U.S.NRC
United States Nuclear Regulatory Commission
Protecting People and the Environment

Software Quality Assurance Plan (SQAP)

Development of RASCAL 4.3.x

Revision 1

Issue Date: September 2016

Prepared by:
J.J. Tomon[1]
J.A. Kowalczik[1]
G.F. Athey[2]

[1]Office of Nuclear Regulatory Research &
Office of Nuclear Security and Incident Response
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555-0001

[2]Athey Consulting
P.O. Box 178
Charles Town, WV 25414-0178

Concurrence and Approvals:

John J. Tomon — Digitally signed by John J. Tomon DN: cn=John J. Tomon, o=U.S. Nuclear Regulatory Commission, ou=Office of Nuclear Regulatory Research, email=John.Tomon@nrc.gov, c=US Date: 2016.09.13 10:41:17 -04'00'	September 13, 2016
RES Technical Monitor/Contracting Officer Representative	Date
Jeffrey Kowalczik — Digitally signed by Jeffrey Kowalczik DN: c=US, o=U.S. Government, ou=U.S. Nuclear Regulatory Commission, ou=NRC-SW, cn=Jeffrey Kowalczik, 0.9.2342.19200300.100.1.1=200003358 Date: 2016.09.14 07:40:00 -04'00'	September 13, 2016
NSIR Technical Monitor	Date
George F. Athey — Digitally signed by George F. Athey Date: 2016.09.14 09:35:48 -04'00'	September 13, 2016
Lead Software Developer (Contractor)	Date

TABLE OF CONTENTS

LIST OF TABLES

ABBREVIATIONS

AC	Athey Consulting
BWR	boiling-water reactor
CM	Configuration Management
COR	Contracting Officer's Representative
NRC	Nuclear Regulatory Commission
NSIR	Office of Nuclear Security and Incident Response
PNNL	Pacific Northwest National Laboratory
PWR	pressurized-water reactor
RAMP	Radiation Protection Computer Code Analysis and Maintenance Program
RASCAL	Radiological Assessment System for Consequence Analysis
RDT	RASCAL Development Team
RES	Office of Nuclear Regulatory Research
SNL	Sandia National Laboratory
SQA	Software Quality Assurance
SQAP	Software Quality Assurance Plan
TM	Technical Monitor
UI	user interface

1.0 PURPOSE

The purpose of this Software Quality Assurance Plan (SQAP) is to define the software quality assurance (SQA) activities that will be followed during the course of software development and deployment of minor, incremental releases to the Radiological Assessment System for Consequence AnaLysis (RASCAL) 4.3.x. This plan defines the quality assurance activities and identifies the documentation that will be created and maintained during the RASCAL 4.3.x software development process. The goal of the plan is to provide adequate confidence that the software development process is controlled, and that the software products will meet established requirements.

1.1 Scope

RASCAL has been under continual development by the U.S. Nuclear Regulatory Commission (NRC) for over 25 years, with RASCAL 4.3 being released on September 29, 2013. Legacy versions of the RASCAL code, including RASCAL 4.3, implemented various elements and levels of SQA, but never in a formalized SQAP.

The scope of this SQAP is to document changes being made to RASCAL 4.3 in producing RASCAL 4.3.x. RASCAL 4.3.x is an incremental, minor update that addresses problems or issues discovered subsequent to the release of RASCAL 4.3. RASCAL 4.3 serves as the legacy, baseline version from which minor software updates are developed. The software update RASCAL 4.3.1 was released in December 2014 and the software update RASCAL 4.3.2 was released in July 2016.

This SQAP captures the SQA activities on going from RASCAL 4.3 to 4.3.x; it does not document or include a detailed review of the RASCAL 4.3 legacy code that remains within RASCAL 4.3.x. Consequently, the various SQA activities documented in this SQAP are discussed at a level that is commensurate to minor software updates.

2.0 REFERENCE DOCUMENTS

Table 1 provides a complete list of documents used or referenced in the development of this SQAP. Included in this list are guidance documents that inform this plan as well as other documents directly referenced by this plan. The documents in Table 1 are grouped by "Document Type" to provide an indication of their relevance to this SQAP.

Table 1 Reference Documents Utilized in the Development of this SQAP

Reference Document	Document Type
U.S. Nuclear Regulatory Commission. 1993. Software Quality Assurance Program and Guidelines. NUREG/BR-0167 (ML012750471).	General SQA
U.S. Nuclear Regulatory Commission. 2012. Software Quality Assurance for RES-sponsored Codes, RES Office Instruction PRM-12, March 5, 2012, (ML12132A176).	General SQA
U.S. Nuclear Regulatory Commission. 2014. "RES and NSIR Coordination Process for Releasing RASCAL Updates."	RASCAL SQA
Athey. G.F. et al. 2015. RASCAL 4.3 User's Guide (Draft). Available from https://www.usnrc-ramp.com.	RASCAL-Reference
McGuire, S.A. et al. 2007. RASCAL 3.0.5: Description of Models and Methods. NUREG-1887. U.S. Nuclear Regulatory Commission. Washington, D.C.	RASCAL-Reference
Ramsdell, J.V., Jr., et al. 2012. RASCAL 4: Descriptions of Models and Methods. NUREG-1940, U.S. Nuclear Regulatory Commission. Washington, D.C.	RASCAL-Reference
Ramsdell, J.V., Jr., et al. 2015. RASCAL 4.3: Descriptions of Models and Methods. NUREG-1940, Supplement 1, U.S. Nuclear Regulatory Commission. Washington, D.C.	RASCAL-Reference
Ramsdell, J.V., Jr., et al. 1983. MESOI Version 2.0: An Interactive Mesoscale Lagrangian Purr Dispersion Model With Deposition and Decay. NUREG/CR-3344, U.S. Nuclear Regulatory Commission, Washington, D.C.	Historical Technical Basis
Ramsdell. J.V. Jr., et al. 1988. The MESORAD Dose Assessment Model. NUREG/CR-4000. Vol.2: Computer Code. U.S. Nuclear Regulatory Commission, Washington, D.C.	Historical Technical Basis
Scherpelz, R. I., et al. 1986. The MESORAD Dose Assessment Model. NUREG/CR-4000. Vol.1: Technical Basis. U.S. Nuclear Regulatory Commission, Washington, D.C.	Historical Technical Basis
Start, G.E. and L.L. Wendell. 1974. Regional Effluent Dispersion Calculations Considering Spatial and Temporal Meteorological Variations. NOAA Tech. Memo. ERL ARL-44. U.S. Department of Commerce.	Historical Technical Basis

3.0 ROLES AND RESPONSIBILITIES

This section identifies the specific organizational element that is responsible for performing each task. The roles and responsibilities for the RASCAL project are listed in Table 2.

Table 2 RASCAL Project Roles and Responsibilities

Role	Responsibility	Responsible Person
Contracting Officer's Representative (COR)	The COR provides final approval and acceptance of each release after all activities have been completed.	John Tomon, NRC (RES)
Technical Monitors (TM)	The TM will provide guidance and approval prior to each implementation release (as applicable), which consists of reviewing and ensuring that all documentation has been completed as required. The SQAP will also provide a new risk determination and new version/scope implementations /changes, as applicable.	John Tomon, NRC (RES) Jeff Kowalczik, NRC (NSIR)
Lead Software Developer	The Lead Software Developer will perform a similar role to the Software Developers; additionally, the Lead Software Developer will be responsible for software configuration management, software backups, and certain records collection, maintenance, and retention.	George Athey, Athey Consulting (AC)
Software Developers	The Software Developers design and implement software code. They resolve problems and verify that all corrections are effective. This includes software updates to the application.	George Athey, AC Jeremy Rishel & Fred Rutz, Pacific Northwest National Laboratory (PNNL) John Fulton, Sandia National Laboratory (SNL)
Software Testers	The Software Tester will create/update all test procedures/test cases and provide to reviewers (as applicable). The software test will execute tests and document test results.	Jeff Kowalczik, NRC (NSIR) John Tomon, NRC (RES) George Athey, AC Jeremy Rishel, PNNL John Fulton, SNL Select NRC Dose Assessment Analysts within the NRC

4.0 SOFTWARE QUALITY ASSURANCE REQUIREMENTS

RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3.

Quality and integrity of the RASCAL 4.3.x software will be ensured by:

- Executing the activities and reviews and preparing the documentation contained within this SQAP.
- Evaluating and planning for the development of software (Section 6.0).
- Identifying and implementing a software development methodology (Section 6.1).
- Identifying software tools and techniques for the project (Section 6.2).
- Establishing and implementing configuration management activities (Section 6.3).
- Implementing software requirement activities (Section 7.0).
- Implementing software design activities (Sections 8.0 and 9.0).
- Establishing and implementing an issue reporting process (Section 11.0).
- Performing software test activities (Section 10.0).
- Performing periodic assessment and reviews of the software and as required by the project (Section 13.0).
- Planning for the software release to the customer (Section 14.0).
- Maintaining this document to incorporate software changes (Section 16.0).
- Maintaining project records related to the software (Section 16.0).

4.1 Documentation

To ensure that the implementation of the software satisfies the requirements, the documentation described in this SQAP is required as a minimum.

It is required to maintain software documentation for project records. Documentation does not necessarily require paper forms of the actual “documents” but rather the ability to locate the evidence of work as needed. This type of documentation could include using electronic records (e.g., RASCAL “Change Logs” in Word or PDF format) or other software/web-based (e.g., SNL RASCAL Collaboration SharePoint) tools.

5.0 SOFTWARE DETERMINATION, DESCRIPTION & HISTORY

This section provides the RASCAL software determination and a description of the program, including its developmental history.

5.1 Software Determination

RASCAL is used by the NSIR as a confirmatory tool for making independent dose and consequence projections during radiological incidents and emergencies. Since RASCAL is used as a confirmatory tool, it is not considered safety software; therefore, the software is considered non-safety, with a low to medium determination. The SQA work activities included in this SQAP are discussed at a level that is commensurate with this level of determination.

5.2 Software Description

RASCAL is a suite of assessment tools for NRC staff use in the confirmatory evaluation of radiological events at U. S. nuclear power facilities. The tools include modules to:

- estimate the release to the atmosphere from reactors, spent fuel pools and casks, and fuel cycle facilities,
- estimate the transport, dispersion, deposition and early phase doses to individuals from releases to the atmosphere,
- acquire and process meteorological data for use by the transport, dispersion, and deposition modules, and
- estimate the early and intermediate phase doses to individuals from surface contamination.

It includes a radionuclide database that contains information for over 800 radionuclides, and a facility database that contains information about commercial power reactors and other commercial nuclear facilities both within the United States and at select international sites. Additionally, it includes two tools to add flexibility to the source term calculations. Finally, RASCAL includes a meteorological data downloader—called MetFetch—to automate the download of observations and forecasts for use within a RASCAL simulation. All of these modules and tools are all available from within the main RASCAL user interface.

5.3 Software History

The initial version of RASCAL (RASCAL v1.3) was published in 1989 and was based on earlier NRC codes that existed at that time (e.g., MESORAD [1986], MESOI [1983]), and ultimately had its roots in earlier DOE codes (e.g., MESODIF, Start and Wendell 1974). More recent versions of RASCAL include RASCAL v3.05 (NUREG-1887, 2007), RASCAL 4.2 (NUREG-1940 2012), and RASCAL 4.3 (NUREG-1940, Supplement 1, 2015). These versions of RASCAL, which were developed by NSIR, all retain a significant amount code from earlier RASCAL versions. Some of the code can be traced back to MESOI. Portions of the computational codes have been written in Microsoft Fortran, Digital Fortran, Compaq Fortran, and Intel Visual Fortran

under standards FORTRAN-77 and Fortran-90. The current version of RASCAL has been compiled using the Intel Visual Fortran compiler. The RASCAL user interface is primarily coded in Visual Basic 6.0, although, Microsoft VB.NET has been used more recently in developing some of the RASCAL tools. Radionuclide and nuclear facility data used by RASCAL are stored in Microsoft Access databases.

In NSIR, code development priorities were established by the NRC Program Manager (PM) with input from RASCAL users in the NRC Operations Center, NRC Regional Offices, Agreement States, industry, and the RASCAL developers. Table 3 shows the major changes in recent versions of RASCAL. Code documentation has generally consisted of a technical basis document that describes the methods RASCAL uses for calculations and a User's Guide book that provides a worked set of problem to train NRC staff on the use of RASCAL tools. The technical basis document, while neither a requirements document nor a design document, provides many of the details that would be found in these documents. It also contains comparisons of computational results between the old and new versions of the codes and describes verification and validation efforts at a high level. The problems in the User's Guide also contribute to code verification. All of the User's Guide problems have been reworked for each new version of RASCAL. Changes in answers to the problems are noted. Changes that are not expected based on code changes are explored until they are explained or a coding problem is identified.

The current development of the RASCAL tools is managed by RES The purpose of this document is to establish RES's SQA guidelines for the minor updates leading to the release of RASCAL 4.3.x. The applicability of this document is limited to the changes made to RASCAL 4.3 in producing RASCAL 4.3.x. It does not include a review of the RASCAL 4.3 legacy code that remains in RASCAL 4.3.x.

Table 3 Evolution of Recent RASCAL Changes

RASCAL Code Component	Version of RASCAL				
	RASCAL 3.05 (2007)	RASCAL 4.2 (2012)	RASCAL 4.3 (2013)	RASCAL 4.3.1 (2015)	RASCAL 4.3.2 (2016)
Source Term	Base Case	Minor changes	Added SOARCA LTSBO	Updated the LTSBO core release fractions	Resolved an issue with missing noble gases in coolant releases
			Added custom inventory option	Calibrated the BWR Mark 1 saturated wet well reduction factors for LTSBO	
	NUREG-1465 LOCA	NUREG-1465 LOCA	Added importance and balance	Revised pressure/hole size model	Revised the UF6 STDose model
			Revised pressure/hole size release	Revised NUREG-1465 LOCA	
			Expanded import, export, & merge capabilities	Resolved the spent fuel pool GUI and full core off-load option	
Atmospheric Transport and Dispersion (ATD)	3 domains (10, 25, & 50 miles)	Time-based dispersion with low speed adjustment	Added 100 mile domain	New output files are created by the plume and puff models	Resolved an issue with HF gas deposition velocity associated with wind speed and stability class.
	48 hour analysis limit	Dispersion parameters estimated by turbulence			
	Gaussian Puff and Plume	Temporal and spatial varying dep. velocities		Resolved an issue where the plume and puff models were not correctly building the list of depositing nuclides	Resolved an issue with the calculation of UO2F2 exposure and deposition
	Pasquill-Gifford Dispersion with low speed adjustment	Iodine 25% part, 30% I_2, 45% CH_3I	Extend analysis to 96 hours	Resolved an issue with the depletion calculation in the plume mode	
	Constant deposition velocities & wet deposition by washout	Revised decay scheme to include short-lived daughters implicitly			
Dose Calculation	TEDE, Cloudshine, CEDE, Groundshine using ICRP 26/30 methodology	Added option of using ICRP 60/72 dose coefficients	Added child thyroid	Dose labels now reflect the ICRP dosimetry system in use in RASCAL	Resolved an issue where the conversion of HF exposure to parts per million (ppm) for the lung pathway
	Finite Cloudshine			Updated GUI for consistent wording to describe dose calculations	
	Limited radionuclide set	Added all radionuclides in FGR 12 with half-lives > 10 minutes		Changed "dose conversion factor" to "dose coefficient"	
Meteorological Processor	Manual data entry	Manual data entry	Manual data entry or download from Internet using MetFetch	Added the capability to the Meteorological Data Processor to read stability class information in a RASCAL-ready import file	Revised MetFetch to correct several data issues with forecast and observational meteorological data from the NWS and NWS server security issues
				Revised MetFetch to correctly handle a "NIL" in records from some forecast files	
				Resolved the issue validating wind speeds units of kilometers per hour	

6.0 SOFTWARE PLANNING

RES is responsible for development and maintenance of the RASCAL code. RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3. The RASCAL software planning is focused on fixing issues and errors identified in the baseline (RASCAL 4.3) software; no new scope or software features are being added in these minor, incremental updates, except where it is practical and useful.

6.1 Software Development Methodology

The development methodology being used for the minor, incremental updates to RASCAL (4.3.x) can be considered incremental, and serve only to update or fix those sections of the code that are identified as being in error. The Software Developers are responsible for making the code updates and, depending on the severity of an issue or error, a new minor version of RASCAL may need to be released.

The following steps will be used when updating the RASCAL software:

1. Identify the need for the software change. The basis for the determination will come from the SNL RASCAL Collaboration (RASCAL_Help@nrc.gov) SharePoint site, either as an error report or request for change (Section 11).

For an error report, the following steps should be completed:

a. The Software Developer should use the submitted documentation as a basis for determining if an error exists within the software. As needed, the COR may need to communicate with the user to gather any additional information needed by the Software Developer. Based upon the submitted documentation, the Software Developer should be able to fully describe the condition(s) that lead to the error.

b. The Software Developer should identify the relevant code that is in error and identify candidate changes to fix the problem.

c. The Software Developer should confer with the TM to:

    i. Discuss options available for fixing the error; consider the effort versus benefit and verify that the details and extent of the changes are well understood.

    ii. Discuss any relevant technical issues; solicit outside expertise, as needed, to define the error fix.

    iii. Get approval to make the fix.

For software changes that are not errors (bugs), but are intended to make the software easier to use or debug, the following steps should be completed:

a. The Software Developer should use the submitted documentation as a basis for determining if a software change is warranted. As needed, the COR may need to communicate with the user to gather any additional information needed by the Software Developer.

b. The Software Developer should consider the software changes needed to implement the update, including possible approaches and conflicts with other parts of the software.

c. The Software Developer should confer with the TM to get approval to make the change.

2. If the TM approves a software change, the Software Developer should:

a. Create a RASCAL "Change Log ID" and "Change Log" (Appendix A) to document the relevant software changes and develop the necessary tests for validating the software change.

b. Make the necessary changes to the software code.

c. Perform defined tests and document the outcome within the "Change Log".

d. Communicate the results of the "Change Log" back to the TM for tracking within the SNL RASCAL Collaboration (RASCAL_Help@nrc.gov) SharePoint site.

3. The COR will communicate all findings and issue resolutions to the end user.

6.2 Software Tools

Software tools being used by the Software Developers in the development of RASCAL 4.3.x include:

- Microsoft Visual Basic 6 – used to maintain the older user interface programs that have not been transitioned to newer development tools.
- Microsoft Visual Studio – used to develop new Visual Basic and FORTRAN components; includes built in Git for version control.
- Intel Visual Fortran Studio XE for Windows – used with Visual Studio for the calculation programs.
- InstallAware – used to build the RASCAL installation program.
- Git for Windows (http://www.git-scm.com/download/win) – used for building and maintaining repositories for the programs and files not developed under Visual Studio (e.g. VB6 and data files).

- Macrium Reflect (from Paramount Software) – for performing software backups.
- MadCap Flare – help authoring tool used to create the online help files (chm format).
- Microsoft Access – used to build the radionuclide and facility databases.
- Text editor (e.g., UltraEdit or Notepad++) – used to build and edit the text data files.

6.3 Software Configuration Management

Configuration management is the responsibility of the Lead Software Developer. The primary purpose for configuration management is to ensure:

- RASCAL versions can be uniquely identified.
- Specific RASCAL versions of deliverables can be reproduced (software, data, and information product deliverables).
- Unintended and/or conflicting changes are prevented.
- Unintended use is prevented.

The following steps will be used by the Lead Software Developer to ensure configuration management of the RASCAL software:

1. A root "\Projects" folder will be created on the Lead Software Developer's computer to store all the RASCAL code; this computer will follow the backup plan defined in Section 6.4.
2. At the start of a minor version update, a subfolder will be created named "RASCAL 4.3.x Dev", where the "x" refers to the minor version number of the software update; this folder will contain the entire project and all the files related to the development of RASCAL 4.3.x.
3. Within this folder, separate folders will be created for each component of the RASCAL program (EXE or DLL). Folders will also be created for the supporting data files and databases used by the RASCAL software.
4. For each component folder, a Git repository will be created. For programs created within Visual Studio, the built-in Git tools should be used. For programs developed with VB6 and other files, such as databases and data file, the stand-alone version of Git should be used.
5. At the start of development, each element to be tracked in version control will be committed to the appropriate depository folder and clearly labeled as the initial starting point. Changes to the code will be formally tracked using the "RASCAL Change Log" (Appendix A) which documents the basis for the code change. When code changes are committed to the code repository, the change log number should be included as part of the commit message.

The Lead Software Developer is the responsible custodian who will ensure software configuration management. Other Software Developers who are performing code changes must receive the code from the Lead Software Developer before performing code updates. During development it is acceptable for Other Software Developers to maintain the interim version control for their portions of the code. Upon completion, all updates will be supplied to the Lead Software Developer for final configuration management. The changes must include a completed "Change Log" (Appendix A), which provides the basis for the code change.

6.4 Software Backup Plan

A software backup plan has been created by the Lead Software Developer to assure minimal progress would be lost in the event of a hardware failure. Backup software (i.e., Macrium Reflect—from Paramount Software) is installed on both the primary and secondary development computers. A dedicated backup hard drive is also installed on both primary and secondary development computers to provide a separate, physical device for backup storage. Each computer also contains an optical drive capable of creating CD, DVD, and Blu-Ray disks. Finally, there are external hard drives with USB and/or eSATA interfaces to which backups can be made for offsite storage.

The following steps outline a general software backup plan that will be followed by the Lead Software Developer:

1. At the beginning of every month, a full backup will be created using the Macrium Reflect software. The backup will consist of the main "\Projects" folder on the primary development computer; it will be made to a separate hard drive in the same computer.
2. At a weekly interval, incremental backups of the main "\Projects" folder will be made.
3. At the end of a month, a copy of the full and incremental backups will be made to an external hard drive and stored in folder labeled for that month. One-year of projects will be available on the hard drive.
4. Every 4-6 months, a clone of all hard drives will be made and stored off site.

The following steps outline a project-related backup plan that will be followed by the Lead Software Developer:

1. On the primary development computer, setup a scheduled, daily backup of the RASCAL 4.3.x development folders.
2. When actively making coding changes to RASCAL 4.3.x, burn these files to DVD and mail a copy of the DVD to the TM or the COR at a monthly frequency. If no active code changes or development is being made to RASCAL 4.3.x then the frequency may be set to some interval agreed upon between the COR and the Lead Software Developer.
3. On any secondary development computer, manually create backups whenever changes have been made.

7.0 SOFTWARE REQUIREMENTS

RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3. No new scope, software features, or software functionality is being added to RASCAL in these minor, incremental updates, except where it is practical and useful for software debugging. Thus, the RASCAL 4.3.x software requirements are being defined as a result of reported issues with the legacy software.

Section 11.0 of this SQAP discusses how RASCAL software issues are reported to RES; these issues are tracked within the SNL RASCAL Collaboration (RASCAL_Help@nrc.gov) SharePoint site. Issues that require a change to the code (Section 6.1) are assigned a formal "Change Log ID" and "Change Log," which are used to track issue resolutions requiring software changes. The RASCAL "Change Log" is used to define and document software requirements, including the reason for the code change, the location within the software where the code change was made, and all test cases used for the verification and validation of the code change.

Appendix B is an example exhibit that summarizes "Change Log Items" for RASCAL 4.3.1 and 4.3.2. Each item that lists a "Change Log ID" has an associated "Change Log," which defines the requirements associated with a specific software change. The Lead Software Developer, with the help of the Software Developers, is responsible for maintaining a "Change Log" for each issue requiring a software change. The Lead Software Developer will maintain a summary list of all "Change Log" items (similar to Appendix B) for all minor, incremental (4.3.x) software updates to RASCAL.

7.1 Reviews

As discussed in Section 6.1, the TM, in consultation with the Software Developers, must approve all software changes. Understanding and defining the software requirements associated with the code change is part of this approval process. The "Change Log" will document the basis for the code change and any associated requirements. A completed "Change Log" is required when the code is submitted to the Lead Software Developer for configuration management; the Lead Software Developer will review the "Change Log" to ensure that all sections have been completed.

8.0 SOFTWARE DESIGN & IMPLEMENTATION

RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3. The RASCAL software design and implementation builds upon existing legacy code within RASCAL 4.3. For consistency, the same software design structure (i.e., executables, DLLs, databases, and data files) and programing languages used to develop RASCAL 4.3 will be used to develop minor, incremental RASCAL software updates. The software is intended for use on Windows operating systems, including XP, Windows 7, and Windows 8.

The basic software design of RASCAL is a set of user interface (UI) tools that gather information from the user, call DLL files to perform certain calculations, and then display the results. In some cases, the software utilizes existing data (text or database files) to populate the UI or in a specific calculation.

The RASCAL executable (i.e., RASCAL43x.exe) is the main executable UI for accessing both "Primary" and "Additional" tools within the RASCAL software; all the tools are accessible from within the main RASCAL UI.

"Primary" tools accessible from within the RASCAL UI include:

- STDose UI – the main program for the Source Term to Dose model.
    - STDose can be used to start the following standalone programs:
        - Radionuclide Data Viewer
        - MetProc UI
        - MetView
    - STDose calls the following calculation modules, as needed:
        - STCalc
        - SFPCalc
        - UF6Calc
        - TADPlume
        - TADPuff
        - UF6Plume
        - Importance
        - Rx_Inventory_S
    - STDose calls the following DLLs, as needed:
        - ImportSourceTerm
        - DisplayReactorActivityBalance
        - DisplayImportance
        - CreateSTDoseCaseName
- FMDose UI – the main program for the Field Measurement to Dose model.
    - FMDose can be used to start following standalone programs:

        - Radionuclide Data Viewer
    - FMDose calls the following calculation module, as needed:
        - FMDCalc

- Radionuclide Data Viewer – standalone interface allowing the viewer to see contents of the nuclide database used within RASCAL.
- DecayCalc UI – the main program for the Decay Calculator tool.

"Additional" tools accessible from within the RASCAL UI include:

- CreateReactorInventory – used to create a custom base inventory file.
- RASCAL_ST_Merge_Export_Tool – used to combine and export source terms.
- MetFetch – used to download meteorological observations and forecast data from the internet.

8.1 Reviews

As discussed in Section 6.1, the TM, in consultation with the Software Developers, must approve all software changes. Understanding and defining the software design and implementation associated with the change is part of this approval process. The "Change Log" will document the software design change and its implementation within RASCAL. A completed "Change Log" is required when the code is submitted to the Lead Software Developer for configuration management; the Lead Software Developer will review the "Change Log" to ensure that all sections have been completed.

9.0 CODE DEVELOPMENT

RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3. The RASCAL code development utilizes and builds upon existing legacy code within RASCAL 4.3. For consistency, similar coding standards and naming conventions will be used to develop the minor, incremental (4.3.x) RASCAL software updates.

9.1 Code Standards & Naming Conventions

As discussed in Section 5.3, RASCAL has a long history of development. RASCAL 4.3 retains a significant amount of code from these earlier RASCAL versions. These versions did not adhere to a specific code standard or naming convention; consequently, there are different standards and conventions used throughout. For consistency and traceability, minor, incremental (4.3.x) code updates should reasonably attempt to follow the coding standards and naming conventions used in the legacy code sections/routines being updated. This will help to ensure readability of the code and aid in future debugging efforts.

9.2 Code Testing

Section 6.1 describes the software development methodology for the minor, incremental (4.3.x) code updates being made to RASCAL. For issues requiring a "Change Log," code testing will be performed by the Software Developer to confirm the code changes are functioning as designed and implemented. With regards to code testing, the "Change Log" includes relevant sections for documenting:

- Code tests – describe the tests developed and executed to verify that the code change works as designed.
- Test Results – write up the test results, including tables, screenshots, and other documentation to support, clarify, and interpret the test results.

The code tests, along with a completed "Change Log," will be submitted to the Lead Software Developer for configuration management; the Lead Software Developer will review the "Change Log" to ensure that all sections are complete.

9.3 Code Reviews

This SQAP describes software quality assurance involving minor code changes (RASCAL 4.3.x) to legacy (RASCAL 4.3) software. Because the changes are minor, formal code reviews are not required. However, a Software Developer may informally request a peer review to help locate an error or to verify a code change has been correctly implemented.

10.0 SOFTWARE TESTING

RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3. Section 9.2 discusses tests that are performed on code updates that are made as a result of software errors found by the RASCAL user community; these tests accompany a "Change Log" that describe and validate the code change is functioning as intended. The following section describes two additional test plans—"User's Guide Testing" and "Group Testing"—that will be used to verify the overall RASCAL software is operating as intended.

10.1 Software Test Plans

The following sections outline two test plans that will be implemented to verify that the minor, incremental (4.3.x) RASCAL software updates are operating as intended. The first test involves "User's Guide Testing"; the RASCAL User's Guide problems will be reworked in RASCAL 4.3.x and the results will be compared and any differences will be explained. The User's Guide problems provide a good basis for testing, since they have been developed over a period of many years and their solutions are well documented. The second test utilizes "Group Testing," whereby a small group of experienced RASCAL users will define problems based upon their operational experience to exercise the RASCAL 4.3.x software.

10.1.1 User's Guide Testing

User's Guide Testing will be performed by the COR, in consultation with the Software Developers, to verify consistency of the RASCAL 4.3.x output with sample problems contained in the RASCAL User's Guide. The User's Guide problems provide a good basis for software testing, since they have a long history of development and their solutions are well documented. Each User's Guide problem will be reworked and compared to the previous solution. Changes in answers to the problems will be noted. Changes that are not expected based on the minor, incremental code changes will be examined until they can be explained or a coding problem has been identified.

User's Guide Testing has the following requirements:

- For each problem in the User's Guide, obtain the RASCAL 4.3 results, including the problem's case summary, source-term summary, max value table, and screenshots of plume and puff model footprints (if applicable).
- Installed version of RASCAL 4.3, with the saved cases used in creating the User's Guide problems.
- Installed version of RASCAL 4.3.x, on an independent (i.e., non-development) computer using the candidate release installer.

User's Guide Testing involves the following steps:

1. Based on the User's Guide problem to be run, are any differences anticipated?
    a. In the user interface.
    b. In the source term calculation.
    c. In the generated wind fields.
    d. In the transport, diffusion, dose calculation.
2. Setup and run the User's Guide problem with RASCAL 4.3.x; save the following:
    a. PDF versions of the source term summary and max value table.
    b. Screenshots or PDF copies of plume and puff footprints.
    c. The new case file.
3. Compare the following:
    a. Source term total release.
    b. Source term by radionuclide; look at a range of half-lives and radionuclide groups.
    c. Plume dose values.
    d. Puff dose values.
4. Document the following:
    a. Pass – no differences.
    b. Pass – differences, but expected and explained.
    c. Fail – differences, cannot explain at this time.
    d. Fail – differences, resolution known.
5. If a problem comparison results in a "Fail", the issue will need to be submitted to RASCAL_Help@nrc.gov for formal tracking and resolution (Section 11.0).

10.1.2 Group Testing

Group Testing will be performed by select Software Testers who are experienced RASCAL users; Software Testers will be selected by the TM and the COR. The Software Testers will define select tests based upon their operational experience to exercise the RASCAL 4.3.x software.

Group Testing has the following requirements:

- Installed version of RASCAL 4.3.x, on an independent (i.e., non-development) computer using the candidate release installer.
- A document recording each test conducted.

Group Testing involves the following steps:

1. Define and document a scenario to be examined. This may include details on a specific type of plant or specific set of conditions to be examined; it should include a description of expected results.
2. Setup and run the case.
3. Examine the results and document whether expectations were met.
    a. Do the code results make sense? Can they be explained?
4. Document the following
    a. Pass
    b. Fail
        i. No obvious issue (e.g., crash), but solution appears to be in error.
        ii. Crashed or generated an error.
5. If the defined test results in a "Fail", the issue will need to be submitted to RASCAL_Help@nrc.gov for formal tracking and resolution (Section 11.0).

10.2 Test Reviews

Before software release, the TM and the COR must review and accept "User's Guide" and "Group" test results for overall acceptance of RASCAL 4.3.x.

11.0 ISSUE REPORTING

Issue reporting, including RASCAL-related questions, comments, or programming errors are submitted by RASCAL users via the "RASCAL Support" (https://www.usnrc-ramp.com/content/rascal-support) link on the U.S. NRC Radiation Protection Computer Code Analysis and Maintenance Program (RAMP) website (https://www.usnrc-ramp.com/). The "RASCAL Support" link on the RAMP website provides RASCAL users with options of either submitting a request to the RASCAL_Help@nrc.gov email account via the "RASCAL Support Request" page or to reach out to the RASCAL user community via the RASCAL Forum boards. Also, RASCAL users can use the "RASCAL Support" link to access the RASCAL Frequently Asked Questions (FAQ) to find answers to issue related to RASCAL 4.3.x.

11.1 RASCAL Support on the RAMP Website

The "RASCAL Support" link allows RASCAL users to reach out to the RASCAL user community via the RASCAL Forum boards or to directly submit an email request to the RASCAL Development Team (RDT) via the "RASCAL Support Request" page.

11.1.1 RASCAL Forum Boards

The RASCAL Forum boards are monitored by the Lead Software Developer and COR to ensure that RASCAL user's question and requests are satisfactorily answered in a timely manner. The subject areas of the RASCAL Forum boards cover the following topics:

- Error Reports
- Model Questions
- General Emergency Preparedness Questions
- General RASCAL Usage Questions

11.1.2 RASCAL Support Request Page

The "RASCAL Support Request" page opens a form by which RASCAL users can submit request to the RASCAL_Help@nrc.gov email account which is managed by the COR. Issues are normally reported by the RASCAL user community, but they may also be reported by the RDT (i.e., COR, TM, Lead Software Developer or Software Developers) to formally document and track issues found during software development, planned updates to the software, or to add a new software feature.

Both the RASCAL Forum boards and the "RASCAL Support Request" page allow the RASCAL user to attach files that will assist the RDT in documenting and troubleshooting the reported error. Standard RASCAL case files (".std" and ".fmd") as well as various other types of files (".txt," ".xml," ".csv," ".xlsx," ".exe," "gz," ".tz," ".pdf," ".doc," ".docx," ".ppt," ".pptx," ".jpg," ".png" ".jpeg," ".gif," and ".msg") can be attached to both the forum and support request email.

11.2 Issue Documentation

Once the COR has received the necessary issue documentation, the issue is entered into the SNL RASCAL Collaboration (RASCAL_Help@nrc.gov) SharePoint site for formal tracking and resolution. Relevant fields used for issue tracking and resolution include:

- Unique issue ID
- Date issue received
- RDT staff contact managing the issue resolution
- Requestor name and contact information
- Request type (code errors; model questions; documentation issues; distribution questions)
- Request scope details and description
- Error code generated by the RASCAL software, if applicable
- Resolution status (in progress, completed, future code version)
- Resolution priority (low, medium, high)
- Resolution deliverable

After the issue is entered into the SNL RASCAL Collaboration (RASCAL_Help@nrc.gov) SharePoint site, it is routed by the COR to the responsible RDT member for resolution; the responsible team member may include the TM and/or a Software Developer.

Responses to issues that can be resolved without a code change (e.g., a model, distribution or documentation question) will be communicated to the COR, who will enter the resolution into the SNL RASCAL Collaboration (RASCAL_Help@nrc.gov) SharePoint site and communicate a response to the requestor by email. If the issue requires a code change, a "Change Log" will also be generated by the Software Developer to formally track and document the code modification (Section 6.1). Depending on the severity of an issue, a new version of RASCAL may need to be released (Section 14.0).

12.0 USER DOCUMENTATION

Earlier versions of RASCAL included various NUREGs that described the operation and bases of the code; Section 2.0, lists many of these documents. RASCAL 4.3.x documentation includes:

- RASCAL 4.3: Descriptions of Models and Methods. NUREG-1940, Supplement 1 (Ramsdell et al. 2015) – describes the technical bases for changes made in models and methods in going from RASCAL 4.2 to RASCAL 4.3 and 4.3.1. NUREG-1940 provides the technical bases for the RASCAL computation codes describe in RASCAL 4.2.
- RASCAL 4.3 User's Guide (Draft) (Athey et al. 2015) – provides problems designed to familiarize the user with the RASCAL software through hands-on problem solving.

These documents will be revised, as appropriate, to reflect any changes made to the software as a result of the minor code updates in RASCAL 4.3.x. Additionally, these documents will be made available to all RASCAL users on the "RASCAL Technical Documents Download Page" on the RAMP website (https://www.usnrc-ramp.com/).

13.0 SOFTWARE REVIEWS

RASCAL 4.3 serves as the baseline version from which minor, incremental (4.3.x) code updates are being made. These updates are in response to issues and errors identified by the RASCAL user community subsequent to the release of RASCAL 4.3. Reviews of RASCAL 4.3.x software requirements (Section 7.1), design and implementation (Section 8.1) and testing (Section 9.3) are discussed in this SQAP at a level that is commensurate to the minor software updates being performed. These reviews are internal; no independent reviews are being performed.

This SQAP will be reviewed by the COR and TM annually to ensure applicability or when changes to the RASCAL software project warrant a new review.

14.0 SOFTWARE RELEASE PLANNING

Appendix C, "RES and NSIR Coordination Process for Releasing RASCAL Updates" has been developed by the NRC and it describes coordination activities between the RES and NSIR for releasing RASCAL updates. The purpose of Appendix C is to identify and describe the criteria, and associated organizational responsibilities, for releasing RASCAL updates to either limited groups of users, as a limited distribution release, or to all users, as a general distribution release. Appendix C will be used by the COR for RASCAL software release planning.

Presently, the RASCAL 4.3.x software is distributed to all RAMP members via the RAMP website (https://www.usnrc-ramp.com/). RASCAL users interested in obtaining a copy of the RASCAL 4.3.x software must meet the registration requirements for obtaining the RASCAL Code from the RAMP website. The RAMP website provides a single point-of-access for distributing the RASCAL software and maintaining a master list of RASCAL users. At the time of issuing a new RASCAL update (4.3.x), the COR will provide RAMP website administrator with a new RASCAL 4.3.x installation program and an install procedure; source files are not included in a release. The RAMP Team will then email the master list of RASCAL users and indicate how to obtain/download the RASCAL 4.3.x update from RAMP.

14.1 Other Deliverables

In addition to software releasing planning, the Lead Software Developer will provide the following deliverables to NRC RES prior to software release:

- source code,
- software executable(s) and install,
- databases,
- software "Change Logs" and associated test cases and
- Updates to NUREGs (Section 12.0).

These deliverables will be provided to the TM and the COR by the Lead Software Developers on DVD in support of software release management.

15.0 MAINTENANCE AND SUPPORT

Maintenance and support on the RASCAL software is provided by RES on an ongoing basis. The primary means of communication for RASCAL maintenance and support is through the "RASCAL Support" link on the RAMP website and the options of either contacting the RASCAL_Help@nrc.gov email account via the "RASCAL Support Request" page or to reach out to the RASCAL user community via the RASCAL Forum boards. The RASCAL user community's questions and requests are collected, tracked, prioritized, and dispositioned by the COR in a similar manner to RASCAL issue reporting (Section 11.0).

16.0 RECORDS COLLECTION, MAINTENANCE, AND RETENTION

Documentation in support of meeting the objectives of this SQAP will be maintained by the Lead Software Developer, including:

- code,
- installs,
- documentation (NUREGs and Help Files),
- change logs and supporting test cases and
- User's Guide and Group Testing Documentation.

All digital records (e.g., Word documents, Excel spreadsheets) will be archived by the backup process discussed in Section 6.4. These records will be retained for a period of one year as part of the rotating backups to hard drives and for at least five years as part of the backup DVD creation. The records will be stored in Lead Software Developer's office and in the RASCAL Project Files maintained by the COR. As discussed in Section 14.1, the Lead Software Developer will provide RES with electronic copies of certain records to meet software release planning SQA requirements.

Paper records (e.g., hand written notes, code markups, and annotated screenshots from testing) will be retained for a period of at least five years. Only those documents that are deemed to have relevance to an SQA activity will be retained. These documents will be stored in the Lead Software Developer's office.

17.0 TRAINING

This SQAP is the primary document for defining the SQA activities to be completed for the release of RASCAL 4.3.x. The RDT should be familiar with the SQA activities discussed in this SQAP; no other internal training activities necessary to meet the objectives of this SQAP.

APPENDIX A – RASCAL CHANGE LOG TEMPLATE

RASCAL Change Log	**Change ID:**	
	Base version:	
	Reason for change:	☐ **bug / fault** ☐ **update** ☐ **new feature**
Location of change	**RASCAL area:**	
	Runtime file:	
	Code files:	
References	**NRC Issue ID:**	
	Other:	
Timeline of events in the change process	**Person**	**Date**
Issue identified		
Code change development complete		
Testing complete		
Changes reviewed and approved		
Code committed to version control		
Released with RASCAL version x.x.x		

Background

Describe why the change is needed. If an error/fault, describe how it was discovered and the steps to see the problem. Include enough detail so it will be clear how to tell when the problem is fixed. If an update or new feature, describe the need.

Discussion

If an error/fault, describe the results of the evaluation of the code. Be specific as to what in the code is causing the problem. If new or update, provide some details about the design of the changes.

Code changes

Describe the changes made. Be specific as to files and subroutines or functions. As practical, include line numbers or code snippets.

Code tests

Describe the tests developed and run to verify that the change works as designed.

Test Results

Write up the test results. Tables, screenshots, etc. inserted to support and clarify.

Other reference material or information

Put here anything else relevant that helps understand the changes.

Supporting files

List the names and as needed the location of any files that provide supporting information. This may include RASCAL saved cases, screenshots, spreadsheets, and documents.

Additional notes

Put anything here to help understand the change.

APPENDIX B – LIST OF CHANGE LOG ITEMS FOR RASCAL 4.3.1 & 4.3.2

RASCAL 4.3.x - List of Change Log Items — Last updated: 2016-09-01

Change Log ID	RASCAL Component	Description	NRC SharePoint ID #	Athey FogBugz Case	Code Changes Complete	Change Log Complete	QA Complete	Update to Tech Doc Needed	Release Version
1	SFPCalc	Spent fuel - core offload - default inventory	83, 97	472	yes	yes	yes		4.3.1
2	SFPCalc	Spent fuel - core offload - custom inventory	83, 97	473	yes	yes	yes		4.3.1
3	STDose UI	SF pool batch validation	125	474	yes	yes	yes		4.3.1
4	STDose UI	Dose vs time export crash when operation canceled	126	475	yes	yes	yes		4.3.1
5	STDose UI	STDose - Validation of nuclide in "Iso in Fire" ST	95	476	yes	yes	yes		4.3.1
6	STDose UI	Balance file summary	127	477	yes	yes	yes		4.3.1
7	Facility DB	Reactor power uprates	128	478	yes	yes	yes		4.3.1
8	Decay data	Updates to Verified R4_Chains.dat	131	479	yes	yes	yes		4.3.1
9	ImportSourceterm	Updates to improve functionality and build in missing daughters	48, 50, 110, 152	480	yes	yes	yes	Yes	4.3.1
10	STCalc	Pressure / hole size method update	129	481	yes	yes	yes	Yes	4.3.1
11	STDose UI	Export ST missing date and time when filtered	133	482	yes	yes	yes		4.3.1
12	STCalc	NaN and 32hr wet well RDF issue fix	2	483	yes	yes	yes		4.3.1
13	TAD codes	Total deposition mix display fails import source term	130	484	yes	yes	yes		4.3.1
14	STDose UI	Update SF pool drained screens	132	485	yes	yes	yes		4.3.1
15	STDose UI	SF pool drained - add delay after SD	134	486	yes	yes	yes	Yes	4.3.1
16	Facility DB	Review and update climate data	65	487	yes	yes	yes		4.3.1
17	Met Proc	Load stability class from RASCAL-Ready if present	135	488	yes	yes	yes		4.3.1
18	MetProc	Fix retention of imported precip codes	136	489	yes	yes	yes		4.3.1
19	LT_SBO.dat	Add low level values for SOARCA core release fractions	137	490	yes	yes	yes		4.3.1
20	STDose UI	Import file name not shown in case summary on reload	138	491	yes	yes	yes		4.3.1
21	MetFetch	Error downloading buoy offshore met forecast data automatically	107	492	yes	yes	yes		4.3.1
22	NuclidesToAddStar	New data file to be used by Import5T DLL	119	493	yes	yes	yes		4.3.1
23	LT_SBO.dat	Update BWR core release fractions with latest SOARCA values	150	494	yes	yes	yes	Yes	4.3.1
24	STDose UI	Wording / label changes for ICRP60	139	495	yes	yes	yes		4.3.1
25	FMDose	Wording / label changes for ICRP60	140	496	yes	yes	yes		4.3.1
26	Importance DLL	Wording change: submersion to cloudshine	141	497	yes	yes	yes		4.3.1
27	Importance Display	Wording change: submersion to cloudshine	142	498	yes	yes	yes		4.3.1
28	STDose UI	Pressure/hole leak screen; update predefined hole sizes	151	509	yes	yes	yes		4.3.1
29	Help files	Update help files as needed	143	499	yes	yes	yes		4.3.1
30	MetProc	Allowed range of speeds when using km/h too small	115	500	yes	yes	yes		4.3.1
31	Merge / Export tool	Show "busy" cursor when loading or merging	144	501	yes	yes	yes		4.3.1
32	Merge / Export tool	Update so cases with imported ST get release height	145	502	yes	yes	yes		4.3.1
33	Merge / Export tool	Fix XML export of imported Fukushima source term	146	503	yes	yes	yes		4.3.1
34	STDose UI	3 part version number on footprints	147	504	yes	yes	yes		4.3.1
35	STDose UI	ST plots - long duration source terms displayed incorrectly	148	505	yes	yes	yes		4.3.1
36	STDose UI	ST differences with core recovered = NO vs damage = vessel melt through	149	506	yes	yes	yes	Yes	4.3.1
37	Installer	Installer not correctly installing folders (error code 76)	41, 43, 79, 86, 87, 88, 94, 100, 108, 109, 111, 112, 113, 123	507	yes	yes	yes		4.3.1
38	STDose UI	Check 4.2 to 4.3 met (error code 13)	29, 35, 40	511	yes	yes	yes		4.3.1
39	Data files	Create topo and Zo files for special sites (error code 76)	41, 78, 82	510	yes	yes	yes		4.3.1

RASCAL 4.3.x - List of Change Log Items

Last updated: 2016-09-01

Change Log ID	RASCAL Component	Description	NRC SharePoint ID #	Athey FogBugz Case	Code Changes Complete	Change Log Complete	QA Complete	Update to Tech Doc Needed	Release Version
40	STDose UI	Sourceterm summary display of importance: submersion to cloudshine	153	518	yes	yes	yes		4.3.1
41	Facility DB	Update the UTM coordinates for Hanford, INEL, and ORNL	154	519	yes	yes	yes		4.3.1
42	LT_SBO.dat	Update PWR core release fractions with latest SOARCA values	157	522	yes	yes	yes	Yes	4.3.1
43	STCalc	Revisions to the balance file	155	520	yes	yes	yes		4.3.1
44	STCP1.dat	Add late in-vessel to LOCA sourceterm	156	521	yes	yes	yes	Yes	4.3.1
45	DisplayBalance	Fix problem with printing balance file	159	523	yes	yes	yes		4.3.1
46	TADPlume	Add output of plume characteristics	163	524	yes	yes	yes		4.3.1
47	TADPuff	Add output of puff characteristics	162	525	yes	yes	yes		4.3.1
48	STDose UI	Fix calculation timer to work correctly when using imported sourceterm	164	515	yes	yes	yes		4.3.1
49	Merge / Export tool	Update to handle 3 digit version # and standardize "Creator" values	N/A	527	yes	yes	yes		4.3.1
50	STCalc	Update method for sourceterm based on containment radiation monitor	167		yes	yes	yes	Yes	4.3.1
51	STCalc	Update to STCalc to output release time steps when release rate is zero	170	526, 529	yes	yes	yes	Yes	4.3.1
52	MetFetch	Update to correctly display download time-out error	165		yes	yes	yes		4.3.1
53	MetFetch	Update to correctly display error when user does not have write permission	177		yes	yes	yes		4.3.1
54	SFPCalc	SF damaged underwater - code does not handle all user inputs	174		yes	yes	yes		4.3.1
55	STDose UI	SF damaged underwater - update controls for default vs custom fuel	174		yes	yes	yes		4.3.1
56	TADPlume	Fix depletion issue	182		yes	yes	yes		4.3.1
57	STDose UI	Fix File not Found error 53 for sites not in database	101, 188		yes	yes	yes		4.3.1
58	Facility DB	Update Laguna Verde met stations	171		yes	yes	yes		4.3.1
59	CreateSTDoseCaseName	Update code to use 24h clock for time string in file name	196		yes	yes	yes		4.3.1
60	STDose UI	Numeric table subscript error when clicking outside grid	212		yes	yes	yes		4.3.1
61	STDose UI	Fix problem with file saving	214		yes	yes	yes		4.3.1
62	Merge / Export tool	Fix problem displaying merge files with more than 6.75 days of data	217		yes	yes	yes		4.3.1
63	STCalc	Fix wet well status handling	216		yes	yes	yes		4.3.1
64	Merge / Export tool	Update code that enables-disables buttons	222		yes	yes	yes		4.3.1
65	STDose UI	Fix issue with I-131 air conc display in SI units	223		yes	yes	yes		4.3.1
66	DecayCalc	Fix issue with nuclide name list not working	227		yes	yes	yes		4.3.1
67	STDose UI	Increase allowed UO2 in fire MAR value	207		yes	yes	yes		4.3.1
68	STCalc	Update to allow different RDFs for BWR Mk1 wetwell with LTSBO	228		yes	yes	yes		4.3.1
69	Facility DB	Update the Surry design pressure	234		yes	yes	yes		4.3.1
70	RASCAL	Update each UI with 4.3.2 version number	N/A		yes	yes	yes		4.3.2
71	MetFetch	Fix missing button issue when screen is scaled	331		yes	yes	yes		4.3.2
72	MetFetch	Fix to use updated access methods for NWS observation data	328, 329		yes	yes	yes		4.3.2
73	Facility DB	Modify to incorporate the South Africa sites	301		yes	yes	yes		4.3.2
74	MObsProc	Use the version that fixed the southern hemisphere issue	317		yes	yes	yes		4.3.2
75	STCalc	Missing noble gas in coolant releases	333		yes	yes	yes		4.3.2
76	UF6Plume	Fix conversion of HF exposure to PPM for lung pathway	319		yes	yes	yes		4.3.2
77	UF6Plume	Fix issue with deposition velocity used with HF gas	319		yes	yes	yes	Yes	4.3.2
78	UF6Plume	Fix issue with calculation of UO2F2 exposure & deposition	319		yes	yes	yes		4.3.2
79	UF6Plume/UF6Calc	Fix issues with HF concentrations	319		yes	yes	yes		4.3.2
80	Site folders	Add in the folders for SAFARI-1 and Koeberg (South Africa)	301		yes	yes	yes		4.3.2
81	Installer	Update for 4.3.2 release	N/A		yes	yes	yes		4.3.2

RASCAL 4.3.x - List of Change Log Items

Last updated: 2016-09-01

Change Log ID	RASCAL Component	Description	NRC SharePoint ID #	Athey FogBugz Case	Code Changes Complete	Change Log Complete	QA Complete	Update to Tech Doc Needed	Release Version
82	Help Files	Update all to replace RASCAL_Help email with RAMP web address	N/A		yes	yes	yes		4.3.2
83	MetFetch	Add option to use high-resolution forecasts	329		yes	yes	yes		4.3.2
84	MetFetch	Reject bad forecasts	329		yes	yes	yes		4.3.2
85	MetFetch	Update task scheduling	329		yes	yes	yes		4.3.2
86	MetProc	Fix DST time adjustment	328		yes	yes	yes		4.3.2
87	MetFetch	Fix conversion from UTC to LST	328		yes	yes	yes		4.3.2

APPENDIX C – RES & NSIR COORDINATION PROCESS FOR RELEASING RASCAL UPDATES

1.0 INTRODUCTION

This is a draft version of a procedure that describes coordination activities between the Office of Nuclear Regulatory Research (RES) and the Office of Nuclear Security and Incident Response (NSIR) for releasing updates to the Radiological Assessment System for Consequence Analysis (RASCAL) computer code. Comments and feedback on this document will be incorporated into a final version, and become part of a Configuration Management Plan for the RASCAL computer code.

The final version of this document is intended to be a living document and changed periodically to incorporate lessons learned during implementation. It should be placed under configuration management and the respective changes managed accordingly.

1.1 Purpose

The purpose of this document is to identify and describe the criteria, and associated organizational responsibilities, for releasing updates to RASCAL to either limited groups of users, as a limited distribution release, or to all users, as a general distribution release. This document is a component of an overall configuration management plan that describes the processes for ensuring that routine and non-routine software changes occur within an identifiable and controlled environment.

1.2 Background

In September 2013, NSIR transferred RASCAL to RES for code development and maintenance. RASCAL version 4.3 was developed by NSIR and issued by RES on September 29, 2013, and serves as the baseline version of this software for further development, along with its associated level of software quality assurance. Further changes to RASCAL are made through formal change control procedures, which include approval or rejection of proposed changes and issuance of the code to users. This document addresses these considerations by establishing a consistent, cross-organizational process for releasing future updates to RASCAL.

Historically, planned updates of RASCAL have been made approximately every 18 months to incorporate new features and feedback from users. However, some RASCAL updates were made in less time to address software faults. Both limited and general distribution releases were made available to all users, based on the software distribution practice in effect at the time.

1.3 Scope

The scope of this document is the identification of an agreed-upon process for releasing updates to RASCAL. A well-defined software release process is integral to the configuration management system for RASCAL.

1.4 References

The following references were used in preparation of this document:

- *Software Quality Assurance for RES-sponsored Codes*, RES Office Instruction PRM-12, March 5, 2012, (ML12132A176).
- *Software Quality Assurance Program and Guidelines*, NUREG/BR-1067, February 1993. ML, (ML012750471).

2.0 OVERVIEW OF THE CONFIGURATION MANAGEMENT PLAN FOR RASCAL

Configuration management is the responsibility of the Lead Software Developer and it is discussed in more detail in section 6.3 of the RASCAL 4.3.x SQAP.

3.0 CRITERIA FOR RELEASING UPDATES TO RASCAL

Software maintenance includes modification of a software product after its release to correct faults, to improve performance, or other attributes. The urgency of releasing a new version of the software is commensurate with the ability of the software to perform its intended function.

Some faults (i.e. software bugs or errors) may require speedy resolution if they cause a code crash during an essential function. In such situations, an urgent update (aka "hotfix" or "quick fix engineering update") of RASCAL may be appropriate in order to correct a single significant problem in the software, and may not be released to all users. Alternatively, an urgent update could be issued to all users (aka "patch") to the software to fix the known software fault.

In contrast to urgent software updates, relatively minor faults in the software may be scheduled for correction during a planned update several months after a major revision of the software is released or later, as part of the next major scheduled software update.

3.1 Urgent Updates to RASCAL

3.1.1 Criteria for Urgent Updates

The following criteria are applied for making an urgent update to RASCAL:

A. The code crashes when the user is performing an essential function.

   Examples include the inability to complete a calculation within the model domain (e.g., 96 hours and 100 miles) for any BWR or PWR accident scenario, including spent fuel pools.

B. The code calculates a result that would significantly impact decision-making for recommending protective actions.

Examples include underestimation or overestimation of dose by a factor of 10 for any BWR or PWR accident scenario, including spent fuel pools.

C. The code provides outputs that are inconsistent for the same input parameters.

Examples include erroneous outputs (dose tables, dose maps, air or ground concentrations) that are inconsistent and would likely cause confusion to the Dose Assessment Analyst during a drill or actual emergency.

3.1.2 Timeliness of Urgent Updates

RES and NSIR staffs should communicate regularly on the nature and extent of software faults that are identified by internal (NRC) and external users. RES staff is notified of software faults and implementation issues from users through email notification via RASCAL_Help@nrc.gov. If NSIR staff becomes aware through their use of the software or their contacts in the emergency preparedness and response community of a software fault that would require an urgent update to RASCAL, they should notify the RES Contracting Officer's Representative (COR) and Technical Monitor immediately.

An urgent release to correct a significant software fault should be accomplished within one month of its discovery, depending on the complexity of the fault. The resolution of the fault should include troubleshooting, code and user interface corrections, and verification by the contractor prior to its release.

3.1.3 Scope of Release for Urgent Updates

When an urgent update to RASCAL is necessary, RES will coordinate with NSIR to recommend whether the distribution of the urgent update should be a limited distribution release, such as for the NRC Operations Centers only, or a general distribution release for all RASCAL users. RES will implement the distribution approach agreed upon by this process. Additionally, at the time of the urgent update, if it is decided that the release will be a limited distribution release then changes made to the software should be incorporated into the next scheduled update that is released to all RASCAL users. This process should be used for each urgent update of RASCAL.

3.2 Scheduled Updates to RASCAL

3.2.1 Criteria for Scheduled Updates

The following criteria are applied for making a scheduled update to RASCAL:

A. The code crashes when the user is performing a non-essential function.

Examples include a software crash when using a feature in RASCAL that is not related to an essential dose calculation.

B. The code calculates a result that would not significantly impact decision-making for recommending protective actions.

Examples include underestimation or overestimation of dose by a factor of less than 10 for any boiling-water reactor (BWR) or pressurized-water reactor (PWR) accident scenario.

3.2.2 Timeliness of Scheduled Updates

Similar to Section 3.1.2 above, RES and NSIR staffs should communicate regularly on the nature and extent of software faults that are identified by internal (NRC) and external users. If there is a software fault that should be included in the next scheduled update to RASCAL, NSIR should inform the RES COR and Technical Monitor within one week.

A scheduled update to correct a software fault should be accomplished within six months of its discovery, depending on the complexity of the fault. The resolution of the fault should include troubleshooting, code and user interface corrections, and verification by the contractor prior to its release. Additionally, scheduled updates will include all software coding changes made for any urgent updates completed since the last scheduled update.

3.2.3 Scope of Scheduled Updates

Similar to the process outlined in Section 3.1.3 above, RES will confer with NSIR on distribution of scheduled software updates. RES will implement the distribution approach agreed upon by this process.