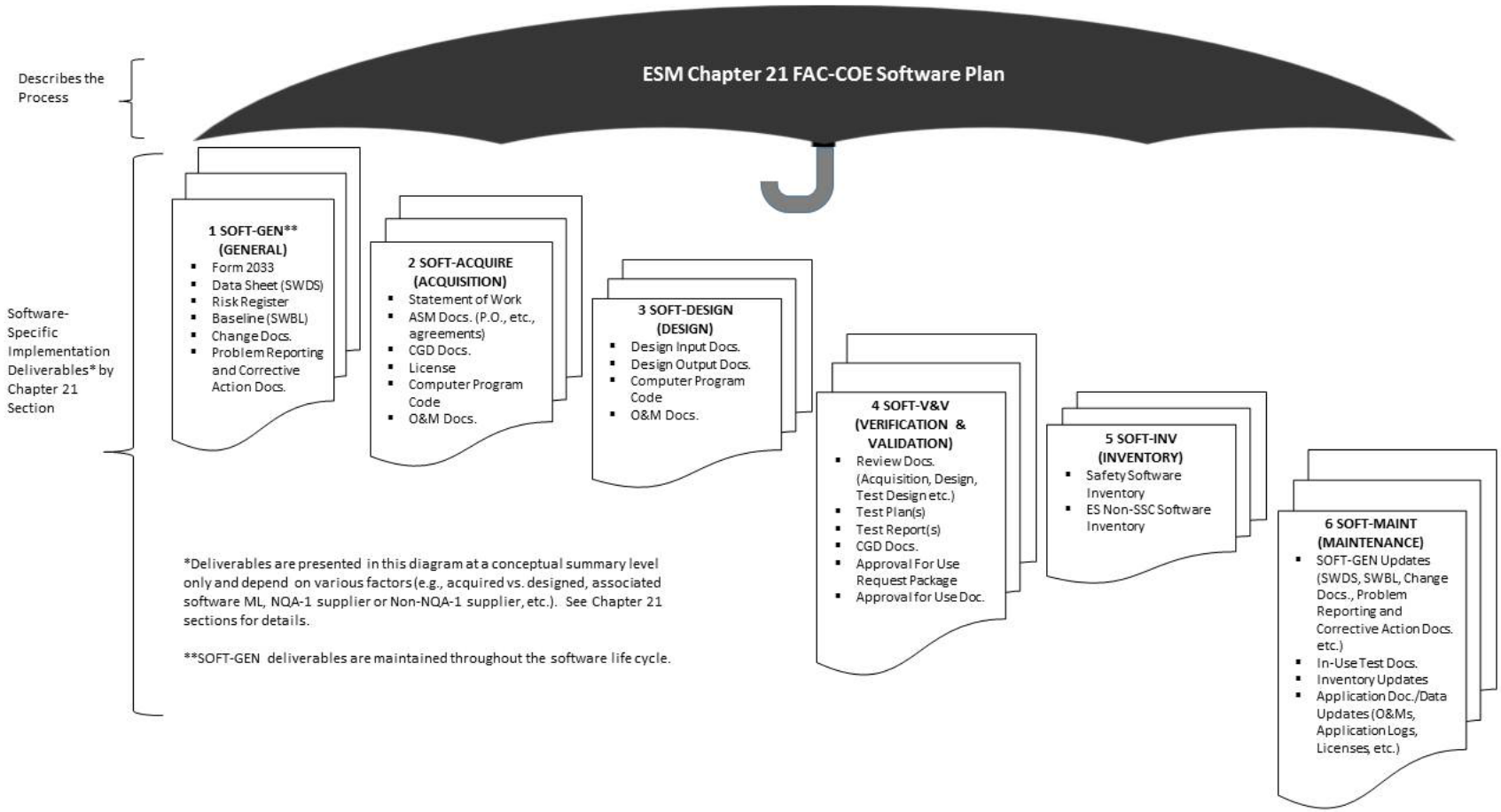


Figure 21.1-3 Chapter 21 Concept, Sections, and Deliverables

H. Roles, Responsibilities, Authorities, and Accountabilities: See SOFT-GEN Appendix B, *SO and SRLM Decision Diagram (Guidance)* when determining these key positions.[13]

I. "Or Equivalent": When this term is used, the SRLM determines equivalency. If questions arise regarding equivalency, the Chapter 21 Point of Contact (POC) will resolve.

J. Chapter References: See SOFT-GEN, Appendix C, *Chapter 21 Reference Listing* for hyperlinked references. Additional hyperlinks are provided in key locations throughout the chapter for convenience.

K. Forms:

1. LANL personnel: Endeavor to use chapter forms as-is and report issues and improvement ideas to the Chapter 21 POC.[14]

2. LANL subcontractors: For SSC software, use chapter forms or POC-approved Subcontractor forms to satisfy chapter requirements. For Non-SSC software, use either Subcontractor's own forms or integrate, adapt, and reformat the chapter forms. As determined by the Chapter 21 POC, either approach is acceptable so long as key functions, data, and approvals are retained.

3. In all cases, LANL Form 2033 must be used as-is.

4. Forms usage is summarized in Table 21.1-1 below:

Table 21.1-1 Form Usage Summary Guidance
See Appendix A for definitions of acronyms

Form	When Form is Initially Developed	When Form is Revised
2033	At earliest practical time but before acquisition and/or design	When the ***application*** of the software program significantly changes
SWDS	Earliest practical time but before acquisition and/or design as applicable	As required by SRLM to adequately manage software and/or per doc. control process[15]
SWBL	At earliest practical time but before acquisition and/or design	▪ Before acceptance testing ▪ After acceptance testing/SWAU ▪ After/with approved changes
SCP or SWNCP	When changes are needed	NA (usually not revised)
SWAU Docs. (e.g., SCP, Ch 15 Cx, PD115 Readiness Docs.)	▪ Prior to initial release (SWAU Form or Readiness Docs.) ▪ With changes (Change Docs.)	NA (usually not revised)

[13] The LANL Facility Design Authority (DA) is always authorized to act or sign for one or more DA Reps (FDARs).

[14] Forms are designated as samples which means minor aspects may be modified so long as all functions, key data, and approvals are retained. Alternatively, SRLMs may direct that other, comparable forms be used (with prior written permission of the chapter POC).

[15] Initial revision may have TBD for some parameters (e.g. use and maintain items) that are not known in the early planning stage and will be determined at a future date.

L. Chapter Revision: So long as SD330 compliance is maintained, the Chapter 21 POC is authorized to approve and issue revised forms, references, and SOFT-GEN Att. 1 independently of associated sections.[16]

1.3 Section SOFT-GEN Purpose and Applicability

A. Purpose: SOFT-GEN provides general software information and requirements to implement Chapter 21. These general requirements include how to: (a) identify software and make software determinations (e.g., Form 2033), (b) implement configuration management (e.g., baselines and software changes), (c) report problems/take corrective actions, and (d) perform software risk management.

B. See Table 21.1-2 for a summary of SOFT-GEN.

C. Applicability of SOFT-GEN: See the Chapter 21 applicability subsection above and the following:

1. For "Simple and Easily Understood" software that is used in the design of SSCs and that is individually verified per SOFT-V&V subsection 9, only the software identification and determination requirements of SOFT-GEN apply. If not individually verified per SOFT-V&V but rather pre-verified per SOFT-V&V subsection 6 or 7, then all requirements of SOFT-GEN apply.

   See other sections for applicability of the respective section for this software (e.g., SOFT-INV requires this type of software to be inventoried).

2. SOFT-GEN applies to subcontractors providing design agency or analysis services to LANL[17].

[16] Chapter 21 compliance to source document is determined by the Chapter 21 POC and periodically independently assessed to ensure compliance is retained. Other chapter section revisions follow the revision process described in ESM Chapter 1 – General, Section 100.

[17] SSC and Non-SSC software resulting from and/or used in the SSC design must be identified and software determinations completed as stated herein.

Table 21.1-2 Section SOFT-GEN Summary for SSC and Non-SSC Software
(This table is a summary only and does not include all requirement details. See text for details.)

Activity No.	SQM Activity	ML[1]		Implementation Detail			Reference
		1, 2, 3	4	How	When	Who[2,3]	ESM Ch. 21 Ref.[4]
1	Identify software ownership/software and complete software determinations	R	Gr	▪ Identify SO and SRLM ▪ Identify software and develop initial and updated software summaries ▪ Determine reasonable probability ▪ Complete and retain Form 2033 ▪ *RLMs issue data call for software*	▪ Earliest practicable time ▪ Before software design, acquisition ▪ *Identify software at least annually for software data call*	▪ SO (D) ▪ SRLM (R, A) ▪ FDAR (R, A) ▪ *RLM (D) for data call*	SOFT-GEN, 2
2	Prepare Software Data Sheet (SWDS)	R	-	▪ Develop Software Data Sheet (SWDS), SOFT-GEN-FM01	▪ Earliest practical time ▪ Before design, acquisition as applicable ▪ Maintain throughout software lifecycle	▪ SO (D) ▪ SRLM (R, A)	SOFT-GEN, 3.0
3	Develop and maintain software baseline	R	-	▪ Develop software baseline (SWBL) using SOFT-GEN-FM02 ▪ Implement a baseline labeling system ▪ Maintain baseline to ensure the most recent approved configuration	▪ Earliest practical time ▪ Before/after acceptance testing ▪ Before software approval for use (SWAU) ▪ After approved changes ▪ For ML-1/ML-2 designed software, at major design activity	▪ SO (D) ▪ SRLM (R, A)	SOFT-GEN, 4.0

Table 21.1-2 Section SOFT-GEN Summary for SSC and Non-SSC Software
(This table is a summary only and does not include all requirement details. See text for details.)

Activity No.	SQM Activity	ML[1]		Implementation Detail			Reference
		1, 2, 3	4	How	When	Who[2,3]	ESM Ch. 21 Ref.[4]
4	Control software changes (change control)	R	Gr	▪ For SSC software, prepare changes per AP-341-519 and/or AP-341-507-FM01; ▪ For Non-SSC software, prepare SWNCP per SOFT-GEN-FM03 ▪ Document changes (including comments in computer program code where possible) ▪ Make doc. changes per governing document control process ▪ Revise software baseline	▪ In a timely manner ▪ When required to ensure proper performance ▪ Per Table 21.1-4	▪ Per Table 21.1-4 or equivalent	SOFT-GEN, 4
5	Perform problem reporting and corrective action	R	Gr	▪ Before approval for use (SWAU), use P330-6, P322-4, less formal methods (e.g., bug lists, comments in program) Or Equivalent ▪ After SWAU, use P330-6, P322-4 or Equivalent	▪ Throughout software lifecycle	▪ SO (D) ▪ SRLM (R, A)	SOFT-GEN, 5
6	Perform software risk management	R	Gr	▪ Implement Chapter 21 ▪ Develop and maintain risk register (Template 3046 or equivalent); preferably as part of the SWDS	▪ Earliest practical time ▪ Before design, acquisition as applicable ▪ Maintain throughout software lifecycle	▪ SO (D) ▪ SRLM (R, A)	SOFT-GEN, 6

Notes:

[1] ML = Associated Management Level as per Form 2033. R = Required. Gr = Required but graded. "-" = Not required.

[2] D = Develop; R = Review; A = Approve.

[3] SO = Software Owner. RLM = Responsible Line Manager; SRLM = Software Responsible Line Manager. FDAR = Facility Design Authority Representative.

[4] Ref. = ESM Chapter 21 section name and subsection number (e.g., Section SOFT-GEN, Subsection 3, Software Identification and Determination).

2.0 SOFTWARE OWNERSHIP, IDENTIFICATION, AND DETERMINATION

2.1 Software Ownership/Software Identification (ML-1 through ML-3 only)

A. How

1. Using Appendix B, *SO and SRLM Decision Diagram (Guidance)*, identify Software Owners (SOs) and a Software Owner Responsible Line Managers (SRLMs).

Note: For ML-4 software, this chapter does not require software identification/summary; however, it may be invoked by the SRLM.

2. For Non-SSC (e.g., used for design and/or analysis), identify and summarize the software.[18]

   Develop a software summary includes the following: (a) software name, (b) software functional description (what it does), (c) software application (when/where/how used), (d) whether the computer program can be changed other than through replacement of the software (replacement includes software upgrades), and (e) whether the software is "Simple and Easily Understood" software that will be individually verified each time the software is used.

Note: **SOFT-GEN webpage references include an example software summary template**.

Guidance: SRLMs or Divisions should broadcast a software data call on a periodic (at least annual) basis to ensure software is identified and managed as required by this chapter.

3. For SSC software, identify and summarize the software as described above with the following clarification:

   a. Analyze the SSC to determine whether the SSC has software. Visually inspect and/or review vendor information (product specifications, catalog data, operations and maintenance manuals, etc.). Identify to a practical extent and to a level that the software can be controlled. *For example, for a control system module that has multiple pieces of software on a single module and the software cannot be modified other than through replacement of the entire module, then identifying the software to the module part number is appropriate.* Do not introduce a potential for adversely affecting manufacturer warranties, SSC damage, or unintended software changes by disassembling, etc.

4. Provide the summary to the LANL Software Responsible Line Manager (SRLM).[19]

B. When.

1. For Non-SSC software, provide the summary to the LANL SRLM at the earliest practicable time and prior to software acquisition or design (or use, if already on hand).

2. For SSC software, provide draft summaries no later than the 60% project design deliverables. When a required field on the software summary is not known, indicate TBD or similar term. Deliver the final summary (complete without TBDs)

[18] The software summary is not a controlled document or quality record. The responsible LANL SRLM uses this information to ensure that software determinations and subsequent required actions in this chapter are completed.
[19] LANL project engineer function must ensure review by appropriate SMEs and/or FDAR.

with the 90% detailed design deliverables. For a deferred[20] SSC design, include the final summary in the deferred design submittal(s).

C. Who. The Software Owner (SO) is responsible for software identification. See Appendix B for guidance on determining the SO. During the design phase, the design agency shall act for the SO. *SRLMs or Divisions should broadcast a software data call on a periodic (at least annual) basis to ensure software is identified and managed as required by this chapter.*

2.2 Software Determination

A. How. Use Table 21.1-3 and the text that follows it to complete software determinations.

1. Ask the following question as shown in Table 21.1-3:

Table 21.1-3 The Reasonable Probability Software Question

Excluding personal productivity and collaboration software that does not provide calculation output (e.g., e-mail software, presentation software)[1], is or will the software be used in connection with[3] the design, analysis, and/or operation of:

- a nuclear (including radiological) facility (Ref. LANL Nuclear Facility List and Conduct Of Operations Resources websites);
- an accelerator, live-firing range, biological hazard facility, high explosive facility, or moderate- or high- chemical hazard facility[2] as determined using Safety Basis Procedure (SBP)111-1, Facility Hazard Categorization and Documentation; or
- LANL's Essential Functions (EFs) as described in SEO-COOP-006, *Continuity of Operations (COOP) Plan.*

Notes:

[1]Personal productivity and collaboration software that does not provide calculation output is non-safety, commercially controlled software and is excluded from this chapter.

[2] From SBP111-1, a facility is defined as equipment, structure, system, process, or activity that fulfills a specific purpose except for utility distribution structures.

[3]Use must be associated with ML-1, ML-2 or ML-3 items. The term "in connection with" broadly captures software that directly or indirectly is used in relation to design, analysis or operation. Operation includes SSC operation (e.g., valve actuation) and administrative operation (e.g., combustible-load-tracking software).

If the answer to the question is "yes" or "not sure" and the associated SSC ML is ML-1 through ML-3 or unknown, then complete Form 2033. If the associated SSC is ML-4, then do not complete Form 2033 unless required by the SRLM or Subcontract.

By default, all software is non-safety and commercially controlled (analogous to ML-4) unless there is a reasonable probability that the software could be safety software or risk significant software.

Guidance: Examples of commercially controlled, non-safety software include: Microsoft PowerPoint® presentation graphics program, Microsoft Outlook® messaging software and, when used in ML-4 applications, Caesar II, SASSI.

[20] Deferred refers to design performed after permit is granted (ref. IBC-2015 107.3.4 and 202 and ESM Chapter 16, IBC-GEN.

2. Complete Form 2033 using the instructions on the form. See form instructions for which revisions of the form may be used. If the software is planned to be used as "Simple and Easily Understood" software used in the design of SSCs (see definitions), then note it on the form.

Guidance on MLs, SRLs, and Type.: For Non-SSC software, use the highest ML of the SSC(s) being designed/analyzed. For SSC software, use ML of the SSC unless the software has a lower associated ML level than the SSC's highest ML level (In this latter case, the reason needs to be documented. This may necessitate an ML determination using AP-341-502 when the software may be a lower ML level than the system it supports.[21] Example: Flowmeter credited for confinement while its embedded software is not). In addition, in both cases, the 2033 software SRL and Type can be at a lower level than the associated SSC in some instances

Guidance on bundling: Form 2033 allows multiple software items to be included (often it is the support software), and SOFT-INV allows SSC software to be inventoried as a software collection (bundled); see SOFT-INV Section 3.0, A. 5. Bundling may work better if all software is the same ML level relative to Form 2033 and subsequent Ch. 21 requirements (may cause confusion if not). Example of where bundling might be considered: A control panel that has a PLC and a touchscreen and separate programs for these components, those separate programs, both ML-2, could be given a single SWID (and 2033, etc.)

3. Obtain a software identification number (SWID) in accordance with AP-341-402, *Engineering Document Management in Operating Facilities. A link to the SharePoint numbering utility is on the Ch 21 webpage.* Ensure the SWID is on the Form 2033. (Unless a SWID location is indicated on the form, place the SWID in the "Reference No." location on the form along with the record number). Ensure the SWID is on subsequent software documentation. Retain the completed Form 2033 as a record in accordance with the SRLM's governing records management process.

B. When.

1. Complete at the earliest practical time and before software design or acquisition. For SSC software, submit draft Form 2033 with the 60% detail design submittal. When a required field on the Form 2033 is not known, indicate TBD or similar term. Provide Form 2033 forms (complete without TBDs) with the 90% detailed design submittals. For a deferred SSC design, include the final Form 2033 in the deferred design submittal(s).

2. If the application of the software significantly changes, review the original determination and complete a new Form 2033 as required. *Guidance: It is recommended that Form 2033s be reviewed at least every three years or other time period that is commensurate with the associated risk.*

   Note: Changes made to the software (e.g., software version, SO, SRLM) do not require a new Form 2033 to be completed unless the change significantly affects the application. The software inventory must be reviewed and updated as required. See SOFT-INV.

[21] Current AP-341-502 excludes software but revision is planned; proceed with determination in the meantime if the associated ML cannot be inferred.

C. Who. The SRLM is responsible for software determinations. Form 2033 instructions require the SO to develop, and the SRLM and FDAR to review and approve the form.

3.0 SOFTWARE DATA SHEET (SWDS) (ML-1 THROUGH ML-3 ONLY)

A. How. Prepare a Software Data Sheet (SWDS). See SOFT-GEN-FM01, *Software Data Sheet Form (SWDS)*. One SWDS can be used for multiple installations. Process and retain the data sheet in accordance with the SRLM's governing document control/records management processes *(e.g., EDMS, and for ES-Div, associated Desktop Instruction).*

Ensure the correct SWID (obtained when completing Form 2033) is part of the SWDS record number and subsequent software documentation.

Note: Software data sheets are not required for ML-4 software; however, they may be used at the SRLM's discretion.

B. When. Complete a SWDS at the earliest practical time and before design and/or acquisition as applicable (e.g., at end of planning).[22] Revise the SWDS as required to maintain sufficient accuracy as determined by the SRLM, to adequately manage the software.

Note: *SRLM should review the data sheet at least every two years.*

C. Who. At a minimum, the SO develops and the SRLM reviews and approves the SWDS.

4.0 CONFIGURATION MANAGEMENT (CM)

A. Maintain software configuration management (CM) in accordance with this Subsection[23]. This section includes developing a baseline of configuration controlled items and the change control processes to make controlled changes to the baseline.

4.1. Develop and Maintain a Software Baseline (SWBL) (ML-1 through ML-3 only)

A. How.

1. Develop and maintain a software baseline.

From App. A definition, a baseline is a specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for use and further development, and that can be changed only by using an approved change control process. (Ref. NQA-1).

Use SOFT-GEN-FM02, *Software Baseline Form (SWBL)*.

Note: A software baseline includes the computer program (code and [configuration] data) and the computer program documentation. For a mature

[22] Ref. IEEE 12207. At the end of the planning phase, SWDS is proof of a plan and management commitment to the project. Initial revision may have TBD for some parameters (e.g. use and maintain items) that are not known in the early planning stage and will be determined at a future date.

[23] See ANSI/IEEE Std. 828, *IEEE Standard for Configuration Management in Systems and Software Engineering* for guidance.

baseline, one may think of this as the information and computer program files that are needed to run the software for a specific application.

ML-4 software does not require a software baseline unless it is required by the SRLM.

2. For SSC software: Ensure the software baseline is consistent with the SSC technical baseline documents, as applicable. See *Engineering Standards Manual Chapter 1, Section Z10, General*; AP-341-616, *Technical Baseline Change During Design;* and AP-341-405, *Identification and Control of Technical Baseline, Variances, Alternate Methods, and Clarifications in Operating Facilities.*

    a. As applicable, ensure configurations (e.g., bounding set points) are established in accordance with AP-341-613, *Instrumentation Set Point Control.*

3. Define the baseline configuration items needed to run the program. Include, as applicable[24]:

    a. **Documentation** - software design requirements; instructions for computer program use including installation and test requirements; system description; test plans and results; characteristics derived from regulatory requirements and commitments; calculations and analyses; support software documentation.

    b. **Computer program(s)** - source, object, back-up, data,[25] configuration, and support software files needed to run the program. Configuration files include limits, settings, bounding set points, etc. As applicable, include the file/directory attributes (e.g., for Window systems and servers) or authorities (e.g., for Unix/Linus systems). These are the required access settings for reading, writing and executing for reading, writing and executing for users, owners, etc.

4. Develop and control the software baseline in accordance with the SRLM's document/electronic document control process *(e.g., EDMS)*. The SRLM's approved document control process is used, as a minimum, to control both the documents and the computer program files that comprise the baseline. The document control process must ensure only authorized access and changes are made to software baselines. Ensure the correct SWID (obtained when completing Form 2033) is part of the SWBL record number and subsequent software documentation.

5. Update the software baseline after software changes. Ensure that the baseline defines the most recent approved software configuration.[26] Ensure work is performed on computers, servers, etc. using documentation and computer

---

[24] Applicability is the determination that the activities and processes are actually used. For example, if the work scope does not include or contribute to design, then design requirements do not apply and the related design procedures are not used [Ref. SD330, *LANL Quality Assurance Program*].

[25] This is the data needed to run the program. This can include data such as limits, bounding set points, and databases.

[26] In the event the in-use computer program fails, becomes corrupted, or other problems occur, the baseline may be used to re-establish operation

programs that match (mirror) the baseline as described and controlled in the SRLM's document control process.

6. Identify the associated hardware needed to run the computer program in the system design description document(s) or directly in the baseline.

7. When "Support Software" (including software tools) is used for new or modified software, one must evaluate, review, accept for use and place it under configuration control as well. Note however, that software tools that do not affect the performance of the software need not be placed under configuration control. Identify the software/tool in the baseline and manage the configuration. Changes to software tools must be evaluated for impact on the software product to determine the level of reviews and retesting that will be required. Vendor-supplied software used with hardware to configure control systems is an example of support software.

8. Implement a baseline labeling system in accordance with the following:

   a. Uniquely identify each configuration item (documentation or computer program).

   b. Identify changes to configuration items by revision.

   c. Provide the ability to uniquely identify each configuration of the revised software available for use.

   d. For baseline document labeling, use the SRLM's approved document numbering system (e.g., AP-341-402, *Engineering Document Management in Operating Facilities).*

   e. Include baseline label information within the computer source code where possible. For each logical block or class, include a brief description of its function, the name of the person writing the description and the date the description was added. For code changes following SWAU, update the header comments with the revision number (e.g., 1.1, 1.2...) and the SCP number driving the change.

   f. For off-the-shelf software (OTS) or other (existing) software with a compliant labeling configuration, use the OTS or existing software labeling configuration.

   g. For other software not described in (f), use Table 21.1-4, *Computer Program File Labeling System*, or as approved by the Chapter 21 POC.

   h. Ensure the correct SWID (normally obtained when completing Form 2033) is integrated into the baseline labeling where possible.

Table 21.1-4 Computer Program File Labeling System			
Prior to Approval for Use[1]		After Approval for Use	
Major Change[2]	Minor Change[2]	Major Change	Minor Change
A, B, C…	A.1, A.2, A.3…	1, 2, 3…	1.1, 1.2, 1.3…

[1] Software approval for use (SWAU) constitutes that the software requirements have been satisfied (including testing, user's manuals, etc.) and the software is ready to be used in it intended operating environment.

[2] See definitions of Major Change and Minor Change in App A, Definitions.

B. When.

1. For ML-1 and ML-2 designed software, establish software baselines at the completion of each major software design activity.

2. Establish software baselines as soon as practical. At the latest, establish before formal acceptance testing/commissioning. Pre-acceptance testing may be performed to support verification of the baseline prior to (final) acceptance testing. Maintain baselines (configuration items) under configuration management until software retirement.

C. Who. The SO prepares the baseline. The SRLM and others as required in the governing document control process, review and approve the software baseline.

4.2. Control Software Changes

A. How.

1. ML-1 and ML-2 Safety Software: For computer program changes (revisions) to software, use SOFT-V&V to V&V the software change and/or commercial grade dedication (CGD), e.g., per *AP-341-703*, to dedicate the change. Unless a more conservative approach is required by the LANL SRLM, use the following graded approach for dedicating software changes:

   a. Dedicate Major Changes (see Definitions).

   b. Dedicate Minor Changes (see Definitions) if the changes affect the critical characteristic(s) that pertain to the functionality of the computer program and as applicable, the associated SSC.

2. For all ML levels, see Table 21.1-5 and apply the following to Major and Minor software changes:

   a. For designed software, institute design control measures commensurate with those applied to the original design.

   b. Document changes and include the following in the change documentation:

      1) a description of the change;

      2) rationale for the change;

      3) identification of affected software baselines, including documents to be updated; and

      4) evaluation and approval of the change (see SOFT-V&V for evaluation methods/criteria).

Table 21.1-5 SOFT-GEN Software Change Summary by Software Management Level (ML)
(This is a summary only and does not include all requirement details. See text for details.)

For the following types of software changes	Provide the following deliverable	SSC Software ML[1]				Non-SSC Software ML[1]			
		1	2	3	4	1	2	3	4
Any change made before or after approval for use (SWAU). Includes Less than Minor, Minor, and Major changes.	Documented comment in the computer program code where feasible	R	R	R	R	R	R	R	R
Minor and Major Changes made during design implementation (e.g., fielding or startup) and before SWAU	AP-341-519 change documents[2]	R	R	Gr	Gr	-	-	-	-
Minor and Major Changes made once software is baselined	Revised baseline[3]	R	R	R	-	R	R	R	-
Minor and Major Changes made after SWAU	SSC: Software Change Package (SCP) Form AP-341-507-FM01[4]	R	R	Gr	-	-	-	-	-
	Non-SSC: SWNCP Form, SOFT-GEN-FM03	-	-	-	-	R	R	Gr	-

[1] See Form 2033 for a crosswalk between MLs and software type, category and software risk levels. R = Required. Gr = Required but graded. "-" = Not required.

[2] Includes Field Change Notices (FCNs), Field Change Requests (FCRs), and Design Revision Notices (DRNs) as described in AP-341-519 unless explicitly superseded by another administrative procedure.

[3] Use SOFT-GEN-FM02, *Software Baseline Form (SWBL)*. Include documents and computer program files.

[4] See ESM Chapter 15, *Commissioning* and AP-341-801, *Post Modification/Post Maintenance Testing* for related testing requirements of SSC software changes.

c. Perform appropriate verification activities and validation (V&V) of changes (see SOFT-V&V). Perform appropriate acceptance testing for the change.

d. Incorporate the changes in documentation, and maintain traceability of the change to the software design requirements.

   1) Include the (1) initiation, evaluation, and disposition of a change request, (2) control and approval of changes prior to implementation, and (3) requirements for retesting (e.g., regression testing) and acceptance of the test results.

e. Perform the following:

   1) maintain a current status of configuration items and control configuration changes until they are incorporated into the approved product baseline;

   2) maintain the status of changes that are proposed and approved, but not implemented; and

3) provide notification of this information to affected organizations.

f. For changes to documentation only, use the governing document control process.

g. Implement the processes described and/or referenced in this chapter to ensure that interfaces are controlled/managed such that unintended consequences do not occur.

h. As applicable, ensure bounding configuration changes are established (Ref. AP-341-613, *Instrumentation Set Point Control)*.

i. Update the software baseline.

3. For less than minor computer program changes, ensure they are made by a competent individual knowledgeable in the software. If feasible, add a brief description, name of who is making, justification that the change does not modify performance functions (all MLs), and the date that the less than minor change is made in the computer program code at the time the change is made. *If it is not feasible to document the less than minor change in the code, then document in associated operational logbooks, etc.*

B. When. Control changes per Table 21.1-5 in a timely manner and when required to ensure proper performance.

C. Who.

1. For SSC software, see AP-341-519 for design phase and AP-341-507 for operations phase unless explicitly superseded by another procedure. For Non-SSC software, see SOFT-GEN-FM03, Non-SSC Software Change Package Form (SWNCP).

2. The organization responsible for the original software design and others as deemed necessary by the SRLM must evaluate, and approve software changes unless an alternate organization has been given the authority by the SRLM to approve the changes.

3. As determined by the SRLM, only those that are knowledgeable in the computer program code may make changes to the computer program code.

5.0 PERFORM PROBLEM REPORTING & CORRECTIVE ACTION (PR&CA)

A. How.

1. For ML-1 through ML-4 software, perform software problem reporting and corrective action throughout the software life cycle. For software where suppliers provide/post errors, review errors/notifications at a frequency commensurate with the risk and frequency of use. *A review of problem reports prior to use in calculations and/or a minimum of once per year is recommended.*

2. For ML-4 software, less-formal methods (e.g., bug lists, use of comment fields within the computer program, software-tailored tools such as TeamForge and trac) or formal methods (see below) may be used throughout the software lifecycle.

3. For ML-1 through ML-3 software, less-formal methods may be used through Approval for Use (SWAU). After (SWAU), formal methods must be used. Individually or in combination, P330-6, *Nonconformance Reporting*, P322-4,

Issues Management, or equivalent process as determined by the SRLM, must address the following:[27]

a. Describe the evaluation process for determining whether a reported problem is an error (see definitions) or other type of problem (e.g., user mistake).

b. Define the responsibilities for disposition of problem reports, including notification to the originator of the results of the evaluation.

c. When a problem is determined to be an error (see definitions), provide, as appropriate, for

1) how the error relates to appropriate software engineering elements;

2) how the error impacts past and present use of the computer program;

3) how the corrective action impacts previous development activities; and

4) how the users are notified of the identified error, its impact, and how to avoid the error, pending implementation of corrective actions.

d. Problem reporting and corrective action must include methods for documenting, evaluating, and correcting software problems. *This should include an estimate of the level of effort and corrected software release date.*

e. Conditions adverse to quality shall be identified promptly and corrected as soon as practicable.

f. In the case of a significant condition adverse to quality, the cause of the condition shall be determined and corrective action taken to preclude recurrence.

g. The identification, cause, and corrective action for significant conditions adverse to quality shall be documented and reported to appropriate levels of management.

h. The verification of completed corrective actions.

B. When. Perform throughout the software life cycle.

C. Who. The SO performs software problem reporting and corrective action on behalf of the SRLM. The SRLM is responsible to ensure it is performed as required by the associated procedure (e.g., P330-6 on NCRs).

6.0 PERFORM SOFTWARE (PROJECT) RISK MANAGEMENT

A. How.

1. *Software risk management as required by O414.1D focuses on the risks to successfully complete the software project; it does not focus on the risks of potential failure of the software.*[28] Software risk management applies to all phases of the software project life cycle. Achieve overall software risk management through implementation of the NQA-1 based, systematic processes of this chapter and the supporting quality assurance program. *That is, successful implementation of this chapter should result in successful completion of a software project. The*

[27] The SWDS may be used to describe an equivalent as long as the PR&CA requirements are satisfied.
[28] Based on DOE G 414.1-4, *Safety Software Guide for Use with 10 CFR 830 Subpart A, Quality Assurance Requirements, and DOE O 414.1C, Quality Assurance.*

SRLM should determine if additional software project-level software Risk Management is needed for ML-4 software (e.g., large and important projects).

2. For ML-1 through ML-3 software, address software-specific risks (risks specific or unique to the software implementation). Manage software-specific risks by (a) assessing and (b) controlling the risks.

   a. With a graded approach approved by the SRLM and Chapter POC, implement risk management based on industry accepted methods such as those described in DOE SQAS21.01.00-1999*, Software Risk Management–A Practical Guide;* DOE G413.3-7, *Risk Management Guide; and/or* DOE G 414.1-4, *Safety Software Guide for Use with 10 CFR 830 Subpart A, Quality Assurance Requirements, and DOE O 414.1C, Quality Assurance*.

3. Risk assessment includes risk identification, analysis, and prioritization to ensure that the necessary resources are available to mitigate them. Risk control includes risk resolution and tracking.

4. Resolve risks using risk avoidance, mitigation, and/or transference.

5. Document and maintain software-specific risks on a risk register (also referred to as risk list) in the SWDS of if desired, in a separate document. Use Template 3046, Software Risk Register from the LANL Forms Center or equivalent. Attach the risk register to the SWDS is the best practice; create a unique SWRR number only if not attaching.

B. When. Develop a SWRR at the earliest practical time, before design/acquisition as applicable, and perform risk management throughout the software life cycle.

C. Who. The SO performs software risk management on behalf of the SRLM. The SRLM is responsible to ensure it is performed, and reviews and approves the risk register as part of the SWDS.

7.0 APPENDICES, ATTACHMENTS AND FORMS

Appendices:
- APPENDIX A: Chapter 21 Definitions, Responsibilities, and Acronyms
- APPENDIX B: Chapter 21 SO and SRLM Decision Diagram (Guidance)
- APPENDIX C: Chapter 21 Reference Listing

Attachments:
- ATTACHMENT 1: Chapter 21 Summary Table

Forms:
- SOFT-GEN-FM01: Software Datasheet Form (SWDS)
- SOFT-GEN-FM02: Software Baseline Form (SWBL)
- SOFT-GEN-FM03: Non-SSC Software Change Package Form (SWNCP)

APPENDIX A: CHAPTER 21 DEFINITIONS, RESPONSIBILTIES, AND ACRONYMS

Only key definitions are listed. See PD340, *Conduct of Engineering for Facility Work*; SD330, *Los Alamos National Laboratory Quality Assurance Program*; and documents referenced therein for additional definitions.

Table 21.1-A1 Chapter 21 Definitions and Responsibilities

Item	Definition/Responsibility
acceptance testing, also known as software validation	The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements, and, to identify differences between expected and actual results in the operating environment. (Ref. NQA-1).
Acquired software	Software that is generally supplied through basic procurements, two-party agreements, or other contractual arrangements. Acquired software includes off-the-shelf (OTS) software such as operating systems, database management systems, compliers, software development tools, and commercial calculational software and spreadsheet tools. Downloadable software that is available at no cost to the user (referred to as freeware) is also considered acquired software. (Based on DOE G 414.1-4).
Acquired Non-SSC software	Non-SSC software that is acquired software where the code cannot be changed other than through replacement. This may also be referred to as Read-Only Non-SSC software. Replacement includes replacement with a subsequent software version or upgrade. (Definition developed for this chapter).
Administrative controls	Administrative controls mean the provisions relating to organization and management, procedures, record keeping, assessment, and reporting necessary to ensure safe operation of a facility. (Ref. 10CFR 830)
Associated Management Level (ML)	The highest management level (ML) of an SSC that is associated with the software function. The software function must be required for, and/or the software failure will have an adverse effect on the SSC. (adaptation of Form 2033 guidance)
baseline	A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for use and further development, and that can be changed only by using an approved change control process. (Ref. NQA-1). ***Note:*** See SOFT-GEN-FM02, *Software Baseline Form (SWBL)*. A software baseline includes the computer program (code and [configuration] data) and the computer program documentation. In layman's terms, one may think of this as the information and computer program files that are needed to run the software for a specific application.
Bounding set points	Bounding set points are those that are bounding or limiting values required by or needed to satisfy safety basis requirements, protect equipment/systems from operational damage, or other limiting values for the proper intended operation of the software. Changes to operational set points within the minimum or maximum values would not constitute a change, but rather would be considered operational use of the software. Example: Bounding set points may be minimum or maximum pressure or tank level values.
Commercially Controlled (CC) software	As determined using Form 2033, *Safety/Non-Safety Software Determination, Categorization, and Software Risk Level (SRL)*, software that is not, or will not be safety software or risk significant software. Such software may be acquired (including off the shelf (OTS) software, freeware, or designed software. Examples: Personal productivity software (e.g., Microsoft PowerPoint, Oracle Project Primavera, MS Outlook, etc.); typical business accounting systems, facility personnel comfort temperature control systems. (Based on P1040).
computer program	A combination of computer instructions and data definitions that enables computer hardware to perform computational or control functions. (Ref. NQA-1).

Table 21.1-A1 Chapter 21 Definitions and Responsibilities	
Item	**Definition/Responsibility**
computer program listings	A printout [e.g., pdf] or other human readable display of the source and, sometimes, object statements that make up a computer program. (Ref. ISO/IEC/IEEE 24765).
configuration item	A collection of hardware or software elements treated as a unit for the purpose of configuration control. (Ref. NQA-1).
configuration management	The process of identifying and defining the configuration items in a system (i.e., software and hardware), controlling the release and change of these items throughout the system's life cycle, and recording and reporting the status of configuration items and change requests. (Ref. NQA-1).
control point	A point in the software life cycle at which specified agreements or control (typically a test or review) are applied to the software configuration items being developed, e.g., an approved baseline or release of a specified document or computer program. (Ref. NQA-1).
Critical software	Software whose proper performance is critical to the expected performance of a safety SSC, a defense-in-depth SSC, or the safety of the nuclear facility. (Ref. DOE-STD-1073).
design analyses	Calculations and/or experiments associated with design. (Based on NQA-1).
designed (or developed) software	Software that is designed or developed for a specific (custom) application. It may be developed by DOE or one of its Management and Operator contractors or contracted with a software company through the procurement process. ***Note:*** Includes the following from DOE G 414.1-4. (a) custom-developed software, (b) configurable software, and (c) utility calculation software. (Definition developed for this chapter and based on DOE G 414.1-4).
Designed Non-SSC software	Non-SSC software where the computer program can be changed other than through replacement. Replacement includes replacement with a subsequent software version or upgrade. (Definition developed for this chapter).
Designed SSC software	SSC software where the computer program can be changed other than through replacement of the computer program and/or the associated SSC. (Definition developed for this chapter).
engineered (engineering) control	Controls that eliminate or reduce exposure to a hazard through use or substitutions of engineered machinery or equipment. (Based on DOE G 450.4-1C).
Engineering Services Software Coordinator	Individual assigned to maintain the software inventory for ES Division, and to perform other duties as assigned (e.g., per desktop instruction).
error	An error is a condition deviating from an established baseline including deviations from the current approved computer program and its baseline requirements. (Ref. NQA-1). An error is something which requires a software change (major or minor). Examples of errors include (a) if a computer program used for design of a structural member provides incorrect design output, and (b) if a computer program turns on heating instead of cooling at high temperature settings.
Firmware	The combination of a hardware device, computer programs, and data that reside as read-only software on that device. The firmware (sometimes referred to as embedded software) can perform very limited functions such as keypad controls, or can provide significant function and control capabilities for control rod drives or safety systems. (Ref. NQA-1).
freeware	Software that is available for use at no cost or for a nominal, usually voluntary fee. (Ref. Merriam-Webster Dictionary).

Table 21.1-A1 Chapter 21 Definitions and Responsibilities	
Item	**Definition/Responsibility**
function	A function is a task that must be performed. A function statement describes the capability necessary for a facility, system, or component to fulfill its mission. Describe a function using verb/noun combination such as “filter particulate.” A function transforms inputs to desired outputs. In the example of “filter particulate,” the function transforms an input fluid containing particulates into two outputs, the fluid without particulates and the particulates. A function describes what must be done, not how. Every function has at least one requirement associated with it.
hazards controls	Hazard controls mean measures to eliminate, limit, or mitigate hazards to workers, the public, or the environment, including: (1) Physical design, structural, and engineering features; (2) Safety structures, systems, and components; (3) Safety management programs; (4)Technical safety requirements; and (5) Other controls necessary to provide adequate protection from hazards. (Ref. 10CFR830).
Less-Than-Minor Computer Program Change	A change that is not a major or minor computer program change and: ▪ adds, deletes, and/or modifies ML-4 performance function code, ▪ adds, deletes, and/or modifies code that does not modify a Performance Function (all MLs), or ▪ imparts changes without adding, deleting or modifying design and/or analysis output values (all MLs). Examples: Modify code to increase the ramp time on an ML-4 softstart pump. Install security patch/service pack updates. An OTS software patch that includes a code change to prevent a screen from “freezing” or loading slowly (all MLs). Add/modify code clarifying notes (all MLs). Modify code to produce multiple reporting formats (all MLs). (Definition developed for this chapter).
Major Computer Program Change	A change that: ▪ the Software Responsible Line Manager (SRLM) or computer program supplier designates as a Major Change, ▪ adds or deletes an ML-1, ML-2 or ML-3 SSC “Performance Function” (including bounding set point changes), ▪ modifies ML-1 or ML-2 SSC performance function code, excluding clarifying notes, ▪ adds, deletes or modifies design and/or analysis output values of ML-1, ML-2 or ML-3 calculations, ▪ recodes to another language, or ▪ modifies a significant number of lines of code. ***Note:*** A Major OTS computer program change is often indicated with increment increase in version number (e.g., change from version 1 to 2). An evaluation of the software however, is required to determine whether the version release is a Major Change. Examples: A change from Delta V control system software from version 7.0 to version 8.0. A change that adds code to implement an interlock functional performance requirement that an ML-3 laser system cannot be activated until area doors are locked. A change that modifies code on ML-2 ventilation system backdraft damper so that damper closure does not slam shut and potentially damage the damper assembly. A change in the algorithm or databased used for calculating the water flow rate in an ML-3 fire protection piping system design. A change in coding language from C to C++. A version change where 40% of the lines of code were modified. (Definition developed for this chapter).
Major Document Change	A document change that is not a minor document change. A major document change includes revisions, changes, or modifications to a document (e.g., procedure, work instruction, drawing, etc.) which impact the effective implementation of the requirement(s). (Based on P1020-2).

Table 21.1-A1 Chapter 21 Definitions and Responsibilities	
Item	**Definition/Responsibility**
Minor Computer Program change	A change that is not a major computer program change and: ▪ adds or deletes an ML-4 SSC Performance Function (including bounding set point changes), ▪ modifies ML-3 SSC Performance Function code, excluding clarifying notes, or ▪ adds, deletes or modifies design and/or analysis output values of ML-4 calculations ***Note:*** A minor OTS software change is often indicated with a fractional increase in version number (e.g., 1.1 or 1.01). An evaluation of the software however, is required to determine whether the version release is a Minor Change. Examples: Add code to implement automatic pump shut-off performance requirement on ML-4 sump low-level alarm. Modify code to fix a coding error on an ML-3 heating/cooling system so that cooling, rather than heating activates at high temperatures. Change the algorithm for calculating the current that flows in an ML-4 electric power system under abnormal conditions. (Definition developed for this chapter).
Minor Document Change	A document change, as defined by the governing document control program, that includes but is not limited to inconsequential editorial corrections, grammatical and spelling changes, organizational name and acronym changes, and similar type changes. (Based on P1020-2).
Non-NQA-1 qualified supplier	A supplier that did not develop and maintain the software in accordance with an NQA-1 quality assurance program. (Definition developed for this chapter).
model	Simplifications of the real world constructed to gain insights into select attributes of a particular physical, biological, economic, engineered, or social system. (Ref. EPA/100K-09/003, *Guidance on the Development, Evaluation, and Application of Environmental Models*.)
Non-safety software	As determined using Form 2033, software that is not otherwise determined to be safety software. Non-safety software includes risk significant and commercially controlled software. (Ref. P1040).
Non-SSC software	Software used in design, analysis and/or for administrative control. This software does not physically monitor and/or control SSCs. Examples: Piping system design/analysis software (CAESAR II®), fire protection system design software (SprinkCAD), area lighting calculation software, spreadsheets used to perform structural load calculations, safety analysis software used to perform dispersion modeling, software used to track facility combustible loading, and software used to track Technical Safety Requirement (TSR) implementation. (Definition developed for this chapter).
Otherwise-Acquired Software	Software that was not acquired, developed and/or maintained in accordance with an NQA-1 quality assurance program. This software may be from entities internal to LANL entities external to LANL (e.g., other DOE sites, U.S. EPA, etc.) This includes existing software (also referred to as in-use or legacy software).
operating environment	A collection of software, firmware, and hardware elements that provide for the execution of computer programs. (Ref. NQA-1)). It is also the location and conditions (environment) where the software will be used or operated to meet is intended function. (Based on P330-8).
performance function	A function that is required to satisfy item performance. (Definition developed for this chapter).
performance function code	The computer program language (code) that is required to satisfy item performance. (Definition developed for this chapter). The performance function code is only those lines of code that affect the performance function.

Table 21.1-A1 Chapter 21 Definitions and Responsibilities	
Item	**Definition/Responsibility**
Read-Only SSC software	SSC software where the computer program cannot be changed other than through replacement of the computer program and/or the associated SSC. (It is sometimes referred to as embedded software or firmware; however, for this Chapter, the term Read-Only SSC software is used.) ***Note:*** Read-only software includes software where limits and/or set points can be configured (e.g., via keypad entries) without modifying the computer program (code). (Definition developed for this chapter).
Regression testing	Selective retesting to detect errors introduced during modification of the computer program or to verify that the modified computer program still meets its specified requirements. (Ref. NQA-1).
Risk Significant (RS) software	Software that is, or will be, used for any of the purposes for which safety software is used (see Form 2033), only such purposes are in or for an accelerator, live-fire range, biological hazard facility, explosive hazard facility, or a moderate- or high- hazard chemical facility; or failure of the software would prevent LANL from performing Essential Functions as described in SEO-COOP-006, *Continuity of Operations (COOP) Plan*. Commercially controlled software is not risk significant software. (Ref. P1040).
Safety and Hazard Analysis and Design Software (SHADS)	Safety software that is used, or will be used, to classify, design, or analyze nuclear (including radiological) facilities. This software is not part of an SSC, but helps to ensure the proper accident or hazards analysis of nuclear (including radiological) facilities or an SSC that performs a safety function. This is Non-SSC software, safety software, and is categorized as SHADS. (Ref. P1040).
Safety Management and Administrative Controls Software (SMACS)	Safety software that performs, or will perform, a hazard control function in support of nuclear (including radiological) facility radiological Safety Management Programs (SMPs) or TSRs; and/or this is software that performs, or will perform, a control function in support of a nuclear (including radiological) facility necessary to provide adequate protection from nuclear (including radiological) facility radiological hazards. It supports eliminating, limiting, or mitigating nuclear hazards to workers, the public, or the environment as addressed in 10 CFR 830, 10 CFR 835, *Occupational Radiation Protection*, and the Department of Energy Acquisition Regulation (DEAR) Integrated Safety Management System (ISMS) clause 48 CFR 970.5223-1, *Integration of Environment, Safety, and Health into Work Planning and Execution*. (Ref. O414.1D).
Safety software	Software that includes any of the following: SSS, SHADS, or SMACS. Both SSC software and Non-SSC software can be safety software. (Ref. P1040).
Safety System Software (SSS)	Safety software for a nuclear (including radiological) facility that performs, or will perform, a safety function as part of an SSC and is cited in either (a) a DOE-approved documented safety analysis, or (b) an approved hazard analysis per DOE P 450.4A, *Integrated Safety Management Policy* and 48 CFR 970-5223-1, *Integration of Environment, Safety, and Health into Work Planning and Execution.* This is SSC safety software and is categorized as SSS. ***Note:*** References implemented at LANL as described in PD110, *Safety Basis*. See DOE-approved documented safety analyses at LANL Safety Basis Document Lists (SBDLs). Analyses include Documented Safety Analyses (DSAs), Preliminary Documented Safety Analyses (PDSAs), Bases for Interim Operations (BIOs), etc. (Based on P1040).
Simple and Easily Understood (Non-SSC) software	Software that satisfies the following criteria: a. The software is used in the design of SSCs; b. The results of the computer program can be easily confirmed through hand calculations; c. A person technically qualified in the subject can review and understand the program and the supporting calculations; and, d. The software can be individually verified with each use (e.g., calculation). (Based on NQA-1).

Table 21.1-A1 Chapter 21 Definitions and Responsibilities	
Item	**Definition/Responsibility**
software	Computer programs and associated documentation and data pertaining to [needed for] the operation of a computer system. (Ref. NQA-1).
software approval/ approved for use (SWAU)	An approval that constitutes that the software requirements have been satisfied (including installation and operating instructions), and the software is ready to be used in the intended operating environment. (Ref. P1040).
software change	A software change is an addition, deletion and/or modification to software. (Definition developed for this chapter).
Software Coordinator	Individual(s), designated by division management, providing coordinating and/or administrative functions in support of chapter compliance (e.g., inventory and associated reporting). *ES Division has an ES-Div Software Coordinator. ES-Software@lanl.gov*
software design requirement	A requirement that impacts or constrains the design of a software system or software system component. (Based on ISO/IEC/IEEE 24765).
software engineering	(a) the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software (b) the study of approaches as in (a) (Ref. NQA-1)
software life cycle	The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and, sometimes, retirement phase. These phases may overlap or be performed iteratively, depending on the software development approach used. (Ref. NQA-1)
software design verification:	The process of determining if the product of the software design activity fulfills the software design requirements. (Ref. NQA-1).
software engineering elements	(a) software acquisition method(s) for controlling the acquisition process for software and software services; (b) software engineering method(s) used to manage the software life-cycle activities; (c) application of standards, conventions, and other work practices that support the software life cycle; (d) controls for support software used to develop, operate, and maintain computer programs. (Ref. NQA-1).
Software Owner (SO)	Responsibilities (performs or causes to be performed, see others in chapter): • Provides the software information and Form 2033, *Safety/Non-Safety Software Determination, Categorization, and Software Risk Level (SRL)* and obtains review and concurrence of the form in accordance with this document. • Reviews and approves the software project planning documentation. • Owns the software and supports the SRLM in complying with the requirements of this document. • Prepares the approval for use documentation that describes the intended use and any associated limitations, access controls, etc., for using the software.
Software Point of Contact (SPOC)	That individual selected by division management to act as software owner for specific software or multiple Non-SCC software programs. Ideally it's the main or super-user of each program, but can be another user or even a single individual for a group or division.
Software Responsible Line Manager (SLRM)	Responsibilities (performs or causes action below to be performed, see others in chapter): • Manages and maintains software in accordance with this document to ensure it operates as intended. • Determines reasonable probability, and as applicable, • As applicable, acquires software and/or software services. • Except for SQA associated with using the software, is responsible for the SQA of the software. • V&Vs the software.

	reviews, signs and retains Form 2033 as a record. • Identifies, documents, approves, controls, and maintains safety and risk significant software owned by the SRLM that is new software or in-use software at LANL nuclear (including radiological) facilities. • Provides software inventory information and any changes thereto to QPA-IQ and ES. • Applies the appropriate amount of SQA rigor in software planning and implementation. • Reviews and approves the software project planning documentation. • Identifies and ensures reviews are performed by competent individuals or groups other than those who developed and documented in the original software design (but who may be from the same organization). • Ensures software owning organization personnel managing or working to this document are adequately trained and as required, qualified. • Approves software for use. • Completes in-use tests in the operating environment. • Retires software, including removal of safety software from software inventories.
software tool	A computer program used in the development, testing, analysis or maintenance of a program or its documentation. Examples include vendor-supplier configuration tools, conversion tables, comparators, cross-reference generators, compilers, CASE (Computer-Aided Design Software Engineering) tools, configuration and code management software, decompilers, disassemblers, editors, flowcharters, monitor test case generators, and timing analyzers. (Based on NQA-1).
Software User (SU)	Responsibilities (performs or causes to be performed, see others in chapter): • Reports software errors and problems. • Uses software within software limitations and in accordance with this document.
Software User Responsible Line Manager (SU RLM)	Responsibilities (performs or causes actions below to be performed, see others in chapter): • Supports completion of in-use tests in the operating environment. • Ensures software users and software user organization personnel managing or working to this document are adequately trained, and as required, qualified.
SSC software	Software that controls and/or monitors system, structures and components (SSCs) and is running and interacting with its environment in real time. SSC software may be safety or non-safety software. Examples: Building Automation Control System (BAS) software, process gas monitoring and control system software, fire alarm control panel (FACP) software, continuous air monitor (CAM) software, seismic switch software, and uninterrupted power supply (UPS) software. (Based on TR. No. 397, *Quality Assurance for Software Important to Safety*, IAEA, 2000).
support software	Software or a program that aides in the development, maintenance, or use of other software or provides general application-independent capability (Ref. ISO/IEC/IEEE 24765). Support software includes software tools and system software (Ref. NQA-1). ***Note***: SSC and Non-SSC software may have support software.
system software	An element of support software, the computer programs used to provide basic or general functionality and facilitate the operation and maintenance of the application computer program. Examples include lower level software layers, assemblers, interpreters, diagnostics, and utilities. (Based on NQA-1).
test case	A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. (Ref. NQA-1).

testing (software)	The process of: (a) operating a system (i.e., software and hardware) or system component under specified conditions (b) observing and recording the results (c) making an evaluation of some aspect of the system (i.e., software and hardware) or system component in order to verify that it satisfies specified requirements and to identify errors (Ref. NQA-1)
test plan (procedure)	A document that describes the approach to be followed for testing a system or component. Typical contents identify the items to be tested, tasks to be performed, and responsibilities for the testing activities. (Ref. NQA-1).
toolbox code	Software that is listed on the DOE Safety Software Quality Assurance Central Registry (Registry). (Ref. P1040).
validation (software)	The process of exercising or evaluating a system or system component by manual or automated means to ensure that it satisfies the specified requirements and to identify differences between expected and actual results in an operating environment (Ref. NQA-1); and providing evidence that the software, and its associated products, satisfies system requirements allocated to software at the end of each life cycle activity, solves the right problem (e.g., correctly models physical laws, implements business rules, uses the proper system assumptions), and satisfies the intended use and user needs (Ref. DOE O 414.1D).
verification	The act of reviewing, inspecting, testing, checking, auditing, or otherwise determining and documenting whether items, processes, services, or documents conform to specified requirements (Ref. NQA-1); and providing objective evidence that the software and its associated products conform to requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (e.g., acquisition, supply, development, operation, and maintenance); satisfy standards, practices, and conventions during life cycle processes; successfully complete each life cycle activity; and satisfy all the criteria for initiating succeeding life cycle activities (e.g., building the software correctly) (Ref. O 414.1D).

Table 21.1-A2 Chapter 21 Acronyms	
Acroynm[1]	**Definition**
A	Approve
AFU	Approval for Use
ASCE	American Society of Civil Engineers
AP	Administrative Procedure
ASM	Acquisition Services Management
ASME	American Society of Mechanical Engineers
BAS	Building Automation System
BIO	Basis for Interim Operations
CAM	Continuous Air Monitor
CGD	Commercial Grade Dedication (Dedicated)
CM	Configuration Management
CoE	Conduct of Engineering
COOP	Continuity of Operations Plan
Cx	Commissioning
D	Develop
DA	Design Authority
DAG	Design Agency
DCF	Design Change Form
DOE	(United States) Department of Energy
DRN	Design Revision Notice
DSA	Documented Safety Analysis
EF	Essential Function
ES	Engineering Services
ES-Div	Engineering Services Division
ESM	Engineering Standards Manual
ESSC	Engineering Services Software Coordinator
FAC-COE	Facility Conduct of Engineering
FACP	Fire Alarm Control Panel
FCR	Field Change Request
FDAR	Facility Design Authority Representative
FDD	Facility Design Description
G	Guide
Gr	Grade
IEEE	Institute of Electrical and Electronics Engineers
IESL	LANL Institutional Evaluated Supplier List
IWR	Interim Work Request
LANL	Los Alamos National Laboratory
LCxA	LANL Commissioning Authority
ML	Management Level
NA	Not Applicable
NQA-1	ASME NQA-1-2008/NQA-1A-2009, Quality Assurance Requirements for Nuclear Facility Applications, Part I and Part II
OCIO	Office of Chief Information Officer

Table 21.1-A2 Chapter 21 Acronyms	
Acroynm[1]	**Definition**
O&M	Operations and Maintenance
OTS	Off the Shelf
PDSA	Preliminary Documented Safety Analysis
PFD	Process Flow Diagram
P&ID	Process and Instrumentation Diagram
POC	Point of Contact
R	Required or review (see context)
RE	Responsible Engineer
RLM	Responsible Line Manager
SB	Safety Basis Division
SBP	Safety Basis Procedure
S/CI	Suspect/Counterfeit Item
SDD	System Design Description
SHADS	Safety and Hazard Analysis Software
SMACS	Safety Management and Administrative Controls Software
SME	Subject Matter Expert
SO	Software Owner
SOO	Sequence of Operations
SOW	Statement of Work
SPOC	Software Point of Contact
SRLM	Software Responsible Line Manager
SSC	Structure, System, or Component
SSS	Safety System Software
SRL	Software Risk Level
SU	Software User
SWAU	Software Approve (Approval) for Use
SWDD	Software Design Documentation (or Document)
SWBL	Software Baseline
SWDS	Software Data Sheet
SWHA	Software Hazards Analysis
SWID	Software Identification Number
SWNCP	Non-SSC Software Change Package
SWRS	Software Requirements Specification
SWYRS	Software System Requirements Specification
SWTM	Software Traceability Matrix
SWTP	Software Test Plan
SWTR	Software Test Report
TA	Technical Area
UPS	Uninterrupted Power Supply
V&V	Verify and Validate (or verification and validation)

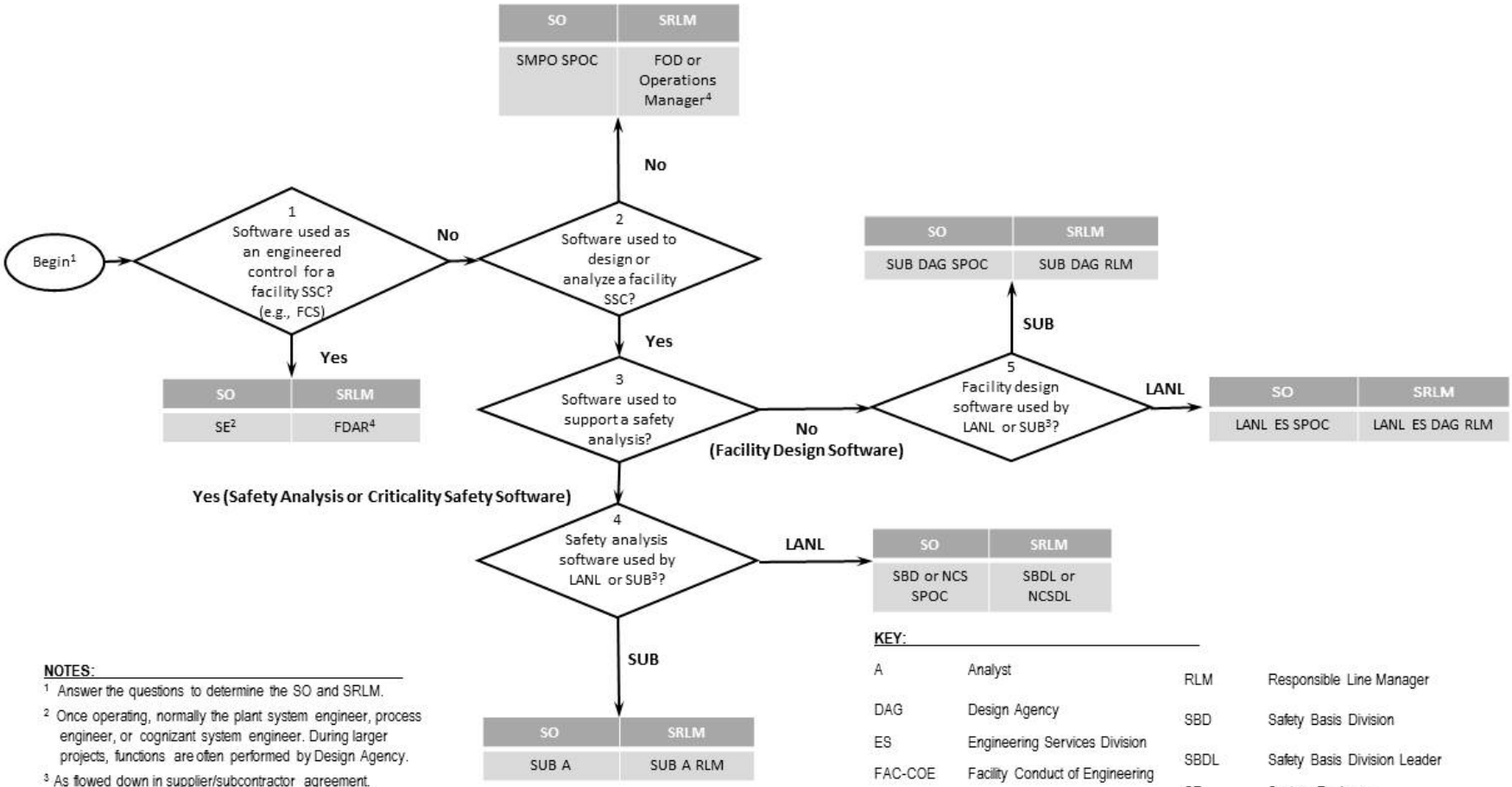
[1] Only key acronyms are listed. See PD340, *Conduct of Engineering for Facility Work* and documents referenced therein for additional meanings.

APPENDIX B: CHAPTER 21 SO AND SRLM DECISION DIAGRAM (GUIDANCE)

Use Figure 21.1-B1 to identify the SO and SRLM. For clarification, contact the Facility Design Authority Representative (FDAR).

- For software used within a single facility/application and/or under a single FDAR, the associated FDAR has the ultimate authority to determine the appropriate SO and SRLM.
- For software used among multiple facilities/applications where there may be multiple FDARs, the LANL Facility Design Authority (DA) has the ultimate authority to make the determination.
- *For fire alarm control panel (FACP) software, the system engineer is the SO. Fire Protection Division Office (FP-DO) provides support to the SO to promote chapter compliance.*

Figure 21.1-B1 Chapter 21 SO and SRLM Decision Diagram (Guidance)



NOTES:

[1] Answer the questions to determine the SO and SRLM.

[2] Once operating, normally the plant system engineer, process engineer, or cognizant system engineer. During larger projects, functions are often performed by Design Agency.

[3] As flowed down in supplier/subcontractor agreement. Criticality Safety decisions use same general logic as SBD.

[4] For software used across multiple FODs, the SMP Program Owner should be the SRLM.

[5] For clarification, contact the FDAR who has the ultimate authority to determine the appropriate SO and SRLM .

[6] See SOFT-GEN App. A, Definitions for administrative and engineered control definitions.

KEY:

A	Analyst	RLM	Responsible Line Manager
DAG	Design Agency	SBD	Safety Basis Division
ES	Engineering Services Division	SBDL	Safety Basis Division Leader
FAC-COE	Facility Conduct of Engineering	SE	System Engineer
FCS	Facility Control System	SMPO	Safety Management Program Owner
FDAR	Facility Design Authority Representative	SO	Software Owner
FOD	Facility Operating Director	SPOC	Software Point of Contact
MAR	Material at Risk	SRLM	Software Responsible Line Manager
NCS	Nuclear Criticality Safety	SSC	System, Structure, Component
NCSDL	Nuclear Criticality Safety Division Leader	SUB	Subcontractor

APPENDIX C: CHAPTER 21 REFERENCE LISTING

Most recent document revision (including revised document numbering and/or titles) applies unless otherwise stated. *Most national standards are available to LANL personnel via links; others must purchase.*

10 CFR 830, *Nuclear Safety Management*

10 CFR 835, *Occupational Radiation Protection*

48 CFR 970-5223-1, *Integration of Environment, Safety, and Health into Work Planning and Execution*

ANSI/ANS-10.7, *Non-Real-Time, High-Integrity Software for the Nuclear Industry – Development Requirements*

ASME NQA-1-2008/NQA-1A-2009, *Quality Assurance Requirements for Nuclear Facility Applications*, Part I and Part II

ASME NQA-1-2015 Part III, Subpart 3.2-2.14, *Quality Assurance Requirements for Commercial Grade Items and Services, Commercial Grade Computer Programs, and Software Services*

ASQ *Certified Software Quality Engineer Handbook*, L. Westfall, Am. Soc of Quality Pres.

DOE G 413.3-21, *Cost Estimating Guide*

DOE O 414.1D, *Quality Assurance*

DOE Safety Software Quality Assurance Central Registry

DOE G 414.1-4, *Safety Software Guide for Use with 10 CFR 830 Subpart A, Quality Assurance Requirements, and DOE O 414.1C, Quality Assurance* (Revision A pending)

DOE G 450.4-1C, *Integrated Safety Management System Guide*

DOE P 450.4A, *Integrated Safety Management Policy*

DOE SQAS21.01.00-1999, *Software Risk Management–A Practical Guide*

DOE-STD-1073, *Configuration Management*

DOE-STD-1195, *Design of Safety Significant Safety Instrumented Systems Used at DOE Nonreactor Nuclear Facilities*

EPA/100/K-09/003, *Guidance on the Development, Evaluation, and Application of Environmental Models, www.epa.gov/crem*

IAEA TR. No. 397, *Quality Assurance for Software Important to Safety*, IAEA, 2000

IEEE

ANSI/IEEE Std 7-4.3.2-2010, *IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations*

ANSI/IEEE Std 828, *IEEE Standard for Configuration Management in Systems and Software Engineering*

ANSI/IEEE Std 26514, *Systems and Software Engineering—Requirements for Designers and Developers of User Documentation*

ANSI/IEEE Std 29148, *Systems and Software Engineering – Life Cycle Processes – Requirements Engineering*

IEEE Std 344, *Standard for Seismic Qualification of Equipment for Nuclear Power Generating Stations*

IEEE Std 830, *IEEE Recommended Practice for Software Specifications*

IEEE Std 1012, *IEEE Standard for System and Software Verification*

IEEE Std 1016, *IEEE Standard for Information Technology – System Design – Software Design Descriptions*

IEEE Std 1016.1, *IEEE Guide to Software Design Descriptions*

IEEE Std 1028, *IEEE Standard for Software Reviews and Audits*

IEEE Std 1666, IEEE *Standard for Standard System C Language Reference Manual*

IEEE Std 12207, *Systems and Software Engineering – Software Life Cycle Processes*

ISO/IEC/IEEE 24765, *Systems and Software Engineering – Vocabulary*

ISO/IEC/IEEE 29119, *Software and systems engineering—Software testing*

ISO/IEC/IEEE 29148, *Systems and Software Engineering-Life Cycle Processes-Requirements Engineering*

ANSI/ISA S84.00.01, *Functional Safety: Safety Instrumented Systems for the Process Industry Sector*

LANL (internal-only unless noted)

AP-341-402, *Engineering Document Management in Operating Facilities*

AP-341-405, *Identification and Control of Technical Baseline, Variances, Alternate Methods, and Clarifications in Operating Facilities*

AP-341-502, *Management Level Determination and Identification of Quality Assurance and Maintenance Requirements*

AP-341-504, *Temporary Modification Control*

AP-341-507, *SSC Control Software Change Package*

AP-341-510, *Field Walk-down, Data Gathering, and Inspections*

AP-341-519, *Design Revision Control*

AP-341-605, *Calculations*

AP-341-601, *Functions and Requirements Document*

AP-341-602, *Requirements and Criteria Document (RCD)*

AP-341-611, *System Design Descriptions*

AP-341-613, *Instrumentation Set Point Control*

AP-341-616, *Technical Baseline Change During Design*

AP-341-620, *Review of LANL Produced Design Documents*

AP-341-621, *Design Authority Technical Review*

AP-341-622, *LANL Review of Designs Produced by External Agencies*

AP-341-703, *Commercial Grade Dedication*

AP-341-801, *Post Modification/Post Maintenance Testing*

AP-341-802, *System Health Reporting*

AP-341-901, *Performing Vital Safety System Assessments*

AP-350-406, *Startup and Commissioning*

ASM 3400.00.0410, *Goods or Services Requiring Special Review/Approval*

ASM Website

Conduct of Operations Resources

CIO-P100, *Cyber Security Risk Management*

SEO-COOP-006, *Continuity of Operations (COOP) Plan*

ES Division Office (ES-Div) Website

ESD Training Program Plan, ESD-PLAN-001

ESDO-AP-002, *Engineering Services (ES) Software Inventory Instructions*

Form 2033, *Safety/Non-Safety Software Determination, Categorization, and Software Risk Level (SRL)*

Information System Security Officer (ISSO)

LANL Institutional Supplier List (IESL)

LANL Nuclear Facility List

LANL Safety Basis Document List

LA-14167-MS, Los Alamos National Laboratory, *Concepts of Model Verification and Validation,* Thacker et al., 2004

P204-1, *Controlled Unclassified Information*

P218, *Cyber Security Controls*

P322-4, *Issues Management*

P330-2, *Control and Calibration of Measuring and Test Equipment (M&TE)*

P330-3, *Quality Audits*

P330-5, *Special Processes*

P330-6, *Nonconformance Reporting*

P330-8, *Inspection and Test*

P330-9, *Suspect/Counterfeit Items (S/CI)*

P341, *Facility Engineering Processes Manual*

P342, *Engineering Standards*

P343, *Facility Engineering Training and Qualification Manual*

P781-1, *Conduct of Training*

P805, *Export Control*

P840-1, *Quality Assurance for Procurements*

P1020-2, *Laboratory Document Control*

P1040, *Software Quality Management*

P1011, *Computer Hardware, Software, and Services: Exceptions to Acquisition, Maintenance, and Support*

PD110, *Safety Basis*

PD115, *LANL Readiness Program*

PD210, *Cyber Security Program*

PD328, *LANL Assessment Program*

PD340, *Conduct of Engineering for Facility Work*

PD370, *Conduct of Engineering for Research and Development (R&D)*

QPA-DO-FSD-001, *LANL Safety Software Inventory Procedure*

SBP-15-351, *Design Basis or Safety Basis Change Review*

SBP111-1, *Facility Hazard Categorization and Documentation*

SBP112-3, *Unreviewed Safety Question (USQ) Process*

SBP113-3, *Unreviewed Safety Issue (USI) Process*

SD130, *Nuclear Criticality Safety Program*

SD330, *Los Alamos National Laboratory Quality Assurance Program*

STD-342-100, *Engineering Standards Manual* (available externally)

Merriam-Webster Dictionary

TeamForge (LANL-internal)

Trac