arseam

T9S900D970-CMPA_REDACTED.pdf

01/06/20 09:17 AM

Xerox® WorkCentre® 7855

REVISIONS

REV	DESCRIPTION	DATE	APPROVED

GENERAL ATOMICS
ELECTROMAGNETICS

16969 MESAMINT STREET
SAN DIEGO, CA 92127

DRAWN K.FALAMINIANO	DATE 1/22/14	TRIGA INL SOFTWARE CONFIGURATION MANAGEMENT PLAN					
CHECKED Angela Garza	1-23-14						
ENGR JAY ZHANG	01/24/14						
ENGR REV Ryan Brown	1-24-14						
MFG ENGR							
PROJ MGR J Hoffman	1/23/14	SIZE A	CAGE CODE 4XB60	DRAWING NUMBER T9S900D970-CMP			REV A
QUAL MGR K.Boesec	1/24/14						
RELEASE M. SMITH	1-24-14			DRW LEVEL 3	SCALE NONE		SHEET 1 OF 13

TABLE OF CONTENTS

1. INTRODUCTION

1.1 PURPOSE

This Software Configuration Management (SCM) Plan (SCMP) provides the guidelines to be used to manage changes to the TRIGA software at General Atomics Electronic Systems, Inc. (GA-ESI). The intended audience for this document includes and is not limited to Project Management, Software Quality Assurance, and Software Engineering personnel.

This SCMP will be used to ensure compliance to the SCM requirements as listed in the Idaho National Laboratory (INL) Statement of Work (SOW).

1.2 SCOPE

Software Configuration Management functions will be performed as described in this document throughout the TRIGA Software Development Life Cycle (SDLC). This SCMP will be used to track and control changes in project documentation, software source code, software build artifacts, test tools, and test artifacts as described in Section 3.1 of this document.

This SCMP is used in conjunction with GA-ESI Configuration Management (CM) operating procedures. There are no known limitations to this plan. No assumptions have been made with respects to this plan.

1.3 REFERENCE DOCUMENTS

Reference ID	Issued By	Document ID	Document Title	Revision
RD.1	IEEE	IEEE Std 828	IEEE Standard for Software Configuration Management Plans	2005
RD.9	ASME	NQA-1	Quality Assurance Requirements for Nuclear Facility Applications	2000
RD.2	INL	SOW-8330	Replacement of the NRAD Instrumentation and Control Console	0
RD.3	GA-ESI	T9S900D970-SWP	TRIGA INL Console Software Development Plan	A
RD.4	GA-ESI	T9S900D970-SWP-2	TRIGA INL Channels Software Development Plan	A
RD.5	GA-ESI	T9S900D97003	Software Quality Assurance Plan	A
RD.6	GA-ESI	OP-4.0-130	Engineering Change Notice	AA
RD.7	GA-ESI	OP-6.5-110	Standard Configuration Identification	C
RD.8	GA-ESI	OP-6.1-100	Configuration Management Systems	E

2. SCM MANAGEMENT

2.1 ORGANIZATION

The personnel involved in the development of the TRIGA software are identified in the TRIGA INL Console Software Development Plan (RD.3). The organization of the development team includes members from Project Management, Software Engineering, Software Quality Assurance, and Software Test. Members from each functional group make up the Software Change Control Board (SWCCB).

The table in Section 2.2 identifies the roles involved in the SDLC and how each role is responsible for following and/or implementing the SCM activities outlined in this plan.

2.2 SOFTWARE CM RESPONSIBILITIES

Role	Responsibilities
Project Management	Responsible for ensuring the SCM process is implemented. Defines scope, resources and schedule for SCM activities.
Software Development Manager	Responsible for ensuring software development activities follow software configuration management practices as outlined in this document throughout the development cycle.
Software Configuration Management Engineer	Creates and maintains the project SCMP to coincide with the current software development practices. Oversees all SCM activities. Performs software builds and software release activities.
Software Quality Assurance Engineer	Performs audits to ensure adherence to CM and SCM procedures. Performs audits to verify release preparedness.
Software Engineer	Updates and adds to the existing software baseline. Ensures modifications made to source code and/or product manuals are version controlled.
Software Test Manager	Ensures that test documentation and artifacts are up-to-date and is supplied to SCM for document control.
Software Requirements Engineer	Responsible for creating and maintaining software requirements specifications. Ensures software requirements and specifications are under document control.
Software Change Control Board	Reviews and dispositions changes to the current software baseline.

2.3 APPLICABLE POLICIES, DIRECTIVES, AND PROCEDURES

See Section 1.4 of this document.

This SCMP is written utilizing the IEEE Std 828-2005 template as a guideline.

2.4 MANAGEMENT OF THE SCM PROCESS

See Section 2.2 of this SCMP for the roles and responsibilities for the management of the SCM process.

Independent surveillance of the SCM activities to ensure compliance with this SCMP will be performed by the Software Quality Assurance (SQA) Engineer.

The identification, assessment, and plans for the mitigation of risks associated with the performance of SCM activities will be handled by project management.

3. SCM ACTIVITIES

Software Configuration Management is the discipline of controlling and tracking changes made to a software system throughout the SDLC. SCM is applicable and not limited to software requirements, design, source code, and project documentation. The following subsections describe the activities involved with SCM, including:

- Configuration identification
- Configuration control
- Configuration status accounting
- Configuration evaluations and reviews
- Release management and deliveries

3.1 CONFIGURATION IDENTIFICATION

A Configuration Item (CI) is any component of a system, including documentation, which will be under the control of CM. These items are identified, recorded and managed within a Configuration Management System (CMS) and maintained throughout the lifecycle of the project.

3.1.1 Configuration Items

Configuration Items for a software system consist of software process plans, specification documentation, software source code, test documentation, technical manuals and version description documentation. The Software Configuration Item (SCI) structure to support the TRIGA software consists of the following:

3.1.1.1 Software Process Plans

Software process plans describe the project specific processes that will be used during the SDLC of the TRIGA project. The process plans outline the details of managing milestones and deliverables, software configuration management, and software quality assurance. Software process plans for the TRIGA project are identified in reference documents RD.3 and RD.4.

3.1.1.2 Specification Documentation

Specification documentation state the functions and capabilities that a system must provide and the constraints that are applied to the system. Specifications can be documented in several ways including; textual descriptions, graphical models or drawings, and mathematical models. Specifications for the TRIGA project are identified in reference documents RD.3 and RD.4.

3.1.1.3 Technical Manuals

Technical manuals describe the operation, installation, administration, and maintenance of the TRIGA system. Technical manuals for the TRIGA project include:

- Operating and Maintenance Manuals
- Programmer Reference Manuals

3.1.1.4 Test Documentation

Test documentation describes the test plans, tools, test beds, approach, methods, test cases, test procedures and test reports that support the Verification and Validation effort. Test documentation for the TRIGA project is identified in reference documents RD.3 and RD.4.

3.1.1.5 Software Code Base

The software code base consists of files that make up the software application. These files consist of source code, scripts, configuration and other files required to create builds and executable files for the TRIGA software.

3.1.2 Naming Configuration Items

3.1.2.1 Documentation Naming

3.1.2.1.1 External Project Documentation

External project documentation refers to documentation that is to be released and can be released to customer(s). External project documentation is identified as GA-ESI drawings and is assigned drawing part numbers.

3.1.2.1.2 Internal Project Documentation

Internal project documentation refers to documentation that is intended to be used internally and not to be released to customer(s). Internal project documentation may include project specific processes, internal audit reports, meeting minutes, etc. File names of these documents shall be defined by the author of the document. Document names shall be meaningful in order to clearly identify the purpose of the document.

3.1.2.2 Software Naming

3.1.2.2.1 Software Versioning Scheme

Software versions will have the basic structure:

"XX.YY.ZZ" (Decimal digits separated by periods)

XX is the major version

YY is the minor version

ZZ is the build number

The "XX.YY" portion of the version is fixed for a given production cycle of an existing product. This is generally referred to as the product development "Phase". The "ZZ" portion will increment from "01" for each successive SCM build, and will zero out each time a new production cycle begins (new "XX.YY"). Build number increments can increase by more than one. For new development, the "XX" portion of the version shall be fixed at "00" until the initial release of "01" is produced, and the "YY" minor version value can increment as phases/reviews are completed. Typically the first system test build would have a version value of "00.03.01".

A minor release would be built to resolve a specific issue with the behavior of an existing product (as in: "01.00" increments to "01.01"). The software changes are clearly minor. A major release may occur

annually or some multiple of that to, where significant changes to the behavior of the software are introduced (for example: "01.05" increments to "02.00").

Build number is always present, regardless of the phase of development. The first release of a new product could have the complete version string "01.00.00". There can be no incrementing the build number after formal release. (Build "01.00.00" cannot increment to "01.00.01" – the next valid number would be "01.01.01", implying the first build of a new product phase.)

3.1.2.2.2 Software Source Code

Software source code refers to all files required to create and build the software system. Software source code file names and its corresponding built output will be defined by the Software Engineer developing the software.

3.1.2.2.3 Software Label

A software label is a marker in the software version control system used to snapshot and capture significant moments in time during the SDLC. Software labels capture revisions made to the software source code and can be used to identify software builds, milestones, releases, etc. A software label shall be identified as *<SOFTWARE_NAME>_<VERSION>*. A sample software label for the TRIGA Console may be similar to *CONSOLE_01.01.00*.

Project	Release Label
Console	CONSOLE_03.04.00
NLW-1000	NLW1000_ 01.00.00
NMP-1000	NMP1000_ 01.00.00
Console (FAT Version)	CONSOLE_03.03.10
NLW-1000 (ATP Version)	NLW1000_00.00.15
NMP-1000 (ATP Version)	NLW1000_00.00.17

3.1.3 Acquiring Configuration Items

The TRIGA development effort utilizes several areas and/or tools to store and maintain CIs. The table below identifies the mechanism for storage, what CIs the storage mechanism controls, and information for accessing the storage area.

Storage	Configuration Items	Storage Information
DocCenter	- Software executables - Software libraries	[redacted]
	- External project documents - VDD	[redacted]
SharePoint	- Internal project documents (i.e. audits, reports, meeting minutes) - Draft revisions	[redacted]
Perforce	- Software source code	[redacted]

GA-ESI Network Storage	- Software builds - Software defect tracking database	[redacted]

NOTE: The X: drive is a location on the network as defined by GA-ESI's Document Center.

3.1.3.1 Console Software Source Code

Main development for the Console will occur on trunk. The project path is defined in Perforce as *//depot/TRIGA/console/trunk*. The Console source code as stored in Perforce supports multiple customers and is isolated such that customer specific code does not affect other customers. This is accomplished through the use of sub-directories named formatted as *<CUSTOMER_IDENTIFIER>_dir*. For example, the under *//depot/TRIGA/console/trunk/tbas_ckeys* customer specific code for the INL emulator is stored under the sub-directory *emuInl_dir*.

3.2 CONFIGURATION CONTROL

Configuration control defines the process for requesting, evaluating, approving or disapproving, and implementing changes to baselined CIs. Changes can include but are not limited to defect, enhancements and new requirements.

A baseline provides a static reference point to a grouping of CIs that make up a system at a given point in time. Baselines establish a version of the software configuration which serves as the basis for further development. After a baseline has been established, changes scope can only be performed through a formal change request process as identified in Section 3.2.1. See the table below for baseline descriptions.

Baseline	Description
Requirements	The point where the initial requirements for the system have been identified.
Functional	The point where the all functions of the system have been identified and have been allocated to the system components in which they will be performed.
Developmental	The current state of the project CIs during the development phase.
Product	The final version of the system.

3.2.1 Requesting, Evaluating and Approving Changes

Figure 1 below identifies the standard workflow for making changes to a defined baseline.

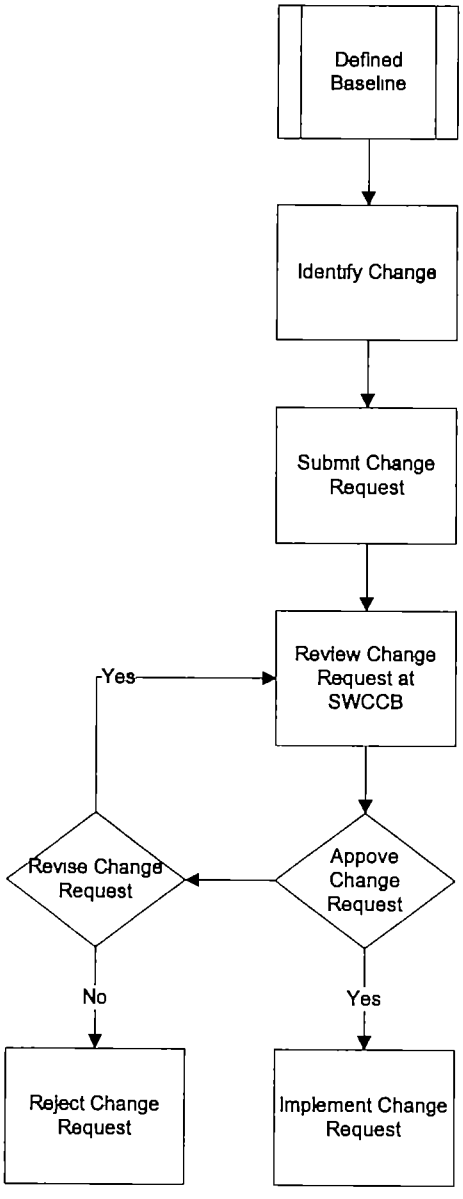

Figure 1: Requesting, Evaluating and Approving Changes

3.2.2 Implementing Changes

3.2.2.1 Implementing Documentation Changes

Releasable customer documents following the initial release will be handled through Engineering Change Notice (ECN) process as described in RD.6.

3.2.2.2 Implementing Software Changes

3.2.2.2.1 Software Defect Changes

Defects discovered during the testing phase will be recorded in the software defect tracking tool database as a "Defect". When the issue is created, the following information shall be collected:

- Issue Type – Defects will be entered as Defect
- Affects Version/s – The software version the defect was found in
- Summary – A brief overview of the problem
- Description – A detailed description of the problem (i.e. expected output)
- Steps to reproduce – Steps used to reproduce the problem

3.2.2.2.2 Feature Changes

Changes to the software due to changes in requirements will be recorded in the software defect tracking tool database as a "Requirement". When the issue is created the following information shall be collected:

- Issue Type – Feature changes will be entered as Requirement
- Affects Version/s – The software version the feature changes affects
- Summary – A brief overview of the feature change
- Description – A detailed description of the change (i.e. Requirement and revision of the requirement specification to be changed)

3.3 CONFIGURATION ACCOUNTING STATUS

The SCM engineer is responsible for the recording and reporting of software configuration status. For software product builds performed by the SCM engineer, a configuration status report will be generated identifying the built software version, included Software Change Requests (SCRs), known software limitations, and additional developer notes associated with each SCR.

Configuration status reports will be controlled as release notes and stored on the project's SharePoint site. A copy of the release notes will be provided with each build in the designated build area located on the GA-ESI network. Refer to the table in Section 3.1.3 for the base location of where builds are stored on the GA-ESI network.

3.4 CONFIGURATION EVALUATION AND REVIEWS

Configuration evaluations and reviews will be used as the mechanism to evaluate a baseline. The SCM engineer along with the SQA engineer will schedule audits, on an as needed basis, to determine the extent to which the physical and functional characteristics of a CI are met. At a minimum, configuration reviews should take place upon definition and completion of the Requirements and Product Baselines.

3.5 Release Management And Delivery

The standard software release management and delivery process will be used.

4. SCM SCHEDULES

Software Configuration Management activities will coincide with the dates and milestones as defined in the TRIGA project schedule. The table listed below identifies the planned dates and duration for each SCM activity as listed in Section 3 of this SCMP. Due to the fluidity of the project schedule, planned dates and duration have been listed relative to dates as define in the project schedule.

Software Configuration Management Activity	Planned Dates	Duration
Configuration identification	From start of project	Throughout SDLC
Configuration Control	From start of project	Throughout SDLC
Configuration Status Accounting	From date of initial software build	Throughout SDLC
Configuration Evaluations and Reviews	One week after definition of baseline to be audited	One day for each audit
Release Management and Deliveries	Two weeks prior to release	Until release of project

5. SCM RESOURCES

This section identifies and describes the tools and equipment used to support the configuration management activities during the software development process.

5.1 Software Build Tools

The software and software build tools used during the development of the TRIGA software is captured in the document entitled TRIGA Software and Tools located on the TRIGA SharePoint site.

5.2 Software Build Environment

Build Machine	Operating System	Architecture
[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	[redacted]

6. SCM PLAN MAINTENANCE

The Software Configuration Manager is responsible for the development and maintenance of this plan. This SCMP will be updated as needed throughout the SDLC to ensure that the current SCM policies are relevant and adequate for the development process. Updates to this plan will follow the same SCM guidelines as outlined in this document.

APPENDIX A: DEFINITIONS AND ACRONYMS

❖ DEFINITIONS

Term	Definitions
Baseline	A version of a configuration item which serves as the basis for further development and may be changed only through formal change control procedures. [Adapted from IEEE 610.12-1990].

❖ ACRONYMS

Acronym	Description
CI	Configuration Item
CM	Configuration Management
GA-ESI	General Atomics Electronic Systems, Inc.
PM	Project Management
SCM	Software Configuration Management
SCMP	Software Configuration Management Plan
SDLC	Software Development Lifecycle
SWCCB	Software Change Control Board
SDP	Software Development Plan
SDM	Software Development Manager
SQA	Software Quality Assurance
SQE	Software Quality Engineer
SCME	Software Configuration Management Engineer
SE	Software Engineer
STM	Software Test Manager
STE	Software Test Engineer
SRE	Software Requirements Engineer
VDD	Version Description Document
SOW	Statement of Work
INL	Idaho National Laboratory