

<i>UF/NRE</i> <i>UFTR</i>	<i>QUALITY ASSURANCE DOCUMENT</i>	<i>Project ID: QA-1</i>	
		<i>Revision 0</i>	<i>Copy 1</i>
		<i>Page 1 of 41</i>	

Project Title: *UFTR DIGITAL CONTROL SYSTEM UPGRADE*

UFTR-QA1-05, Software Safety Plan (SSP)

Prepared by,

Dr. Gabriel Ghita

 (Signature)

Date: *02/17/10*

Reviewed by,

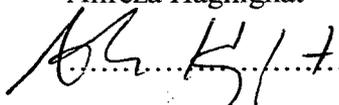
Prof. James Baciak

 (Signature)

Date: *02/17/10*

Approved by,

Alireza Haghghat

 (Signature)

Date: *2/17/10*

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 4 of 41</i>

TABLE OF CONTENTS

1. Introduction.....8

1.1 Purpose.....8

1.1.1 Software Safety Requirements.....8

1.1.2 TXS Platform Qualification11

1.2 Scope.....12

2. Definitions, acronyms and abbreviations, and references13

2.1 Definitions13

2.2 Acronyms and Abbreviations15

2.3 References16

2.3.1 UF Documents16

2.3.2 AREVA NP Inc. Documents16

2.3.3 Industry Standards17

2.3.4 NRC Documents.....17

3. Software safety management19

3.1 Organization and responsibilities19

3.1.1 Project Organization19

3.1.2 Responsibilities.....19

3.1.2.1 UF Project Manager.....19

3.1.2.2 Software Development Lead19

3.1.2.3 Software Development Group.....20

3.1.2.4 IV&V Group.....20

3.1.2.5 Quality Assurance Project Auditor21

3.2 Resources21

3.3 Staff qualification and training.....21

3.4 Software life cycle.....21

3.5 Documentation requirements.....22

3.5.1 Software Project Management22

3.5.2 Software Configuration Management.....22

3.5.3 Software Quality Assurance.....22

3.5.4 Software Safety Requirements.....22

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 5 of 41</i>

3.5.5	Software Safety Design.....	22
3.5.6	Software Development Methodology	23
3.5.7	Test Documentation	23
3.5.8	Software Verification and Validation Plan.....	23
3.5.9	Reporting Safety Verification and Validation.....	24
3.5.10	Software User Documentation	24
3.5.11	Results of Software Safety Requirements Analysis	24
3.5.12	Results of Software Safety Design Analysis.....	24
3.5.13	Results of Software Safety Code Analysis	24
3.5.14	Results of Software Safety Test Analysis	24
3.5.15	Results of Software Safety Change Analysis	24
3.6	Software safety program records	24
3.6.1	Scope of Safety Program Records	24
3.6.2	Responsibility for Software Safety Program Records.....	25
3.6.3	Maintenance of Software Safety Program Records.....	25
3.6.4	Open Item Tracking System	25
3.7	Software configuration management activities	25
3.8	Software quality assurance activities	26
3.9	Software verification and validation (SV&V) activities	26
3.10	Tool support and approval.....	26
3.10.1	Tool Approval.....	26
3.10.2	Installation of Upgrades to Approved Tools	27
3.10.3	Withdrawal of Previously Approved Tools	27
3.10.4	Identification of Limitations	27
3.11	Previously developed or purchased software	27
3.12	Subcontract management.....	27
3.13	Process certification	28
4.	Software safety analyses.....	29
4.1	Software safety analyses preparation.....	31
4.2	Software safety requirements analysis.....	32

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 6 of 41

4.2.1	Application Software Requirements Phase and Requirements Traceability Matrix.....	32
4.2.2	Verification and Validation (V&V).....	32
4.3	Software safety design analysis.....	33
4.4	Software safety code analysis - Automatic Code Generation.....	33
4.5	Software safety test analysis.....	33
4.5.1	SIVAT Testing.....	33
4.5.2	Test Phase - Factory Acceptance Testing (FAT).....	34
4.6	Software safety change analysis.....	34
4.7	TXS Cyber Security	34
4.7.1	TXS Features that Minimize Cyber Security Vulnerabilities.....	35
4.7.2	TXS Monitoring and Service Interface (MSI).....	35
4.7.3	TXS Computer Message Transfer Protocols.....	36
4.7.4	Access Control.....	36
4.7.5	Remote Access Control.....	37
4.7.6	TXS Software Development Process Controls	37
4.7.7	Process Controls for Software Developed and Maintained by AREVA NP GmbH.....	37
4.7.8	Process Controls for Software Developed and Maintained by UFTR.....	37
5.	Post development	39
5.1	Training.....	39
5.2	Deployment	39
5.2.1	Installation TXS.....	39
5.2.2	Startup and transition – Commissioning.....	39
5.2.3	Operations support	39
5.3	Monitoring	40
5.4	Maintenance	40
5.5	Retirement and notification	40
6.	Plan approval	41
6.1	Responsibilities	41

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 7 of 41</i>

6.2 Updates.....41

6.3 Change Approval41

6.4 Change Distribution.....41

<i>UF/NRE UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 8 of 41</i>

1. Introduction

1.1 Purpose

This University of Florida Training Reactor (UFTR) Software Safety Plan (SSP) outlines the process for achieving high functional reliability and design quality for the safety-critical Application Software for the replacement of the UFTR's analog reactor protection system (RPS). Software safety is defined as "freedom from software hazards" (IEEE Std 1228-1994, /29/), where a software hazard is "a software condition that is a prerequisite to an accident" (IEEE Std 1228-1994, /29/).

The UFTR replacement RPS is planned to use the TELEPERM XS (TXS) platform and application software designed and developed to provide appropriate RPS safety functions. Planned and documented software safety analysis activities, defined within this plan, will be employed to ensure the achievement of safety objectives such that safety system software development is consistent with the defined system safety analyses. The Software Safety Analysis activities are Defense-in-Depth and Diversity (D3) Analysis (4.1), Software Requirements Traceability Matrix (RTM) (4.2.1), Verification and Validation (V&V) (4.2.2), Software safety design analysis (4.3), Automatic Code Generation (4.4), SIVAT Testing (4.5.1), Factory Acceptance Testing (FAT) (4.5.2), and Software Safety Change Analysis (4.6). For this application, the defined system safety analysis is reported in the UFTR Final Safety Analysis Report (FSAR), /16/, and UFTR Supplemental SAR (SSAR), /17/. The software safety analysis activities ensure that:

- System safety requirements as specified in UFTR Technical Specifications have been met correctly (this includes requirements and assumptions of the system safety analysis);
- Software elements that can affect safety are identified;
- Safety problems and resolutions identified in these analyses are documented.

This Plan is prepared following the guidance defined in the IEEE Std 1228-1994, "Standard for Software Safety Plans," /29/, and NUREG-0800 BTP HICB-14, /37/.

1.1.1 Software Safety Requirements

NUREG 1537 Part 2, February 1996, "Guidelines for Preparing and Reviewing Applications for the Licensing of Non-Power Reactors," /40/, indicates that "hardware and software for computerized systems should meet the guidelines of IEEE Standard 7-4.3.2-1993, "Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations", /21/, and RG 1.152, Revision 1, "Criteria for Digital Computers in Safety Systems of Nuclear Power Plants,"/36/, and software should meet the guidelines of ANSI/ANS 10.4-1987, /30/, applicable to non-power reactor systems." ANSI/ANS 15.15-1978, /32/, and the ANS 15.20, /33/, are indicated as being useful as general guides for the design, implementation, and evaluation of I&C systems for non-power reactors and should (but not shall) be used where applicable.

<i>UF/NRE UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 9 of 41</i>

Reference /37/: NUREG-0800, "Standard Review Plan", Chapter 7, Branch Technical Position HICB 14, "Guidance on Software Reviews for Digital Computer Based Instrumentation and Control Systems", Revision 4, June, 1997, Section 3.1.i, "Software Safety Plan" and Section 3.2.a, "Safety Analysis Activities" is applicable to power reactors but the software safety principles contained in this reference are applied to this Plan. In addition, the reference /38/: NUREG-0800, "Standard Review Plan", Chapter 7, Branch Technical Position HICB 19, "Guidance for Evaluation of Defense-in-Depth and Diversity in Digital Computer-Based Instrumentation and Control Systems," Revision 4, June, 1997 is also applicable to power reactors but is referenced and used as appropriate for the UFTR application.

The software safety analysis activities defined in this Plan are based on the importance to safety of the functions to be provided in the UFTR Replacement RPS. The basis for the safety significance of the RPS functions is the safety analysis reported in the UFTR FSAR (Chapter 15), /16/, and UFTR SSAR (Chapter 13), /17/. Several accident scenarios are considered in the aforementioned reports; below, we describe a few severe scenarios:

Loss of Coolant Accident (LOCA)

The first indication of a LOCA will come from the flow rate monitor within the primary loop return line. Once this rate falls below prescribed LSSS levels, the reactor is tripped, and the control blades drop to prevent excessive heating within the core. If these sensors do not operate correctly, the LOCA may also be detected by water level monitors, area radiation monitors, temperature sensors within the fuel boxes, and at inlet and outlet of the primary and secondary coolant loops. Any of these sensors will initiate the control blade drop trip. Operator initiated trip of the reactor is also be feasible since each of these sensors are linked to indicators within the control room.

It should also be emphasized that insertion of the control blades is not necessary, and the reactor will shut itself down because of negative coefficient of reactivity caused by moderator void effect. The FSAR Appendix 13C /16/ shows that in the event of a LOCA, the fuel plate temperature rise will not approach temperatures of even half the melting point of aluminum. It must be concluded then, that a loss of cooling flow accident in no way represents a hazard to core structural integrity of the UFTR.

Control Blade System Malfunction

The blades are sustained in a raised position by means of a motor, acting through an electromagnetic clutch. Interruption of the magnet current results in a decoupling of the motor drive from the blade drive shaft, causing the blades to fall back into the core in a failsafe arrangement. In case of a loss of power, a manual scram, or any scram signal from the instrumentation system, the electromagnets are

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 10 of 41</i>

de-energized and the system fails safe by gravity dropping of the blades into the core.

The only way in which the blades could fail to fall into the reactor during a reactor scram would be through either failure of the circuits to de-energize the electromagnetic coupling, or a mechanical failure of the blade drives or jamming in the shroud. The operator can manually trip the reactor or turn off the power in the case of circuit malfunction. In the event of blade jamming, or combined circuit and operator failure, the reactor is shut down by the inherent design of the reactor described in FSAR Chapter 4, "Reactor," /16/, and by the water dump trip acting as an alternate shutdown mechanism. Additional mechanism for reactor shutdown is provided by the dumping of moderator/coolant via the rupture disk.

Loss of Power

If the power source on which the RPS functions is lost during operation, the electromagnets that hold the control blades out of the core are de-energized and the system fails safe by gravity dropping of the blades into the core. This sort of trip can also be initiated by shutting of the power from the building's circuit breaker. In the event of blade jamming, the reactor is shut down by the inherent design of the reactor and by the water dump trip acting as an alternate shutdown mechanism. Additional technique for reactor shutdown is provided by the dumping of moderator/coolant via the rupture disk.

Large Reactivity Addition

This type of accident occurs due to low reactor period and causes a high rate of power increase. This unsafe reactor condition would first be detected by the low power level NIs in both safety channels (BF3 and IC). If these fail to trip the reactor, the high power level NIs would detect power levels above the LSSS. The next method of defense is the system of 6 temperature sensors that measure core temperature, followed by the temperature sensors placed in the primary and secondary loops. Even if these systems do not respond at all, the reactivity excursion will not occur. The following is a summary of SSAR submitted for the UFTR HEU-LEU Fuel Conversion project, /17/:

a. Unprotected Sudden Insertion of 0.6% $\Delta k/k$

The unprotected insertion of 0.6% $\Delta k/k$ was modeled for 300 seconds to show that the power does not rise again after suppression of the initial power spike. However, the core does reach an equilibrium power level of about 600 kW.. Under these conditions, the coolant reaches the saturation temperature and boiling occurs in the uppermost nodes of the coolant channel. The peak temperatures of about 108°C in the fuel and cladding for the LEU core is well below the incipient melting temperatures of 582°C for the Al-6061 cladding of the LEU silicide fuel.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 11 of 41</i>

b. Sudden Insertion of the Maximum Allowed Excess Reactivity

For a step insertion of 1.4% $\Delta k/k$ the total energy release would be < 6.1 MWs and the maximum temperature of the cladding would be less than 300°C providing much conservatism. Thus, even without action of the reactivity control system, the UFTR can tolerate the sudden ejection of a maximum reactivity worth experiment, or equivalent reactivity insertion, without any fuel damage in the LEU fuel assembly design.

1.1.2 TXS Platform Qualification

The TXS technology is a mature and fully integrated nuclear safety system. The TXS hardware is fully qualified safety-related equipment. The TXS operating system software and Function Block library are developed and maintained using a life-cycle process (as described in the TXS Topical Report, /18/). The TXS Application Software is generated by the TXS Specification And Coding Environment (SPACE) software development tool. The TXS technology and development process have been reviewed and accepted by the NRC as stated in the NRC SER, /39/, on the TXS Topical Report, /18/. UFTR uses SIVAT (a simulation-based validation tool) for testing of the Application Software generated by the SPACE tool to detect errors that could prevent the software from fulfilling its safety function. The TXS technology is implemented with processes that remove risks associated with integration of software and hardware.

The risks associated with first-of-a-kind engineering work are minimized through the use of qualified software development tools and structured engineering analyses. The use of the TXS object-oriented automated code generation tool, i.e., SPACE, supports the development of high quality software with a less complex process, which minimizes the potential for human error and reduces the inherent risk in the development of the Application Software. SIVAT testing of the Application Software generated by the SPACE tool is used to detect errors that would prevent the software from fulfilling its safety function. These tools support the development of high quality software. Together, these two (2) software tools provide reasonable assurance that software errors are minimized. SIVAT testing, coupled with FAT are sufficient to ensure that there are no software hazards. The software risk is further minimized through the V&V process, which includes Application Software requirements traceability.

A D3 Analysis was performed in order to ensure that adequate defense-in-depth has been provided in the design (Section 4.1).

Section 4 of this Plan describes the software safety activities in detail.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 12 of 41</i>

1.2 Scope

The SSP is prepared considering the guidance in IEEE Std 1228-1994, /29/, and NUREG-0800 BTP HICB-14, /37/. The priority of this Plan is to support the intent and goal of IEEE Std 1228-1994, /29/, in producing the highest quality product.

This Plan applies to all safety-critical project-specific software items and related documentation for the design or modification of the TXS software developed for the UFTR RPS replacement project.

This SSP does not apply to the TXS system platform software development. The platform software is developed and maintained by AREVA NP GmbH (Germany) on a project-independent (generic) basis and is controlled by AREVA NP GmbH procedures. The applicable AREVA NP GmbH procedures for the TXS platform software have been reviewed and accepted by the NRC as stated in the NRC SER, /39/, on the TXS Topical Report, /18/. The TXS system platform software is purchased for the UFTR project as safety-related and is controlled via UFTR QA procedures.

It should be recognized that the responsibility to produce a “safe” Application Software product is not separate from the responsibility to produce a “quality” product, or a “functional” product. A single organization is accountable for both of these responsibilities. Each of these responsibilities is the task of the project team as defined in this Plan.

UF/NRE UFTR	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 13 of 41</i>

2. Definitions, acronyms and abbreviations, and references

2.1 Definitions

Certificate of Conformance (CoC)

A document signed or otherwise authenticated by an authorized individual certifying the degree to which items or services meet specified requirements.

Configuration Item (CI), [IEEE Std 610.12-1990, /25/]:

An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process.

Configuration Management (CM), [IEEE Std 610.12-1990, /25/]:

A discipline applying technical and administrative direction and surveillance to:

- Identify and document the functional and physical characteristics of a configuration item;
- Control changes to those characteristics;
- Record and report change processing and implementation status;
- Verify compliance with specified requirements.

Configuration Status Accounting, [IEEE Std 610.12-1990, /25/]:

An element of configuration management, consisting of the recording and reporting of information needed to manage a configuration effectively. This information includes a listing of the approved configuration identification, the status of proposed changes to the configuration, and the implementation status of approved changes.

Criticality Analysis, [IEEE Std 1012-1998, /27/]:

A structured evaluation of the software characteristics (e.g., safety, security, complexity, performance) for severity of system failure, system degradation, or failure to meet software requirements or system objectives.

FunBase

A design tool that administrates the naming of software modules, parameters, signals, data tables, and other entries in the design so that each entity is uniquely and consistently named and properly connected in the Application Software.

Open Item

A potential discrepancy, a potential improvement, or a possible anomaly from the required status or condition discovered during the phases of a TXS project.

Previously Developed Software, [IEEE Std 1228-1994, /29/]:

Software that has been produced prior to or independent of the project for which the Plan is prepared, including software that is obtained or purchased from outside sources.

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 14 of 41

Safety-Critical Software, [IEEE Std 1228-1994, /29/]:

Software whose inadvertent response to stimuli, failure to respond when required, response out-of sequence, or response in combination with other responses can result in a basis accident;

SIVAT, [TXS SIVATL, /19/]:

SIVAT allows the functionality of the I&C system engineered in SPACE to be tested by simulation. Simulation is based on the code generated by the FDG code generator and the RTE code generator. This enables engineering errors to be detected at an early stage. The objective of the test is to verify that the requirements have been translated into function diagrams without errors, and that the software automatically generated from these function diagrams provide the functionality required in terms of input and output response. The tests cover interface to the RTE, use of correct function blocks and whether they have been correctly connected and parameterized. The failure of I/O modules, processing modules and data messages can be simulated. The tests are run using scripts that define the input signals of the I&C system and the simulation run. The test results are recorded in log files and plots for further evaluation. Process models can also be linked into the simulator to perform closed-loop tests.

Software Life Cycle, [IEEE Std 610.12-1990, /25/]:

The period of time that begins when a software product is conceived and ends when the software is no longer available for use.

Software Requirements Traceability Matrix

A matrix that provides a method that can be used to trace and document software requirements have been met. It provides a complete view of software requirements to be tested, and it traces specific software requirements through all activities of software development to verify that the requirements were met. Additionally, the RTM formally documents the process and provides documented evidence that can be useful in verifying that safety requirements and licensing commitments were met.

SPACE, [TXS SPACEI, /20/]:

SPACE engineering system comprises the tools used for engineering and maintenance of the TXS I&C software. Engineering in this context refers to the overall process of creating and testing the I&C software. The SPACE tools are:

- Specification of the I&C functions and hardware topology;
- Automatic code generation;
- Software authentication (*reflist* and *scanmic*);
- Software Loading;
- Load Analysis tool;
- Database administration.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 15 of 41</i>

Validation, [*IEEE Std 610.12-1990, /25/*]:

The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies requirements.

Verification, [*IEEE Std 610.12-1990, /25/*]:

- The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase;
- Formal proof of program correctness.

V&V (Verification and Validation), [*IEEE Std 610.12-1990, /25/*]:

The process of determining whether:

- The requirements for a system or component are complete and correct;
- The products of each development phase fulfill the requirements or conditions imposed by the previous phase;
- The final system or component complies with the specified requirements.

2.2 Acronyms and Abbreviations

BTP	Branch Technical Position
CoC	Certificate of Conformance
CI	Configuration Item
CM	Configuration Management
CRC	Cyclic Redundancy Check
D3	Defense-in-Depth and Diversity
FAT	Factory Acceptance Test
FDE	Function Diagram Editor
FDG	Function Diagram Group
FRS	Functional Requirements Specifications
GSM	Graphic Service Monitor
HICB	Human Instrumentation and Control Branch
IV&V	Independent Verification and Validation
I/O	Input Output
I&C	Instrumentation and Control
QA	Quality Assurance
RPS	Reactor Protection System
RTD	Resistive Temperature Detector
RTE	Run Time Environment
RTM	Software Requirements Traceability Matrix
SAT	Site Acceptance Test
SCMP	Software Configuration Management Plan
SDD	Software Design Description

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 16 of 41

SER Safety Evaluation Report
SIVAT Simulation Based Validation Tool
SPACE Specification and Coding Environment
SRS Software Requirements Specification
SSP Software Safety Plan
SQAP Software Quality Assurance Plan
Std Standard
SVVP Software V&V Plan
TXS TELEPERM XS
UF University of Florida
UFTR University of Florida Training Reactor

2.3 References

2.3.1 UF Documents

- /1/ UFTR-QAP, "Quality Assurance Program (QAP)"
- /2/ UFTR-QAP-01-P, "Conduct of Quality Assurance"
- /3/ UFTR-QA1-QAPP, "Quality Assurance Project Plan (QAPP)"
- /4/ UFTR-QA1-01, "Software Quality Assurance Plan (SQAP)"
- /5/ UFTR-QA1-02, "Software Configuration Management Plan (SCMP)"
- /6/ UFTR-QA1-03, "Software Verification and Validation Plan (SVVP)"
- /7/ UFTR-QA1-06.1, "Software Test Plan – SIVAT Plan"
- /8/ UFTR-QA1-06.2, "Factory Acceptance Test Plan – FAT Plan"
- /9/ UFTR-QA1-07, "Software Installation Plan"
- /10/ UFTR-QA1-10, "Software Training Plan"
- /11/ UFTR-QA1-100, "Functional Requirements Specifications (FRS)"
- /12/ UFTR-QA1-103, "Diversity and Defense-in-Depth (D3) Analysis"
- /13/ UFTR-QA1-105, "Cyber Security"
- /14/ UFTR-QA1-108, "Software Requirements Traceability Matrix (RTM)"
- /15/ UFTR-QA1-109, "Software Library and Control"
- /16/ University of Florida Training Reactor SAR, 2009
- /17/ UF SSAR HEU-LEU, "SUBMITTAL REPORT to Cover Analyses of University of Florida Training Reactor (UFTR) Conversion from HEU to LEU Fuel," Second Version, August 2009

2.3.2 AREVA NP Inc. Documents

- /18/ AREVA NP Inc. Document No., 38-1288541-00, Topical Report EMF-2110(NP) (A) Revision 1, "TELEPERM XS: A Digital Reactor Protection System"

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 17 of 41</i>

- /19/ AREVA NP Inc. Document No., 01-5044046-01, "TELEPERM XS SIVAT-TXS Simulation Based Validation Tool (Version 1.5.0 and higher) User Manual TXS-1047-76-V2.1"
- /20/ AREVA NP Inc. Document No., 01-1007858-00, "TELEPERM XS Engineering System SPACE (for TXS Software Release 3.0.0 or Higher for LINUX) Overview TXS-1026-76-V3.0/03.03"

2.3.3 Industry Standards

- /21/ IEEE Standard 7-4.3.2-1993, "Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations"
- /22/ IEEE Std 603-1998, "Standard Criteria for Safety Systems for Nuclear Power Generating Stations"
- /23/ IEEE Std 610.12-1990, "Standard Glossary of Software Engineering Terminology"
- /24/ IEEE Std 829-2008, "Standard for Software Test Documentation"
- /25/ IEEE Std 830-1998, "Recommended Practice for Software Requirements Specification"
- /26/ IEEE Std 1008-1987, "Standard for Software Unit Testing"
- /27/ IEEE Std 1012-1998, "Standard for Software Verification and Validation"
- /28/ IEEE Std 1016-1998, "IEEE Recommended Practice for Software Design Descriptions"
- /29/ IEEE Std 1228-1994, "Standard for Software Safety Plans"

2.3.4 NRC Documents

- /30/ ANSI/ANS 10.4-1987, "Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry"
- /31/ ANSI/ANS-15.8-1995; R2005 (R=Reaffirmed), "Quality Assurance Program Requirements for Research Reactors"
- /32/ ANSI/ANS 15.15-1978, "Criteria for the Reactor Safety Systems of Research Reactors"
- /33/ ANS-15.20, "Criteria for the Reactor Control and Safety Systems of Research Reactors"
- /34/ 10 CFR Part 50 Appendix B, "Quality Assurance Criteria for Nuclear Power Plants and Fuel Reprocessing Plants"
- /35/ 10 CFR Part 100, "Reactor Site Criteria"
- /36/ RG 1.152, Revision 1, "Criteria for Digital Computers in Safety Systems of Nuclear Power Plants"
- /37/ NUREG-0800, "Standard Review Plan", Chapter 7, Branch Technical Position HICB-14, "Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems", Revision 4,

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 18 of 41

June, 1997, Section 3.1.i, "Software Safety Plan" and Section 3.2.a, "Safety Analysis Activities"

- /38/ NUREG-0800, "Standard Review Plan", Chapter 7, Branch Technical Position HICB-19, "Guidance for Evaluation of Defense-in-Depth and Diversity in Digital Computer-Based Instrumentation and Control Systems", Revision 4, June, 1997
- /39/ AREVA NP Inc. Document No., 38-9033245-000, Safety Evaluation by the Office of Nuclear Reactor Regulation Siemens Power Corporation Topical Report EMF-2110(NP), "TELEPERM XS: A Digital Reactor Protection System, Project No. 702," May 5, 2000
- /40/ NUREG 1537 Part 2, February 1996, "Guidelines for Preparing and Reviewing Applications for the Licensing of Non-Power Reactors"
- /41/ Regulatory Guide 1.172, Rev. 0, September 1997, "Software Requirements Specifications for Digital Computer Software Used In Safety Systems of Nuclear Power Plants"

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 19 of 41</i>

3. Software safety management

3.1 Organization and responsibilities

3.1.1 Project Organization

The software safety management activities are executed by the following organizations:

- 1) The Software Development Lead administratively and for Technical Leadership, reports to the UF Project Management (PM).
- 2) An Independent Verification and Validation (IV&V) team. This team also reports to the PM.
- 3) The UFTR Quality Assurance Project Auditor provides surveillance and audits of the software development activities per the UFTR “Quality Assurance Program (QAP), /1/. The Project Auditor reports to the PM.

3.1.2 Responsibilities

3.1.2.1 UF Project Management

The UF-PM has the following responsibilities in the conduct of the SSP:

- Coordinate software safety tasks within the overall context of the system safety program;
- Coordinate safety task planning with other organizational components or functions, such as development, system safety, software quality assurance, software reliability, software configuration management, V&V, and software testing;
- Obtain, allocate, and monitor resources for effective implementation of the SSP;
- Participate in audits of the SSP implementation;
- Coordinate technical issues related to software safety with the UFTR Software Development Lead;
- Ensure training in methods, tools, and techniques used in software safety tasks for the project and V&V personnel, including documentation per the UFTR QAP, /1/;
- Communicate any safety concerns in accordance with the UFTR QAP, /1/.

3.1.2.2 Software Development Lead

The Software Development Lead is responsible for providing the administrative and technical leadership for the software safety activities to include:

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 20 of 41

- Responsibility for the development of the SSP (this document) and any needed revisions;
- Responsibility for the overall conduct of software safety activities;
- Technical direction to members of the Software Development Group for software safety activities;
- Providing the Software Development Group project specific training in accordance with the UFTR “Software Training Plan,” /10/, and ensuring that the training is documented per the UFTR QAP, /1/;
- Communicating any safety concerns in accordance with the UFTR QAP, /1/.

3.1.2.3 Software Development Group

The Software Development Group is responsible for:

- Developing and implementing the safety-related Application Software per the software development methodology and the associated safety analyses per Section 4 of this Plan;
- Initiating SSP change requests via an Open Item Form;
- Testing any safety critical features;
- Communicating any safety concerns in accordance with the UFTR QAP, /1/.

3.1.2.4 IV&V Group

The IV&V team is responsible for:

- Ensuring independence between the IV&V team and the design organization;
- Responsibility for the overall conduct of V&V software safety activities;
- Providing the IV&V team project specific training in accordance with the UFTR “Software Training Plan”, /10/, and ensuring that the training is documented per the UFTR QAP, /1/;
- Preparing and maintaining the RTM during the course of the project;
- Communicating any safety concerns in accordance with the UFTR QAP, /1/;
- Performing all additional V&V activities as defined in the UFTR Software Verification & Validation Plan (SVVP), /6/.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 21 of 41</i>

3.1.2.5 Quality Assurance Project Auditor

The Quality Assurance Project Auditor is responsible for performing surveillances, audits, and other activities in accordance with the UFTR QAP, /1/.

3.2 Resources

The resources used in this SSP include engineering personnel, the SPACE engineering tool, and the SIVAT test tool. The PM, the Software Design Team, and the IV&V team determine the personnel resources required to implement the SSP activities. The project schedule shall be used to monitor and allocate resources.

3.3 Staff qualification and training

Employee Training is conducted and documented in accordance with the UFTR QAP, /1/. All employees in the UFTR project working on safety-related projects are trained in accordance with the UFTR “Software Training Plan”, /10/. Ongoing training is performed to ensure that all personnel are qualified to perform the project work in a technically proficient and quality manner.

This training qualifies UFTR employees to perform the following activities:

- a) Software engineering – use of SPACE tools
- b) Use of SIVAT tool
- c) Use of TXS documentation
- d) Define safety requirements
- e) Design and implement safety-critical portions of the system
- f) Perform software safety analysis tasks
- g) Test safety-critical features
- h) Audit SSP implementation
- i) Perform process certification

It is the responsibility of the PM to assess the skill levels and assign qualified individuals accordingly. Ongoing training requirements are fulfilled by additional training which is set forth by the PM and the Software Development Team. Work group specific requirements shall be specified in the UFTR, “Software Training Plan,” /10/. All project personnel assigned to work on any activity in the Software Life Cycle process shall complete training on the SSP.

The Software Design Team shall maintain personnel training records in accordance with the UFTR QAP, /1/, for all project personnel following any revision to the SSP. The IV&V team shall be trained in accordance with the UFTR SVVP, /6/, in addition to the SSP.

3.4 Software life cycle

The Software Life Cycle phases are listed below and the software safety activities are linked to the appropriate phases:

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 22 of 41

- Basic Design Phase includes the Defense-in-Depth and Diversity (D3) analysis, and the initiation of the Application Software RTM;
- Detailed Design Phase includes continuation of the RTM and the SIVAT Testing;
- Testing Phase includes the FAT and the continuation of the RTM;
- Installation Phase is the installation of the system at the UFTR. Software safety activities during this phase include analysis of any software changes made after the FAT and incorporation of any changes into the analyses listed in Section 4.1, Software Safety Analyses Preparation. In addition, a test report of any regression testing that is required for validating the changes is generated;
- Commissioning Phase and Final Documentation Phase also include updating all documentation to incorporate any changes generated during the Post Installation Testing.

The Project Configuration Management methodology is defined in the UFTR Software Configuration Management Plan (SCMP), /5/, and applies to all phases of the project.

3.5 Documentation requirements

The following sections specify the documents to be prepared and their contents.

3.5.1 Software Project Management

Software Project Management Documentation Requirements are incorporated in accordance with the UFTR "Quality Assurance Project Plan (QAPP)," /3/.

3.5.2 Software Configuration Management

See the UFTR SCMP, /5/.

3.5.3 Software Quality Assurance

See the UFTR "Software Quality Assurance Plan (SQAP)," /4/.

3.5.4 Software Safety Requirements

The Software Safety Requirements shall be incorporated in the Software Requirements Specifications (SRS) document. The SRS shall be written by the Software Development Group and shall satisfy the requirements of IEEE Std. 830-1998, /25/, as endorsed by RG 1.172, /41/. The Software Development Group shall assure that all of the specification requirements, functional requirements and software requirements are incorporated into the software design. The Software Development Group shall trace the requirements from the FRS into the SRS in accordance with the UFTR RTM, /14/.

3.5.5 Software Safety Design

The Software Safety design features shall be described in the Software Design Description (SDD) document. The TXS Application Software SDD shall be

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 23 of 41</i>

written by the Software Development Group and shall meet the intent of IEEE Std. 1016-1998, /28/. The SDD is a written representation of the TXS RPS application software, utilizing the FunBase tool, created to facilitate analysis, planning, and implementation in SPACE.

The TXS Application Software SDD is comprised of various views of the application software, including an overview of the system architecture, and a top-level presentation of the important system functions. The SDD includes a library of all the standard SPACE design entities used in the software design. The SDD lists the important entities in the design, including the functional logic modules and their input and output modules. Other views include database tables listing the changeable parameters and information signals, together with some of their attributes, and communication interfaces.

The Software Development Group shall trace the requirements from the SRS in accordance with the UFTR RTM, /14/.

3.5.6 Software Development Methodology

Safety-related software for the TXS Application Software shall be coded using the SPACE tool. The design for implementation of the requirements from the SRS shall be translated into the SDD. The information in the SDD shall be coded into the SPACE database using the SPACE Function Diagram Editor (FDE) tool. The code shall then be automatically generated by the Function Diagram Group (FDG) and Run Time Environment (RTE) Code generators from the SPACE database.

The code contained in the SPACE database shall then be tested using the SIVAT tool. This methodology shall be used for creating the safety-related Application Software and checking for deficiencies in the software that could prevent it from fulfilling its safety function. Errors shown by the SIVAT tool shall be corrected in the SPACE diagrams and retested until the results are shown to be satisfactory.

The SIVAT test planning, test procedures and test reports are prepared according to the UFTR “Software Test Plan – SIVAT Plan,” /7/, which uses the guidance of IEEE Std 829-2008, /24/, and IEEE Std 1008-1987, /26/.

3.5.7 Test Documentation

The FAT planning, procedures, and reports are prepared according to the UFTR, “Factory Acceptance Test Plan – FAT Plan,” /8/, which uses the guidance of IEEE Std 829-2008, /24/.

3.5.8 Software Verification and Validation Plan

Information regarding how software safety will be verified and validated is outlined in the UFTR SVVP, /6/, which uses the guidance of IEEE Std 1012-1998, /27/, (Refer to Section 3.9).

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 24 of 41</i>

3.5.9 Reporting Safety Verification and Validation

Information documenting the results of the safety V&V activities performed by the IV&V team shall be recorded and reported in the IV&V Report generated at the end of each phase. The IV&V activities are defined in the UFTR SVVP, /6/, which uses the guidance of IEEE Std 1012-1998, /27/, (Refer to Section 3.9 for additional information).

3.5.10 Software User Documentation

Information that may be significant to the safe installation, use, and maintenance of the software shall be prepared in accordance with the UFTR QAP, /1/, and the UFTR SQAP, /4/.

3.5.11 Results of Software Safety Requirements Analysis

Refer to Section 4.2.

3.5.12 Results of Software Safety Design Analysis

Refer to Section 4.3.

3.5.13 Results of Software Safety Code Analysis

Refer to Section 4.4.

3.5.14 Results of Software Safety Test Analysis

Refer to Section 4.5.

3.5.15 Results of Software Safety Change Analysis

Refer to Section 4.6.

3.6 Software safety program records

3.6.1 Scope of Safety Program Records

The software safety program records shall not be separate from the other records generated by the project. Neither are they handled differently than any other quality records on the project. Per IEEE Std 1228-1994, /29/, software safety program records shall include:

- 1) Results of analyses, including IV&V, performed on requirements, design code, test, and other technical documentation. Specifically, the analyses/reports include the D3 Report, the RTM, the IV&V reports, the Test Summary Report (SIVAT), and the FAT Summary Report.
- 2) Information on suspected or confirmed safety problems in the prerelease or installed system. This information is included in the analyses/reports listed above.
- 3) Results of audits performed on software safety program tasks. Audit reports are separate from the design reports, but are handled in the same

<i>UF/NRE UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 25 of 41</i>

way, i.e. the audit reports are not exclusively on safety program items, but on the items included in the UFTR SQAP, /4/

- 4) Results of safety tests conducted on all or any part of the entire system. The results of SIVAT Testing are included in the Test Summary Report. The results of the FAT are included in the FAT Summary Report.
- 5) A record of training provided to software safety program personnel. These records are maintained in accordance with the UFTR QAP, /1/.
- 6) Results of any certifications performed. For the Application Software generated at the UFTR, no certifications shall be given, apart from the Certificate of Conformance (CoC) for the entire system that it meets the UFTR specification (see Section 3.13). AREVA NP GmbH provides the CoC's for software tools.

The records shall be entered into the UFTR document control system for maintenance and storage and shall be sufficient to certify that the processes and tasks specified in this Plan have been carried out satisfactorily.

3.6.2 Responsibility for Software Safety Program Records

It is the responsibility of the project personnel who generate the records to process the software safety program records into the UFTR document control system except for audits performed by Quality Assurance. Quality Assurance shall process the documentation from the audits.

3.6.3 Maintenance of Software Safety Program Records

The UFTR document control procedures for safety-related documents apply for all UFTR RPS project documentation produced. All project documentation, including the documentation of the safety aspects of the software, shall be entered as records into the UFTR document control system for maintenance and storage. These records are sufficient to certify that the processes and tasks specified in this Plan have been carried out satisfactorily.

3.6.4 Open Item Tracking System

A potential discrepancy, a potential improvement, or a possible anomaly from the required status or condition discovered during the phases of the UFTR project is processed according to the "Procedure for Conducting the QA," /2/.

3.7 Software configuration management activities

The Application Software Configuration Management activities are defined, implemented, and managed in accordance with the UFTR SCMP, /5/. The SCMP provides the method and tools to identify and control the UFTR Application Software developed for the UFTR project.

UF/NRE UFTR	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 26 of 41</i>

Configuration control activities request, evaluate, approve or disapprove, and implement changes to the Application Software. Changes encompass both error correction and enhancement.

The project work breakdown structure addresses the production, review, approval, and control of the Application Software. Schedule reporting tracks the completion of the Application Software throughout the project including additional activities documented to make changes.

3.8 Software quality assurance activities

The SQAP, /4/, provides the measures to ensure the developed software conforms to the project technical requirements, rules, and standards.

The SQAP, /4/, describes the tools to be used and methodology to be followed in developing and maintaining software to be used for the design of TXS Application Software.

3.9 Software verification and validation (SV&V) activities

The Application Software V&V activities are defined and implemented in accordance with the project-specific SVVP, /6/.

The UFTR SVVP, /6/, specifies the V&V activities to be performed during software planning, development, and implementation that will demonstrate high levels of quality and confidence in the software being developed. The V&V activities provide traceable documented evidence that a high level of quality and a low level of risk have been achieved. The SVVP provides the methods and tools to determine whether the Software Configuration Items of the UFTR RPS Project conform to the project requirements for the TXS Application Software.

3.10 Tool support and approval

The TXS software tools are as follows: SPACE (including FDE, automatic code generators, *scanmic*, *reflist*, *cpupload*, *netload*, GSM, and other tools which make up the applicable version of SPACE), Qt Designer, and SIVAT. SPACE is the core software tool which is used to create and generate code for the Application Software, Qt Designer is used to create the project-specific screens used in the GSM, and SIVAT is the tool used to test the Application Software in a simulated environment.

The UFTR software engineers are the users of the tools described above and shall be trained in their use. Use of the TXS software tools is approved and controlled by the UFTR SCMP, /5/.

3.10.1 Tool Approval

The TXS software tool SPACE (Engineering System SPACE, /18/) has been reviewed and accepted by the NRC as stated in the NRC SER, /39/, on the TXS Topical Report, /18/, and found to be an acceptable platform for the development of I&C applications. All tools used for the development and testing of the UFTR

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 27 of 41</i>

software are qualified for use by AREVA NP GmbH. The Application Software generated by the TXS engineering tool SPACE uses qualified software modules from a function block library.

Configuration of the tools is controlled by AREVA NP GmbH. These tools are purchased as certified quality-related products. No process for selecting configuration management tools is specified since the configuration management tools for the TXS technology have already been selected.

The PM shall ensure the current approved version of each tool is used for the project. The PM's approval is documented through the release of the Software Configuration Items List at each phase.

Project tools, such as FunBase, do not require V&V or testing to qualify their use because the end product is extensively reviewed and the tools are not used in the online operation of the system.

3.10.2 Installation of Upgrades to Approved Tools

Upgrades to approved tools are controlled by the UFTR SCMP, /5/.

3.10.3 Withdrawal of Previously Approved Tools

Withdrawal of an approved tool is controlled by the Software Design Team.

3.10.4 Identification of Limitations

The inadvertent introduction of software hazards by project tools is mitigated by the proper use of the tools and the proper use of techniques for the software configuration management, software quality assurance, and V&V as described in the respective plans.

3.11 Previously developed or purchased software

The TXS system platform software development is performed by AREVA NP GmbH on a project-independent (generic) basis and is controlled by the AREVA NP GmbH Software Quality Assurance (SQA) program. The process for approval for use of this software is covered by the UFTR SCMP, /5/. The TXS system platform software is reviewed and accepted by the NRC as stated in the NRC SER, /39/, on the TXS Topical Report, /18/.

The online software, code generators, and function block library portions of the TXS system platform software shall be purchased as a qualified safety-related product. Additional tools such as GSM, *hwparams*, Gateway (GW), and Qualified Display System (QDS) shall be purchased as qualified non-safety-related products.

3.12 Subcontract management

The UFTR RPS project does not use software developed by a subcontractor. Instead, the TXS system platform software is purchased from AREVA NP GmbH as a qualified safety-related product. The TXS system platform software is reviewed and

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 28 of 41</i>

accepted by the NRC as stated in the NRC SER, /39/, on the TXS Topical Report, /18/. If circumstances require a safety software purchase from a subcontractor, safety software shall be purchased from an approved vendor whose program meets ANSI/ANS-15.8-1995; R2005 (R=Reaffirmed), "Quality Assurance Program Requirements for Research Reactors," /31/, criteria in accordance with the requirements of the UFTR QAP, /1/, and the associated software requirements.

If circumstances require a non-safety software purchase from a subcontractor, non-safety software shall be purchased that meets the requirements of the UFTR QAP, /1/, and the associated software requirements.

3.13 Process certification

Certificates of Conformance (CoCs) are required for certification of safety-critical software products used by or produced for the UFTR project. They are in accordance with the processes specified in this Plan.

The CoCs shall be prepared by a project team member and reviewed and processed in accordance with the UFTR QAP, /1/. The CoCs shall be provided on the system and include the software and hardware.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 29 of 41</i>

4. Software safety analyses

The UFTR approach to software safety analysis is predicated on two (2) important foundational elements:

- 1) A design control process for safety-related work, as required by the ANSI/ANS-15.8-1995; R2005 (R=Reaffirmed), "Quality Assurance Program Requirements for Research Reactors," /31/;
- 2) The use of the NRC-approved TXS object-oriented automated code generation tools for the development of the Application Software.

IEEE Std 1228-1994, /29/, provides guidance regarding SSPs and software safety analysis activities. This Plan provides basis for the software safety analysis activities that are performed for the UFTR RPS application software. The application software runs on the qualified TXS platform. The following safety analysis activities shall be utilized to meet the objectives of this SSP:

- 1) D3 Analysis
- 2) Application Software RTM
- 3) Verification and Validation (V&V)
- 4) Automatic Code Generation
- 5) SIVAT Testing
- 6) Factory Acceptance Testing (F²AT)
- 7) Cyber Security

The safety analysis activities are concentrated on the application software rather than the TXS platform operating software on the basis that the TXS platform has been fully qualified as an integrated platform. The TXS platform was designed to provide a stable, predictable and reliable system. The TXS platform software (non-application software for system operation and communication) has been designed using principles that ensure determinism or principles by which the order of facts perfectly defines the conditions for existence of a phenomenon such that the phenomenon must occur if the conditions or order of facts are satisfied.

Because the TXS platform was designed to provide a stable, predictable and reliable system and there was a generic qualification process, which included independent validation testing, it is argued that software hazards associated with the generic TXS platform have been reasonably and adequately addressed. In addition, the TXS system uses a comprehensive set of self-monitoring tests to monitor system performance for internal faults, as described in the TXS Topical Report, /18/. Lastly, the TXS operating system has a base of substantial nuclear operating experience to validate performance that has provided opportunities to identify latent errors.

The use of the qualified TXS object-oriented automated code generation tools minimized the inherent risk in the development of the Application Software as well as minimized the potential for human error. The software safety analysis methodology for TXS projects is

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 30 of 41</i>

based on the use of the pre-qualified TXS platform and software engineering tools. The IEEE Std 1228-1994, /29/, necessitates various specific activities that were simplified by use of the TXS platform. The safety analysis activities specified in IEEE Std 1228-1994, /29/, are discussed below.

Section 4.1, “Software safety analysis preparation” of IEEE Std 1228-1994, /29/, addresses issues related to the D3 Analysis. The RPS Replacement Project Software for the UFTR provides safety functions to shutdown the reactor in case of an accident. The reactor trips or safety functions of the replacement RPS software were not individually analyzed; instead, the approach taken was to assume that all the RPS safety functions are necessary and must function in case of an accident. The required safety functions for the application software are defined in the UFTR “Functional Requirements Specifications (FRS),” /11/. Therefore, the software hazardous states are the loss of any safety function (which results in a reactor trip). Some possible causes of a hazard and where the causes are addressed (analysis paragraph) are:

- 1) Incomplete or incorrect incorporation of safety requirements (4.2.1, 4.2.2, 4.5.1, 4.5.2)
- 2) Mistakes in coding/programming (4.1, 4.2.2, 4.4, 4.5.1, 4.5.2)
- 3) Mistakes in timing of safety actions (4.3, 4.5.2)
- 4) Failure of system hardware or software (4.1, 4.3, 4.5.2)
- 5) Failure to address identified problems (4.2.2, 4.5.1, 4.5.2)

Section 4.2, “Software safety requirements analysis” of IEEE Std 1228-1994, /29/, requires a definition of the software safety-related activities that will be carried out as part of the software requirements phase of development. The requirements of this section are addressed by the RTM (4.2.1), the V&V (4.2.2), and the FAT (4.5.2) safety analysis activities.

Section 4.3, “Software safety design analysis” of IEEE Std 1228-1994, /29/, requires a definition of the safety-critical activities that will be carried out as part of the software design phase of software development. The design techniques and practices covering the partitioning of the software into design elements are addressed by the use of Automatic Code Generation (4.4). The evaluation of compliance of the software design with the system safety requirements is addressed by the RTM (4.2.1), by the V&V activities (4.2.2), by the SIVAT Testing (4.5.1), and by the FAT (4.5.2). Formal reviews and inspections of the design software are addressed in the V&V activities (4.2.2).

Section 4.4, “Software safety code analysis” of IEEE Std 1228-1994, /29/, requires a definition of the software safety-related activities that will be carried out as part of the coding phase of software development. The software hazards associated with first-of-a-kind application engineering work were minimized through the use of qualified software development tools, the use of a qualified Function Block library which provides a large experience base for the standard modules, the use of the object-oriented automated code generation tool (SPACE, /20/) which mitigated an important human error source by

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 31 of 41</i>

eliminating conventional software development and code generation, and the use of the SPACE tool to eliminate both errors of translation and the introduction of complexity by engineers trying to optimize application coding. The use of the SPACE tool supported the development of high quality software with a less complex process, which eliminated several of the software hazards associated with manual coding. Therefore the Section 4.4.4 of IEEE Std 1228-1994, /29/, requirement is met by the safety analysis activity of Automatic Code Generation (4.4).

Section 4.5, "Software safety test analysis" of IEEE Std 1228-1994, /29/, requires a definition of the software safety-related analysis activities that will be carried out as part of the software testing phase of software development. This requirement is addressed by the FAT (4.5.2). The safety analysis activity that will show testing coverage for all software safety requirements is the RTM (4.2.1).

Section 4.6, "Software safety change analysis" of IEEE Std 1228-1994, /29/, requires a definition of the software safety-related analysis activities that will be carried out in response to changes made in assumptions, specifications, requirements, design, code, equipment, test plans, environment, user documentation, and training materials. This requirement is addressed in Section 4.6, Software Safety Change Analysis. The high level system design is defined by TXS Topical Report requirements, /18/, which includes system architecture requirements for generic software safety elements and by the System Functional Description, which defines application specific system safety requirements. The TXS Topical Report, /18/, also addresses the basic architecture, software integration, and development process. The interfaces between the software and the rest of the system are analyzed during the actions taken to prepare for FAT.

4.1 Software safety analyses preparation

The high level system design is defined by TXS Topical Report requirements, /18/, that includes system architecture requirements for generic software safety elements and by the System FRS, /11/, which defines application specific system safety requirements. The TXS Topical Report, /18/, also addresses the basic architecture, software integration, and development process. These documents identify software safety-related actions that will be required of the software to prevent the system from exiting the allowed operating region (see SSAR, /17/),

A D3 Analysis (See UFTR "Diversity and Defense-in-Depth (D3) Analysis," /12/) shall be provided in accordance with the NUREG-0800, Standard Review Plan (SRP) Chapter 7, Appendix 7-A, Branch Technical Position (BTP) HICB-19, /38/. This analysis shall determine whether the UFTR can withstand any Design Basis Accident presented in the Safety Analysis Report (SAR), /16/, without exceeding 10 CFR Part 100, /35/, dose limits, in accordance with the requirements of the NRC's BTP HICB-19, /38/.

This analysis is the bounding analysis for software failures as it assumes a total failure of all software on all non-diverse channels of the systems. The D3 analysis is

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 32 of 41</i>

credited here because the results of this analysis impact the succeeding phases of software development.

4.2 Software safety requirements analysis

4.2.1 Application Software Requirements Phase and Requirements Traceability Matrix

During the Software Safety Requirements Phase as defined in IEEE Std. 1228-1994, /29/, the RPS replacement system functional requirements shall be defined and documented. The safety-related software activity and analysis, which is carried out as part of the software requirements phase, is the translation of the safety requirements from the UFTR functional specification to the SRS. Relevant software design constraints and guidelines are addressed. This activity is performed and reviewed by the Software Development Group and the SRS is released by the PM according to the UFTR QAP, /1/. The SRS is prepared for the UFTR RPS Replacement system using the guidance of IEEE Std. 830-1998, /25/.

The Application Software RTM is generated by the IV&V team to track the safety-related requirements (and non safety-related requirements) from the UFTR RPS functional specification to the SRS. The RTM is used to verify compliance with the project specifications. The RTM report is prepared by the IV&V team and released by the IV&V team. The IV&V team performs these required activities in accordance with the UFTR SVVP, /6/.

All identified discrepancies shall be processed as Open Items per the UFTR Procedure for Conducting QA, /2/. Baselines were established for control of the design in accordance with the UFTR SCMP, /5/.

4.2.2 Verification and Validation (V&V)

Information regarding how software safety is verified and validated is documented in the UFTR SVVP, /6/ using the guidance of IEEE Std. 1012- 1998, /27/, (Refer to Section 3.9).

The V&V activities provide an independent process to ensure the verification of an accurate translation during each software development phase and the validation that the software product fulfills the requirements for the specific intended uses for which it is developed. The IV&V activities take place throughout the Project phases to provide a method of proving the design which is independent and distinct from the design efforts. The software life-cycle design activities normally take place over an extended time period (approximately one year) and can result in a number of revisions to design documents. Most IV&V activities may be repeated and in some cases performed and reported after software development activity had progressed to the succeeding phase.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 33 of 41</i>

4.3 Software safety design analysis

Software design analyses are performed during the preparation of the SDD to ensure that each requirement from the SRS is satisfied in the SDD. The preparation process of the SDD identifies the safety-critical software design elements. The review process of the SDD includes the software safety design analysis as part of its criteria. Design elements used in the SDD are predefined in the TXS SPACE tools. Classification, relationship, and design techniques for the design elements have been identified and addressed during the TXS platform development, /18/. An evaluation of compliance of design (i. e., SDD) to system safety requirements is fulfilled during review of the SDD. The RTM is updated to track the requirements from the SRS to the SDD. The IV&V team performs the required activities in accordance with the UFTR SVVP, /6/.

4.4 Software safety code analysis - Automatic Code Generation

A SDD shall be prepared that implements the requirements of the SRS. The Application Software code is generated from the SDD by utilizing the SPACE Function Diagram Editor tool and the automatic code generators. The result of this coding is then run through pre-qualified, automatic code generators that were fully qualified and purchased as safety-related. This activity reduces the likelihood of a manual coding error that may pose a risk that is adverse to safety.

The IV&V team performs independent review of the translation of SRS requirements into the SDD and the translation of the SDD into the Code document in accordance with the UFTR SVVP, /6/. All identified discrepancies are processed as Open Items per the UFTR QAP, /1/, and the UFTR Procedure for Conducting QA, /2/. Baselines shall be established for control of the design in accordance with the UFTR SCMP, /5/.

4.5 Software safety test analysis

4.5.1 SIVAT Testing

The SIVAT Testing activity (tool) is used only as an engineering design tool for the UFTR Replacement project. SIVAT testing validates that the correct software modules have been properly used and that the functionality of the application software meets the software requirements and the UFTR functional specifications. SIVAT can simulate various TXS malfunctions to verify that the response to these faults is as intended. The SIVAT Testing allows the Application Software to be fully tested in a simulated environment. The application code contained in the SPACE database is tested using the SIVAT tool during the Detailed Design Phase. This methodology is used for checking for deficiencies in the software that could prevent it from fulfilling its safety function. The SIVAT Test Plan, /7/, is prepared using the guidance of IEEE Std. 829-1983, /24/, and IEEE Std. 1008-1987, /26/.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 34 of 41</i>

4.5.2 Test Phase - Factory Acceptance Testing (FAT)

The IV&V Team is responsible for the FAT activity. The FAT validates that the functionality of the system meets the system requirements in the fully integrated system. Application software integration (with hardware) is performed prior to FAT and is effectively checked during FAT by functional testing. This testing satisfies the integration and functional test requirements of IEEE Std. 1012-1998, /27/, for the UFTR. The testing during the FAT validates that the functionality of the application software meets the software safety requirements.

The Software Safety FAT is performed to ensure that each software requirement is tested and met. Software safety requirements for testing are followed in the Test Plan. The review process identifies that the safety requirements are met in the Test Procedure. Risk associated with the implementation of design will be addressed in the Test Reports. Acceptance Test Reports are planned to document the test results and to provide basis documenting that no new software hazards remain in the tested system.

The FAT Plan, /8/, is prepared using the guidance of IEEE Std. 829-1983, /24/, and the UFTR Project SVVP, /6/. The Application Software RTM, maintained by the IV&V Team, shall be updated to trace safety requirements to the Test procedures. The RTM is used to verify the relationship between the software tests and the software requirements in order to show that the requirements have been addressed by one or more acceptance tests.

Discrepancies identified during FAT are processed as Open Items per the UFTR QAP, /1/, and the UFTR Procedure for Conducting QA, /2/. Baselines shall be established for control of the design in accordance with the UFTR SCMP, /5/.

4.6 Software safety change analysis

Software configuration management and change control shall be applied to all software design documents and code. Control is established through the implementation of the configuration identification, the change control, and status accounting functions in accordance with the UFTR SCMP, /5/, and safety analysis activities described in Sections 4.1 through 4.5 as they are applicable to the scope of the software change. Any changes to software may require repetition of FAT (4.5.2) activity in the form of regression testing.

4.7 TXS Cyber Security

The TXS platform has features to address potential security vulnerabilities in the life cycle phases of the system and software. The Cyber Security requirements and features shall be embedded within the software and hardware requirements specific to the UFTR TXS system and the platform features of the TXS system (see Topical Report, /16/) to ensure software safety is not compromised by Cyber attack.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 35 of 41</i>

4.7.1 TXS Features that Minimize Cyber Security Vulnerabilities

The TXS system has a number of features that minimize or eliminate cyber security vulnerabilities which are described in the TXS Topical Report, /18/. The TXS application software is reactor-specific and developed with the NRC-approved tools, /31/. The TXS operating system software and function block library are reactor-independent, pre-developed, and NRC-approved software qualified for safety applications. The TXS operating system, function block library, and application software is custom and proprietary software. The use of the TXS system software is limited to nuclear applications. Consequently, the software does not have the same cyber security vulnerabilities as open-access industrial software or widely distributed public software.

The TXS system does not use Transmission Control Protocol/Internet Protocol (TCP/IP) software or message routing services, which are software features that are commonly exploited in cyber attacks. As such, the TXS software strategy eliminates important transmission vectors for external cyber attacks (e.g., control system attacks, malevolent code infection, and denial of service attacks).

The TXS system is designed to execute program code from Flash Erasable Programmable Read Only Memory (FEPRM). The program code on the FEPRM can only be changed from the TXS Service Unit (SU). The TXS system is designed without the use of external data storage media. Consequently, the TXS system has no memory storage media where malevolent code can hide or replicate. These design features eliminate the vulnerability to cyber attacks caused by malevolent code. The integrity of the code installed on the FEPRM is routinely checked for integrity as part of the cyclic test features. This design feature ensures that any damaged code is promptly identified and isolated.

4.7.2 TXS Monitoring and Service Interface (MSI)

The standard TXS system is designed to be installed within a physically secure area. As described in the TXS Topical Report, /18/, the MSI is designed as a qualified data transmission barrier between the UFTR RPS safety function computers on one side and the TXS SU, TXS QDS and the TXS GW on the other side. The MSI is designed with the same TXS operating principles as the UFTR RPS computers.

The TXS GW is designed to export the TXS system information to outside non-safety systems. The standard TXS GW feature of sending routine signaling messages and a limited set of commands back to the MSI is not used in the design of the UFTR system. The TXS GW is isolated from the safety functions via the MSI computer as described in the TXS Topical Report, /18/.

The TXS SU is used by authorized personnel to monitor and test the TXS system, to diagnose system alarms and failures, and to make parameter and software changes. The TXS safety function computers are not directly accessed by

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 36 of 41</i>

the TXS SU; instead the TXS RPS computers are only accessed via the the MSI which provides communication independence.

4.7.3 TXS Computer Message Transfer Protocols

Communication between the TXS computers is designed with logical point-to-point connections. Separate logical MicroNET communication channels are provided for each message. The TXS system is designed for strictly cyclic data transfer via network connections, using a predefined package size and a constant bus load, with checksum and message age monitoring. The TXS system is designed to operate with a fixed cycle time. Each TXS computer reads input data and writes output data once per cycle. This design feature eliminates the vulnerability to network data storms or cyber attacks (i.e., denial of service attacks).

The message size is determined individually for each message at the time of code generation by the TXS automated code generation tool. The TXS design uses a standard message frame that consists of a standard header and a fixed size data section. The integrity of all messages received from other TXS computers is checked as part of the cyclic operation.

Service commands sent by TXS SU in service messages are addressed to an individual TXS computer. Service commands from the TXS SU are accepted for execution by a TXS computer only when the message is verified to have the correct ID, the receiving computer is ready for a new service message, the message is new, the service command code is valid, and the service command is valid for execution in the CPU operational mode. Invalid service messages are rejected without further action or response. Errors occurring during the execution of a permitted command are reported by means of an error code in the response to the service message.

These design features, when coupled with the lack of software features that are commonly exploited in cyber attacks, eliminate additional cyber security concerns (e.g., control system attacks and spoofing). TXS GW communication is designed with a static memory structure using linear dual-port memory software interface to transmit data values and status information. The TXS GW computer reads data according to the external interface independently of the TXS side.

The message transfer protocol used in the design of the TXS system ensures that messages sent to the TXS computers are secure, valid, correct, and current. Messages received from the TXS SU are checked twice: once by the MSI and again by the TXS RPS computers. These features eliminate the vulnerability to network data storms or cyber attacks (i.e., denial of service attacks).

4.7.4 Access Control

The standard TXS access control design features provide reasonable assurance that the Cyber Security program objectives of confidentiality, integrity, and availability are met. Multiple layers of access control features are provided for the TXS SU to prevent unauthorized access to the TXS computers. The standard

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 37 of 41</i>

TXS GW design has redundant logical barriers for access to the TXS computers through the use of the MSI and TXS GW computers. The standard TXS GW access controls can be augmented in the UFTR design by the use of an additional hardware feature for one-way access control, which provides an additional layer of access control from the TXS GW. MSI allows the TXS QDS to send analog signals to the TXS RPS computers and to receive analog and binary signals from the TXS RPS computers.

4.7.5 Remote Access Control

The UFTR shall not have remote access capability.

4.7.6 TXS Software Development Process Controls

The TXS software consists of the TXS system code (produced by AREVA NP GmbH) and the TXS application code (produced for the UFTR). All the TXS codes are fully documented. No third party safety software is used in the TXS system. Software Configuration Management and change control is applied to all documents and codes. Control is established through the implementation of the configuration identification, the change control, and status accounting functions in accordance with the UFTR SCMP, /5/.

4.7.7 Process Controls for Software Developed and Maintained by AREVA NP GmbH

The standard TXS system is described in the TXS Topical Report, /18/, which describes both the design features and development process for the TXS hardware platform and operating system. The cyber security controls for the development of the TXS operating system software and function block library software are discussed in the TXS Topical Report, /18/.

4.7.8 Process Controls for Software Developed and Maintained by UFTR

The development process and control for the TXS application software for U.S. projects are described in the TXS Topical Report, /18/, and shall be augmented by the UFTR TXS Cyber-Security, /13/, features and Software Library and Control, /15/, features. The procedure controls for the TXS application software for U.S. projects shall include the following requirements that provide a secure software development infrastructure:

- Engineering servers are placed in a secure area with access only given to the Software Supervisor or a delegate;
- Engineering servers are equipped with login/password controls to only allow access to the Software Supervisor or delegate;
- All computers are equipped with login/password protection;

UF/NRE UFTR	Prepared by		Reviewed by		QA-1, UFTR-QA1-05	
	Name:		Name:		Revision 0	Copy 1
	Date :	Initials:	Date :	Initials:	Vol. 1	Page 38 of 41

- Virus protection and firewalls are installed on engineering servers if paths other than through the corporate virus protected/fire-walled network exist;
- All personnel must be trained on Export Control requirements and sign the corresponding agreement (see UFTR QAPP, /3/);
- All TXS software items (including software received from AREVA NP GmbH) shall be controlled in a Software Library, where virus scanning is performed each time software is returned.

When software is installed, Cyclic Redundancy Check (CRC) checks are used to ensure the software matches exactly the code entered into the software. The functionality of all application code and corresponding security measures are tested during FAT.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 39 of 41</i>

5. Post development

5.1 Training

Training requirements for system commissioning, transition, operation and use, and traceability from training requirements to training documents are provided by the UFTR (See UFTR “UFTR Software Training Plan,” /10/).

5.2 Deployment

At the completion of Site Acceptance Test (SAT), the tracking and approval of changes shall be processed per the UFTR SQAP, /4/, and SCMP, /5/.

5.2.1 Installation of TXS

Requirements for the installation of the new UFTR TXS system software are defined in the UFTR “Software Installation Plan,” /9/. The generation and retention of the appropriate records of the installation shall be performed under the UFTR QAP, /1/.

SCM during installation shall remain under UFTR control per the UFTR SCMP, /5/. Any changes made to the software during the installation process shall be reviewed by UFTR.

5.2.2 Startup and transition – Commissioning

Commissioning of the new UFTR RPS shall be per the UFTR approved procedures. The work, including the generation and retention of the appropriate records of the installation shall be performed under the UFTR QAP, /1/. The requirements for commissioning of the new UFTR RPS system will be developed by UFTR to meet all site specific requirements.

The procedures developed by UFTR must address the following elements from IEEE Std 1228-1994, /29/, that are applicable to the UFTR project:

- Startup of the new system;
- Validation of results from the new system.

SCM during startup shall remain under UFTR control per the UFTR SCMP, /5/. Any changes to the software during post modification testing and the unit startup shall require UFTR review and approval prior to implementing the change as required by the UFTR SQAP, /4/.

5.2.3 Operations support

Documentation of the hardware and software system including design documentation, user manuals, and maintenance recommendations shall be in a controlled document. It is the responsibility of the UFTR project team and management to ensure that this information is provided in a timely manner to support UFTR operations.

<i>UF/NRE UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 40 of 41</i>

5.3 Monitoring

Monitoring of the operation of safety-critical software and the documenting and reporting of safety concerns after final acceptance shall be in accordance with UFTR Policies and Procedures.

5.4 Maintenance

Software Configuration Management following commissioning of the system shall be per the UFTR SCMP, /5/. Any changes made to the UFTR software following commissioning shall be considered a Design Change and processed per the requirements of UFTR design control procedures.

5.5 Retirement and notification

Any future retirement activities shall be in accordance with UFTR policies and procedures. Engineering support from AREVA NP Inc shall be provided in accordance with the RPS digital control system upgrade contract.

<i>UF/NRE</i> <i>UFTR</i>	<i>Prepared by</i>		<i>Reviewed by</i>		<i>QA-1, UFTR-QA1-05</i>	
	<i>Name:</i>		<i>Name:</i>		<i>Revision 0</i>	<i>Copy 1</i>
	<i>Date :</i>	<i>Initials:</i>	<i>Date :</i>	<i>Initials:</i>	<i>Vol. 1</i>	<i>Page 41 of 41</i>

6. Plan approval

6.1 Responsibilities

The UFTR Software Development Lead is responsible for monitoring the SSP to ensure it meets all the codes and standards required by the UFTR Technical Specifications and SAR.

Each UFTR project team member is responsible for ensuring compliance with the SSP.

6.2 Updates

Updates to the SSP will be made as necessary. Minor or editorial changes identified during a given design phase may be held for issue until the end of that phase.

6.3 Change Approval

Any changes to this Plan must be via a revision and reviewed and approved in accordance with UFTR QAP, /1/. This Plan shall be reviewed periodically throughout the UFTR upgrade project.

6.4 Change Distribution

Following approval of any change to the SSP, each member of Software Development Group and others as identified by the Software Development Lead shall be trained via the electronic Personnel Training Report.