CNWRA *A center of excellence in earth sciences and engineering*

A Division of Southwest Research Institute®
6220 Culebra Road • San Antonio, Texas, U.S.A. 78228-5166
(210) 522-5160 • Fax (210) 522-5155

March 27, 2003
Contract No. NRC–02–02–012
Account No. 20.06002.01.113

U.S. Nuclear Regulatory Commission
ATTN: Mr. James Firth
Office of Nuclear Material Safety and Safeguards
Division of Waste Management
Performance Assessment and HLW Integration Branch
Mail Stop 7C-18
Washington, DC 20555

Subject: Transmittal of Software Validation Test Plan for the Total-System Performance Assessment Version 5.0 Code (IM 06002.01.113.310)

Dear Mr. Firth:

The purpose of this letter is to transmit the subject report in fulfillment of TPA Version 5.0 Software Testing Plan—Letter Report (IM 06002.01.113.310). A preliminary software validation test plan that contained descriptions of the function tests for the utility modules associated with the TPA Version 5.0 code's executive model, as well as descriptions of the system-level tests that may be conducted at a later time, was submitted on February 28, 2003. The draft final software validation test plan includes the tests described in the preliminary report and also provides descriptions of function tests for the TPA Version 5.0 code's consequence modules and associated stand-alone codes.

We estimate that completion of the Phase I and Phase II tests described in the software validation test plan will require the expenditure of 22 person-months of effort by the July 31, 2003 delivery date for TPA Version 5.0 Code, Validated Version—Software (Major Milestone 06002.01.113.330). We can bring to bear 14 person-months of effort during the next four months (March 28, 2003 through July 31, 2003), which is 8 person-months less than required. The options for completing the validation testing include: (i) 8 person-months of effort from NRC TSPAI staff, (ii) reducing the scope of the tests by 8 person-months, and (iii) extending the due date of the MM so that the CNWRA can provide more resources to validation testing. Please let me know which of the three options you prefer or if you there are other approaches you wish to consider.

Washington Office • Twinbrook Metro Plaza #210
12300 Twinbrook Parkway • Rockville, Maryland 20852-1606

If you have any questions about the content and format of the draft final software validation test plan, please direct them to me at (210) 522-5082.

Sincerely yours,

Gordon W. Wittmeyer, Ph.D.
Manager, Performance Assessment

GW/rm
Enclosure

cc:

M. Leach	J. Schlueter	J. Danna	W. Patrick	R. Janetzke
B. Meehan	A. Campbell	W. Dam	CNWRA Directors	L. Howard
D. DeMarco	T. McCartin	C. McKenney	CNWRA Element Managers	S. Mohanty
E. Whitt	C. Grossman	R. Johnson	S. Domine (SwRI QA)	I. Chichkov
J. Greeves	D. Esh	M. Thaggard	P. Maldonaldo	O. Pensado
W. Reamer	R. Codell	J. Peckenpaugh	R. Benke	O. Povetko
L. Kokajko				P. LaPlante
				M. Smith

SOFTWARE VALIDATION TEST PLAN FOR THE TOTAL-SYSTEM PERFORMANCE ASSESSMENT VERSION 5.0 CODE

Prepared for

U.S. Nuclear Regulatory Commission
Contract NRC–02–02–012

Prepared by

Sitakanta Mohanty (CNWRA)
Bruce Goodwin (Consultant)
George Adams (SwRI)
Gordon Wittmeyer (CNWRA)

Center for Nuclear Waste Regulatory Analyses
San Antonio, Texas

Gordon Wittmeyer
Gordon W. Wittmeyer
Performance Assessment, Manager

3/27/2003
Date

ABSTRACT

This software validation test plan describes functional tests that will be conducted to validate the executive module of the Total-system Performance Assessment (TPA) Version 5.0 code as well as a suite of system-level software validation tests that assess the ability of the TPA Version 5.0 code to correctly compute overall risk estimates for other high-level waste disposal systems. Although a certain level of software validation has been performed at the completion of each major revision of the TPA code, these validation efforts were not formalized. Because the TPA Version 5.0 code is likely to be used during the U.S. Nuclear Regulatory Commission (NRC) review of the U.S. Department of Energy license application for a potential repository at Yucca Mountain, this test plan describes both the formal process for systematically testing the validity of the software as well as a limited set of model validation tests whose successful completion will demonstrate that the TPA code is suited for its intended application.

Because there is no single, generally accepted approach for validating software, like the TPA code, which simulates diverse coupled natural processes over time periods of tens of thousands of years, the validation tests will be conducted in phases so that knowledge gained in the execution of earlier phases can be used to improve the robustness and utility of the tests performed in the later phases. Three different phases will be executed for the completion of the validation exercise. Phase I focuses primarily on tests that will provide confidence that the software computes correctly. Phase II involves function-level tests that provide confidence that the model provide reasonable results. Phase III involves comparison of system code with other system codes. All Phase I activities and most of the Phase II activities are expected to be completed by the time of the final delivery of the TPA Version 5.0 code to the NRC in July 2003. Phase III activities will be initiated after July 2003.

This plan also considers the software validation of computer codes that are used by various key technical issue staff to provide inputs to the TPA code and contains tests proposed by those key technical issue staff that were involved in the development of the abstracted models in the TPA Version 5.0 code.

CONTENTS

CONTENTS (continued)

TABLES

ACKNOWLEDGMENTS

This report was prepared to document work performed by the Center for Nuclear Waste Regulatory Analyses (CNWRA) for the U.S. Nuclear Regulatory Commission (NRC) under Contract No. NRC–02–02–012. The activities reported here were performed on behalf of the NRC Office of Nuclear Material Safety and Safeguards, Division of Waste Management. The report is an independent product of the CNWRA and does not necessarily reflect the views or regulatory position of the NRC.

The authors wish to thank M. Thaggard (NRC), R. Benke, and R. Folck (consultant) for their technical reviews and B. Sagar for his programmatic review. Thanks are expressed to R. Janetzke, O. Pensado, M. Smith, R. Benke, O. Povetko, P. LaPlante, R. Fedors, C. Manepally, Bertetti, D. Pickett, G. Cragnolino, J. Winterle, D. Farrell, S. Painter, L. Browning, J. Stamatakos, B. Hill, D. Esh (NRC), and R. Codell (NRC) for providing test ideas and test cases. Finally, the authors extend their thanks to R. Mantooth and A. Ramos for superb secretarial support.

QUALITY OF DATA, ANALYSES, AND CODE DEVELOPMENT

DATA: No new data were used in the preparation of this document.

ANALYSES AND CODES: No computer codes were used in the preparation of this document.

1 SCOPE OF VALIDATION

1.1 Introduction

The U.S. Nuclear Regulatory Commission (NRC) with assistance from the Center for Nuclear Waste Regulatory Analyses (CNWRA) has developed the Total-System Performance Assessment (TPA) Version 5.0 code to be used by the staff to assist their review of the license application for the disposal of high-level waste at Yucca Mountain, Nevada. Because of the important role played by the TPA code, it must be tested prior to the receipt of any license application to ensure that it is reliable and suitable for its intended use. This report presents a test plan whose successful execution will produce the information needed to demonstrate that the TPA Version 5.0 code functions properly and is suited for use in the NRC review of any license application for the Yucca Mountain high-level waste repository.

Software validation is a quality assurance process that ensures the ability of software to meet its intended function by demonstrating that results produced by the software are consistent with specified design requirements. Examples of software validation methods include inspection of source code and quantitative evaluation of implemented mathematical algorithms. The process generally involves the execution of a set of tests constructed and implemented by staff who were not directly involved with the development of the software. This independence helps to ensure that unstated, tacit assumptions made by the developers do not influence the evaluation of test results. A failed validation test may be indicative of problems with the software or with the documentation of the software or with both. A successful validation test, therefore, is one where (i) the software performs as intended and (ii) there are no ambiguities or deficiencies in the documentation of how the code was intended to function.

Several versions of the TPA code have been released since IPA Phase II (Wescott, et al., 1995). Major releases include TPA Versions 3.1.4 (Mohanty and McCartin, 1998a), 3.2 (Mohanty and McCartin, 1998b), 4.0 (Mohanty, et al., 2002a), and 5.0, which remains under development. Each major modification involved substantial changes to the fundamental conceptual models, algorithms implementing the conceptual models, and the sampling strategies used to propagate uncertainty and variability. Each version of the code was checked at the end of its development cycle for reasonableness and computational correctness before being used for system-level or process-level analyses. Subsequent to the completion of three major revisions to the TPA code since IPA Phase II, detailed sensitivity analysis activities were conducted that exercised most of the capabilities built into the TPA code. The TPA Version 3.1.4 code was used to conduct system-level as well as process-level sensitivity analyses (NRC, 1999), while the TPA Version 3.2 and TPA Version 4.1 codes were used only to conduct system-level analyses (Mohanty, et al., 1999; Mohanty, et al., 2002b). Several groups of staff from other key technical issues have exercised various versions of the TPA code to improve their understanding of the risk significance of specific features, events, and processes. In addition, sensitivity analyses conducted by Total System Performance Assessment Integration staff and staff from other key technical issues have used the TPA code to perform process-level sensitivity analyses and specialized analyses to support review comments on the Total System Performance Assessment–Viability Assessment (CRWMS M&O, 1998) and Total System Performance Assessment–Site Recommendation (CRWMS M&O, 2001). In addition, an external review group evaluated the models used in the TPA code Version 3.2 code and the corresponding sensitivity analysis

report. These activities in combination have effectively subjected the TPA code to a fairly stringent examination of the TPA code's ability to produce reasonable results for the conditions of interest at Yucca Mountain. Additional scrutiny of the TPA code has been received by publishing the results from these studies in conference proceedings and in peer-review journals. In spite of all these activities, the validation results have not been formally documented. The goal of the TPA code validation test plan is to formalize the tests that will permit formal documentation of results in a manner that make users and reviewers confident that the TPA code computes correctly and is suitable for its intended application.

Because there is no single, generally accepted approach for validating software, such as the TPA Version 5.0 code, which simulates complex coupled natural processes over periods of tens of thousands of years, the validation tests will be conducted in phases so that knowledge gained in the execution of earlier phases can be used to improve the robustness and utility of the tests performed in the later phases. Three different test phases will be conducted to complete the validation exercise. Phase I focuses primarily on function tests that will provide confidence that the software computes correctly. Phase II involves function-level tests that provide confidence that the models are providing sensible results. Phase III involves comparison of the overall results for the TPA Version 5.0 code with other system codes. All Phase I activities and most of Phase II activities will be completed by the time of the final delivery of the TPA Version 5.0 code to the NRC in July of 2003. Phase III activities will be initiated after July 2003.

1.2 Scope of Validation Tests

This plan outlines two broad classes of software validation tests: (i) function tests and (ii) system tests—the function tests are further divided into those for (i) the TPA executive, (ii) the consequence modules, and (iii) the stand-alone codes.

1.2.1 Function Tests

Function tests evaluate the major components of the TPA code to demonstrate that the selected components produce the correct output, given specified input. For instance, validation of an algorithm that simulates the equations for radionuclide ingrowth and decay would require data on half-lives and initial inventories of radionuclides and involve an independent computational method to determine whether time-dependent inventories are properly calculated in the TPA code.

The TPA Version 5.0 code consists of an executive with several general purpose utilities and a set of consequence modules, some of which spawn the execution of external codes used to simulate specific physical processes.

- The executive provides the probabilistic simulation framework. It samples values of input parameters from specified probability distributions, directs the transfer of these input values and other intermediate data to the consequence modules and controls the generation of output files. The executive also includes a set of utilities that perform functions, such as creating a sequence of input parameter values based on Latin Hypercube or Monte Carlo sampling strategies or calculating the intersection points of a line with a quadrilateral.

- The consequence modules simulate processes and events that could affect the performance of a disposal facility. They are invoked sequentially and use information supplied by the executive consisting of sampled values for input parameters and (possibly) intermediate data provided by a prior consequence module. The consequence modules also pass results back to the executive, such as information required by a subsequent module or estimates of total system performance.

- The TPA code makes use of several external or stand-alone codes that have been developed to simulate specialized processes, such as transport of radionuclides through variably saturated media. Each stand-alone code is invoked by a consequence module or the TPA executive but runs separately from TPA. Data are transferred through intermediate files. For example, a consequence module may first create a file containing the input data required by a stand-alone code and then read an output file created by the stand-alone code. Several stand-alone codes (e.g., NEFTRAN and GENII) are well-established computer codes. Other stand-alone codes were developed by the CNWRA and NRC staff. Some of these stand-alone codes have already undergone their own software validation tests and may not require additional testing. However, validation of the TPA Version 5.0 code will include examination of the modifications of the codes for TPA-specific applications. This many involve inspection of intermediate files to ensure their contents are as expected and that data are correctly transferred between the TPA code and the stand-alone codes.

1.2.2 System Tests

System tests deal with the software as a whole and combine traditional software validation with additional tests to assess the reasonableness of risk estimates obtained from the TPA code. The tests described here involve the use of TPA to replicate existing studies that have been thoroughly documented and reviewed. It is desirable to examine studies conducted by organizations in other countries to enhance the independent aspect of software validation. System tests will require adaptation of the input data for TPA to describe as closely as possible the conditions pertaining to an existing study. These conditions will be somewhat different from the conditions for which TPA has been developed; however, a successful software validation test will lend confidence in the ability of TPA to deal with safety assessments over long time frames.

1.2.3 Range of Input Data

TPA accounts for uncertainty in model parameters following a probabilistic approach. It simulates performance for multiple realizations in which a parameter may be assigned any value from its prescribed probability distribution function. Accordingly, function tests need to examine a full range of feasible input data to examine the robustness of the software as well as its accuracy.

In many instances, it may be sufficient to examine extreme and central values; however, parameter and state values that effect conditional states, which may invoke different internal algorithms, also need to be tested. Algorithms that rely on temporal or spatial discretization may need special care to ensure convergence occurs for all feasible combinations of input data.

In other instances, such as when an output variable is a function of many input parameters (some of which may be correlated), it may be necessary to examine a set of randomly sampled values. These cases include function tests of consequence modules which rely on results generated by one or more previous consequence modules. Such tests might need to examine random sets of results computed by the prior consequences modules.

Finally, the TPA Version 5.0 code will be tested to ensure that invalid data are properly identified and displayed. Some data values may be disallowed for physical reasons (e.g., negative half-lives for radionuclides), other data may be disallowed because of numerical restrictions imposed in the algorithm (e.g., time step size.)

1.3 Test Guidelines

1.3.1 Test Participants

To make the testing as objective as possible, validation tests should be defined by technically qualified staff who have not been not been directly involved with the development of the code being tested. Function tests will require participants with expertise in statistics, computer programming and numerical analysis. Function tests involving stand-alone codes may require specialists or access to specialized code or analytical tools. Each system test will require personnel familiar with the selected system studies. Staff familiar with the overall scope of the tests should review each test to ensure it is complete and comprehensive. These staff may also prepare summary reports.

1.3.2 Input Data and Simulation Results from TPA

All tests will require access to the input data used by TPA. Most tests will require access to intermediate or final results generated by TPA. TPA code simulation will be carried out by developers of the TPA code, following the specifications provided by those who developed the tests.

1.3.3 Documents Describing TPA

The primary source for the theoretical background and functional requirements, as well as the design of the TPA code is "Total-System Performance Assessment (TPA) Version 4.0 Code: Module Descriptions and User's Guide" (Mohanty, et al., 2002a). The Software Requirement Description (SRD) (Janetzke, et al., 2002) is the primary source of information on the changes made to the TPA Version 4.0 (Version 4.1j, in particular) which have led to the TPA Version 5.0 code. Other information may be taken from documents referenced in the user's guide.

In some instances, resolution of uncertainties and ambiguities might require direct examination of the TPA source code. Because this action may undermine the objectivity of the software validation test, it should be noted in the test report.

1.3.4 Acceptance Criteria

The credibility of a software validation test will be enhanced if criteria are defined a *priori* as to what constitutes success. A test will be considered successful if these criteria are met or if discrepancies can be fully explained.

Acceptance criteria may be qualitative or quantitative depending on the nature and purpose of the test. An example of a qualitative criterion is conformance of the source code to established good programming standards. An example of a quantitative criterion is the relative magnitude of the difference obtained by comparing the numerical results from an independent source to the results produced by the TPA code. For this software validation exercise, a difference of 10 percent or less is generally acceptable unless otherwise stated. This criterion may be relaxed if rounding errors in the calculated numbers from the independent source are excessive.

Most of the function tests will use quantitative criteria. The system tests will involve only qualitative criteria because of differences in waste disposal concepts and processes simulated, as well as in the associated codes used to assess the systems. Examples of qualitative criteria include (i) agreement in the relative effectiveness of the engineered barriers, (ii) typical time delays offered by the geosphere transport pathways and (iii) the identification of key radionuclides and important biosphere pathways.

1.3.5 Test Reports

Each test will be identified in a software test form and the results of the test will be described in a self-contained document, preferably following a common layout. These documents will be attached to the software test form and will be submitted to quality assurance records. Each test report will include the following items:

- The title of the test with date and participants
- Overall purpose with description of or reference to the underlying theory and reference to the sources of information used to conduct the test
- Overview of the scope of the test
- Description of (and justification for) the methodology used
- Statement on the expected results from TPA and criteria used to judge success
- Identification of input data used
- Detailed discussion of the implementation, analysis, and conclusions of the test

The conclusions should state clearly whether the test has met its criteria and whether additional testing is recommended. A summary report would provide an overview of the conclusions from the test reports.

1.3.6 Repetition of Tests

Tests with quantitative criteria should be designed to be easily repeatable to simplify retesting if an uncovered deficiency was subsequently corrected. In addition, repeatable tests will facilitate software validation of future versions of TPA.

A test can be made repeatable by augmenting the test report with a complete set of input data and results, copies of spreadsheets and related analytical tools, and references to other codes and their input files used in the test.

2 ENVIRONMENT

2.1 Software

The software to be tested in the validation procedure is the TPA Version 5.0 code, including the executive and all of its subroutines and function subprograms, as well as the following auxiliary codes: ASHPLUME, DSFAILT, EBSFILT, ENV, ENVIN, FAILT, MECHFAIL, NEFMKS, RELEASET, an SNLLHS. A complete listing of the executive subprograms appears in Appendix A. The programs will be compiled using the Sun Microsystems, Inc. SUNWspro FORTRAN Version 4.2 compiler.

The number of source code and include files used to construct the TPA Version 5.0 code exceeds 190 and the number of output files generated by the TPA Version 5.0 code and its associated auxiliary codes exceeds 45, therefore, the names of these files are not listed here. Each test participant will be provided with an addendum listing the names and locations of files relevant to the test that they are required to execute.

2.2 Hardware

All software validation tests will be performed on Sun Microsystems, Inc. computers using Sun Sparc processors running the SunOS Version 5.8 operating system. Specific computers to be used at the CNWRA include the Sun_Fire 280R system, which has 10 Sun ultra Sparc - III CPUs running at 750 MHz, and the Sun_Fire V880 system, which has 8 Sun ultra Sparc - IIIi CPUs running at 1000 MHz.

3 PREREQUISITES

All participants in this software validation procedure will be provided access to an official copy of the TPA Version 5.0 code. The test participant will compile the code using the make file supplied with the code, which invokes the Sun Microsystems, Inc. SUNWspro FORTRAN Version 4.2 compiler (*Makefile4.2*). The test participant must set the environment variables at the UNIX prompt as follows:

```
setenv TPA_DATA $HOME/current_code
setenv TPA_TEST $HOME/current_code
```

Where `current_code` is the directory containing the official copy of the TPA Version 5.0 code provided to the test participant.

4 ASSUMPTIONS AND CONSTRAINTS

There are a number of restrictions that all participants in the software validation testing will be required to adhere to. Currently, only four restrictions have been identified; however, it is likely that others will be identified once testing begins. Current software validation test restrictions include:

- TPA simulation times must be less than or equal to 100,000 years.
- Functional tests of utility and consequence modules will not be conducted in full Monte Carlo mode.
- Interface to the TPA code will be through ASCII text output files and not through graphs or plots.
- No software validation testing will be conducted using Intel-based personal computers.

5 TEST CASES

The function tests described in Sections 5.1, 5.2, and 5.3 and the system tests described in Section 5.4 below cover most aspects of the TPA Version 5.0 code. These tests address most potential software validation issues, but additional tests might be identified as the validation testing proceeds. Table 5-1 shows which tests will be conducted in each phase of the validation testing.

Table 5-1. Summary of Validation Tests

Test-Types		Validation Phases		
		Phase I	Phase II	Phase III
Function Tests	TPA Executive	Computational Correctness		
	Consequence Modules	Computational Correctness	Physical Reasonableness	
	Stand-Alone Codes	Computational Correctness	Physical Reasonableness	
System Tests				Physical Reasonableness

Each software validation test is focused on a specific function, but there are elements common to most tests. These common elements include:

- The TPA code consists of a main program and many subroutines with data transferred through intermediate files and FORTRAN CALL and FUNCTION statements. The software validation test of each function should include detailed examination of all data transfers. The tests should ensure that the expected format of intermediate files is consistent with modules that create, read or change the file, and that CALL and FUNCTION statements reference the desired subroutine and have the correct sequence of arguments.

- All assignment and logical statements should be examined for consistency of dimensions. For instance, in a simple assignment such as A = B x C, the dimension of A should equal the product of the dimensions of B and C.

- For improved coverage and efficiency, some software validation tests might require the development of simple code drivers called stubs that exercise particular functions or subfunctions. Stubs would typically consist of FORTRAN statements that call a TPA subroutine with a set of appropriate arguments. Values for these arguments might be varied to examine ranges and combinations of feasible values. For instance, stubs would be effective in testing the subroutines associated with the ARRAY utility.

5.1 Test Cases for the TPA Executive

The software validation tests described in this section will examine the main underlying functions of the executive. The TPA user's guide identifies six primary utilities (READER, SAMPLER, INVENT, MODULE VARIABLE, SUBAREA and ARRAY) and five secondary utilities (FILEUNIT, FINDELEV, NUMRECIP, PEAKFINDER and RAN) associated with the executive. The following discussion provides a brief description of the purpose of these modules and outlines the scope of software validation tests. Table A–1 in appendix A provides additional information on the details of these tests. Table 5-2 identifies whether the requirements are qualitative or quantitative.

Table 5-2. Validation Tests for the TPA Executive

Test ID	Test Description	Effort (Person—Months)	Criteria for Success
E1	READER • input checks • drift calculator	0.5	 qualitative quantitative
E2	SAMPLER • distribution sampling	0.25	 qualitative
E3	INVENT • data checks • inventory calculations	0.5	 qualitative quantitative

Table 5-2. Validation Tests for the TPA Executive			
Test ID	**Test Description**	**Effort (Person—Months)**	**Criteria for Success**
E4	MODULE VARIABLE • security features	0.25	qualitative
E5	SUBAREA • data checks • calculations	0.5	qualitative quantitative
E6	ARRAY • used correctly • calculations	0.75	qualitative quantitative
E7	FILEUNIT • inspect code	0.25	qualitative
E8	FINDELEV • calculations	0.25	quantitative
E9	NUMRECIP • calculations	0.25	quantitative
E10	PEAKFIND • peak dose time and magnitude	0.25	quantitative
E11	RAN • random numbers without cycling • extraneous code	0.25	quantitative qualitative
E12	IAREADER • overwrites sampled values	0.25	qualitative
E13	EXEC (CHECKPOINT RESTART) • maintenance check.pnt • execution resumed when required	0.25	qualitative qualitative

Most of these utilities have been carried over without change from Versions 3.1.4, 3.2, and 4.0 of TPA, and most have undergone a number of formal or informal tests, which are described in Appendix A. Consequently, it may not be necessary to devise new tests, but merely repeat and document the tests that were applied to earlier versions.

5.1.1 READER Primary Utility Module (E1)

READER is the only subroutine allowed to access the main input file, *tpa.inp*. READER examines the data in *tpa.inp* to trap potential errors and then stores the data into various target variables and arrays. READER also includes a drift calculator that computes the endpoint coordinates of the drifts in the repository. Sample software validation tests will ensure the following:

- Adequacy of error traps including those for global input variables
- Proper assignment of input data to intended targets
- Validity of probability density functions (e.g., ensure nonnegative lower bound for parameters physically limited to non-negative values)
- Accuracy of the drift calculator
- There are no assignment statements that could modify the input data assigned to the target variables and arrays

Most of these tests would likely involve inspection of code and execution of TPA with a variety of (valid and invalid) input data and the acceptance criteria would be qualitative. Tests of the drift calculator might involve use of a spreadsheet, and it is expected the acceptance criteria would be quantitative. These tests would require personnel with a background in FORTRAN coding, and a thorough testing could take approximately two weeks of effort.

5.1.2 SAMPLER Primary Utility Module (E2)

SAMPLER assigns sampled values to variables characterized by probability density functions for each realization. It supports Latin Hypercube and Monte Carlo sampling strategies for 13 different distribution functions. The Latin Hypercube strategy is based on a well-known code developed by Iman and Shortencarier (1984) and is invoked in TPA in a stand-alone subutility called *snllhs.f*.

The functions provided by SAMPLER are crucial to the probabilistic framework, and thorough software validation testing is warranted. The tests will ensure the following:

- Distribution functions have the correct attributes. (e.g., does a large number of realizations yield the expected probability curve for each distribution?)
- The correlation algorithm performs correctly for different combinations of distribution functions
- The correct input is provided to *snllhs.f*. (e.g., invalid cases are disallowed)
- *snllhs.f* produces the expected Latin Hypercube samples for all feasible combinations of the number of random variables and desired sample sizes
- The random number generator produces a sequence of random numbers with the desired properties and with a period that is sufficiently long to preclude the possibility of recycling

Many of these tests would involve independent implementation, using a spreadsheet for example, and the acceptance criteria would be quantitative. Some tests, such as examination of the output from *snllhs.f*, might involve visual examination of plots of sampled parameter values for various sample sizes. These tests will require personnel with a knowledge of statistics. Problems with SAMPLER have mostly been rectified during previously conducted sensitivity studies. Additional tests may take approximately one week of effort.

5.1.3 INVENT Primary Utility Module (E3)

INVENT computes and stores radionuclide-specific information for use throughout TPA. Section 3.3.2 in Mohanty, et al., (2002a) describes the governing equation and solution for computing inventories for 43 radionuclides of interest. Software validation tests will ensure the following:

- Correct association of radionuclide identity and radionuclide properties for different combinations of selected radionuclides
- Radionuclides can be selected in any order
- The same radionuclide can appear twice in two different chains
- Correct inventory calculation for all feasible combinations of half-lives
- Correct inclusion of colloidal radionuclides

Tests on input information can be made with runs of TPA using a modified input file, while inventory calculations can be made using conventional mathematical software. These tests should take approximately two weeks of effort by a numerical analyst.

5.1.4 MODULE VARIABLE Primary Utility Module (E4)

MODULE VARIABLE subroutine stores selected results computed by consequence modules into a database. The results are identified and accessed by a special index. Values can only be stored by the consequence module that generates the result although other consequence modules can scan the data.

Software validation tests will examine the measures used to ensure that these security functions operate as intended. Failure could occur if a consequence module uses an incorrect special index to store its own data or to retrieve data generated by itself or another consequence module. Failure could also occur if the database becomes corrupted by an invalid storage procedure.

Software validation tests will include code review of MODULE VARIABLE module and the consequence modules to determine whether the correct sequence of arguments and expected values of special indices are used throughout. Assignment statements affecting special indices will be checked to ensure that all new values are unique. Finally, the structure and integrity of the database will be evaluated to determine whether records can become corrupted, for instance, by overwriting part of an existing record.

Tests of MODULE VARIABLES is expected to require about one week of effort by an experienced FORTRAN programmer.

5.1.5 SUBAREA Primary Utility Module (E5)

SUBAREA has functions that involve calculation of repository subarea information for use in the consequence modules FAULTO, SZFT, READER and VOLCANO (Appendix C, Mohanty, et al., 2002a). The main input data is the number and coordinates of the subareas and a parameter describing the areal loading of waste packages.

Details concerning the data transformations must be determined. Software validation tests of SUBAREA will be qualitative and quantitative and take approximately two weeks of effort by a numerical analyst to ensure the following:

- The expected data are correctly provided by READER
- There are no unstated input data restrictions (e.g., coincidence of two lines defining the quadrilateral)
- Calculations are valid for all allowed input combinations (e.g., many small subareas)
- Calculations of elements (points, circles and lines) within a specified subarea are performed correctly.

5.1.6 ARRAY Primary Utility Module (E6)

ARRAY collects together 20 subroutines that perform various data transformations (Table C–2, Mohanty, et al., 2002a). Software validation tests of these subroutines will use stubs aimed at the specified functions performed. Development of the code and subsequent software validation tests would require approximately three weeks of effort by a programmer-analyst.

5.1.7 FILEUNIT Secondary Utility Module (E7)

FILEUNIT assigns unique unit numbers to all modules in TPA. Software validation tests will involve code review to ensure that all assignments are distinctive and reference the correct file name. In addition, tests will be conducted to ensure that code modules requesting a unit number are entitled to retrieve a unit number from FILEUNIT and that sufficient unit numbers are available for the code to execute. Approximately one week of effort is required by a programmer.

5.1.8 FINDELEV Secondary Utility Module (E8)

FINDELEV provides elevation of the ground surface for a specified set of universal transverse mercator coordinates. The governing equations must be determined, and software validation will involve independent calculations using a spreadsheet or development of stub code. Approximately one week of effort is required by a programmer-analyst.

5.1.9 NUMRECIP Secondary Utility Module (E9)

NUMRECIP is a collection of four numerical algorithms, taken from a well-known source, that have been adapted for use in TPA (Table C–3, Mohanty, et al., 2002a). Software validation will involve independent calculations using a spreadsheet or development of stub code, depending on the nature of the algorithm being tested. These tests will require one week of effort by a programmer-analyst.

5.1.10 PEAKFIND Secondary Utility Module (E10)

PEAKFIND determines the time and magnitude of peak doses for individual radionuclides and for the total dose summed over all radionuclides simulated, presumably from a set of dose-time points. Software validation will involve inspection of the dose-time data or curves or use of stub code. About one week of effort by an analyst is required.

5.1.11 RAN Secondary Utility Module (E11)

RAN, which is based on the same pseudo-random number used by SAMPLE, generates random values for use in generating Monte Carlo samples and a seismic hazard curve. The software validation test will ensure that the random number generator produces a sequence of random numbers with the desired properties and a period that is sufficiently long to preclude the possibility of recycling. About one week of effort by an analyst will be required.

5.1.12 IAREADER Secondary Utility Module (E12)

A suite of secondary utility modules (identified in the TPA code as ZPORTUNX/ZPORTPC) are designed to facilitate SunOS implementation. However, the personal computer implementation resulted in a substantial number of changes to the IAREADER module. This module reads the input file *ia.dat*. This input file controls the parameter values for importance analysis runs. The file *ia.dat* is only used when the importance analysis flag is set in the *tpa.inp* file. If this flag is set the sampled values from Latin hypercube sampling for a parameter are overwritten by the values in *ia.dat*. The tests will ensure the following:

- Detection of incorrectly formatted *ia.dat*. file
- Detection of illegal parameter names
- Detection of parameter values outside their physical limits
- Detection of duplicate names
- Input values are correctly written to the *lhs.out* file

Most of these tests would involve the execution of TPA with a variety of (valid and invalid) input data, and the acceptance criteria would be qualitative. Approximately one week of effort from a programmer-analyst will be required.

5.1.13 Checkpoint-Restart Module (E13)

Checkpoint-restart is a feature added to the EXEC module that allows the TPA execution to be interrupted and then subsequently resumed. Testing will be conducted to verify that the checkpoint-restart feature allows the code to be resumed when required. Much of the functionality associated

with the checkpoint restart feature is associated with writing a file, *check.pnt,* to disk. Testing will reveal any limitations this file places on the execution speed of the code along with any additional disk space requirements for code execution. Approximately one week of effort from a programmer-analyst will be required.

5.2 Test Cases for the Consequence Modules

This section describes a set of software validation tests designed to evaluate the computational correctness of the consequence modules. In addition, a limited number of model validation tests are defined for the consequence modules that will examine the reasonableness of the abstracted models. These tests are summarized by identification code in Table 5-3.

5.2.1 UZFLOW (C1)

UZFLOW simulates shallow infiltration at the land surface, and percolation of water to the repository horizon, where it affects the near-field environment of the repository. Some of the water that reaches the repository horizon may contact and promote corrosion of the drip shield and waste packages, and subsequently dissolve and transport radionuclides.

The UZFLOW module consists of three sub-models. The first sub-model employs a simple climate model to estimate how the mean annual temperature (MAT) and meal annual precipitation (MAP) vary as functions of time. The second sub-model estimates shallow mean annual infiltration (MAI) as a function of time and location using the time-varying MAT and MAP estimates provided by the first sub-model. The second sub-models also uses information on land surface elevation, soil depth, soil type, rock type, wind speed, and long-wave radiation that is provided by externally-generated input files. The third sub-model simply computes the spatial average of the MAI estimates provided by the second sub-model to produce the time-varying flow rate of infiltrating water moving toward the repository for each repository sub-area. This flow rate does not account for thermal effects from repository generated decay heat. Thermal effects are treated in the NFENV module. The deep percolation is assumed to be the average value of MAI within a subarea.

Software validation of the UZFLOW module will involve an independent implementation (e.g., spreadsheet calculation) of the module and the stand-alone code ITYM used in the second sub-model. Rudimentary model validation tests will compare infiltration estimates from the UZFLOW module to results from other well-established computer codes or field data.

5.2.1.1 Software Validation Tests

- Hand calucation checks will be made of the following: (i) time interpolation of the climate data supplied in the *climato2.dat* file; (ii) calculation of MAP and MAT from the climate function and input data; and (iii) perturbation calculations for MAP and MAT over one time interval. The hand calculation should match those values computed by UZFLOW (C1-1).

- Estimates produced by the infiltration response surface function, which is constructed and evaluated by the ITYM stand-alone pre-processor, will be compared to simulation results from the BREATH computer code (Stothoff, 1995), for several specific cases, including the base case modern climate. The BREATH computer code, which has already undergone formal software validation at the CNWRA (Fedors, 2002), will be used because simple analytical solutions are not readily available to implement the net infiltration conceptual

model used in the TPA code. For specified values of MAP and MAT, the infiltration response surface, when evaluated at a specific spatial location and time, should provide an estimate of MAI that is within 10 percent of that estimated by BREATH using long-term average climate values (C1-2).

- UZFLOW estimates subarea average net infiltration from ITYM output at various spatial resolutions (e.g., 30, 60, or 120-meter square pixels). Net infiltration maps with known subarea net infiltration values can be used to confirm the subarea-averaging process. Net infiltration estimates for each pixel within each subarea will be set to a uniform value with different values for each subarea. If the calculated subarea averages are the same as the uniform value specified at each pixel, the test will be considered successful (C1-3).

- The net infiltration results at large scale (i.e., mountain scale) will be compared against results from more general approaches, such as the Chloride Mass Balance method on analog watersheds. There are several analog watersheds in southern Nevada, such as the 3-springs watershed (Lichty, 1995), with data for which data is available. The test will be considered successful if the results will be within an order of magniture of the USGS results (C1-4).

UZFLOW validation tests will require about three weeks of effort.

5.2.2 NFENV (C2)

NFENV simulates the near field environment around the repository and waste packages for each subarea. It provides estimates of temperature, relative humidity, pH, and concentrations of chloride and carbonate and requires input from a variety of sources. The heat transfer calculations in NFENV yield estimates of the temperature rise in the rock above the repository, the temperature at the inner and outer surfaces of the waste package and the maximum temperature of the spent fuel. The relative humidity is determined from temperature estimates. NFENV also estimates the time-dependent infiltration rate of water in the near field. Most of these functions are relatively detached from each other, and it may be effective to perform several separate functions tests.

5.2.2.1 Software Validation Tests

- The governing equations used to calculate the temperature of the wall of the drift, the waste package outer surface and waste package inner surface temperatures, and the refluxing of water resulting from near-field thermohydrologic conditions will be compared to results using analytical solutions implemented in MATHCAD® or by Excel®. Conditions representing upper and lower limits of temperature and relative humidity will be used to test the thermal and reflux models. Computed results from the TPA code and the analytical solutions should differ by no more than 10 percent (C2-1).

- To investigate model adequacy, (i) the effect of the fixed repository surface boundary condition on the temperature estimates will be qualitatively verified using alternative boundary conditions in the detailed numerical model, (ii) the impact of hydrology coupling on the temperature estimates will be qualitatively investigated, and (iii) the repository edge temperature estimated by the abstracted model will be qualitatively compared with numerical simulation results (C2-2).

- Because temperatures are calculated separately from chemical species and liquid responses, the integration of the three will be evaluated. For example, the concentrated chloride environment at the waste package should not be diluted until temperatures reach a level that can support aqueous flow into the drift (C2-3).

- pH values provided by the NFENV module to EBSFAIL and EBSREL will be verified for consistent use in the latter two modules. The pH values will also be verified for consistency with the chemistry used in the determination of deliquescence relative humidity (C2-4).

NFENV validation tests will require approximately five weeks of effort.

5.2.3 EBSFAIL (C3)

The EBSFAIL module is used to compute the failure time of the waste package due to humid air corrosion, general corrosion, and localized corrosion. Inputs to the code are relative humidity, temperature and chloride concentration as functions of time and the failure time of the drip shield.

5.2.3.1 Software Validation Tests

- EBSFAIL prepares the input data used in the stand-alone code, FAILT, which performs most of the waste package degradation computations. The correctness of data transfer between EBSFAIL and FAILT will be verified (C3-1).

- The FAILT algorithm used to compute the penetration depth of the corrosion front and the failure time will be implemented using software such as Mathematica®. Values of the corrosion potential, critical potential for localized corrosion, and penetration depth of the corrosion front as functions of time computed with the TPA code (for one realization and one repository subarea) will be compared to computations in Mathematica®. The computation of the critical potential for localized corrosion will be checked to ensure that results are computationally correct. Synthetic input data will be used to verify that different degradation modes (i.e., humid air corrosion, general corrosion, and localized corrosion) are properly activated. Differences in the TPA and Mathematica® computations of penetration rate, failure time, erosion potential, and critical potential should be less than 10 percent (C3-2).

- For a Monte Carlo run of the TPA code where localized corrosion does not occur, the distribution of waste package failure times can be approximately computed by

$$t_i = \frac{d}{CR_i} \qquad [6\text{-}1]$$

where

d — thickness of the outer, corrosion resistant waste package wall
CR_i — general corrosion rate sampled for realization i

The general corrosion rate is an input parameter that is sampled within the TPA code from a specified distribution function. The ensemble of failure times computed using Eq. [6-1] should be comparable to the more complex method used within the TPA code to compute the distribution of failure times, provided that localized corrosion does not occur.

Comparison of the resulting failure time distributions (i.e., failure time CDFs) should not result in more than 5 percent difference at any quantile (C3-3).

- Parameters derived from experimental data for different materials such as Type 316 SS, and Alloys 825, 625 and 22 will be used to check if the repassivation potential computed by the TPA code follows the expected trend (C3-4).

- Potential versus pH diagrams will be prepared numerically that include the effects of variations in chloride concentrations, to define domains of localized corrosion and uniform corrosion. Variables such as oxygen partial pressure and temperature will be varied to verify that the expected degradation modes are properly identified by the TPA code (C3-5).

EBSFAIL validation tests will require about five weeks of effort.

5.2.4 DSFAIL (C4)

DSFAIL is used to compute the failure time of the drip shield by general corrosion. DSFAIL prepares the input file for the stand-alone code DSFAILT, which is used to perform drip shield corrosion computations for each subarea. Inputs to the code are the corrosion rate and fluoride concentration as a function of time. The computed drip shield thickness as a function of time is passed to the executive which, in turn, passes it to the SEISMO2 module.

The general corrosion model for the drip shield is straightforward. The failure time is computed as in Eq. [61], with an enhancing factor affecting the corrosion rate that is function of the fluoride concentration. Documentation on the effect of fluoride on Ti grade 7 corrosion rate has been documented elsewhere (e.g., Brossia et al., 2001).

5.2.4.1 Software Validation Tests

- Data transfers between DSFAIL, DSFAILT, the executive, and SEISMO2 will be verified (C4-1).

- The computational algorithm used in the stand-alone module DSFAILT will be implemented in Mathematica®. Synthetic input data (e.g. fluoride concentration versus time) will be used to test the response by DSFAILT to a variety of conditions. Differences between DSFAILT and Mathematica® computations for the failure time of the drip shield should be less than 10 percent (C4-2).

DSFAIL validation tests will require about two weeks of effort.

5.2.5 WELDFAIL (C5)

WELDFAIL is a subroutine invoked by the FAILT stand-alone module. WELDFAIL calculates the failure time of the closure weld area of the waste package due to humid air, general, and localized corrosion. WELDFAIL returns the weld failure time to FAILT which then passes it back to the TPA Executive.

5.2.5.1 Software Validation Tests

- Data transfers between WELDFAIL, FAILT, and the executive will be verified by code inspection (C5-1).

- The algorithm used by the WELDFAIL module to compute the failure time of the closure weld area is similar to the algorithm implemented in FAILT to address corrosion of the body of the waste package. The same test used to evaluate waste package corrosion (C3–2) computations will be used to evaluate corrosion computations of the closure weld area. An implementation in Mathematica® of the corrosion model will be used to evaluate the WELDFAIL implementation. Differences in the computed failure times should be less than 10%. Only a few tests will be performed to verify weld corrosion computations, given the similarity of WELDFAIL to FAILT (C-5).

WELDFAIL validation tests will require about two weeks of effort.

5.2.6 SEISMO (C6)

The SEISMO module calls the MECHFAIL stand-alone code, which currently estimates the number of drip shield failures caused by static and dynamic rock fall loads as well as by seismicity. MECHFAIL uses various abstractions to estimate (i) time varying static rockfall loads, (ii) size and fall height of discrete rock blocks dislodged from the drift ceiling during seismic events, (iii) time of occurrence and magnitude of seismic events, and (iv) the potential failure of the drip shield from static and dynamic rock fall loads.

Currently, the MECHFAIL module has place holders for the model abstractions that approximate the number of waste package failures caused by direct seismic shaking and potential interactions with the drip shield caused by rock fall loads. These abstractions for waste package failure[1], however, have yet to be developed and are, therefore, not part of the TPA Version 5.0 code.

Process-level tests have already demonstrated that the drift failure fraction and drip shield failure fraction calculated within the MECHFAIL module correspond to expected results. System-level tests that were performed verify that the MECHFAIL module will (i) return all drip shields failed over all time steps when the drip shield thickness calculated in DSFAIL is zero, (ii) return drift failure fractions that are reasonable (and passed onto the SEISMO and EXEC modules without error), and (iii) return the same results (i.e., drift failure fraction and drip shield failure fraction) for the first 10,000 years for simulation times of 10,000 and 100,000 yrs. No additional tests of the SEISMO/MECHFAIL module will be performed.

5.2.7 EBSREL (C7)

EBSREL simulates the release of radionuclides from failed waste packages. The stand-alone code RELEASET is interfaced with the TPA executive via EBSREL. EBSREL also controls execution of the EBSFILT stand-alone code used to simulate radionuclide transport through the invert upon

[1]Gute, G.D., G. Ofoegbu, F. Thomassy, S. Hsiung, G. Adams, A. Ghosh, B. Dasgupta, A. Chowdhury, and S. Mohanty. "MECHFAIL: A Total-system Performance Assessment Code Module for Evaluating Engineered Barrier Performance Under Mechanical Loading Conditions." CNWRA 2003-06. San Antonio, Texas: CNWRA. 2003.

which the waste package rests. The input data to computations in RELEASET are the waste package temperature, relative humidity, water flow rates into the waste package, and radionuclide inventories and half-lives. Two water contact modes are considered in RELEASET (bathtub and flow-through) and four possible empirical rate equations (the user may select only one equation) to simulate the dissolution of spent nuclear fuel. Gap and grain boundary inventories are assumed to dissolve as soon as water contacts the spent fuel. A model for glass dissolution is also implemented in RELEASET.

5.2.7.1 Software Validation Tests

- Numerous data transfers occurs between EBSREL, RELEASET, and the NFENV module used to estimate temperatures and flow rates of water into the waste package. Transfer of data between these modules and the executive will be verified by code and file inspection (C7-1).

- Computations of release rates will be performed in Mathematica®, MathCad®, or Matlab® for simple cases (e.g., constant flow rates, constant exposed spent fuel surface area, large inventory, long-lived radionuclides) and the output release rates will be compared to RELEASET computations. Simplified computations will focus on verifying mass balance and faithfulness to equations described in the TPA 4.0 User's Guide (Mohanty, et al, 2002a) and will consider only one radionuclide at a time. Differences in estimated release rates should be less than 10 percent. (C7-2).

- The implementation of solubility constraints will be evaluated to ensure that radionuclide concentrations do not exceed the solubility limit. Calculations will also be performed to evaluate the release rate of radionuclides with low solubility (C7-3).

- Release rates from the engineered barrier subsystem will be computed using different decay chains from those specified in the TPA nominal case. Study of decay chains with several members, different half-lives, and inventory will highlight the response of EBSREL and RELEASET to various limiting cases. Qualitative correctness of release rate trends will be examined with respect to solubility limit, half life, inventory, and the radionuclide position in the decay chain (C7-4).

EBSREL validation tests will require about four weeks of effort.

5.2.8 EBSFILT (C8)

The EBSFILT stand-alone module is used to simulate radionuclide transport through the invert into the unsaturated zone below the repository. This code is interfaced with the TPA executive via EBSREL. EBSFILT uses a straight forward transfer function solution for one-dimensional advective-diffusive transport of radionuclides in a semi-infinite domain under steady-state flow conditions.

5.2.8.1 Software Validation Tests

- The principal input data to EBSFILT are the radionuclide release rates computed by RELEASET. Data transfer between EBSREL and EBSFILT will be verified by code inspection (C8-1).

- The transfer function approach will be implemented using software such as Mathematica. Release rates computed using Mathematica® will be compared to EBSFILT release rates. Differences in predicted outflow specific activities should not exceed 10 percent. Synthetic input release rates and parameters will be used to explore the behavior of EBSFILT under particular cases (e.g., high and low flow rates and diffusion coefficients, and short and long travel paths, short and long-live radionuclides, low and high retardation). Only one radionuclide at a time will be considered for these cases (C8-2).

- A mass conservation test will be conducted by integrating the input release and output release rates over time. If radionuclide decay is negligible, the total mass entering the invert must equal the total mass leaving the invert. If the decay rate is not negligible, the time integrals can be corrected to account for decay. Such tests will establish if there is any additional mass loss in the control volume. Only one radionuclide at a time will be considered in these tests. Mass differences should be less than 1 percent (C8-3).

EBSFILT validation tests will require about four weeks of effort.

5.2.9 UZFT (C9)

The UZFT module simulates the aqueous transport of radionuclides through the unsaturated zone from the repository to the water table for each repository subarea. The main function of UZFT is to perform simple data transformations, such as calculating retardation factors, and to prepare input files for use by the stand-alone code *nefmks.*

5.2.9.1 Software Validation Tests

- The algorithm used to compute transport velocities in the unsaturated zone will be checked using hand calculations or a Mathematica® script for several percolation rates that span the expected range. This test will be considered successful if the UZFT module correctly determines whether water flow is in the matrix or fracture continuum for each hydrostrati-graphic unit, if the saturation is computed correctly, and if the resulting velocity is correct computed (C9-1).

- Spreadsheet or similar calculations will be made to verify conversion from K_d data to R_d values used for retardation in the matrix. This test will be successful if the calculations are within 1 percent or less (C9-2).

- TPA colloid transport results will be compared against analytical results to test, for example, whether the abstracted model in the TPA code is consistent with (e.g., within ± 10 percent), or at least conservative with respect to, analytical model that account for non-equilibrium colloid attachment. TPA results will also be compared with available data from the DOE field (e.g., C-Wells for retardation factors) and laboratory experiments (e.g., LANL lab data

on Pu). Data used directly to develop analytical model or to derive parameters, will not be included in the validation test (C9-3).

UZFT validation will require about four weeks of effort.

5.2.10 SZFT (C10)

The SZFT module simulates the transport of radionuclides in groundwater through saturated zones between each repository subarea to the location of a potential receptor who could intercept contaminated groundwater. The main functions of SZFT are to perform simple data transformations, such as calculation of retardation factors, and to prepare input files for use by the stand-alone code *nefmks*.

5.2.10.1 Software Validation Tests

- Connections or links between the stand-alone data preparation code *nefmks* and the TPA code will be verified (C10-1).

- Spreadsheet or simple decay model calculations will be performed to bound reasonable activity levels of radionuclides for calculated transport times. A simple mass (or more accurately, activity) balance will be done for unretarded radionuclides. Activity levels should be within 10 percent, while mass balance should be within 1 percent (C10-2).

- To check that groundwater travel times from SZFT are correct, the porosity and streamtube width multiplier values will be varied in the input file to ensure that the groundwater travel times output changes proportionally (C10-3).

- The saturated zone streamtube approach will be verified against the three-dimensional site-scale numerical model using MODFLOW. The test is expected to provide a reasonable alternative conceptual model for saturated zone flow. A milestone report on the MODFLOW analysis of the abstracted SZ model in the TPA code will be delivered as an Unsaturated and Saturated Flow Under Isothermal Conditions Key Technical Issue deliverable in April 2003. In this effort, a set of *streamtube.dat* files have been developed that represent reasonable alternative conceptual models for saturated zone flow (C10-4).

- For colloid transport in saturated zone, tests similar to C9-4 for UZFT, will be carried out (C10-5).

SZFT validation will required about six weeks of effort.

5.2.11 DCAGW (C11)

DCAGW estimates the total effective dose equivalent to potential receptors who contact the contaminated water plume. The functions of DCAGW involve relatively modest data manipulations. The first step involves calculation of radionuclide concentrations in well water. These and other required data are passed to the stand-alone code GENTPA, which is derived from the quality-assured code called GENII (Napier, et al. 1988). GENTPA returns radionuclide-and pathway-specific annual intake rates, which DCAGW matches with radionuclide dose coefficients and concentrations to yield radionuclide-specific effective dose equivalents. Validation of the

GENII acquired software (Napier, et al. 1988) will be demonstrated using available documentation of benchmark and code comparison testing from peer reviewed journal articles and technical reports including an international model validation effort (International Atomic Energy Agency, 1995).

5.2.11.1 Software Validation Tests

- Verify that those portions of GENTPA that execute algorithms from the GENII code for DCAGW (i.e., *envin.exe* and *env.exe*) produce the same results as the acquired software (GENII code) (Napier, et al. 1988) when a similar set of input data relevant to the biosphere model implementation in the TPA code is used. Data manipulations in GENTPA that convert the results of the GENII based algorithms to dose will be verified by hand calculation, possibly using a spreadsheet. Another verification test for DCAGW involves inspection of code and examination of data transfers to GENTPA (C11-1).

These tests will require about four weeks of effort.

5.2.12 FAULTO (C12)

The FAULTO module performs a simple calculation of number of disrupted waste packages based on faults of fixed length occurring at sampled times, locations, and orientations with sampled widths and displacements.

5.2.12.1 Software Validation Tests

- For a given set of constant fault parameters (location, orientation, and width) the number of affected waste packages will be calculated by hand, using the repository geometry, and quantitatively compared to the module result. This test will be performed twice, once for a northwest orientation and once for a northeast orientation (C12-1).

FAULTO validation will require about two weeks of effort.

5.2.13 VOLCANO (C13)

VOLCANO determines effects associated with igneous events that disrupt the repository. An igneous event always involves an intrusion that leads to waste package failures in each subarea for use in EBSREL. An igneous event might also involve volcanic eruptions, which ejects a mass of high-level waste into the air for use by ASHPLUMO. The geometric model can be used to mechanistically estimate the number of waste packages entrained in the eruption, or the user can manually input a distributions for this quantity. The number of waste packages failed due to magma intrusion is specified directly by the input of a distribution for this parameter in *tpa.inp*. During the TPA calculation, VOLCANO passes information on the timing of a volcanic event, mass of waste ejected by a volcanic eruption, and number of waste packages failures resulting from igneous intrusion.

5.2.13.1 Software Validation Tests

- The timing of the igneous event is specified as a finite exponential distribution in *tpa.inp*. By describing the occurrence of an igneous event as a Poisson process, separate calculations for sampling the timing of igneous events will be compared to those values passed by VOLCANO. Histogram plots for 10 time bins within the compliance period will be visually inspected for qualitative agreement (C13-1).

- For a given set of constant igneous parameters (dike angle, dike length, dike width, volcanic cone diameter) using the geometric model for extrusive volcanism, the mass of waste ejected during an eruption will be calculated by hand and quantitatively compared to the result passed by VOLCANO. This test is passed if the numerical differences are less than 1 percent relative to the VOLCANO result. This test will be performed twice by simultaneously setting the aforementioned igneous parameters to constants: (1) at their minimum values in *tpa.inp* and then (2) at their maximum values in *tpa.inp* (C13-2).

- This test will verify that the number of waste packages failed due to magma intrusion is not dependent on the geometric model and its parameters (dike angle, dike length, dike width, volcanic cone diameter). Assuming the geometric model is selected and a constant is used for the TPA input parameter related to number of waste packages failed due to magma intrusion, the values of the dike angle, dike length, dike width, volcanic cone diameter will be varied. This test is passed if the number of waste packages failed due to magma intrusion produced by VOLCANO is not numerically affected (C13-3).

- Using the distribution model, the mass of waste ejected during an eruption is specified as a beta distribution in *tpa.inp*. Separate calculations for sampling the mass of waste ejected for extrusive volcanism will be compared to those values passed by VOLCANO. Histogram plots for 10 bins of the ejected mass will be visually inspected for qualitative agreement (C13-4).

- The number of waste packages failed due to magma intrusion is specified as a lognormal distribution in *tpa.inp*. Separate calculations for sampling the number of failed waste packages will be compared to those values passed by VOLCANO. Histogram plots for 10 bins of the number of failed waste packages will be visually inspected for qualitative agreement (C13-5).

- Subarea 2 is specified for the location of the igneous event in *tpa.inp*. This test focuses on extrusive volcanism, due to its greater risk-significance when compared to igneous intrusion. The sensitivity of the extrusive doses to the subarea of the eruption will be investigated. This test is passed if the difference in dose from an eruption in any of the remaining 9 subareas is within a factor of two relative to the dose for an eruption in Subarea 2 (C13-6).

VOLCANO validation will require about six weeks of effort.

5.2.14 ASHPLUMO (C14)

ASHPLUMO simulates the airborne transport of volcanic ash contaminated with high-level waste and the subsequent accumulation of deposited ash and high-level waste as a function of position on the earth's surface. These data pertain to a single volcanic extrusion event and are used in ASHRMOVO and DCAGS to estimate doses resulting from exposure to radionuclides on the earth's surface. ASHPLUMO uses the *ashplume* stand-alone code (Table 5-4). The model for the transport and deposition of volcanic ash has been validated against measured ash thicknesses from actual eruptions with good agreement (Hill et al., 1998). ASHPLUMO calculates the areal deposition of both ash and high-level waste at the receptor location.

5.2.14.1 Software Validation Tests

- Data transfer of those parameters from ASHPLUMO to the stand-alone code *ashplume* and the transfer of results from *ashplume.f* to ASHPLUMO will be checked for correctness (C14-1).

- Simplified calculations will be performed to show that the entire mass of the tephra eruption is being accounted for in the TPA calculations. The deposit thickness at the receptor location will be calculated by hand for a deposition whose thickness decreases exponentially from the eruption source. Given a uniform distribution throughout the deposit, the high-level waste concentration in tephra will be compared with calculated distribution in TPA code. A mass-balance verification of the high-level waste will be performed to determine if any mass is gained or lost .This test is passed if the numerical differences in total mass are less than 1 percent relative to the TPA result (C14-2).

- To test the behavior of the fuel incorporation model, a very high value for the incorporation ratio (greater than 0.9) will be entered to reduce the incorporation of high-level waste into ash and to reduce the areal deposition of high-level waste. This qualitative test is passed if the areal deposition of high-level waste (g cm^{-2}), as reported in *ashplumo.rlt*, is significantly reduced for the very high value of the incorporation ratio (C14-3).

- The relationship between the areal deposition of ash and high-level waste will be investigated. For a fixed time of eruption, it is expected that a simple relationship exists between areal deposition (g cm^{-2}) of ash and high-level waste. This qualitative test is passed if a simple relationship can describe the differences in the areal mass deposition of ash and waste as reported in *ashplumo.rlt* (C14-4).

ASHPLUMO validation will require about four weeks of effort.

5.2.15 ASHRMOVO (C15)

ASHRMOVO calculates areal radionuclide concentrations (radionuclide activity per unit surface area) at the compliance point. The initial concentrations are given by the deposited masses of ash and radionuclide computed by ASHPLUMO. ASHRMOVO takes into account subsequent radionuclide removal by leaching, erosion, and decay and subsequent radionuclide addition by erosion from other locations to yield time-dependent areal concentrations at the compliance point. The governing equations are equivalent to those available in the INVENT utility, but with a potential

leach rate limit. The leach rate calculation is the same as in the GENII code (Napier, et al., 1988) and TPA biosphere calculations.

5.2.15.1 Software Validation Tests

- Correctness of calculations will be verified using an analytical tool, such as MathCad®. Differences in computed areal concentration should be less than 10 percent (C15-1).

- Additional information for model validation will be obtained from published literature (models used in ASHRMOVO are common) and comparisons will be made with other available leaching and erosion models (C15-2).

ASHRMOVO validation will require about three weeks of effort.

5.2.16 DCAGS (C16)

DCAGS estimates dose to a potential receptor from exposure to radionuclides on the ground surfaces. Estimates of external exposure are given as the product of the areal concentration produced by ASHRMOVO and ground surface dose conversion factors. The external dose conversion factors account for the fraction of the year the receptor is exposed. For the inhalation and ingestion pathways as well as for the groundwater pathway, two sets of dose conversion factors are calculated for each realization, one for the current biosphere and one for the pluvial biosphere. The choice of dose conversion factor must be consistent with climate change factors simulated in UZFLOW.

Estimates of inhalation exposure also start from the deposited masses of ash and radionuclides produced by ASHRMOVO and use the mass loading model to estimate radionuclide concentration in the air above the contaminated ash. These estimates include factors to simulate radionuclide losses from wind transport and other processes. Inhalation dose is proportional to radionuclide concentrations in air, with the proportionality constant given by the product of breathing rate, fraction of the year the recipient is exposed to the air, and an inhalation dose conversion factor.

Estimates of ingestion exposure start from the deposited masses of ash and radionuclides produced by ASHRMOVO. The GENTPA code is then used to calculate the amount of each radionuclide that moves from the ground surface through various pathways and is eventually ingested by the receptor. The ingestion exposures are given as the product of the activity of ingested radionuclides and an ingestion dose conversion factor. Intake and exposure calculation algorithms (for all pathways except inhalation) in DCAGS are duplicated in DCAGW therefore similar types of tests are planned.

5.2.16.1 Software Validation Tests

- Verify that those portions of GENTPA that execute algorithms from the GENII code for DCAGS (i.e., *envin.exe* and *env.exe*) produce the same results as the acquired software (GENII code)(Napier, et al. 1988) when a similar set of input data relevant to the biosphere model implementation in the TPA code are used. Data manipulations in GENTPA that convert the results of the GENII based algorithms to dose will be verified by hand calculation, possibly using a spreadsheet. Another verification test for DCAGS involves inspection of code and examination of data transfers to GENTPA (C16-1).

- Verify that the inhalation doses calculated in DCAGS are the same as those obtained by simple hand calculation (C16-2).

- Model validation for GENTPA calculations will be supported by GENII code validation and additional tests including comparison of environmental transport and dose calculations from the TPA Version 5.0 code to results from the RESRAD Version 6.1 code (Yu, et al., 2001). For this comparison the results are not expected to match exactly, but should provide results that are within an order of magnitude of those produced using the TPA Version 5.0 code. The RESRAD family of codes has been in use or development for more than 15 years and comparison with it should provide confidence in the results being produced by the TPA Version 5.0 code. The RESRAD Version 6.1 code is scheduled to undergo separate formal validation under CNWRA TOP–018 procedure to qualify the code for use during licensing reviews, including that for the proposed Yucca Mountain high level waste repository, making comparison to the RESRAD Version 6.1 code meaningful (C16-3).

DCAGS validation will require about three weeks of effort.

Table 5-3. Tests for the TPA Consequence Modules

Test ID	Test Description	Effort (person-months)	Criteria for Success
C1	UZFLOW • Basic calculations • BREATH Computer code • Areal averaging of infiltration • Chloride mass balance	0.75	 quantitative quantitative quantitative qualitative
C2	NFENV • Temperature and RH • Boundary and edge effects • Heat, chemistry, and flow • pH consistency	1.25	 quantitative qualitative qualitative qualitative
C3	EBSFAIL • Data transfer • Corrosion penetration depth • Distribution of failure times • Re-passivation model • Effect of chemical environment	1.25	 quantitative quantitative quantitative qualitative qualitative
C4	DSFAIL • Data transfer • Effect of fluoride on failure	0.5	 quantitative quantitative
C5	WELDFAIL • Data transfer • Weld corrosion model • Failure times	0.5	 quantitative quantitative qualitative

Table 5-3. Tests for the TPA Consequence Modules			
Test ID	**Test Description**	**Effort (person-months)**	**Criteria for Success**
C6	SEISMO (No tests)	NA	NA
C7	EBSREL • Data transfer • Mass balance and release rates • Solubility models • Release models	1.0	 quantitative quantitative quantitative qualitative
C8	EBSFILT • Data transfer • Convolution and transfer function • Mass balance • Diffusive/advective limit cases	1.0	 quantitative quantitative quantitative qualitative
C9	UZFT • Calculating transport velocity • Calculating Rd • Colloid transport model	1.0	 quantitative quantitative qualitative
C10	SZFT • Data transformations for nefmks • Activity balance • Travel times • Streamtube to 3D • Colloid transport model	1.5	 quantitative quantitative quantitative qualitative qualitative
C11	DCAGW • Data transformations • Dose calculations	1.0	 quantitative quantitative
C12	FAULTO • Number of failed waste packages	0.5	 quantitative
C13	VOLCANO • Poisson process timing • Check of erupted mass • Number of WP failures • Distribution of mass ejected • Distribution of WP failed • Dose per subarea	1.5	 qualitative quantitative quantitative qualitative qualitative quantitative

Table 5-3. Tests for the TPA Consequence Modules (continued)			
Test ID	**Test Description**	**Effort (person-months)**	**Criteria for Success**
C14	ASHPLUMO • Data transfer • Mass balance • Waste incorporation test • Areal deposition test	1.0	 quantitative quantitative qualitative qualitative
C15	ASHRMOVO • Concentration calculations • Leaching and erosion models	0.75	 quantitative qualitative
C16	DCAGS • Replication by GENII • Hand calculation of inhalation dose • RESRAD Version 6.1 code	0.75	 quantitative quantitative quantitative

5.3 Test Cases for the Stand-Alone Codes

The input to TPA includes results from external codes such as MULTIFLO and ITYM, and it has been assumed that these external codes have been or will be verified by staff from the respective key technical issues.

During each realization, TPA may invoke the stand-alone codes listed in Table 5-4. Detailed validation of these stand-alone codes is considered to be beyond the scope of the software validation of the TPA code. However, some validation tests are required to ensure that data transfer is correct. Function tests of the stand-alone components of TPA will ensure the following:

- The input file created by the TPA consequence module adheres to the format required by the stand-alone code
- The output file created by the stand-alone code adheres to the format required by the TPA consequence model
- The output information supplied by the stand-alone code is appropriate for the needs of the TPA consequence modules
- There are no numerical convergence problems that could affect the correctness of the output supplied by the stand-alone codes

The first two software validation tests of the input and output files will require a programmer and involve inspection of the code for the consequence modules and for the stand-alone codes.

The last two tests are more involved and would require personnel experienced with the capabilities and limitations of each stand-alone code. It is particularly important to insure such personnel are

Table 5-4. Validation Tests for the TPA Stand-Alone Codes			
Test ID	TPA Consequence Module and Referenced Stand-Alone Codes	Effort (Person—Months)	Criteria for Success
S1	EBSFAIL • failt.f	0.25	qualitative
S2	EBSREL • *releaset.f* (time dependent fow, fmult, and flow multiplication factors) • *ebsfilt.f*	1	qualitative qualitative
S3	UZFT and SZFT • *nefmks.f*	0.5	qualitative
S4	DCAGS, DCAGW • GENTPA	0.5	qualitative
S5	ASHPLUMO • *ashplume.f*	0.75	qualitative
S6	SEISMO2 • *mechfail.f*	–	–
S7	DSFAIL • *dsfailt.f*	0.1	qualitative
S8	SAMPLER • *snllhs.f*	0.25	qualitative

aware of the ranges and combinations of input data that could be supplied. The estimated level of effort for testing the stand-alone codes has not yet been established.

5.4 Outline of System Tests

Three system tests have been identified for software validation of TPA (Table 5-5). Two tests involve using TPA to replicate recent safety assessment studies of nuclear waste disposal carried out in Canada. Both of these assessments have been fully documented and have undergone rigorous national and international peer review. The three test, provisional at this time, involves using the TPA code to replicate results for a generic Japanese repository.

5.4.1 Comparison with the Canadian Safety Assessment

The Canadian Nuclear Waste Management Program completed two safety assessments in 1994 and 1996 on the concept for disposal of nuclear fuel waste. The two assessments formed part of an Environmental Impact Statement submitted to a federal review panel for public and technical review. Technical reviews were conducted by an independent group of experts appointed by the panel, staff from the Canadian nuclear regulators (the Atomic Energy Control Board) and other federal agencies, and by a review team established by the Nuclear Energy Agency.

The concept involves deep geologic disposal in (saturated) crystalline rock in the Canadian Precambrian Shield. The concept also makes use of engineered barriers that include a durable UO_2 waste form, corrosion resistant containers and a surrounding layer of bentonite clay. Two safety assessments, the Environmental Impact Statement and Second Case Study, dealt with different hypothetical implementations.

1. The Environmental Impact Statement study was based on a geosphere which had groundwater transit times of about 10^4 years and an engineered barrier system which included titanium alloy containers emplaced in boreholes drilled in the floor of the repository drifts. The Environmental Impact Statement made use of the safety assessment code called SYVAC3-CC3 and its documentation (Goodwin, et al., 1994) includes extensive coverage of intermediate results from a deterministic simulation.

2. The Second Case Study examined a less effective geosphere where the groundwater transit times were as short as a few decades. The more robust engineered barrier system included copper containers emplaced within the repository drifts. The Second Case Study made use of the safety assessment code called SYVAC3-PR4. Its documentation (Goodwin, et al., 1996) is focused on probabilistic results.

The common component of the codes used in the Environmental Impact Statement and Second Case Study, SYVAC3, has capabilities similar to the TPA executive and its utilities. The CC3 and PR4 are system models that have many of the features found in the TPA consequence modules and stand-alone codes. There are also some important differences; for instance, the characteristics of the disposal site and the engineered barrier systems are quite dissimilar, such that the CC3 and PR4 models do not consider transport through unsaturated media or volcanism.

Two system software validation tests (S1 and S2 in Table 5-5) are suggested in which the input data for the TPA Version 5.0 code would be modified to replicate as closely as possible the conditions pertaining to the two Canadian studies. These two tests would be limited to a single deterministic simulation. However, consideration should be given at the end of these tests on the feasibility and benefits of a probabilistic comparison.

A preliminary consideration of the differences and similarities between the TPA consequence modules and the CC3 and PR4 models indicates two general classes of modifications would be required. The first is a simple change to existing input parameter values for a simulated process; the second involves adjustments to parameters values so as to bypass or ignore a simulated process. Examples include the following:

- UZFLOW—simple change to precipitation and subarea data to yield equivalent flow rates through different repository subareas.

- NFENV—simple change to heat transfer and groundwater composition data and adjustments to ignore the ventilation and boiling (reflux) transients.

- EBSFAIL—simple change to Type 1 failures and adjustments to mimic (for S1) or ignore (for S2) Types 2 and 3 failures (for S1, some Type 3 failures might only require a simple change).

- SEISMO, FAULTO, VOLCANO—adjustments to bypass these processes.
  UZFLOW—simple change to precipitation and subarea data to yield equivalent flow rates through different repository subareas.

- NFENV—simple change to heat transfer and groundwater composition data and adjustments to ignore the ventilation and boiling (reflux) transients.

- EBSFAIL—simple change to Type 1 failures and adjustments to mimic (for S1) or ignore (for S2) Types 2 and 3 failures (for S1, some Type 3 failures might only require a simple change).

- SEISMO, FAULTO, VOLCANO—adjustments to bypass these processes.

- EBSREL—simple changes to input parameters.

- UZFT, SZFT—simple changes to most input parameters with adjustments to ignore matrix diffusion

- DCAGW, DCAGS—simple changes to input parameters.

- ASHPLUMO, ASHRMOVO—adjustments to bypass these processes. Simple changes for these modules and for VOLCANO could be made to mimic the airborne exposure routes in CC3 and PR4, but those exposure routes are relatively unimportant in the Environmental Impact Statement and Second Case Study.

The S1 test could involve comparison of container failure rates (for uniform and crevice corrosion and for defective containers), radionuclide release rates from the waste package, disposal vault and saturated zone, and exposure routes affecting the critical group. The S2 test could involve comparison of radionuclide release rates from the disposal vault and saturated zone (the conditions for this saturated zone more closely resemble the conditions described in the TPA input file), and exposure routes affecting the critical group. The acceptance criteria for system tests S1 and S2 would be largely qualitative in nature. For instance, there should be agreement in the effect of the geosphere in delaying radionuclide transport and in the identity and exposure routes of radionuclides contributing to dose, quantitative comparisons might be feasible for some intermediate calculations such as radionuclide release from the invert.

Table 5-5. System Validation Tests

Test ID	Test Description	Effort (Person—Months)	Criteria for Success
S1	Modify TPA to replicate SYVAC3-CC3	2	qualitative
S2	Modify TPA to replicate SYVAC3-PR4	1	qualitative
S3	Modify TPA to replicate H12	1	qualitative

The system-level software validation tests described above requires personnel with a comprehensive understanding of the underlying premises and capabilities of both system codes. It is estimated that software validation test S1 will require approximately one month of effort by personnel with expertise in SYVAC3-CC3 and an equal amount of effort by personnel with expertise in TPA. The level of effort for the S2 test will be approximately half of the S1 test.

5.4.2 Comparison with the Japanese Safety Assessment

The Japan Nuclear Cycle development group recently completed a comprehensive safety assessment exercise called H12 (JNC, 2000). The Japan Nuclear Cycle H12 concept for nuclear waste disposal involves deep geologic disposal in saturated crystalline bedrock and an engineered barrier system that includes corrosion-resistant copper canisters containing vitrified high-level waste emplaced within repository drifts and surrounded by bentonite clay.

The collection of submodels in H12 share many features with the consequence modules in TPA. Important differences are largely associated with the characteristics of the disposal sites, the engineered barrier system, radionuclide inventory, and biosphere. The proposed system software validation test (S3) would involve the H12 base case, which will require a single base case simulation.

6 NOTES

The TPA software validation tests described above will require approximately one person year of effort from participants with a broad range of expertise. Priorities will be given to those models that are known to significantly influence risk, output uncertainty, and sensitivity. It is expected that once a modicum of experience is gained from conducting a few of the function and system-level validation tests, the remaining tests will be re-prioritized to ensure that resources are focused on the most important tests.

7 REFERENCES

Brossia, C.S., L. Browning, D.S. Dunn, O.C. Moghissi, O. Pensado, and L. Yang. "Effect of Environment on the Corrosion Potential of Waste Package and Drip Shield Materials." CNWRA 2001–03. San Antonio, Texas: Center for Nuclear Waste Regulatory Analyses. 2001.

CRWMS M&O. "Total System Performance Assessment for the Site Recommendation." TDR–WIS–PA–000001. Rev. 00 ICN 01. Las Vegas, Nevada: CRWMS M&O. 2001.

——. "Total System Performance Assessment—Viability Assessment (TSPA-VA) Analyses Technical Basis Document." B00000000–01717–4301–00002. Las Vegas, Nevada: TRW Environmental Safety Systems, Inc. 1998.

Fedors, R. "Software Validation Test Report for BREATH Version 1.2. 2002.

Goodwin, B.W., T.H. Andres, W.C. Hajas, D.M. LeNeveu, T.W. Melnyk, J.G. Szekely, A.G. Wikjord, D.C. Donahue, S.B. Keeling, C.I. Kitson, S.E. Oliver, K. Witzke, and L. Wojciechowski. "The Disposal of Canada's Nuclear Fuel Waste: A Study of Postclosure Safety of In-Room Emplacement of Used CANDU Fuel in Copper Containers in Permeable Plutonic Rock." Vol. 5: Radiological Assessment. AECL–11494–5, COG–95–552–5. Ottowa, Canada. Atomic Energy of Canada Limited. 1996.

Goodwin, B.W., D.B. McConnell, T.H. Andres, W.C. Hajas, D.M. LeNeveu, T.W. Melnyk, G.R.Sherman, M.E. Stephens, J.G. Szekely, P.C. Bera, C.M. Cosgrove, K.D. Dougan, S.B. Keeling, C.I. Kitson, B.C. Kummen, S.E. Oliver, K. Witzke, L. Wojciechowski, and A.G. Wikjord. "The Disposal of Canada's Nuclear Fuel Waste: Postclosure Assessment of a Reference System." AECL–10717, COG–93–7. Ottowa, Canada. Atomic Energy of Canada Limited. 1994.

Hill, B.E., C.B. Connor, M.S. Jarzemba, P.C. La Femina, M. Navarro, and W. Strauch. 1995 eruptions of the Cerro Negro volcano, Nicaragua, and risk assessment for future eruptions. Geological Society of American Bulletin. 110(10): 1231–1241. 1998.

International Atomic Energy Agency. Validation of Models Using Chernobyl Fallout Data from the Central Bohemia Region of the Czech Republic, Scenario CB, First Report of the VAMP Multiple Pathways Assessment Working Group—Part of the IAEA/CEC Co-ordinated Research Programme on the Validation of Environmental Model Predictions (VAMP). IAEA–TECDOC–795. Vienna, Austria: International Atomic Energy Agency. 1995.

Iman, R.L. and M.J. Shortencarier. NUREG/CR–3624, "A FORTRAN 77 Program and User's Guide for the Generation of Latin Hypercube and Random Samples for Use with Computer Models." Washington, DC: NRC. 1984.

Janetzke, R., S. Mohanty, C. Grossman, L. Browning, R. Codell, R. Fedors, B. Hill, O. Pensado, and D. Turner. "Software Requirements Descriptions for the Total-system Performance Assessment Version 5.0 Code—Amendment 1." San Antonio, Texas: CNWRA. 2002.

JNC, 2000. H–12: Project to Establish the Scientific and Technical Basis for HLW in Japan, by the Japan Nuclear Cycle Development Institute.

Lichty, R.W. and P.W. McKinley. "Estimates of Ground Water Recharge Rates for Two Small Basins in Central Nevada. Water Resources Investigations Report." 94-4104. MOL. 19960924.0524. Denver, Colorado: U.S. Geological Survey. 1995.

Mohanty, S. and T.J. McCartin. "Total-system Performance Assessment (TPA) Version 3.1.4 Code: Module Description and User's Guide." San Antonio, Texas. CNWRA. 1998a.

Mohanty, S. and T.J. McCartin. "Total-system Performance Assessment (TPA) Version 3.2 Code: Module Description and User's Guide." San Antonio, Texas: CNWRA. 1998b.

Mohanty, S., T.J. McCartin, and D.W. Esh (coords). "Total-system Performance Assessment (TPA) Version 4 Code: Module Descriptions and User's Guide." San Antonio, Texas: CNWRA. 2002a.

Mohanty, S., R. Codell, J. Mencahca, R. Janetzke, M. Smith, P. Laplante, M. Rahimi, and A. Lozano. "System-Level Performance Assessment of the Proposed Repository at Yucca Mountain using the TPA Version 4.1 Code." CNWRA 2002-05. Rev. 1.0. San Antonio, Texas. CNWRA. 2002b.

Mohanty, S., R. Codell, R.W. Rice, J. Weldy, Y. Lu, R.M. Byrne, T.J. McCartin, M.S. Jarzemba, and G.W. Wittmeyer. "System-Level Repository Sensitivity Analyses Using TPA Version 3.2 Code." CNWRA 99–002. San Antonio, Texas: CNWRA. 1999.

Napier, B.A. "GENII Version 2 Users' Guide." Richland, Washington: Pacific Northwest National Laboratory. September 2002.

Napier, B.A., R.A. Peloquin, D.L. Strenge, and J.V. Ramsdell. "GENII: The Hanford Environmental Radiation Dosimetry Software System, Volume 1: Conceptual Representation." PNL–6584, Vol. 1. Richland, Washington: Pacific Northwest Laboratory. 1988.

NRC. NUREG–1668, "NRC Sensitivity and Uncertainty Analyses for a Proposed HLW Repository at Yucca Mountain, Nevada, Using TPA 3.1. Vol. II: Result and Conclusions." Washington, DC: NRC. March 1999.

Stothoff, S.A. "BREATH Version 1.1—Coupled Flow and Energy Transport in Porous Media. Simulator Description and User Guide." San Antonio, Texas: CNWRA. 1995

Wescott, R.G., M.P. Lee, N.A. Eisenberg, T.J. McCartin, and R.G. Baca, eds. NUREG–1464, "NRC Iterative Performance Assessment Phase 2." Washington, DC: NRC. 1995.

APPENDIX A

APPENDIX A

The following tables are a summary of the work that has been accomplished as well as the work that needs to be completed for validation testing. The tables are organized as follows:

A. Module: Identifier associated with the module.

B. Validation Test or Test ID: Testing that needs to be completed for software validation.

C. Testing Phase: The Validation Test identifies testing for the code module. This testing falls into three categories:

   1. Phase 1: Check for functional correctness. Tests in this category include reasonableness checks, hand calculations, and summary calculations in an Excel® Spreadsheet.

   2. Phase 2: Model validation with analytical results. In this phase, module level output is compared to analytical results from packages such as Mathematica® or MathCad®.

   3. Phase 3: Model validation with outside system-level models. Verifies the code can generate results which are comparable to those generated by outside system-level models. Special input parameters may have to be used for such comparisons

D. Effort (Person-Months): An estimate of the additional time required to accomplish the validation tests.

Table A–1. Summary of Validation Tests for Primary and Secondary Utility Modules Associated with the TPA Executive			
Module	**Validation Test**	**Testing Phase**	**Effort (Person—Months)**
E1: READER	Verify that invalid parameter values for density functions will be trapped. Inspect the code to verify that input variables are actually being used and not overwritten. Also, verify the drift endpoint calculations with the repository design. May want to add checks on nuclide chains to include colloids to verify the chains specified in *tpa.inp* are supported.	I	0.5
E2: SAMPLER	Verify distribution sampling and selected mean values for all distributions but in particular to include new distributions such as User Supplied Piecewise CDF. This testing would verify that the expected probability curve is obtained for each distribution. Verification of beta, logbeta, and iuniform distributions will be required. And, verification that integer and floating point values are sampled to the required precision is needed. In addition, verify that LHS Mode 2 can be invoked and the code will execute correctly using a previously generated sampled parameter file.	I	0.25
E3: INVENT	Using data checks and inventory calculations, verify colloidal radionuclides have been correctly integrated.	I	0.25
	Verify that different combinations of chains and different ordering of radionuclides can be selected and that when these different combinations are selected, both the correct information from the *nuclides.dat* input file is retrieved and inventories are calculated correctly.	I	0.25

Table A–1. Summary of Validation Tests for Primary and Secondary Utility Modules Associated with the TPA Executive (continued)			
Module	**Validation Test**	**Testing Phase**	**Effort (Person—Months)**
E4: MODULE-VARIABLE	Verify the security features are working properly. Specifically, verify that values can only be stored by the consequence module that generates the result. In addition, verify that no other module can overwrite and corrupt the corresponding MODULE-VARIABLE database.	I	0.25
E5: SUBAREA	The SUBAREA module contains a number of subarea specific utility routines that need to be verified in terms of the current repository subarea layout. For this verification, each utility module will have to be invoked with a set of test input. Afterwards, the output will have to be compared to expected values.	I	0.5
E6: ARRAY	Verify individual subroutines correctly perform the required data transformations. Should also verify that the calling routines are using the subroutine correctly. For example, is the caller trying to clear an array of integers or is it trying to clear an array of floating point values, and does the caller use the correct subroutine within ARRAY (In this case, Subroutine ZEROI for integers and Subroutine ZERO for floating point values).	I	0.75
E7: FILEUNIT	Inspect the code to verify that FILEUNIT functions correctly in assigning unique file numbers to individual code modules, that the code modules requesting a unit number are entitled to retrieve a unit number from FILEUNIT, and that sufficient unit numbers are available for the code to execute.	I	0.25
E8: FINDELEV	Perform independent calculations to verify the FINDELEV module returns the correct elevation for the coordinates provided.	I	0.25

Table A–1. Summary of Validation Tests for Primary and Secondary Utility Modules Associated with the TPA Executive (continued)			
Module	**Validation Test**	**Testing Phase**	**Effort (Person—Months)**
E9: NUMRECIP	Perform hand calculations to verify the individual numerical algorithms are returning the correct result.	I	0.25
E10: PEAKFIND	Verify the peak dose time and magnitudes in relation to dose-time data.	II	0.25
E11: RAN	Verify that random number generation has a sufficiently long cycle to ensure that serial correlation does occur.	I	0.25
E12: IAREADER	Verify that importance analysis information is transferred from *tpa.inp* and that input routines are retrieving the importance analysis information. IAREADER takes input information from the *ia.dat* file and overwrites sampled values. Therefore, testing will verify that IAREADER correctly overwrites sampled values for retrieval by other subroutines.	I	0.25
E13: EXEC (CHECKPOINT RESTART)	Verify that updated and new output files are appended correctly on restart. Verify the checkpoint file, *check.pnt*, is created, maintained, accessed, and deleted correctly and that execution can be resumed when required.	I	0.25

Table A–2. Summary of Validation Tests for Consequence Modules			
Module	**Test ID**	**Phase**	**Effort (Person—Months)**
C1: UZFLOW	C1-1	I	0.125
	C1-2	I	0.25
	C1-3	I	0.125
	C1-4	II	0.25
C2: NFENV	C2-1	I	0.25
	C2-2	II	0.25
	C2-3	II	0.25
	C2-4	II	0.50
C3: EBSFAIL	C3-1	I	0.125
	C3-2	I	0.25
	C3-3	I	0.25
	C3-4	II	0.125
	C3-5	II	0.5

Table A–2. Summary of Validation Tests for Consequence Modules (continued)			
Module	**Test ID**	**Phase**	**Effort (Person—Months)**
C4: DSFAIL	C4-1	I	0.25
	C4-2	I	0.25
C5: WELDFAIL	C5-1	I	0.125
	C5-2	I	0.25
	C5-3	II	0.125
C6: SEISMO	No Tests	NA	
C7: EBSREL	C7-1	I	0.125
	C7-2	I	0.5
	C7-3	I	0.125
	C7-4	II	0.25
C8: EBSFILT	C8-1	I	0.125
	C8-2	I	0.375
	C8-3	I	0.25
	C8-4	II	0.25

Table A–2. Summary of Validation Tests for Consequence Modules (continued)			
Module	**Test ID**	**Phase**	**Effort (Person—Months)**
C9: UZFT	C9-1	I	0.375
	C9-2	I	0.125
	C9-3	I	0.5
C10: SZFT	C10-1	I	0.125
	C10-2	I	0.125
	C10-3	I	0.25
	C10-4	II	0.5
	C10-5	II	0.5
C11: DCAGW	C11-1	I	1.0
C12: FAULTO	C12-1	I	0.5
C13: VOLCANO	C13-1	I	0.25
	C13-2	I	0.25
	C13-3	I	0.25
	C13-4	I	0.25
	C13-5	I	0.25
	C13-6	I	0.25

Table A–2. Summary of Validation Tests for Consequence Modules (continued)			
Module0	Test ID	Phase	Effort (Person—Months)
C14: ASHPLUMO	C14-1	I	0.125
	C14-2	I	0.125
	C14-3	II	0.25
	C14-4	II	0.5
C15: ASHRMOVO	C15-1	I	0.25
	C15-2	II	0.5
C16: DCAGS	C16-1	I	0.25
	C16-2	I	0.25
	C16-3	II	0.25

Table A–3. Summary of Validation Tests for Stand-Alone Modules			
Module	**Validation Test**	**Testing Phase**	**Effort (Person—Months)**
S1: EBSFAIL to FAILT	Verify that data from WELDFAIL is correctly transferred back to EBSFAIL and subsequently to RELEASET.	I	0.0 (completed)
	Verify transfer of information to files for use by FAILT.	I	0.25
S2: EBSREL to RELEASET	Verify that the time dependent fow and fmult factors and the flow multipliers are implemented correctly.	II	0.25
	Verify transfer of information to files for use by RELEASET.	I	0.25
	Verify that information generated by EBSREL is passed through EBSFILT correctly. In addition, verify the composite information from the spent fuel and glass waste forms is correctly returned to EBSREL	I	0.5
S3: UZFT, SZFT to NEFMKS	Verify input data to NEFMKS to include colloid information. In addition, verify parameters such as immobile porosity.	I	0.5
S4: DCAGS, DCAGW to GENTPA	Verify transfer of information to and from GENTPA code.	I	0.5
S5: ASHPLUMO to ASHPLUME	Verify the transfer of information between ASHPLUMO and ASHPLUME	I	0.75
S6: SEISMO2	—	II	—

Table A–3. Summary of Validation Tests for Stand-Alone Modules (continued)			
Module	**Validation Test**	**Testing Phase**	**Effort (Person—Months)**
S7: DSFAIL to DSFAILT	Verify dryout chemistry is placed in file *fluoride.dat* for use by DSFAILT module	I	0.1
S8: SAMPLER to SNLLHS	Verify that for the distributions which may be defined in *tpa.inp*, and in particular the new distributions of: User Supplied Piecewise CDF, User Supplied Discrete, iuniform, and logbeta that the values for these distributions are correctly passed to SNLLHS. In addition, verify that invalid input information such as out of bound large sample sizes are trapped.	I	0.25

Table A–4. Summary of System Level Validation Tests			
Test-ID	**Validation Test**	**Testing Phase**	**Effort (Person—Months)**
S1	Replicate results from Canadian SYVAC-C03	III	2
S2	Replicate results from Canadian SYVAC-PR4	III	1
S3	Replicate results from Japanese H12 performance assessment	III	1