ORNL/NRC/LTR-248

Contract Program or Project Title:	Probabilistic Pressure Boundary Integrity Safety Assessment (PISA)
Subject of this Document:	**SIAM-xLPR Version 1.0 Framework Report**
Type of Document:	Letter Report
Authors:	H. B. Klasky P. T. Williams B. R. Bass S. Yin
Date of Document:	September 30, 2010
Responsible NRC Individual and NRC Office or Division	M. T. Kirk Division of Engineering Technology Office of Nuclear Regulatory Research

Prepared for the
U. S. Nuclear Regulatory Commission
Washington, D.C. 20555-0001
Under Interagency Agreement DOE 1886-N653-3Y
NRC JCN No. N6438

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831-8056
Managed and operated by
UT-Battelle, LLC for the
U. S. DEPARTMENT OF ENERGY
under Contract No. DE-AC05-00OR22725

ORNL/NRC/LTR-248

SIAM-xLPR Version 1.0 Framework Report

Prepared by

H. B. Klasky, P. T. Williams, S. Yin, and B. R. Bass
Oak Ridge National laboratory
P. O. Box 2008
Oak Ridge, Tennessee 37831-6285
Managed by
UT-Battelle, LLC
For the U.S. Department of Energy
Under contract DE-AC05-00OR22725

Prepared for
U.S. Nuclear Regulatory Commission

ORNL/NRC/LTR-248

SIAM-xLPR Version 1.0 Framework Report

Manuscript Completed: September 2010
Date Published:

Prepared by
H. B. Klasky, P. T. Williams, S. Yin, and B. R. Bass
Oak Ridge National laboratory
P. O. Box 2008
Oak Ridge, Tennessee 37831-6285
Managed by
UT-Battelle, LLC
For the U.S. Department of Energy
Under contract DE-AC05-00OR22725

Prepared for
Division of Engineering
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001
NRC Job Code N6438

Research Sponsored by the Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission under Interagency Agreement 1886-T778-08 with the U.S. Department of Energy. This paper was prepared by Oak Ridge National Laboratory, P. O. Box 2008, Oak Ridge, Tennessee 37831-6285, managed by UT-Battelle, LLC, for the U.S. Department of Energy, under contract DE-AC05-00OR22725.

This page is intentionally left blank.

SIAM-xLPR Version 1.0 Framework Report

H. B. Klasky, P. T. Williams, S. Yin, and B. R. Bass

ABSTRACT

This document describes the probabilistic framework developed for the Structural Integrity Assessments Modular – Extremely Low Probability of Rupture (SIAM-xLPR) software tool at the Oak Ridge National Laboratory (ORNL), which supports the Nuclear Regulatory Commission's initiative to develop a risk-informed alternative to the current leak before break assessment methodology. The new methodology will provide a tool that can directly assess the probability of rupture in nuclear power plants' primary water piping systems. SIAM-xLPR is a modular-based assessment tool that incorporates a prototype xLPR model assembled from new and existing fracture mechanics models and software. xLPR represents one of the four subsystems currently installed in the ORNL-developed SIAM Problem-solving Environment. The prototype SIAM-xLPR models and software modules are linked within a probabilistic framework developed using open-source languages and software libraries.

This page is intentionally left blank.

FOREWORD

The present document is one of a series of reports summarizing the results of the NRC xLPR project for the work performed during the 2010 fiscal year. The complete list of the titles and the organization authors is as follows:

• xLPR Pilot Study Final Report	• NUREG and EPRI
• GSxLPR User's Manual	• SNL
• SIAMxLPR User's Manual	• ORNL
• xLPR Version 1.0 Report	• xLPR Computational Group
• GSxLPR and SIAMxLPR Comparison Report	• CNWRA
• GSxLPR v1.0 Framework Report	• SNL
• SIAMxLPR v1.0 Framework Report	• ORNL
• xLPR v.1.0 Models/Inputs Report	• xLPR Models/Inputs Group

CONTENTS

Page

LIST OF FIGURES

ABBREVIATIONS

BSD	Berkeley Software Distribution
BWR	Boiling Water Reactor
CM	Configuration Management
DISFRAC	Dislocation-Based Fracture Model
EMC2	Engineering Mechanics Corporation of Columbus
EPFM	Elastic Plastic Fracture Mechanics
EPRI	Electric Power Research Institute
FAVOR	Fracture Analysis of Vessels – Oak Ridge
GPL	General Public License
GNU	Recursive acronym for "GNU's Not Unix" (http://www.gnu.org/)
GUI	Graphic User Interface
IDE	Integrated Development Environment
ISI	In-Service Inspection [used in reference web links but not in text]
LBB	Leak-Before-Break
LOCA	Loss-of-Coolant Accident
NRC	Nuclear Regulatory Commission
NumPy	Numerical Python
ORNL	Oak Ridge National Laboratory
PFM	Probabilistic Fracture Mechanics
PNNL	Pacific Northwest National Laboratory
POND	Probability of Non-Detection
PRA	Probabilistic Risk Assessment
PRAISE	Piping Reliability Analysis Including Seismic Events
PTS	Pressurized Thermal Shock
PWSCC	Primary Water Stress Corrosion Cracking
RPV	Reactor Pressure Vessel
SC	Surface Crack
SCC	Stress Corrosion Crack
SCFM	Scientific Computing Fracture Mechanics
SciPy	Scientific Python
SCM	Software Configuration Management
SIAM-PFM	Structural Integrity Assessments Modular-Probabilistic Fracture Mechanics
SIAM-xLPR	The xLPR application as implemented into SIAM-PFM
SNL	Sandia National Laboratories
SRP	Standard Review Plan
StDev	Standard Deviation
SSE	Safe Shutdown Earthquake
TWC	Through-Wall Crack
US	United States
xLPR	Extremely Low Probability of Rupture
WRS	Weld Residual Stress

ACKNOWLEDGMENTS

This project report, as with all the xLPR reports, represents a synthesis of a collective effort of a national team of experts. For xLPR, in addition to gratefully recognizing the significant support of the US Nuclear Regulatory Commission, very sincere thanks are due to all those persons and organizations who contributed to the project, for the most part on a contribution-in-kind basis. The following diagram hopefully leaves none out:

1. INTRODUCTION

The Extremely Low Probability of Rupture (xLPR) project [1] was launched by the United States (US) Nuclear Regulatory Commission (NRC) to develop a power reactor's piping system assessment methodology to demonstrate compliance with the 10CFR50 Appendix A, GDC-4 requirement [2].

> "*Criterion 4--Environmental and dynamic effects design bases*. Structures, systems, and components important to safety shall be designed to accommodate the effects of and to be compatible with the environmental conditions associated with normal operation, maintenance, testing, and postulated accidents, including loss-of-coolant accidents. These structures, systems, and components shall be appropriately protected against dynamic effects, including the effects of missiles, pipe whipping, and discharging fluids, that may result from equipment failures and from events and conditions outside the nuclear power unit. However, dynamic effects associated with postulated pipe ruptures in nuclear power units may be excluded from the design basis when analyses reviewed and approved by the Commission demonstrate that the probability of fluid system piping rupture is extremely low under conditions consistent with the design basis for the piping."

Organizations participating in the xLPR project are as follows: US-NRC, Oak Ridge National Laboratory (ORNL), Sandia National Laboratory (SNL), Battelle, Structural Integrity Associates, Inc., Electric Power Research Institute (EPRI), and Westinghouse, Engineering Mechanics Corporation of Columbus (EMC^2), Phoenix Engineering Associates, Inc, Exelon, First Energy, Dominion Engineering, Inc., Pacific Northwest National Laboratory (PNNL), and AREVA. Hereinafter the participant organizations of the xLPR project will be named the xLPR consortium. The xLPR consortium has several task groups that decide on models for short-term use in developing the initial version of the code. These task groups are as follows:

- Computational Group
- Models Group
- Acceptance Criteria Group
- Inputs Group
- Project Integration Group

The purpose of this report is to provide an overview of the SIAM-xLPR implementation. The SIAM-xLPR implementation provides a framework for evaluating the interaction among various modules focused on initiation and growth of primary water stress corrosion cracks (PWSCC) in a dissimilar metal pressurizer surge nozzle weld. It focused not only on integrating the modules developed by the xLPR consortium that are part of the piping assessment methodology, but also on developing a unique piece of reusable open-source software that works as a framework and that provides a means for incorporating changes to the model inputs of the piping assessment methodology, and to present and display the resulting outputs generated by it. The implementation of SIAM-xLPR started in November 2009, the alpha version was complete in March 2010 and the beta version in September 2010.

The implementation of SIAM-xLPR was performed at the ORNL site and its development was performed by researchers in the Scientific Computing Fracture Mechanics (SCFM) team. This report includes a description of the Configuration Management practices followed by the SCFM team. Initial sections provide an overview of the SIAM-PFM framework, the xLPR project, and the SIAM-xLPR implementation. Subsequent sections provide a description of the pilot study problem, and presents the SIAM-xLPR results for the pilot study. This is followed by an integrated assessment of the project with conclusions and recommendations.

2. THE SIAM FRAMEWORK

The SIAM-PFM Framework [3] is a problem-solving environment being developed at ORNL for the NRC. The acronym SIAM-PFM (or just SIAM for short) stands for Structural Integrity Assessment Modular – Probabilistic Fracture Mechanics. SIAM-PFM is intended to be a framework within which a wide range of nuclear power plant safety issues can be addressed in a systematic and consistent way by using modern principles of probabilistic risk assessment. Probability techniques are applied to problems in fracture mechanics in order to predict fracture behavior and thus to assess the structural integrity of a variety of nuclear power-plant components that make up the primary pressure boundary. This platform is intended to be readily extensible to different problem classes with the level and methods of user interaction to be determined by discussions with the NRC and potential stake holders.

At present, SIAM-PFM contains four applications; three were developed at ORNL and one was developed elsewhere, as noted below. All of the modules can be operated within the SIAM-PFM framework or independently as stand-alone applications. The four modules include the following:

- **FAVOR – Fracture Analysis of Vessels Oak Ridge.** FAVOR [4,5] is a safety assessment tool for analyzing the effects of pressure and temperature loading due to normal and accident conditions on commercial nuclear reactor pressure vessels (RPVs). The FAVOR code, developed at ORNL for the NRC, addresses the potential for failure by through-wall cracking of the RPV wall when the vessel beltline is exposed to thermal-hydraulic transients, such as pressurized-thermal-shock (PTS) events, during the plant’s operational history. Making use of finite element techniques and modern probabilistic risk assessment (PRA) methodologies, FAVOR estimates the probability of through-wall cracking of a nuclear reactor vessel due to pre-existing flaws in the wall of the vessel.

- **DISFRAC – Dislocation-Based Fracture Model with Extensions to Cleavage Initiation in Ferritic Steels.** DISFRAC is an implementation in code of a theoretical, multi-scale model currently under development for the prediction of fracture toughness of ferritic steels in the transition temperature region. The model accounts for temperature, irradiation, strain rate, and material condition (chemistry and heat treatment) effects. DISFRAC permits fracture safety assessments of ferritic structures with only tensile properties required as input.

- **PRAISE – Piping Reliability Analysis Including Seismic Events.** PRAISE, v04.2, [6,7] evaluates the reliability of welds in nuclear power plant piping systems. The PRAISE code was originally developed to provide a technical basis for the NRC to determine whether it could relax its requirements on the combination of a safe shutdown earthquake (SSE) and a large loss-of-coolant accident (LOCA) for power plant piping components. In addition, PRAISE allows for an estimation not only of the probability of the simultaneous occurrence of a large LOCA and an earthquake, but also of the probability of a large LOCA caused by normal and abnormal loading without an earthquake. The original development of PRAISE provided a probabilistic treatment of the growth of crack-like weld defects in piping due to cyclic loading. This treatment of fatigue-crack growth was later expanded to include the initiation and growth of stress corrosion cracks (SCCs) found in Boiling Water Reactors (BWRs). Additional development for PRAISE, v04.2 [6], expanded the capabilities of the code to include a probabilistic treatment of fatigue-crack initiation.

- **xLPR – Extremely Low Probability of Rupture.** xLPR [8], the subject of this document, is the latest addition to the SIAM-PFM suite. SIAM-XLPR will refer herein to the xLPR application as implemented into SIAM-PFM.

3. THE XLPR PROJECT

xLPR is a methodology for assessing the integrity of nuclear power plant piping systems. Specifically, xLPR aims to provide a method for direct and quantitative assessment of compliance with the 10CFR50 Appendix A, GDC-4 [2] requirement.

xLPR is being implemented as a software tool that predicts how nuclear power plant piping systems degrade over time; it also models the uncertainties both input variables and models and it permits assessment of the effects of mitigation efforts on the system. The implementation that ORNL presents in this report is that of the pilot study, which concentrates on the initiation and growth of PWSCC in a dissimilar metal pressurizer surge nozzle weld (see Fig. 1). Thus, for the pilot study, the degradation mechanism and materials are fixed, but the operating conditions, pipe geometry, weld residual stress characterization, mitigation effects, material properties, and uncertainty characterizations of the inputs to the model can be modified by the user.

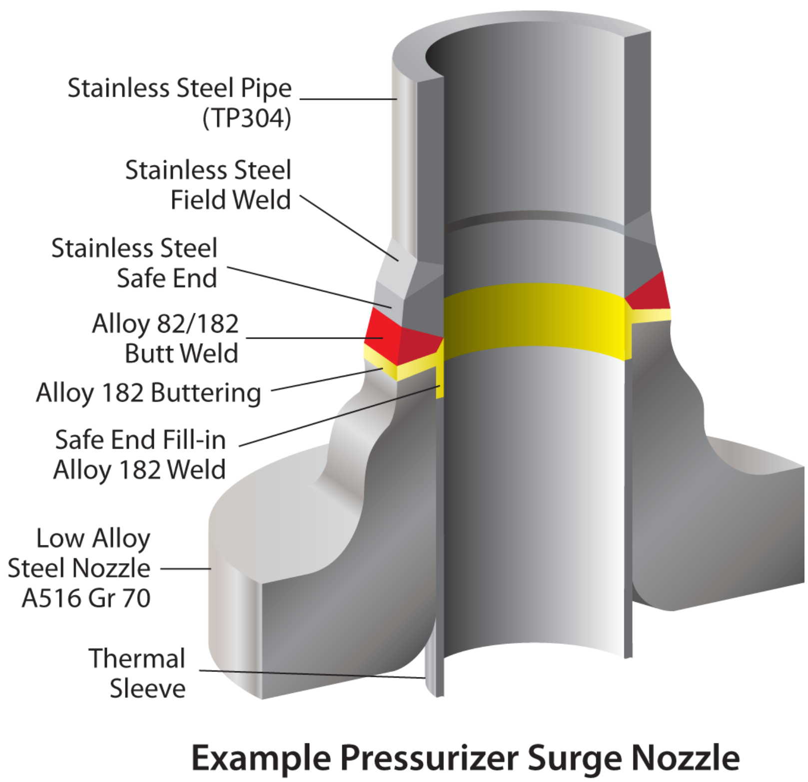

Fig. 1. Dissimilar metal pressurizer surge nozzle weld geometry schematic.

4. THE SIAM-XLPR SYSTEM ARCHITECTURE

4.1 OVERALL FLOW PROCESS

Fig. 2 presents the logical flow of xLPR. Both epistemic and aleatory trial records are sampled according to pre-defined parameters related to load, leaks, and crack initiation. The term “epistemic” uncertainties refer to uncertainties that reflect a lack of knowledge; i.e. the information is not present or available, but in principle could be acquired with enough study or expert judgment. “Aleatory” uncertainties refer to uncertainties that cannot be determined or deduced. The green block at the center of Fig. 2 represents the core of the execution: the “Execute Time Loop” module. This Time Loop module executes the trials for each new set of variables. Appendix A. SIAM-xLPR Framework Design contains additional details.

The details of the Execute Time Loop flow are depicted in Fig. 3, which presents a sequence of modules that model the progression of cracking in piping systems: crack initiation, crack growth, degree of crack stability, and leakage. The figure also shows the progression of mitigation efforts: inspection, detection, and mitigation of PWSCC. During each time step the number of active cracks is determined, and a decision is made as to whether inspection and/or mitigation will be performed. If any crack has initiated within a given time step, its location is identified. If coalescence or merging with another crack or cracks is detected, the new combined crack’s size and location are also determined. All new cracks initiate as internal surface cracks (SC). Over time they may grow through the wall of the pipe and transition to through-wall cracks (TWC). In the case of through-wall cracks, the crack-opening displacement and the leakage rate are calculated. Through-wall cracks can be detected by SIAM-xLPR when the leakage rate is larger than a user-specified LOCA leakage rate; the system also records the predicted time of the failure. For those cracks identified as surface cracks, the system calculates the Probability of Non-Detection (POND), and completes the current time step.

4.2 DESCRIPTION OF MODULES EMPLOYED IN XLPR PROCESS

Only one of the modules employed in the pilot sutdy is executed outside of the TimeLoop module. This module is the Load Module [9] which is executed in the Develop Time History block in Fig. 1. The Load Module is under the direct control of the SIAM-xLPR Framework and is executed after all probabilistic sampling has been taken place as shown in Fig. 2. The Load Module calculates the total axial membrane stress, with and without stress due to SSE and with and without WRS. It also calculates the total bending moment with and without SSE. The load module also determines the WRS fitting coefficients for its assumed 3rd order form, boundary conditions, and constraints.

The purpose of the TimeLoop module shown in Fig.3 is to bring together 10 independent modules and carry out a time integration of the dissimilar metal pressurizer surge nozzle weld. This time integration covers a time period constituting a plant time horizon of approximately 60 years. The size of the discrete time step applied in this integration is a user input with a minimum allowable time step of one month. By the point at which the framework hands execution of the individual trial over to the timeloop, all required probabilistic sampling has already taken place and the timeloop; therefore represents a completely deterministic kernel which, when completed, transfers the results back to the framework and waits for the next realization trial inputs to be passed to it from the framework.

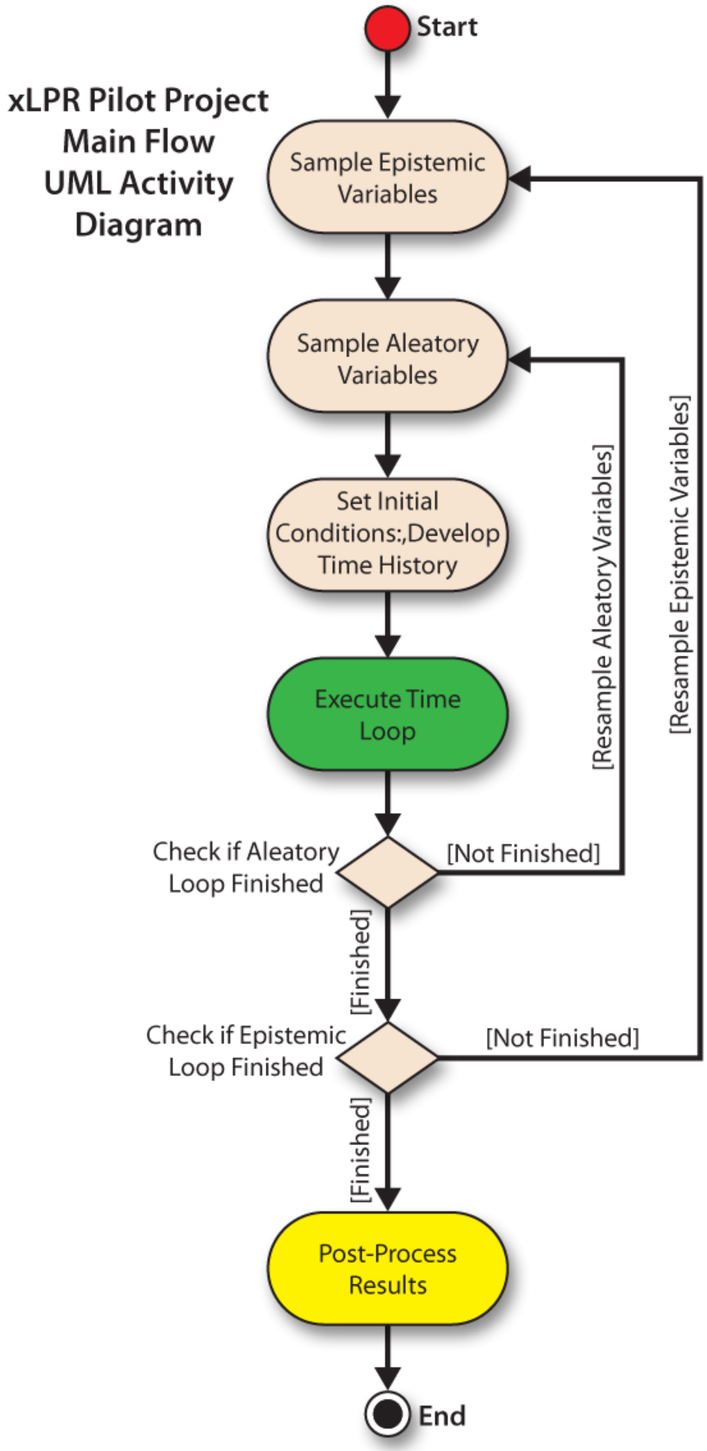

Fig. 2. xLPR's High-Level Execution Flow.

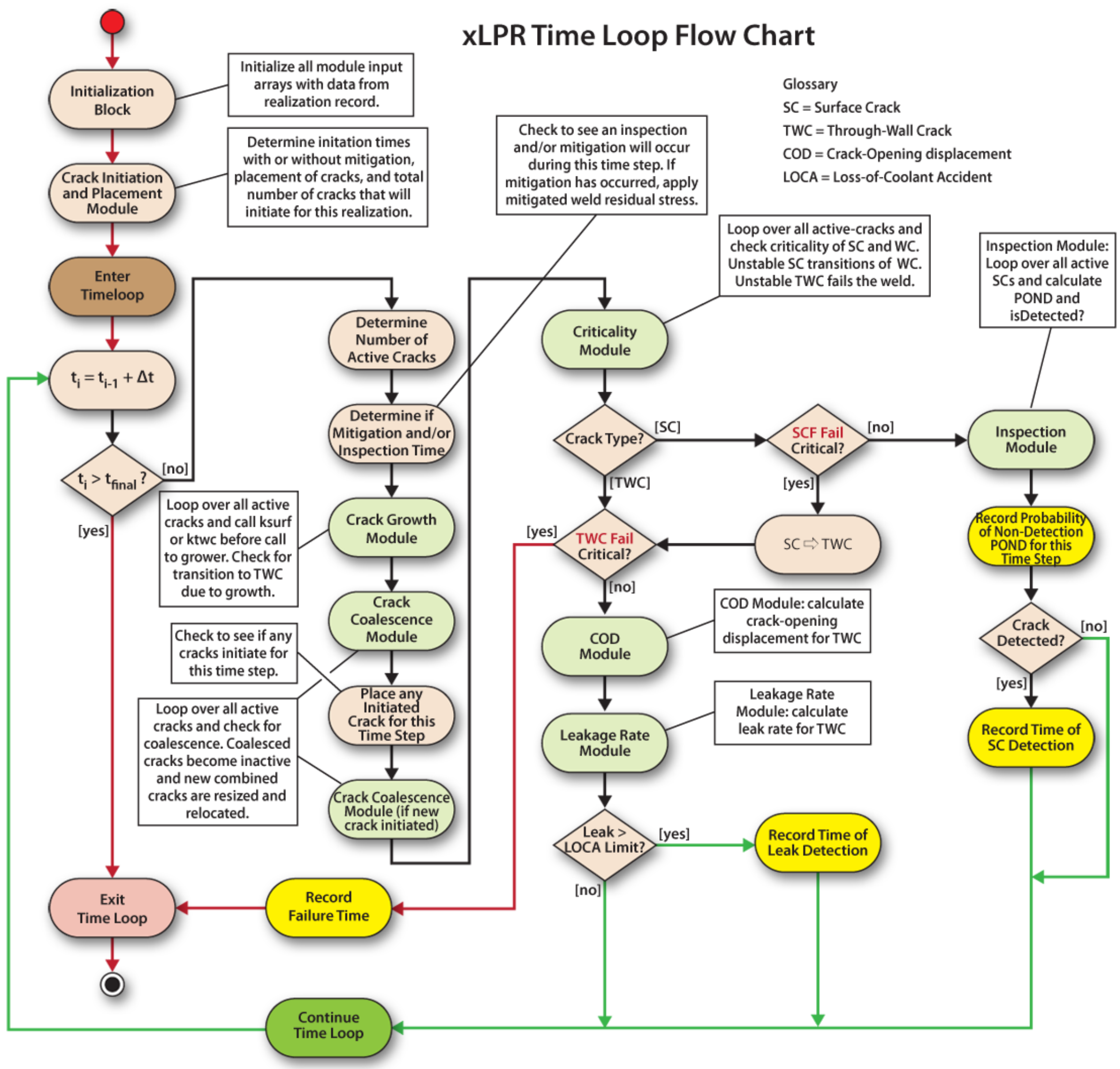

Fig. 3 xLPR's Time Loop Execution Diagram.

The 10 modules under the control of the TimeLoop module are as follows (the descriptive text is taken from the cited *Conceptual Descriptions*):

1. **Crack Initiation and Placement [10]** – The crack initiation module of the xLPR pilot code assigns crack initiation times to cracks assumed to initiate in piping weldments due to PWSCC. The initiation times are determined from a probabilistic model that includes the effects of temperature and stress. Additionally, the module places the initiated cracks at specific locations on the weld perimeter

2. **Driving Forces for Surface Cracks [11]** – The kSurf_v1.0 module calculates the stress intensity factor for a semi-elliptical ID surface crack. The Anderson K-solutions [12] for a circumferential ID surface crack were used in the development of this module. The solutions in this report [12] were generated for *R/t* (inner radius divided by wall thickness) values from 3 to 100, *c/a* (half crack length divided by crack depth) values from 1 to 32, and *a/t* (crack depth divided by wall thickness) values from 0.2 to 0.8. Anderson generated influence functions G0,G1, and G5 (global in-plane bending) using finite element techniques. The influence function G2, G3, and G4 are inferred from the weight function formulas given in [12]. For the case of a circumferential semi-elliptical surface crack, the crack growth at both the deepest (ϕ=90°) and surface (ϕ = 0°) locations are calculated and applied to the initial crack sizes. The finite length surface crack is always assumed to remain semi-elliptical.

3. **Driving Forces for Through-Wall Cracks [13]** – The kTWC_circ_V1.0 module calculates the stress intensity factor for an idealized through-wall crack. The Anderson K-solutions [14] for a circumferential through-wall crack were used to develop this module. These solutions were generated for *R/t* values from 1 to 100 and to crack lengths of about 66 percent of the circumference. The solutions were generated for both the inside and outside surface of the through-wall crack, however; only the G0, G1 and G5 influence functions are available. The through-wall crack K-solutions were curve fit and the coefficients were presented for *R/t* values of 1, 3, 5, 10, 20, 60, and 100. These coefficients are used in this module and linear interpolation is used to predict the coefficients for other *R/t* values. The influence function on both the inside and outside surface of the through-wall crack are calculated, and then averaged to get the K-solution for through-wall crack growth.

4. **Crack Growth [15]** – The Crack Growth Module uses a PWSCC mechanism. The Module calculates the crack growth rate in both the depth and length directions for surface cracks and in the length direction only for through-wall cracks. It also calculates the updated crack depth (for surface cracks only) and updated half crack length using the calculated crack growth rates.

5. **Crack Coalescence [16]** – The Coalescence Module for the pilot study combines two cracks based on the following criterion: (a) For semi-elliptical surface cracks when the distance between the surface cracks becomes less than half the deepest surface crack depth, the cracks will coalesce. The depth of the new crack is equal to the deepest surface crack and the length is equal to the sum of the lengths of each crack plus the distance between them. This criterion is based on Section 8 of [27] and the Section XI, Article IWA-3000 of [17]. (b) Two through-wall cracks will coalesce when the crack tips touch. The length of the new crack is equal to the sum of the lengths of each crack.

6. **Criticality Module for Surface Cracks [18]** – The SC_Fail_v2.0 module assesses the stability of a surface crack in a pipe subjected to combined tension and bending loading. Based on input pipe/crack geometry, pipe material properties and loads, the ultimate load-

carrying capacity of the surface crack is compared with the current loading. A flag is returned that indicates the result of this comparison: predicted failure, yes or no, as well as the ratio of the current applied bending moment to the critical bending moment. The SC_Fail_v2.0 module uses a wrapper subroutine, SC_Fail, for doing the surface crack assessment, providing the user the option to select any of a number of surface crack analyses by an input flag. Presently, two surface crack ultimate-load-capacity predictions are implemented: (a) constant depth surface crack net-section plastic collapse and (b) semi-elliptical surface crack net-section plastic collapse.

7. **Criticality Module for Through-Wall Cracks [19]** – The TWC_Fail_v2.0 module assesses the stability of a through-wall crack in a pipe subjected to combined tension and bending load. Based on input pipe/crack geometry and pipe material properties and loads, the critical crack size of the through-wall crack is compared with the current crack size. A flag is returned that indicates the result of this comparison: predicted failure, yes or no, as well as the ratio of the current crack angle to the critical crack angle. The TWC_Fail_v2.0 module uses a wrapper subroutine, TWC_Fail, for performing the though-wall crack assessment, providing the user the option to select any of a number of though-wall crack analyses by an input flag. Presently, two though-wall crack ultimate load capacity predictions are implemented: (a) ideal through-wall crack net-section collapse and (b) LBB.ENG2 elastic-plastic fracture mechanics (EPFM) through-wall crack solution.

8. **Crack-Opening Displacement Module [20]** – The COD_v2.0 module calculates the crack opening displacement of a through-wall crack pipe subjected to combined tension and bending loading, based on input pipe/crack geometry, pipe material properties and applied loads.

9. **Leakage Rate Module [21]** – The SQUIRTv3.0 (xLPR SQUIRT v1.0) module calculates a two-phase critical flow rate for water leaking from a through-wall crack in a pipe. The experimental and theoretical bases for this module come from the work of Henry and Fauske [22-24]. However, considerations were added by Paul, et al [25] to include additional pressure losses which were not accounted for in the Henry-Fauske models.

10. **In-Service Inspection Module [26]** – For the xLPR Pilot Project only the POND is calculated. An inspection is considered to have been performed on a crack of depth a. The probability of detecting a crack of size a is known. If the crack is detected, the indicated crack depth, $\hat{a}$, is predicted from the inspection sizing uncertainty. The crack will be repaired if its indicated size is greater than a specified depth, a_{rep}. Three types of repair are considered: (1) removal of the crack and re-welding, (2) removal and replacement of the entire weld, and (3) performance of an OD weld overlay. If a repair by crack removal and re-welding is performed, there is a probability P_{rep} that a crack will be introduced. If a crack is introduced, it will have a depth a_o and half-surface length b_o, both of which are random variables. The residual stresses, amongst other things, will be altered by the repair process. The post-repair conditions (other than crack size) need to be defined and considered in the analysis, but this is outside the scope of the inspection module. If the entire weld is removed and replaced, the weld will be treated as being a new weld. If a weld overlay is performed, then the crack will remain and the residual stresses are changed. Material properties in the region beyond the original material will also change.

4.3 INTEGRATION OF MODULES INTO SIAM-XLPR IMPLEMENTATION

ORNL's implementation of the xLPR application has closely followed the xLPR Program Plan [27] by creating Python bindings for the core models developed by the xLPR Pilot Project Computational Group. The framework written in Python applies the Load Module (written in Fortran 95) to construct the input loading data required for the Monte Carlo realizations. These realization records are stored in a database where they can be retrieved during a separate execution to carry out the time-loop deterministic kernel shown in Fig. 2. The deterministic analysis includes the crack initiation and placement, crack driving forces, crack growth, crack coalescence, crack stability, crack-opening displacement, leakage rate, and in-service inspection modules, all programmed in Fortran 95 and linked to the SIAM- xLPR framework through Python bindings.

4.4 DESCRIPTION OF CONFIGURATION MANAGEMENT PRACTICES

SIAM-xLPR development has followed the Software Configuration Management (SCM) process established by the xLPR consortium: all the xLPR frameworks and their associated modules, including inputs, source code and binary files are being organized as subdirectories in a centralized repository implemented using Microsoft SharePoint site. The repository can be accessed remotely through the web. Modifications to the items tracked by the SCM process (e.g., framework, modules source code and xLPR model inputs) are manually tracked and documented on the SharePoint server by using forms in MS Word files. Controlled versions of frameworks, module source codes, and xLPR model inputs can be downloaded from the centralized repository. This centralized repository enables the independent and parallel development of the modules and frameworks across organizational and geographic boundaries. The developer checks out a CM item from the SharePoint server (e.g., module source code) and makes the modifications and uploads the file version to be independently checked and verified together with the Microsoft SCM Forms. The documentation is also checked out, modified and checked back in to be independently verified.

The complexity of the SIAM software framework required the coding of thousands of lines of software and the integration of multiple layers of software over the duration of the current xLPR Pilot Project. Thus, the ORNL team incorporated internally the use of a robust automated revision control tool at the early stages of the xLPR development effort to track the history and evolution of the project. This tool replaces the older (ca 1990s) method of filling in MS Word change document forms with a modern, state-of-the-art system that automatically keeps a log of the change record, including the change's author, date and time, description, and comments. This automated revision control tool was used to manage multiple versions of the large number of files that form the SIAM-xLPR framework and to communicate changes between the ORNL team members during development. This is a common software engineering practice widely used in the software development industry.

Without a robust automated revision control tool, manually managing multiple versions of the large number of files that form the SIAM-xLPR framework and communicating the changes between the ORNL team members would have been impossible. Thus, in agreement with the xLPR consortium, we are using the xLPR SharePoint site, to store framework milestones and module source code updates and to share these advancements with other members of the xLPR consortium.

ORNL adopted the *Mercurial* [28] source control management tool for the xLPR project. Mercurial is an open-source [GNU General Public License (GPL) 2.0] cross-platform, distributed, revision-control tool for software developers. Written in Python and C, its major design goals include: 1) high performance and scalability, 2) decentralized, fully-distributed collaborative development, 3) robust handling of both plain text and binary files, and 4) advanced branching and merging capabilities.

Distributed computing is now a current trend in revision control tools. This approach allows peer-to-peer communications to drop the dependency on a centralized server, which tends to become a bottle neck as projects grow larger and clients need to communicate more frequently with the server. However, it is hard to take full advantage of the distributed capability in today's organizations due to security policies and the risk of allowing peer-to-peer communication between clients. Thus, a compromise is required in order to operate a distributed SCM tool while observing the company's computer security policies. At ORNL, security policies do not allow peer-to-peer communication and this fully distributed implementation of Mercurial was not feasible. Instead, we implemented a host-repository server where clients periodically synchronize their work advancements. All data, metadata, and small incremental changes to the code, however, are stored locally in the clients which speed up the tool's process to save data in the repository. Moreover, if the network goes down, clients can still commit their work and synchronize changes with the main repository only when needed. This compromise between saving small incremental changes to the code locally on the clients and saving only periodically on the server has had a positive impact in our software productivity.

For overall management of the SIAM-xLPR project, ORNL adopted the *Trac* open-source [GNU GPL 2.0 and Berkeley Software Distribution (BSD)] cross-platform, web-based project management and issue-tracking tool. Also written in Python, *Trac* allows hyperlinking information between a computer bug database, revision control, and wiki content. *Trac* allows wiki markup in *issue* descriptions and *commit* messages, creating links and seamless references between bugs, tasks, changesets, files, and wiki pages. *Trac* includes a timeline which shows all current and past project events in order, making the acquisition of an overview of the project and tracking progress very easy. Moreover, *Trac* also includes a roadmap which shows the road ahead, listing the upcoming milestones. In addition, *Mercurial* is installed as a plugin to *Trac* [29], enabling integrated software management and project management all via a web-based interface.

5. DESCRIPTION OF PILOT STUDY PROBLEMS SET

A series of deterministic and probabilistic model problems were developed in ref. [30] for the xLPR Pilot Study. A Case Matrix consisting of 43 cases was created to provide solution results from SIAM xLPR to meet the requirements of [30]. Within this Case Matrix, a naming convention was applied to allow cross-referencing with the model problems described in [30]. This naming convention has the following form:

SIAM_v1.01_S??_< number of trials >_<case number>

where

S?? refers to the relevant section number in [30],

<number of trials > is the total number of realizations equal to the product of the number of epistemic trials times the number of aleatory trials per epistemic trial, and

<case number> is a unique case number for this section in [30].

As an example, Case SIAM_v1.01_S3.1_10000_001 represents the probabilistic base case of Section 3.1 in ref. [30]. See Appendix B for a complete listing of the SIAM-xLPR Case Matrix.

6. EVALUATION OF DETERMINISTIC ANALYSES

Section 2.1 of ref. [30] provides for two deterministic test cases as outlined below.

SIAM_v1.01_S2.1_00001_001 – Deterministic Analysis #1: A single crack that initiates at time=0 years, with no mitigation. The location of the crack is at the point in the weld with the highest stress loading (ϕ = 0 rad). The input data for this case are included with the controlled version of the inputs spread sheet for xLPR. Figures 4 through 9 present selected solution results for this case. At 59 months, the initial surface crack transitions to a through-wall crack which then fails the weld at 79 months.

SIAM_v1.01_S2.1_00001_002 – Deterministic Analysis #2: Three cracks initiate all at time=0 years, with no mitigation. The same problem input as the first deterministic analysis is used except with three cracks. The three cracks are the same initial size as in Deterministic Analysis #1. Their respective locations are ϕ= 0 rad, ϕ= 0.6 rad, and ϕ= -1.0 rad. The input data for this case are included with the controlled version of the inputs spread sheet for xLPR. Figures 10−13 present selected solution results for the case. Crack 1 coalesces with crack 2 at 33 months into the analysis, when combined crack 1/2 now becomes a combined and larger surface crack with a new location. The combined crack 1/2 then transitions to through-wall crack at 53 months and then coalesces with crack 3 at 62 months into the analysis. At 64 months combined cracks 1/2/3 fail the weld.

For the purposes of this verification/benchmarking study, the input conditions shown in Figs. 4−13 for the two deterministic cases are hardwired in SIAM-xLPR and cannot be changed by the user through the Graphical User Interface (GUI).

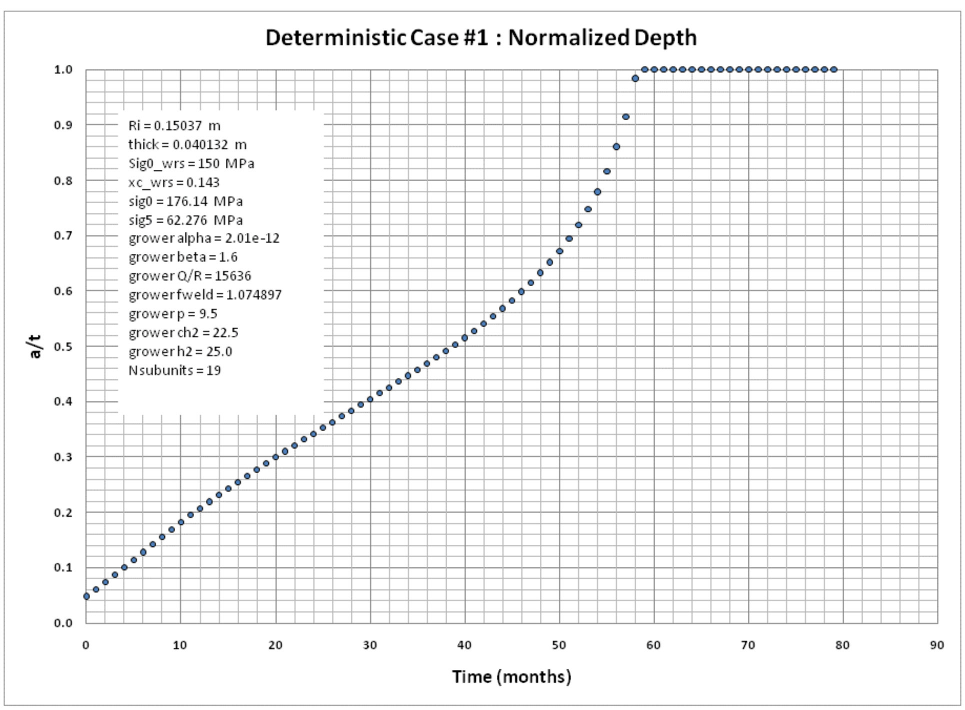

Fig. 4. SIAM_v1.01_S2.1_00001_001 normalized depth of growing crack.

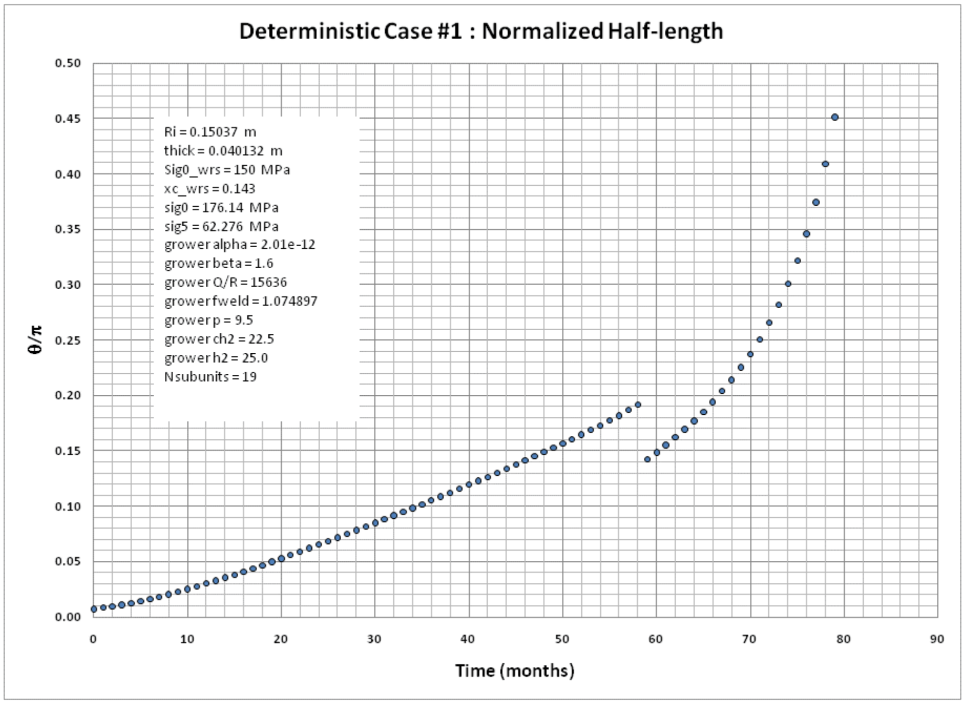

Fig. 5. SIAM_v1.01_S2.1_00001_001 normalized half-length of growing crack.

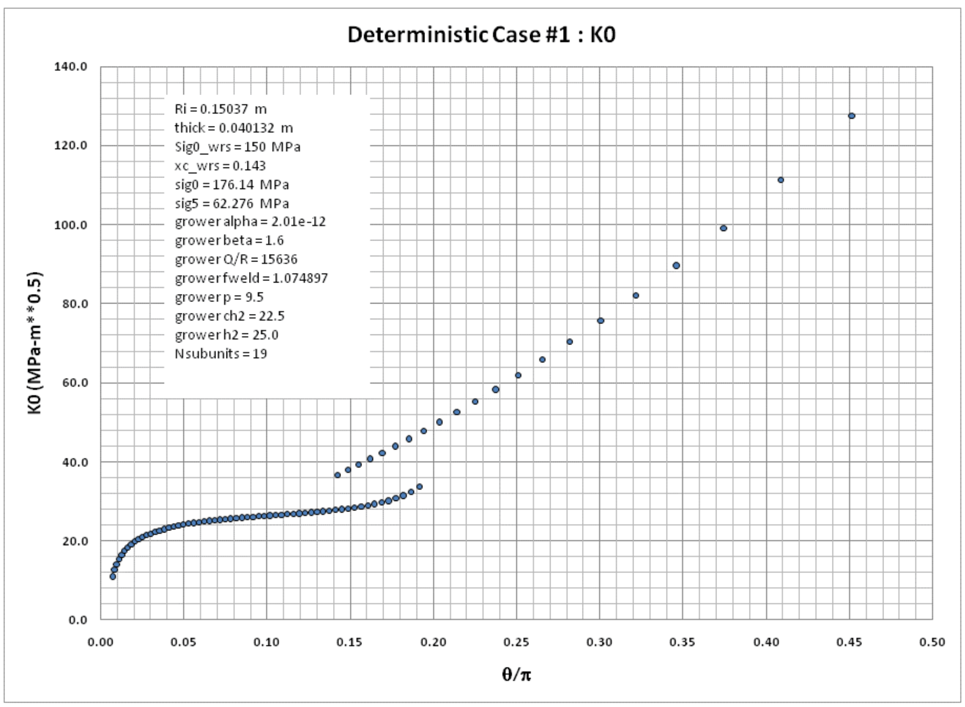

Fig. 6. SIAM_v1.01_S2.1_00001_001 driving force as a function of normalized half-length at highest point of crack near the ID.

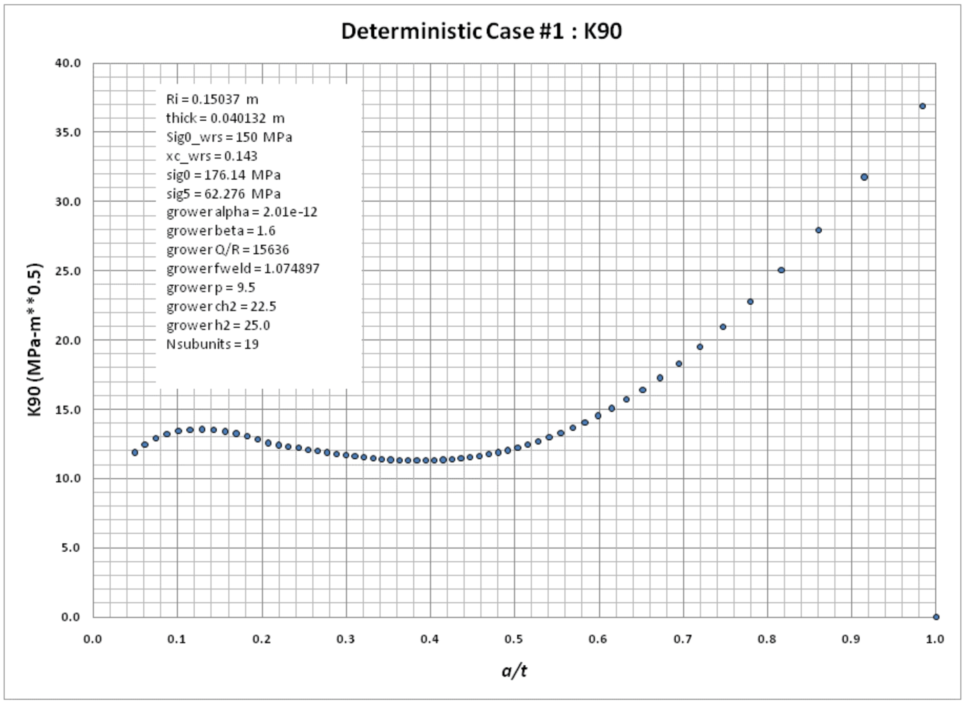

Fig. 7. SIAM_v1.01_S2.1_00001_001 driving force as a function of normalized crack depth at deepest point of crack.

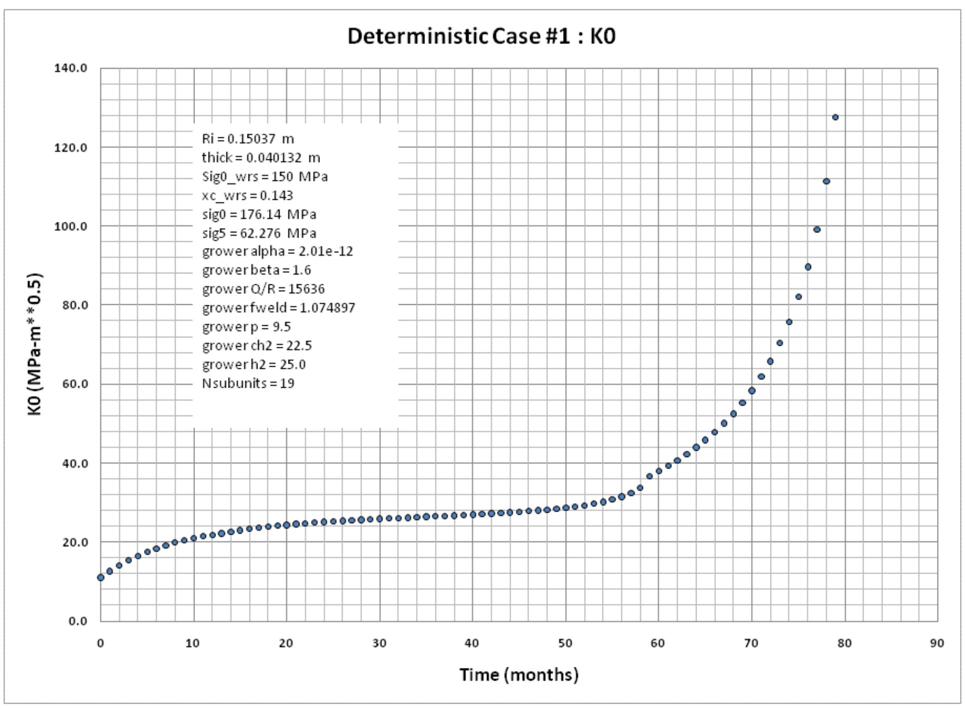

Fig. 8. SIAM_v1.01_S2.1_00001_001 driving force as a function of time at highest point of crack near the ID.

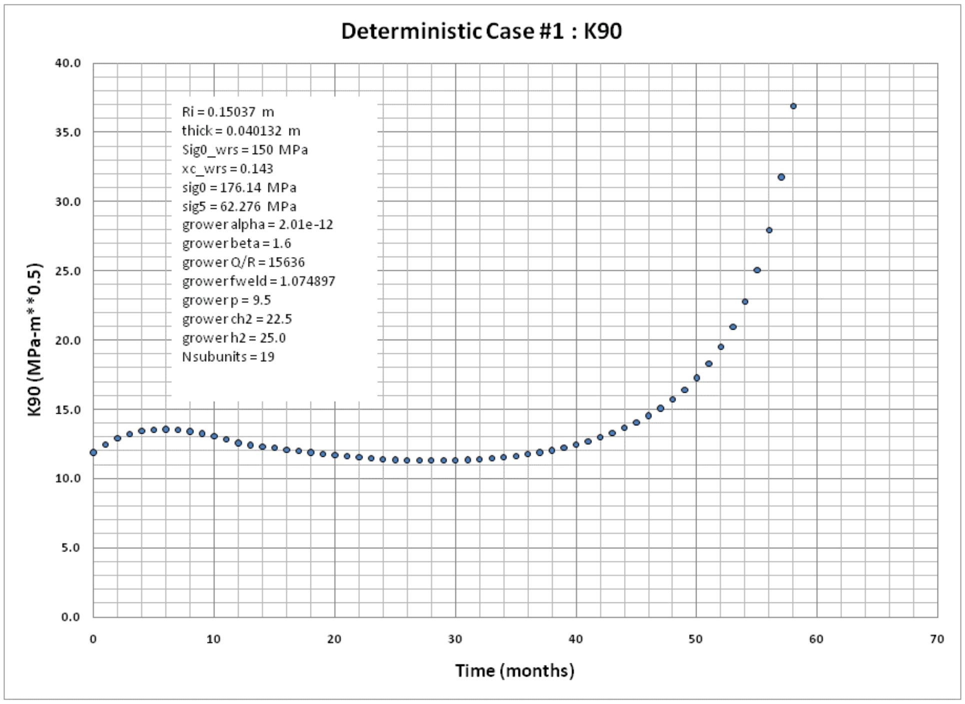

Fig. 9. SIAM_v1.01_S2.1_00001_001 driving force as function of time at deepest point of crack.

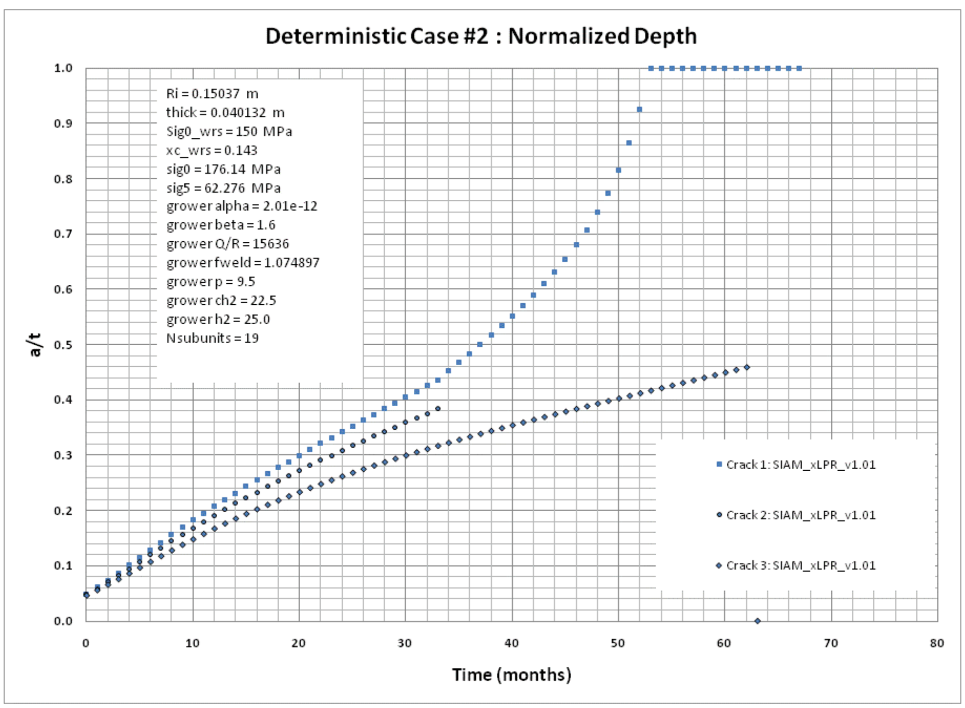

Fig. 10. SIAM_v1.01_S2.1_00001_002 normalized depth as a function of time for three growing cracks.

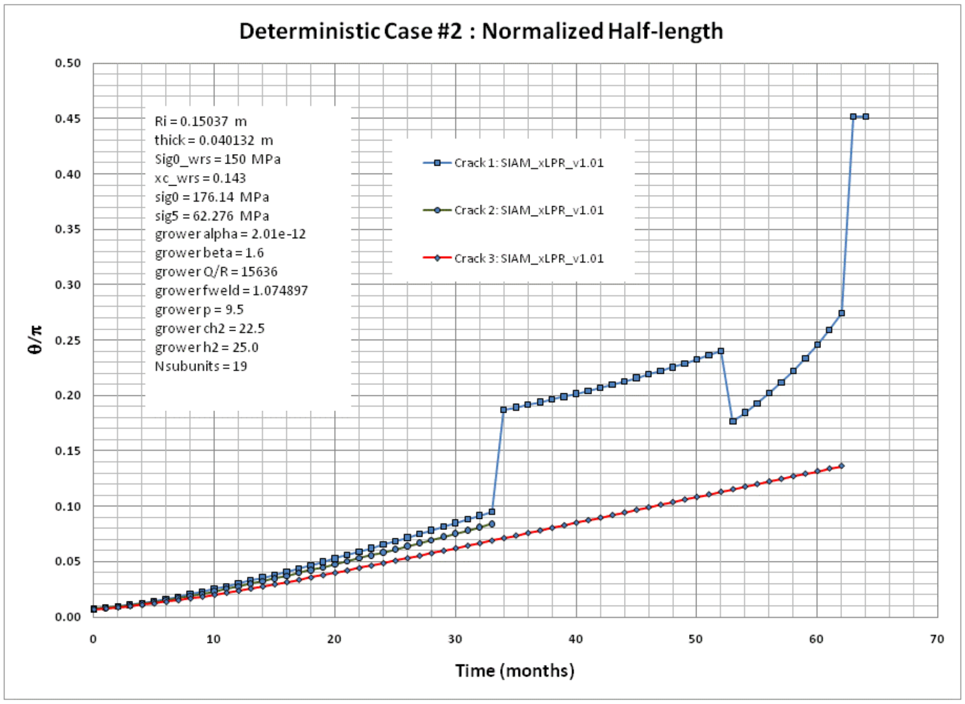

Fig. 11. SIAM_v1.01_S2.1_00001_002 normalized half-length as a function of time for three growing cracks.

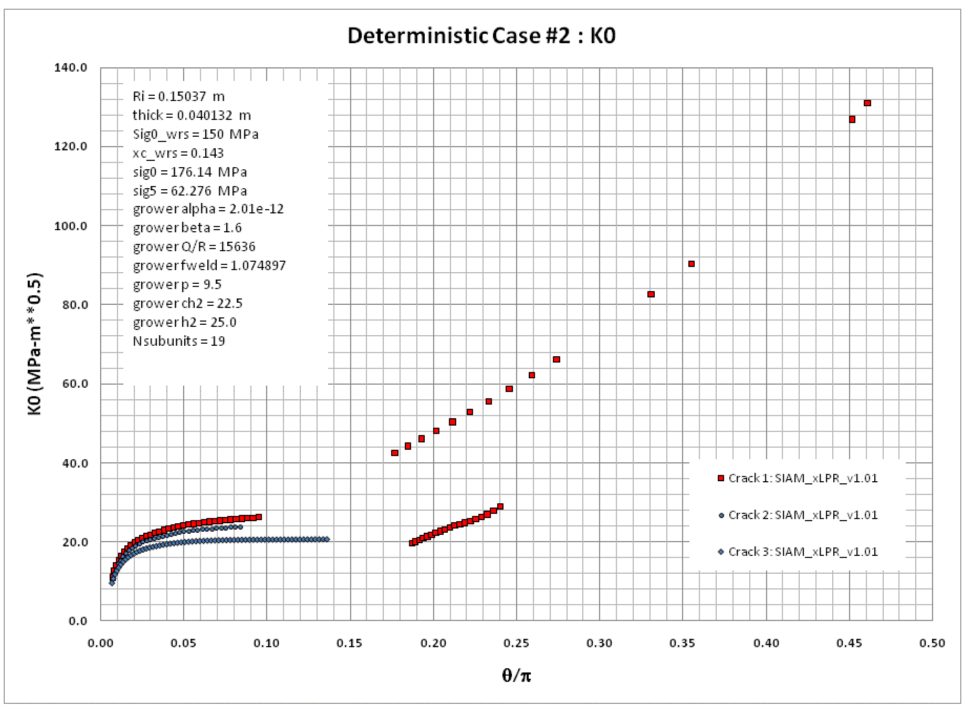

Fig. 12. SIAM_v1.01_S2.1_00001_002 driving force at point near the ID as a function of normalized half-length for three growing cracks.

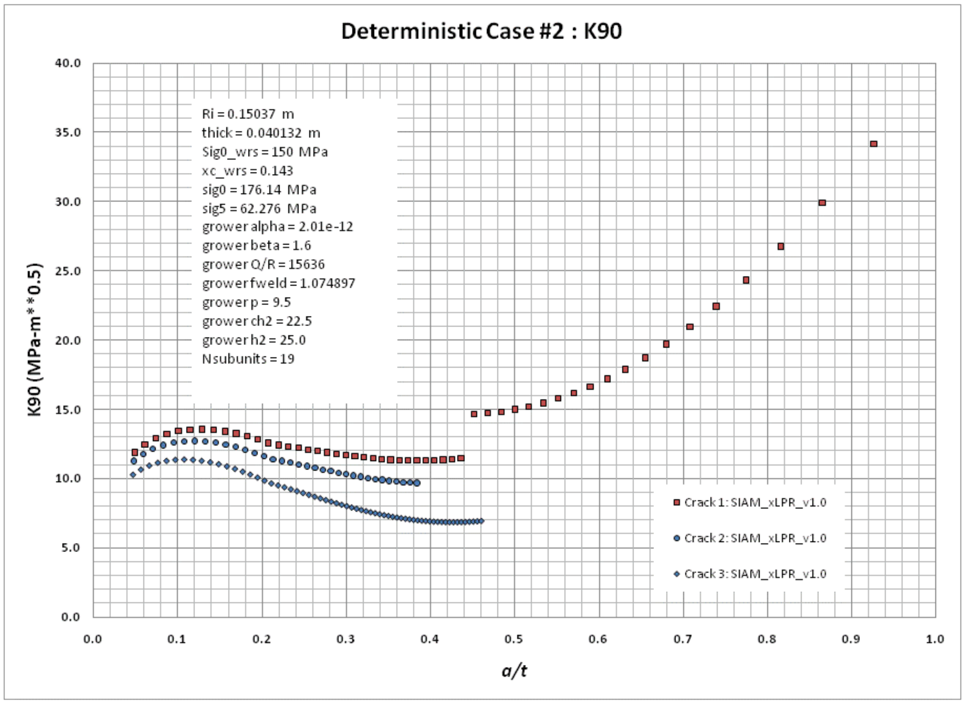

Fig. 13. SIAM_v1.01_S2.1_00001_002 driving force at deepest point of crack as a function normalized depth for three growing cracks.

7. EVALUATION OF PROBABILISTIC ANALYSES

7.1 STABILITY TESTING (SECT. 2.2 IN REF. [30])

Section 2.2 of ref. [30] requires a number of tests aimed at estimating the stability and convergence characteristics of probabilistic runs.

Model stability testing activities include three types of stability tests: *statistical stability*, *temporal stability*, and *spatial stability* or discretization. Collectively, these three tests are referred to as *model stability testing*. Statistical and temporal stability testing are required by ref. [30].

Statistical stability testing involves a number of activities related to demonstrating that a sufficient number of stochastic realizations have been run to achieve a numerically stable mean, including: (1) determining confidence intervals (generating several replicates with different random seeds and using a t-test) around selected output (SIAM_v1.01_S2.2a_05000_001 through SIAM_v1.01_S2.2a_ 05000_003, see Fig. 14 showing a 95% confidence interval about the mean probability of rupture as a function of time); (2) demonstrating numerical accuracy of the mean results by comparing the results of the base case with analyses using more realizations and different random seeds (SIAM_v1.01_ S2.2a_05000_001 and SIAM_v1.01_S2.2a_ 05000_004 through SIAM_v1.01_S2.2a_ 05000_006, see Fig. 15 showing a 95% confidence interval about the mean probability of rupture as a function of time).

Temporal stability refers to the use of an appropriate time step size necessary to achieve a stable solution. The time steps must collectively encompass the range of events and processes. The degree of stability is shown in graphical comparisons of the results of the stability analysis, using time steps as short as one month, two months, six months, and one year (SIAM_v1.01_S2.2b_ 10000_001 through SIAM_v1.01_S2.2b_ 10000_004, see Fig. 16 showing a 95% confidence interval about the mean probability of rupture as a function of time).

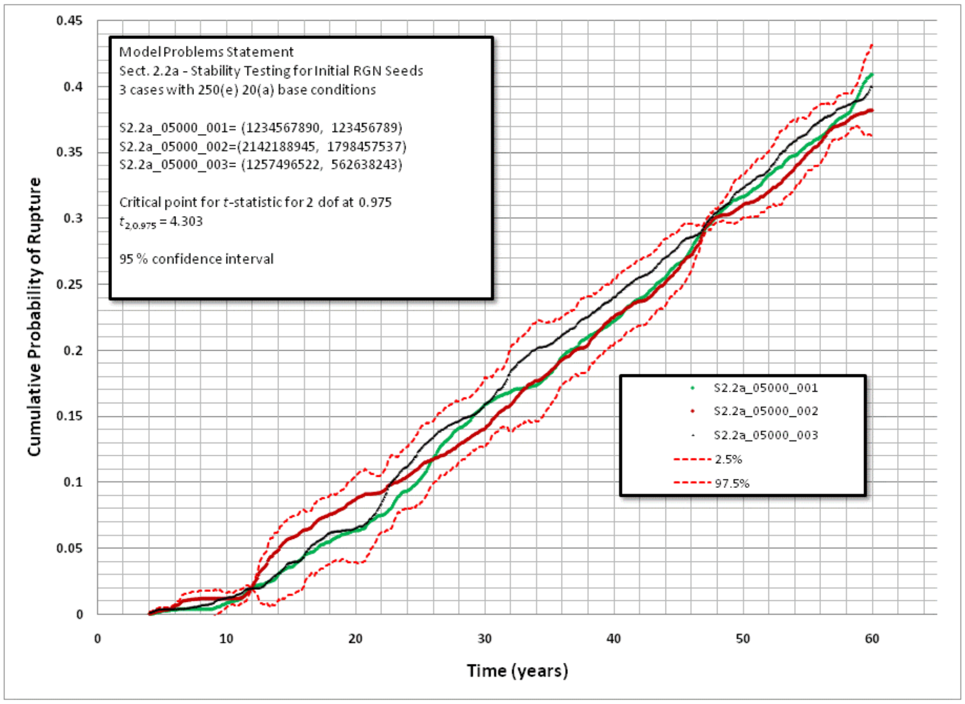

Fig. 14. SIAM_v1.01_S2.2a_05000_001 through SIAM_v1.01_S2.2a_05000_003 showing the results of a t-analysis establishing a 95% confidence interval about the mean probability of rupture as a function of time with varying initial random number generator seeds.

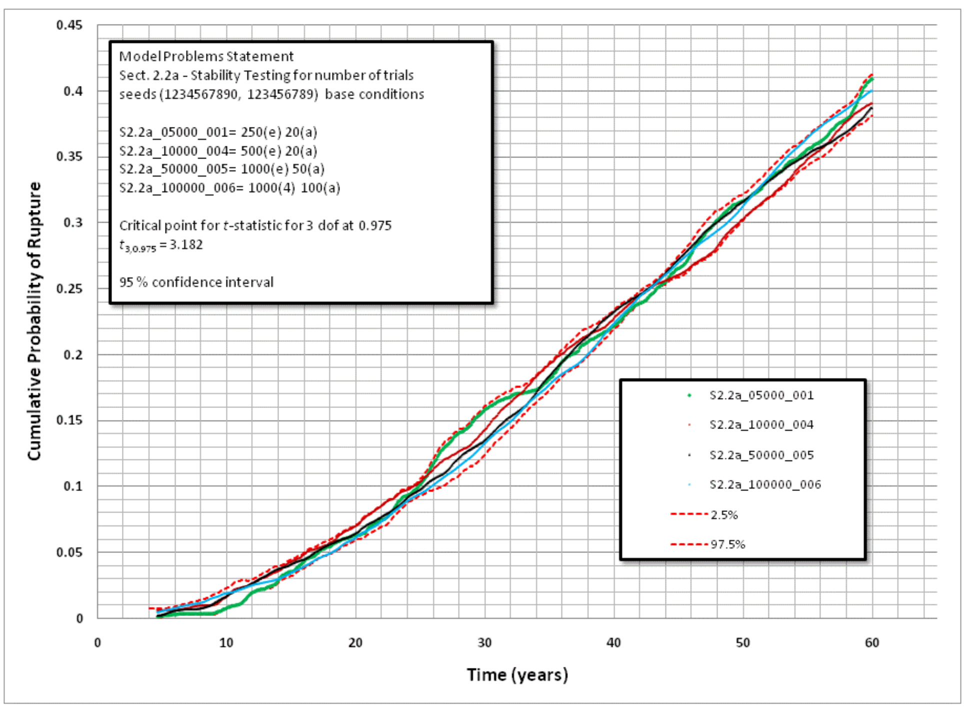

Fig. 15. SIAM_v1.01_S2.2a_05000_001 and SIAM_v1.01_S2.2a_10000_004 through SIAM_v1.01_S2.2a_100000_006 showing the results of a *t*-analysis establishing a 95% confidence interval about the mean probability of rupture as a function of time with varying number of trials.

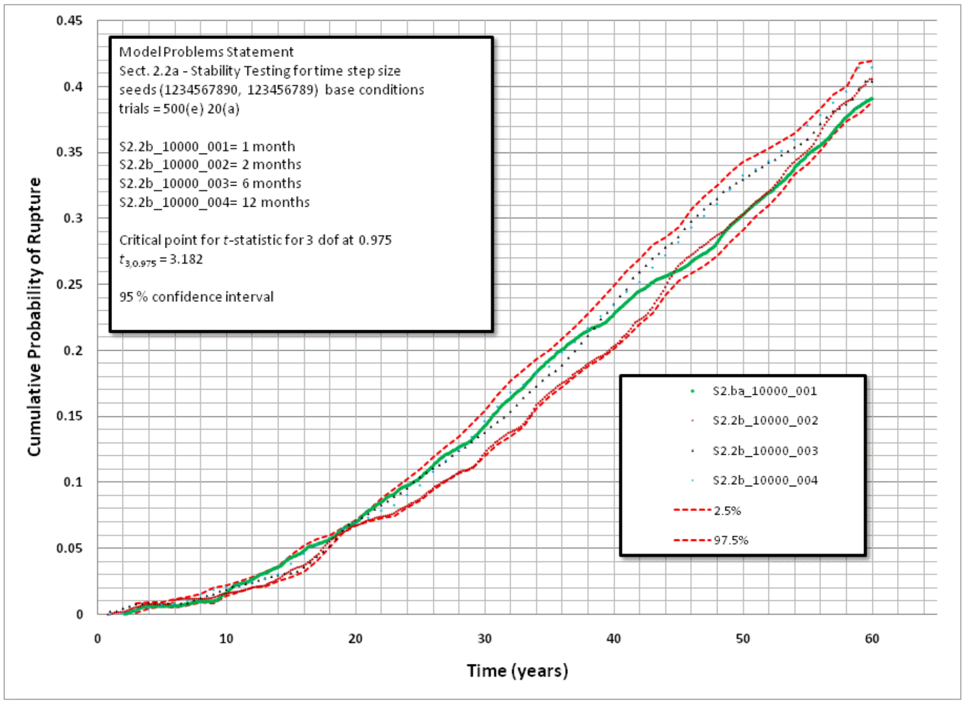

Fig. 16. SIAM_v1.01_S2.b_10000_001 through SIAM_v1.01_S2.b_10000_004 showing the results of a *t*-analysis establishing a 95% confidence interval about the mean probability of rupture as a function of time with time step size.

7.2 PROBABILISTIC BASE CASE ANALYSIS (SECT. 3.1 IN REF. [30])

Two probabilistic base case analyses (SIAM_v1.01_S3.1_10000_001 and SIAM_v1.01_ S3.1_50000_002) were run where the difference between the two was only the sample size using the Monte Carlo method. The base case consists of the surge nozzle geometry, with the appropriate loads and inputs taken from published data. The main driver for PWSCC is the welding residual stress; therefore for the base case the welding residual stresses assumed are shown in Figure 1 of ref. [30]. In this figure, the surge nozzle is assumed to have an ID repair and an Alloy 182 fill-in weld for seating the thermal sleeve. It is assumed that the safe end weld is far away from the dissimilar metal weld.

The outputs to be generated are given in Sect. 3.2.2 of ref. [30]. Figures 17 through 27 present selected results from SIAM_v1.01_S3.1_50000_002 where the mean and 95th percentile are shown with expected values averaged over aleatory uncertainty as described in Sect. 3.2.1 of ref. [30].

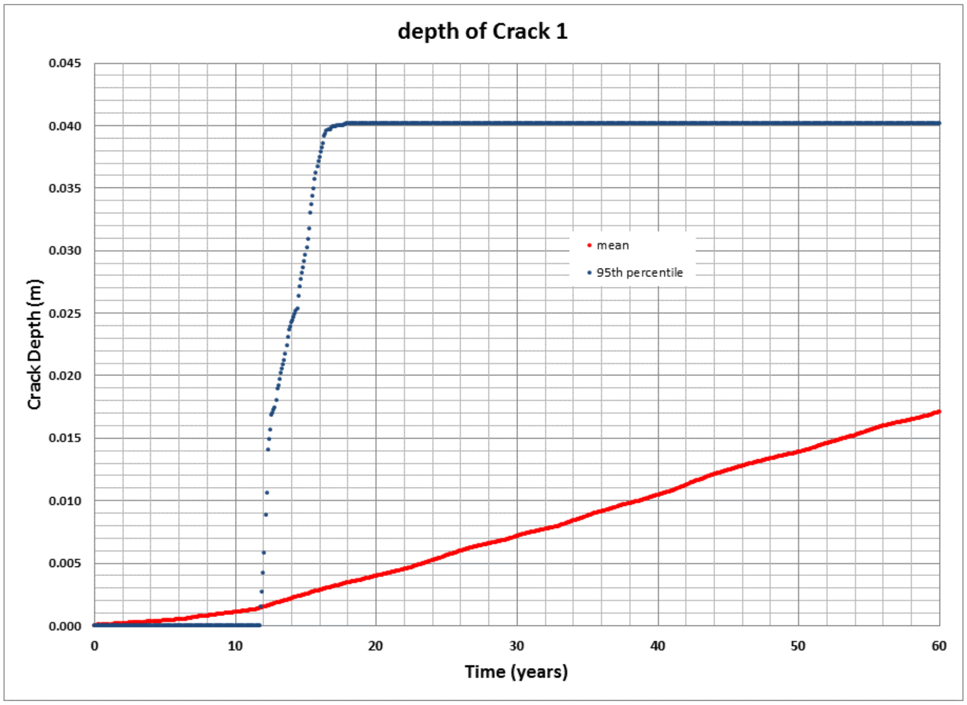

Fig. 17. SIAM_v1.01_S3.1_50000_002 showing time-dependent crack depth (expected over aleatory uncertainty) for crack 1.

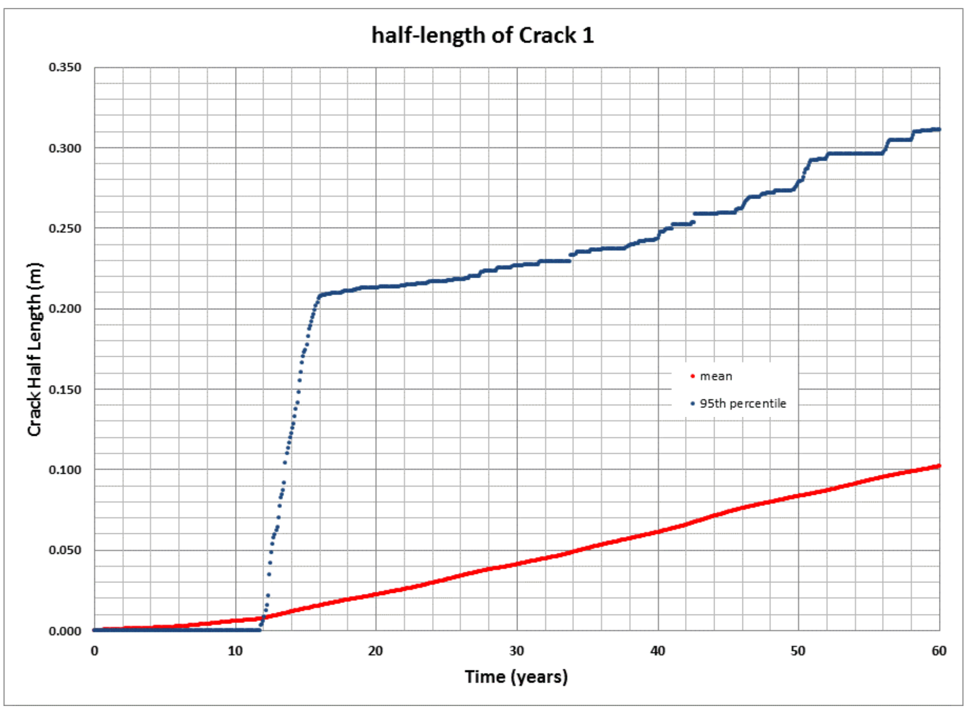

Fig. 18. SIAM_v1.01_S3.1_50000_002 showing time-dependent crack half-length (expected over aleatory uncertainty) for crack 1.

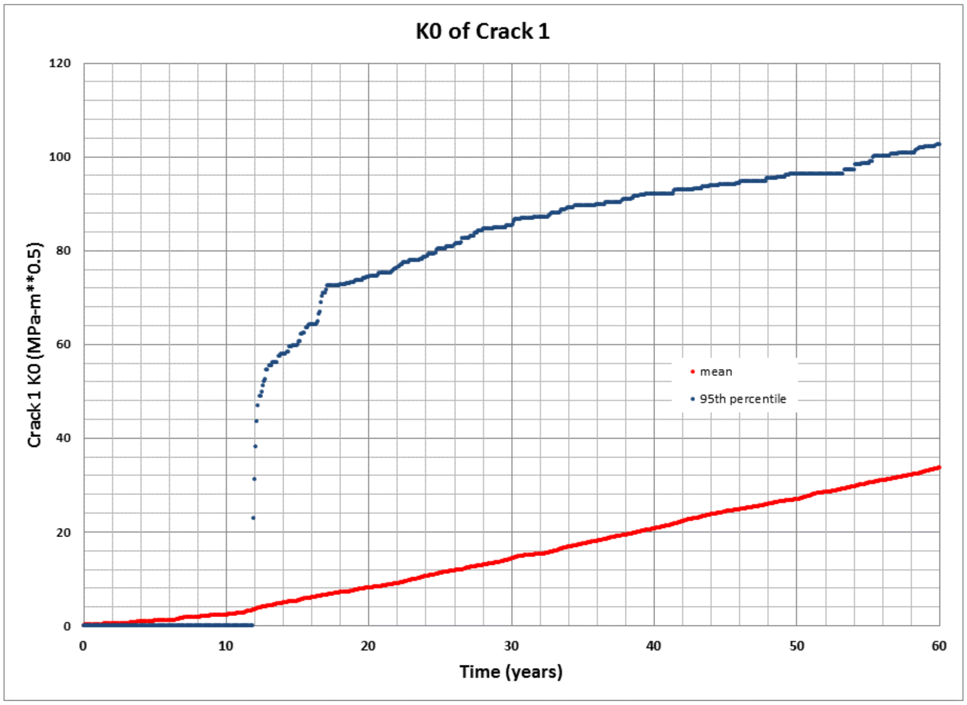

Fig. 19. SIAM_v1.01_S3.1_50000_002 showing time-dependent stress intensity factor at point near the ID (expected over aleatory uncertainty) for crack 1.

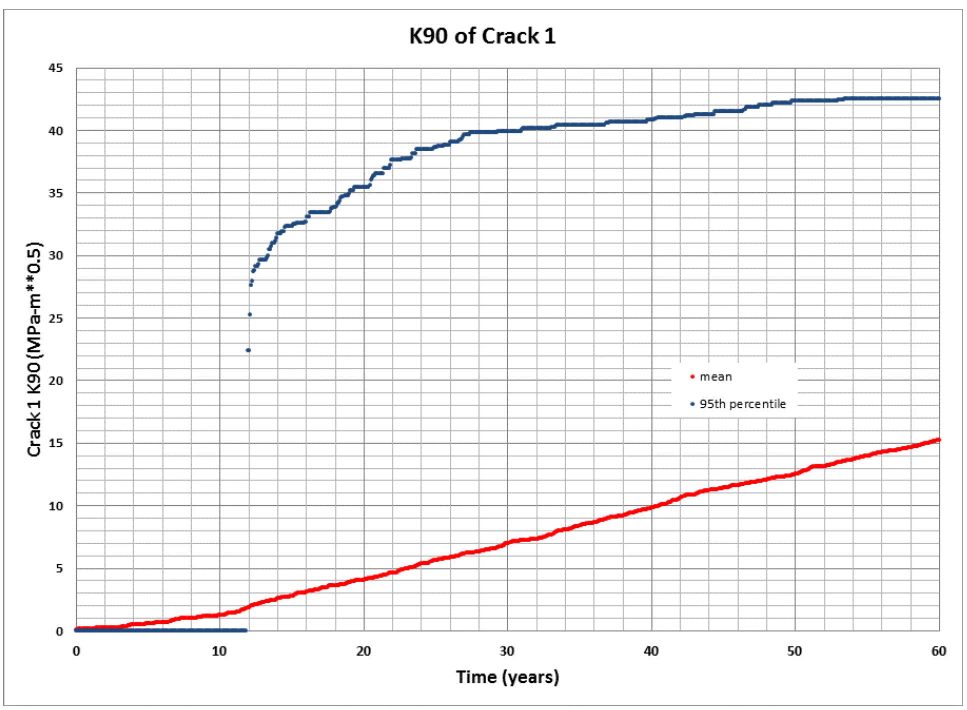

Fig. 20. SIAM_v1.01_S3.1_50000_002 showing time-dependent stress intensity factor at deepest point (expected over aleatory uncertainty) for crack 1.

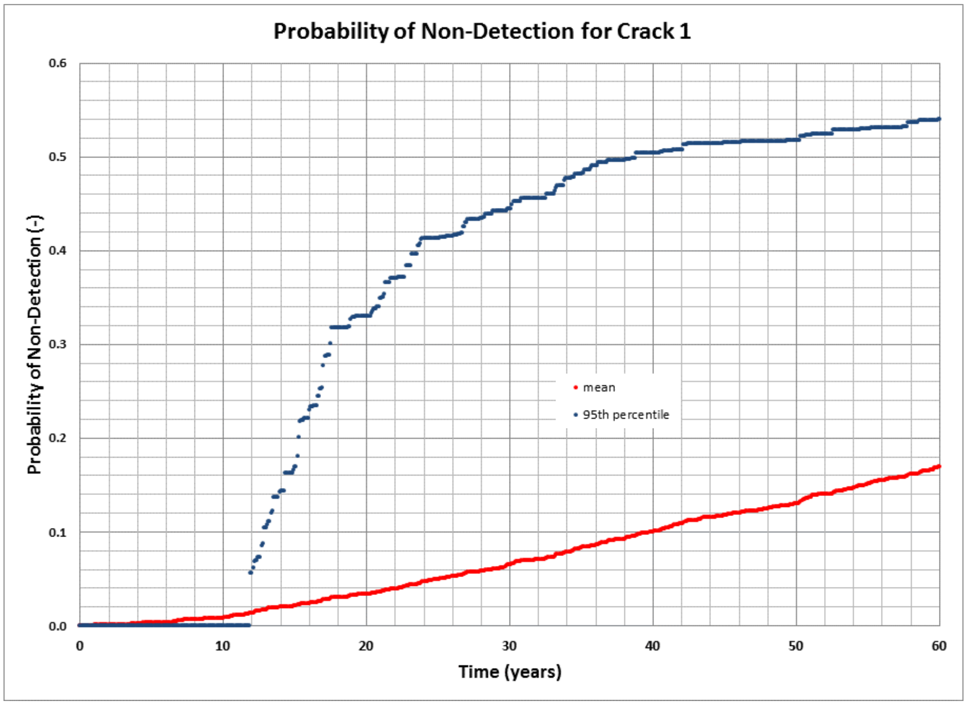

Fig. 21. SIAM_v1.01_S3.1_50000_002 showing time-dependent probability of non-detection (expected over aleatory uncertainty) for crack 1.

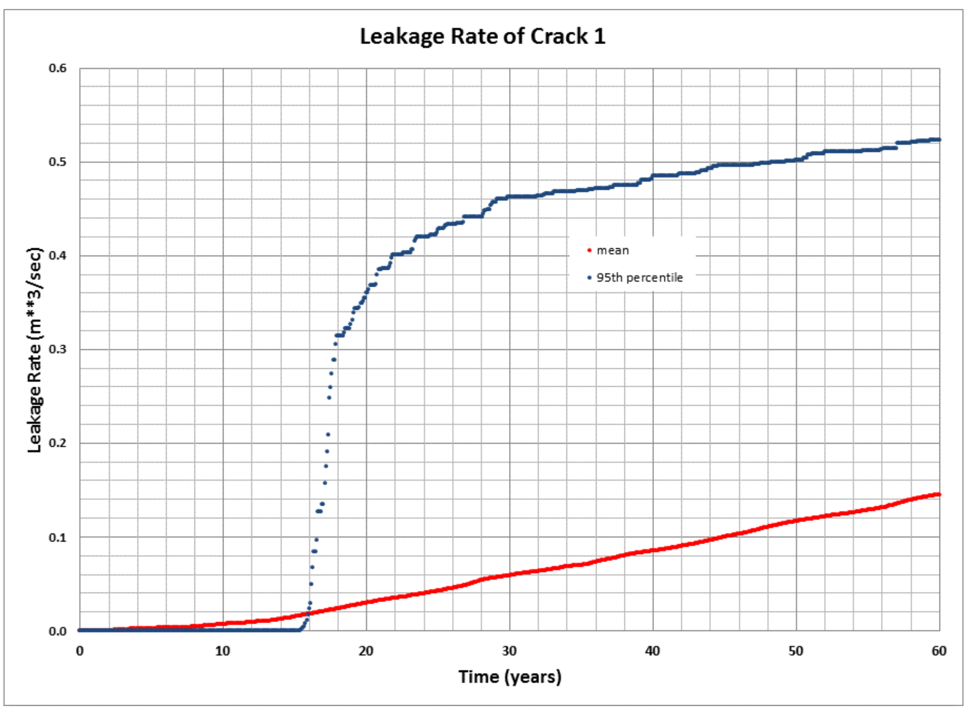

Fig. 22. SIAM_v1.01_S3.1_50000_002 showing time-dependent leakage rate (expected over aleatory uncertainty) for crack 1.

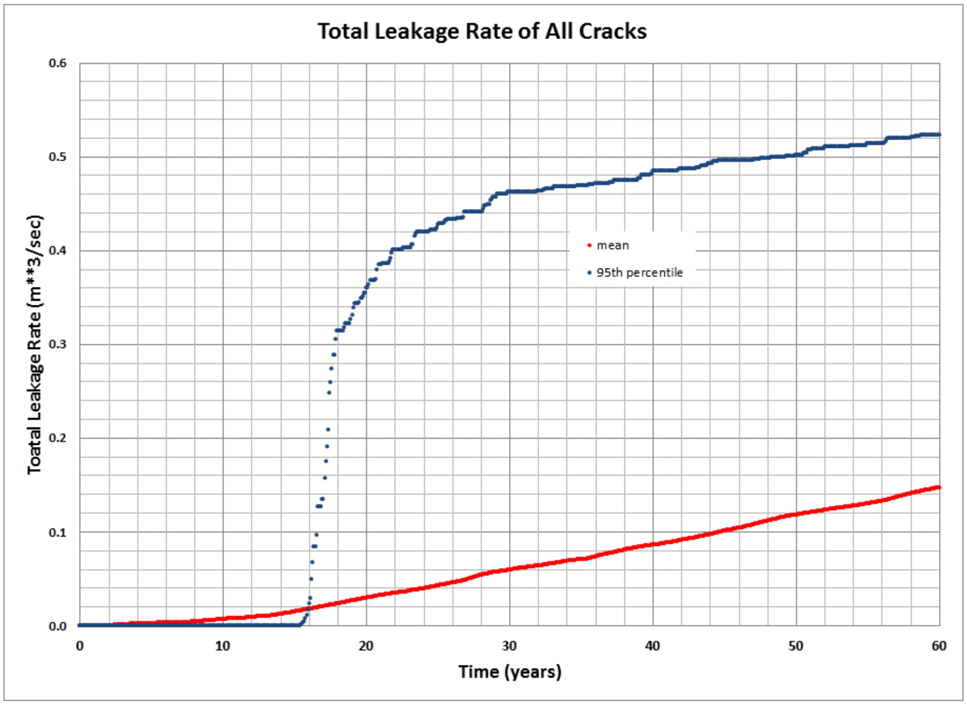

Fig. 23. SIAM_v1.01_S3.1_50000_002 showing time-dependent total leakage rate (expected over aleatory uncertainty) for all cracks.

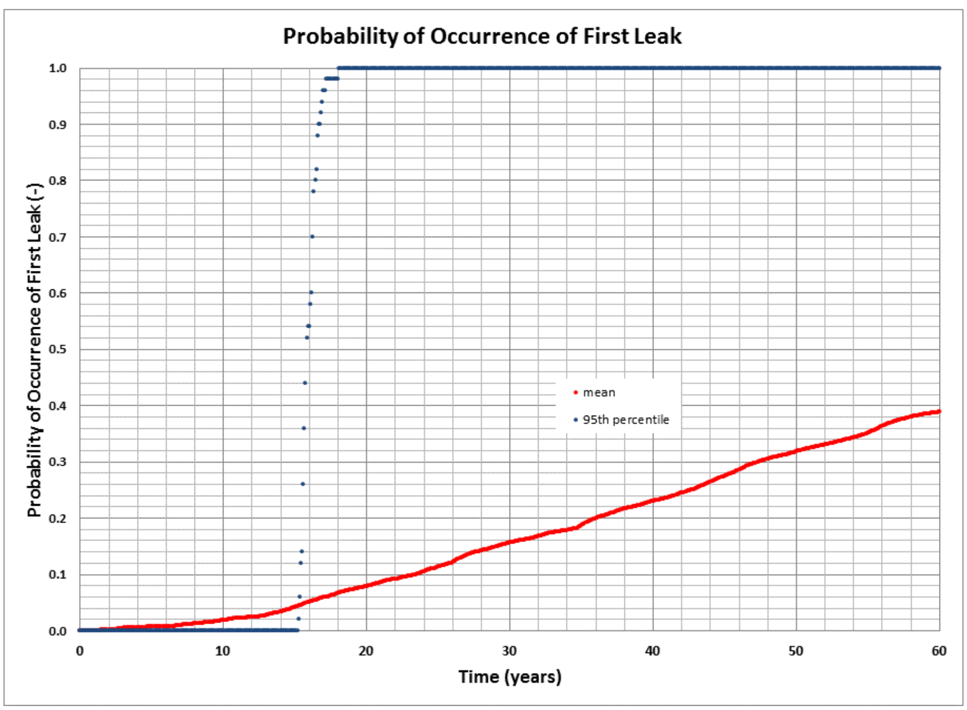

Fig. 24. SIAM_v1.01_S3.1_50000_002 showing time-dependent first leakage probability.

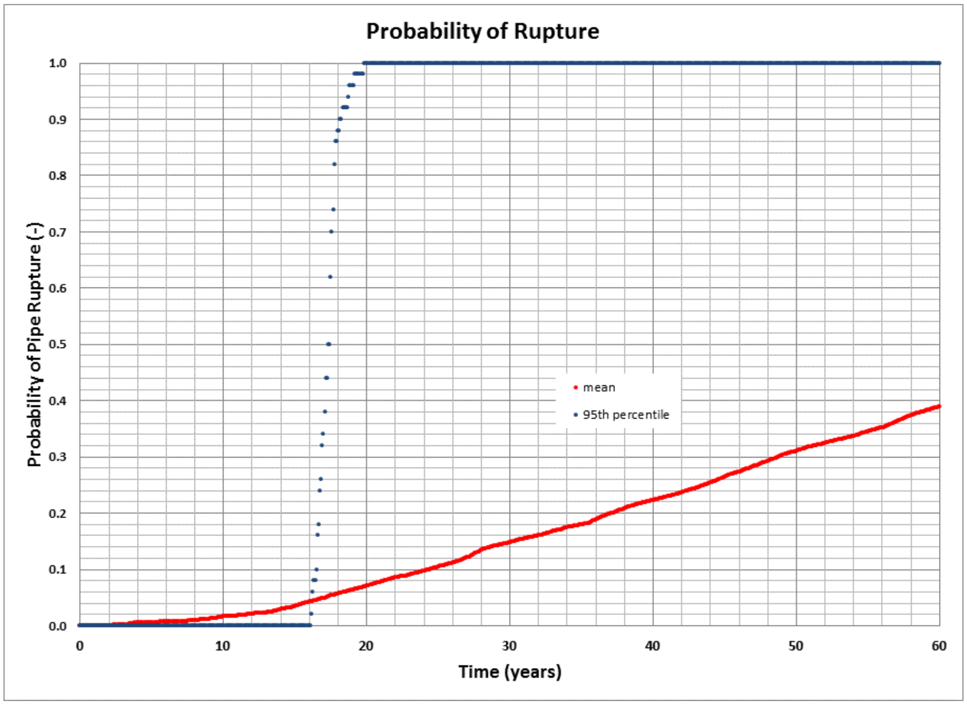

Fig. 25. SIAM_v1.01_S3.1_50000_002 showing rupture probability as a function of time.

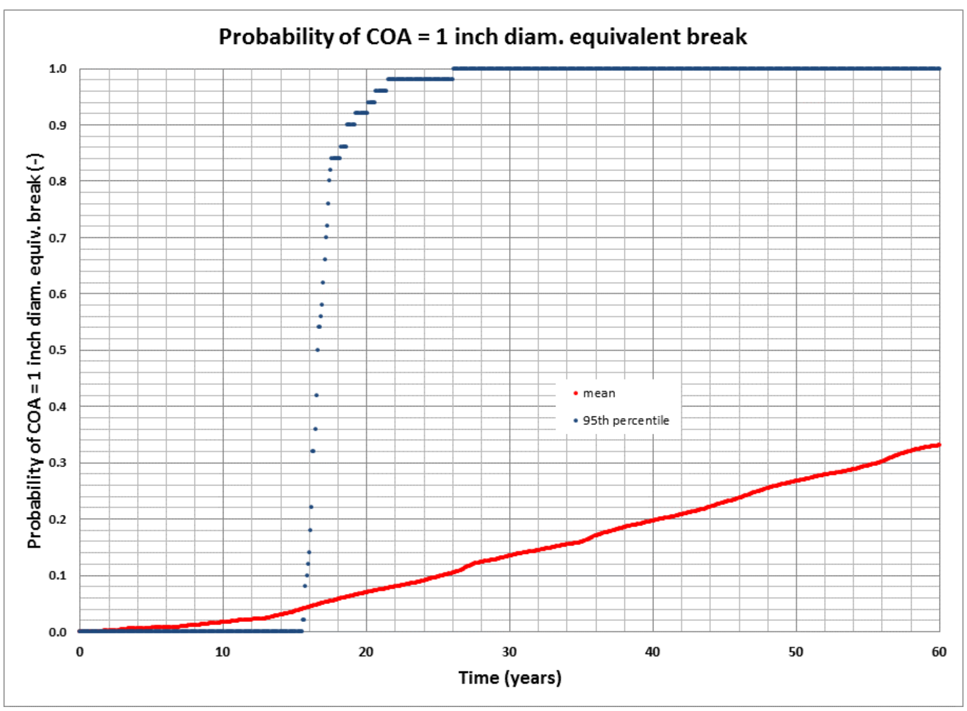

Fig. 26. SIAM_v1.01_S3.1_50000_002 showing time-dependent probability of a COA=1 inch equivalent break diameter (506.71 mm^2).

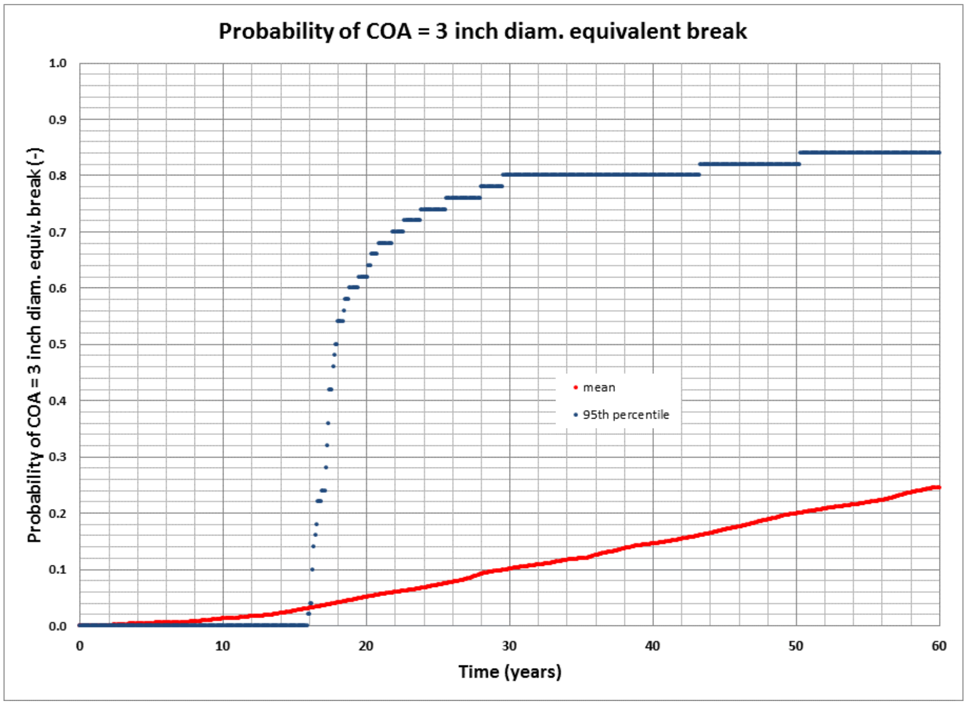

Fig. 27. SIAM_v1.01_S3.1_50000_002 showing time-dependent probability of a COA=3 inch equivalent break diameter (4560.37 mm^2).

7.3 SENSITIVITY ANALYSES (SECT. 4.0 IN REF. [30])

A set of sensitivity analysis are described in ref. [30] to demonstrate the SIAM-xLPR model functionality. These analyses are used to evaluate the impacts of some of the modeling assumptions and various alternative model processes not selected for the base case analysis.

7.3.1 Effect of Safe End Length (Sect. 4.1 in ref. [30])

As described in ref. [30], the stainless steel safe end weld that attaches the safe end to the surge nozzle piping causes a through-thickness bending stress that can reduce the tensile inner diameter stresses at the dissimilar metal weld. The extent of the effect on the dissimilar metal weld is a direct function of the length of the safe end. In the base case for the pilot study, it was assumed that the safe end was long enough that the safe end weld did not affect the stresses in the dissimilar metal weld. This case (SIAM_v1.01_S4.1_50000_002, see Fig. 28) considers a short safe end length. A unique distribution for axial stress component for the epistemic parameter S0_WRS and Xc are used to impose this short safe end condition. It is assumed that the distributions are normal and the parameters are specified by their mean and standard deviation (StDev):

- S0_WRS = -16.2MPa mean and 117MPa StDev (max = 300MPa, min = -300MPa)
- Xc = 0.18 mean and 0.036 StDev (max = 0.5, min = 0.1)

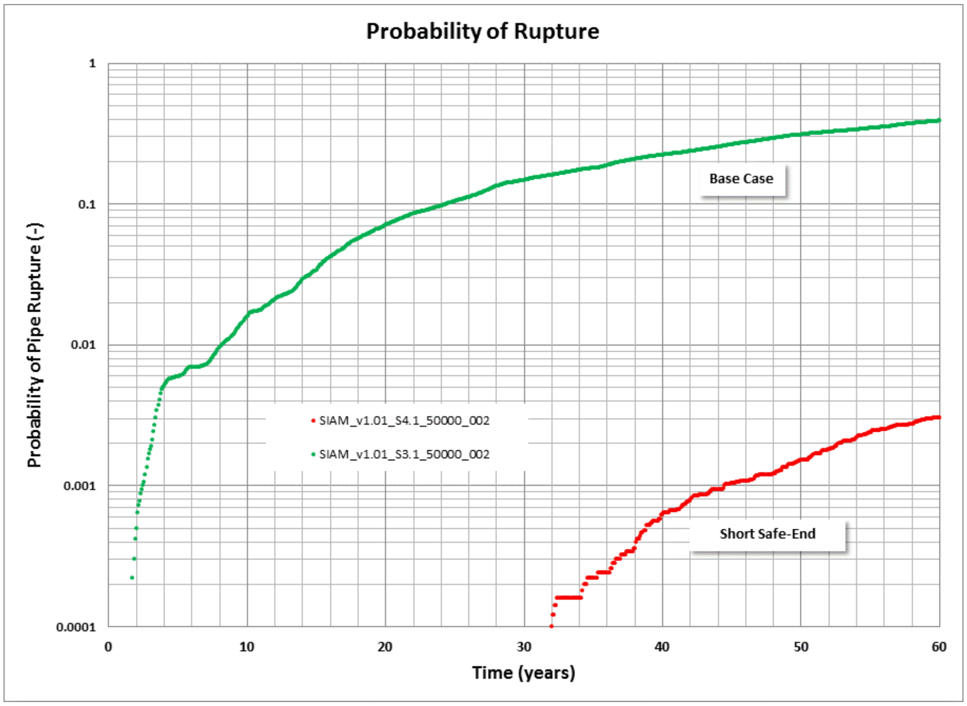

Fig. 28. SIAM_v1.01_S4.1_50000_002 showing rupture probability as a function of time for short safe compared to base case.

7.3.2 Effect of Stress Mitigation (Sect. 4.2 in ref. [30])

Sect. 4.2 of ref. [30] describes three mitigation analysis cases. These cases evaluate different mitigation times, as well as the mitigation effectiveness over the representative distributions for *Sigma0_wrs_mitigated* and *Xc_mitigated*.

SIAM_v1.01_S4.2_50000_007 Mitigation time 10 years

SIAM_v1.01_S4.2_50000_005 Mitigation time 20 years

SIAM_v1.01_S4.2_50000_006 Mitigation time 30 years

The distributions of welding residual stress to be used for the mitigation to apply a normal distribution of two parameters that control the WRS distribution in the model are:

Sigma0_wrs_mitigated = -344.75 MPa mean and 34 MPa StDev (min = -447, max = -242)

Xc_mitigated = 0.38 mean and 0.038 StDev (min = 0.26, max = 0.5)

The results of these analyses are shown in Fig. 29 comparing the mean probability of rupture.

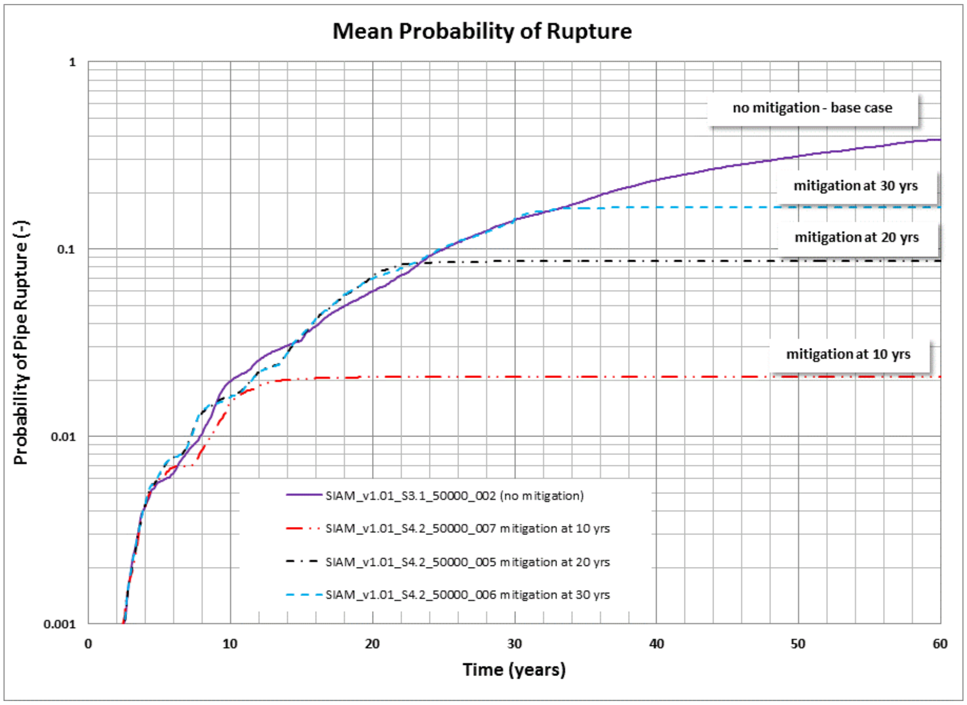

Fig. 29. SIAM_v1.01_S4.2_50000_005 through 007 showing mean rupture probability as a function of time of a mitigation action compared to the base case with no mitigation.

7.3.3 Crack Initiation Model Uncertainty (Sect. 4.3 in ref. [30])

The crack initiation module includes three alternative models for crack initiation where Method 2 has been applied in the base case. For this sensitivity analysis (SIAM_v1.01_S4.3_50000_002), Method 1 was run and compared to the base case. Fig. 30 presents the results of this comparison for the mean probability of rupture averaged over the aleatory uncertainty.

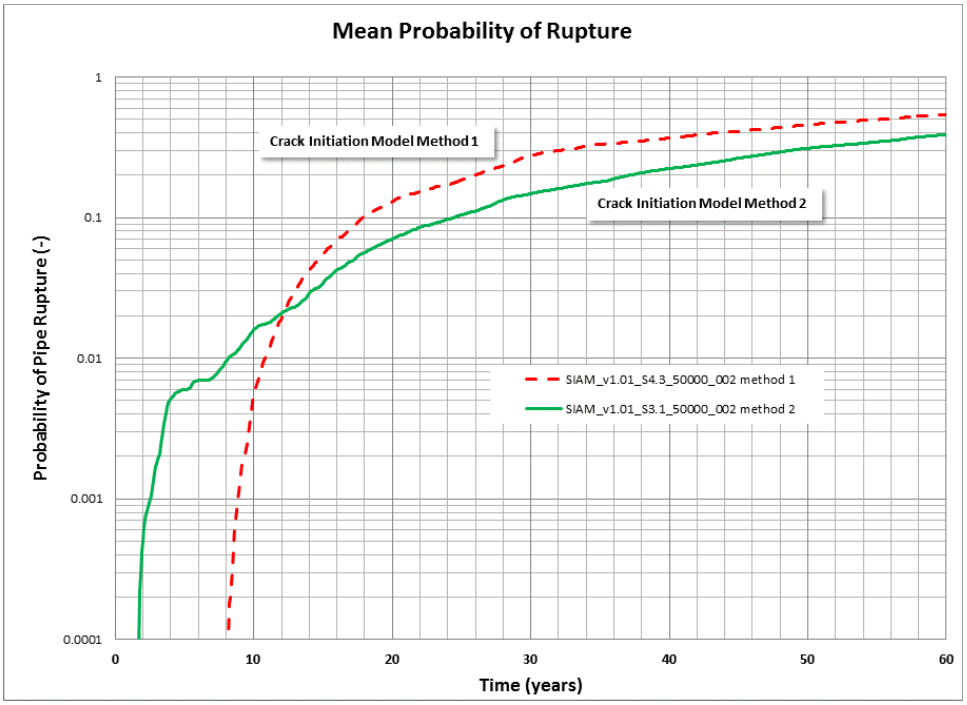

Fig. 30. SIAM_v1.01_S4.3_50000_002 showing mean rupture probability as a function of time of comparing the results of Crack Initiation Method 1 to the base case Method 2.

7.3.4 Chemical Mitigation (Sect. 4.4 in ref. [30])

Two sensitivity cases were run (SIAM_v1.01_S4.4_10000_001 and SIAM_v1.01_S4.4_10000_002) two examine the effects on crack growth of increasing the hydrogen concentration in the primary water system. For the base case (SIAM_v1.01_S3.1_10000_001), the hydrogen concentration was set at 25 cc/kg-STP. For these two sensitivity analyses, the hydrogen concentration was increased to 50 and 80 cc/kg-STP to determine the effect. The results are shown in Fig. 31 for the mean probability of rupture.

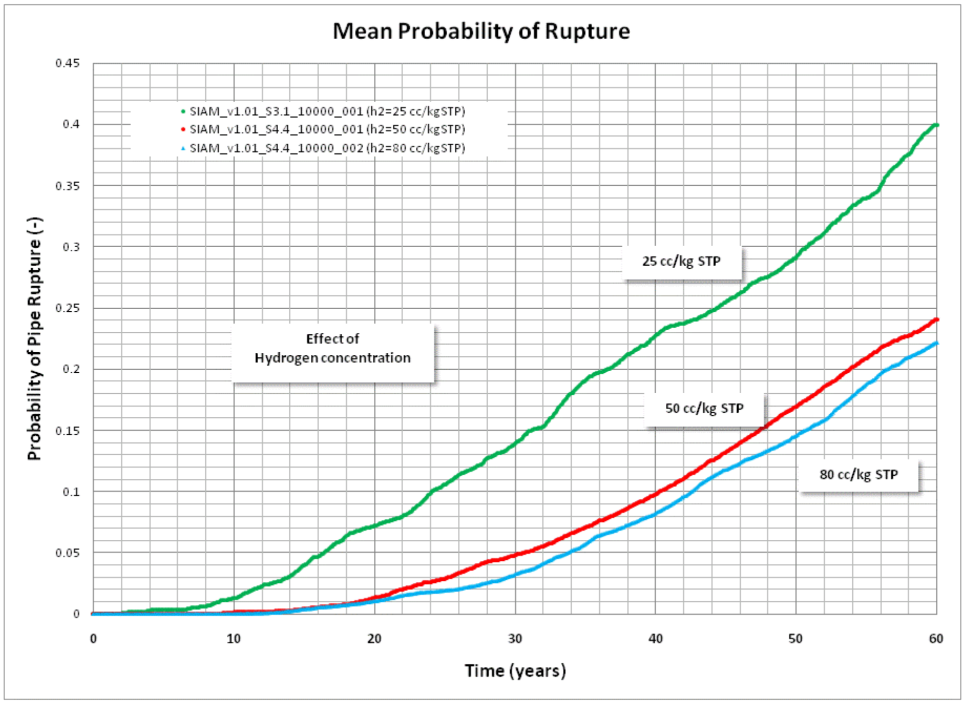

Fig. 31. SIAM_v1.01_S4.4_10000_001 and 002 showing mean rupture probability as a function of time comparing the results of increasing hydrogen concentrations to the Base Case value of 25 cc/kg STP.

7.4 POST-PROCESSING ANALYSES (SECT. 5.0 IN REF. [30])

The base case and sensitivity analyses have been post-processed using a set of tools developed to evaluate the extremely low probability failures. The desired output, defined in Section 3.2 of ref. [30], including inspection and leak detection have been evaluated using post-processing analyses and the post processing code developed by SNL for the xLPR Pilot Study.

7.4.1 Leak Detection Capability (Sect. 5.1 in ref. [30])

In order to demonstrate the code's capability to model the detection of leaks, the output of selected cases were analyzed to demonstrate the effect on the output mean probability of rupture for leak detection limits of 0.1, 1, 10, and 50 gpm. The cases to be analyzed will include

- Base case (SIAM_v1.01_S5.1_1000_001 through 004)
- Short safe end case (SIAM_v1.01_S5.1_1000_005 through 008)

It was not necessary to redo any specific analysis when leak detection capability was changed. It was postulated that once a leak is detected, the weld will be replaced and will not fail again. Therefore all calculations are done assuming that the leaks are not detected, which can lead under certain conditions to pipe rupture. The user can select a detection threshold that will lead to a correction of output data of interest if a leak is detected. Figs. 32 and 33 show the results of these sensitivity cases.

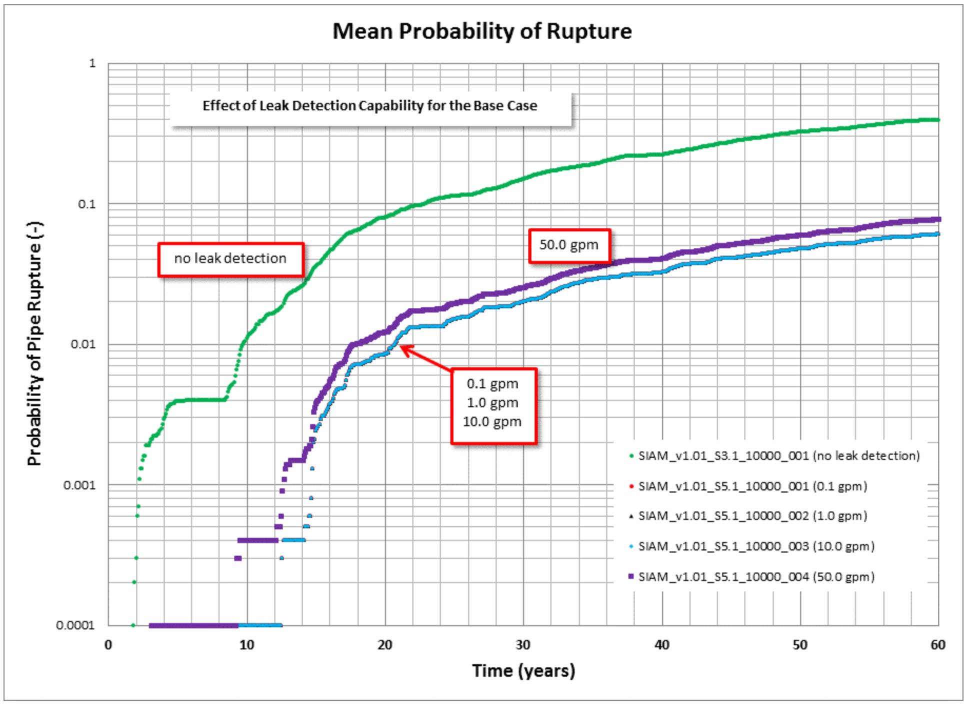

Fig. 32. SIAM_v1.01_S5.1_10000_001 through 004 showing mean rupture probability as a function of time comparing the results of increasing leak detection capability to the Base Case condition of no leak detection.

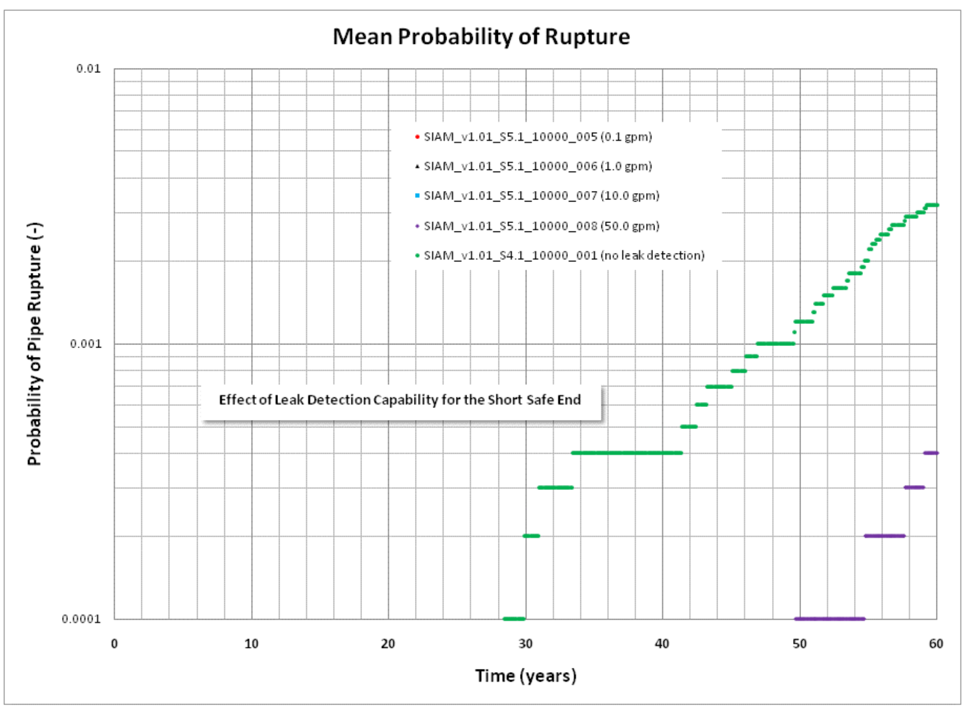

Fig. 33. SIAM_v1.01_S5.1_10000_005 through 008 showing mean rupture probability as a function of time comparing the results of increasing leak detection capability to the safe end condition of no leak detection.

7.4.2 Inspection Schedule (Sect. 5.2 in ref. [30])

The effect of in-service inspections is demonstrated with these sensitivity analyses. Inspections of every 5, 10, 20, and 30 years will be compared to the case of no inspection for the

- Base case (SIAM_v1.01_S5.2_1000_001 through 004, Fig. 34)
- Short safe end case (SIAM_v1.01_S5.2_1000_005 through 008, Fig. 35)

As with the leak detection capability cases, the effects of in-service inspection are addressed using the Post-Processing Utility developed by Sandia National Laboratories.

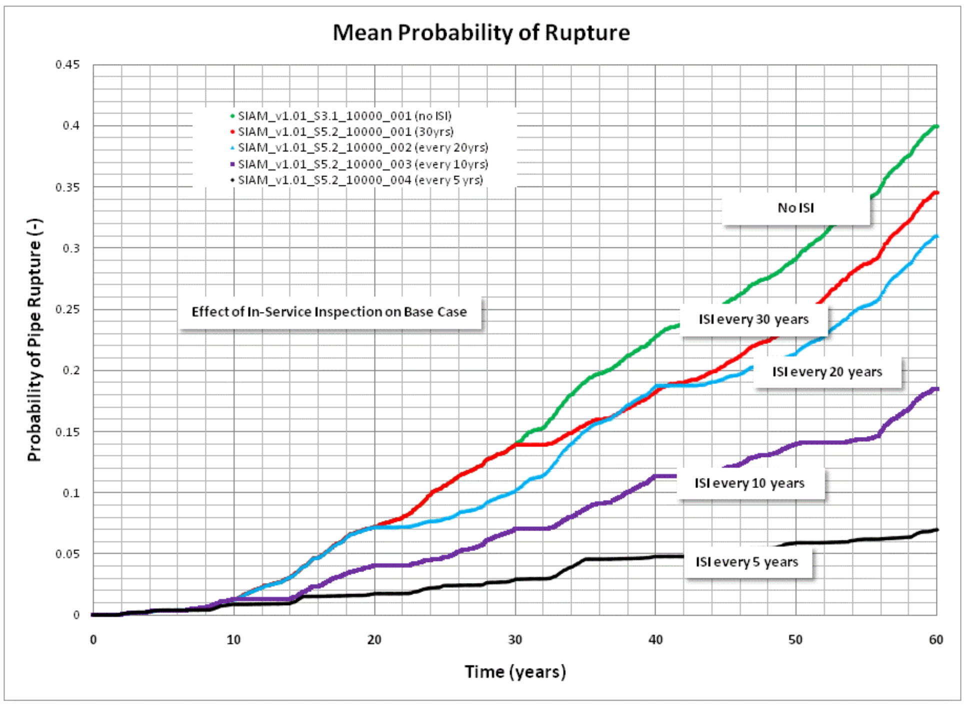

Fig. 34. SIAM_v1.01_S5.2_10000_001 through 004 showing mean rupture probability as a function of time comparing the results of varying the inspection schedule to the base case of no inspections.

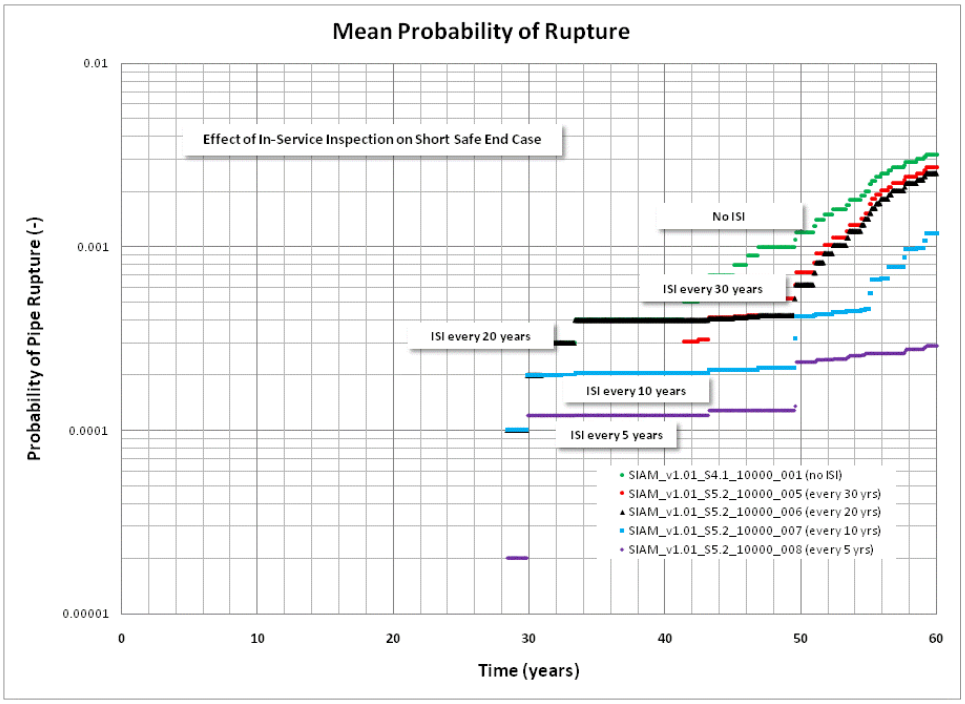

Fig. 35. SIAM_v1.01_S5.2_10000_005 through 008 showing mean rupture probability as a function of time comparing the results of varying the inspection schedule to the short safe end case of no inspections.

8. SELF-ASSESSMENT

With the development of SIAM-xLPR version 1, the ORNL team has contributed fully to the attainment of the goal of the xLPR Consortium to cooperatively develop a software tool that can perform a probabilistic analysis of the primary water piping system of a nuclear power plant. This analysis addresses, quantifies, and characterizes uncertainties in a modern PRA setting and provides a tool that assists in demonstrating compliance with the requirements of 10CFR Part 50, Appendix A, General Design Criteria 4.

The ORNL team has contributed to the xLPR Pilot Project by creating a complete software framework that provides an integration of the computational modules developed by the Models Group and the Inputs Group. A key feature of the SIAM-xLPR probabilistic framework is that it demonstrates that a scientific application at this level of complexity can be successfully developed consisting entirely of open-source software and open-source scientific computing libraries, including both software written by ORNL and software available from the open-source community. Another key element in the development of the SIAM-xLPR framework is that its architectural design is based on the modern paradigm of object-oriented programming. The software industry recognizes object-oriented programming as the optimal methodology to control and manage increasing complexity through the incremental and modular extension of large sophisticated software applications. The adoption of this open-source, object-oriented approach for SIAM-xLPR has served to benchmark and compare results with another approach using a commercial proprietary application, with the goal of assessing the advantages and disadvantages of each approach in the construction of a modular probabilistic framework for xLPR.

The ORNL team has developed and provided analytical output to aid in the assessment of the pressurizer surge nozzle dissimilar metal weld leakage probabilities and to provide relative, order-of-magnitude estimates of piping rupture probabilities. The ORNL team’s work has demonstrated that it is feasible to estimate the expected extremely low probabilities of rupture using the computational approaches and models developed by the teams participating in the xLPR Pilot Project. The ORNL team has contributed to evaluating and testing some of the xLPR kernel modules to establish credibility of the code and the overall xLPR project. The ORNL team’s framework SIAM-xLPR was made available, along with the documentation, so others can execute the xLPR kernel, analyze data, and submit comments and improvements. The ORNL team followed standard SCM practices to maintain the source code and documentation of the work performed. All these tasks were successfully achieved within the ambitious schedule of this project.

A valuable lesson learned during this year came from reassessing the validity of our original estimates for the time and staffing resources required to complete this initial phase of the project. It is now clear that due to the complexity of the application and the aggressive schedules required to meet the goals of the xLPR Pilot Project, it was necessary to compress what should have been a 2 to 3 year software development cycle into less than one year. Future management planning in these areas will benefit from the experience gained during this year’s effort. We also have a better understanding of what future staffing requirements will be necessary and how to better incorporate time for application testing (both unit-testing and integrated testing) into our future software development plans.

9. CONCLUSIONS AND RECOMMENDATIONS

Conclusions:

1. The experience of implementing the open-source version of the xLPR project has been both challenging and rewarding. It has been challenging because it was necessary to build the whole framework from scratch using a multitude of resources: user input, output, and integration of the xLPR modules and utilities. It has been rewarding because the planned open-source system was successfully realized and is now available for installation and download on most versions of the Windows Operating System.

2. It is our conviction that ORNL's xLPR implementation should be designed in such a way that it facilitates our end users' work, fulfills their needs, and represents the best efforts of every member in the consortium. With this idea in mind, we selected our logo and graphic user interface for user input and output. SIAM-xLPR's presentation page represents a piping system combined with the logos of the different participating groups that form the xLPR national consortium. SIAM-xLPR's GUI was implemented and improved by taking into consideration the comments and suggestions of users within this consortium.

3. Analytical results generated by both Commercial and Open Source implementations were compared and benchmarked one against the other. That exercise contributed significantly to (1) providing some verification of those implementations, (2) achieving a better understanding of the integrated xLPR process, and (3) establishing a baseline for future harmonization, tuning, and improvement of the xLPR kernel modules.

4. Implementing an open-source system framework from scratch required a major effort from the ORNL team. Admittedly, we underestimated the level of effort and resources that it required. The experience gained during this Pilot Project phase will help us significantly in the planning process for future phases of the project.

5. The process of developing two implementations pushed both teams to give their best efforts, and the resulting two frameworks represent a high level of quality. The teams remained enthusiastic through the end of the project and the bonds among the xLPR consortium have grown stronger.

6. Having a CM process that involves completing Microsoft Word form files was better than not having any process in place. But teaching the xLPR CM process to members who were not familiar with it proved to be time consuming. This CM process also had other shortcomings, such as not having the advantages of a relational database system. In general, our team found it difficult to query and locate information on the xLPR CM system.

7. An added benefit derived from investing time in defining our core classes is that some of them can be re-used on other projects (for example: Variable, variate, random generator, Poisson, LHS, and more).

8. Python is a well-developed programming language, widely used within the scientific community. We found it to be a robust rapid development language during the creation of SIAM-xLPR.

9. When storing data, one should be careful using the most appropriate data storage mechanism

for scientific data. We spent significant time trying to develop a relational database for storage of results. But as our data model has the shape of a tree, where values hang as leaves from branches, the relational model was found not to be appropriate.

10. When presenting analysis results, one should be careful about using semi-log plots. We found that some subtle issues could be inadvertently masked by the use of such plots. For example, in the mean probability of rupture, at the time of pre-emptive mitigation, there is a sudden and pronounced increase in the slope of the mean cumulative probability of rupture vs. time curve. During the first year after mitigation, the curve begins to roll over and eventually to flatten out. On a semi-log plot, the latter behavior cannot be easily identified.

Recommendations:

1. We need to encourage Module developers to follow ORNL's proposal of FORTRAN programming best practices. If we had done so from the beginning of the project, we could have saved time spent re-organizing and cleaning up coding in several of the modules. This should be a common practice on future developments. Moreover, the computational group should meet to review the best practices document and to improve it. No code should be passed to the computational group if it has not been verified as meeting all the criteria stated in the Programming Practices guideline.

2. We recommend continuing the use of FORTRAN for writing the kernel modules. Otherwise, the complexity of integrating the frameworks will be increased, especially if the selected programming language is not compatible with the ones used in the frameworks. We should keep modules simple and uniform.

3. The developers of modules should provide test cases, inputs and expected outputs, and a README file describing how to call and run the module. Having this information upfront will help us with trouble shooting problems in the future.

4. When we talk about 'clear requirement specifications' we need to ensure that the requirement specifications are clear enough to keep the project advancing forward. However, time expended on the specifications document should not be so excessive that the project fails to meet deadlines.

5. We need to focus on smaller development cycles, where inputs and outputs are progressively presented and tested.

6. We should be mindful of the needs and skills of our final users, in order to provide them with an assessment tool suitable for their work environment and their computer environment.

7. Increasingly complex problems confronted by the xLPR project will require more computational power. We need to think about improving I/O times while focusing on our end user computer environment in mind. For example, our end-user community is expected to be predominately based on Windows personal computers, not high performance computers or UNIX environments.

8. All members of the Computational team, PIB group, and the xLPR consortium should be encouraged to use the two frameworks and provide feedback to the developers of the frameworks and also developers of the modules.

9. Several options for CM software are available to document and track changes of the CM items. We recommend exploring some of the options to take advantage of automating and reducing the paper work and time needed to complete this important aspect of software development.

10. The xLPR implementations need to take advantage of web-based capabilities and new efforts need to be focused on creating a web application version of these implementations.

10. REFERENCES

1. Marjorie EricksonKirk and Mark T. EricksonKirk, *Proposed Pilot Study to Assess the Process Developed to Provide a Technical Basis for Revision of the LBB Methodology, Scope of Work: Industry/NRC Pilot Study to Assess LBB Methodology*, January 14, 2009.

2. Title 10 Code of Federal Regulations (10CFR50), Appendix A, GDC-4.

3. P. T. Williams, S. Yin, T. L. Dickson, and B. R. Bass, "Development of a Modular Computational Platform for Passive Pressure-Bearing Components in Nuclear Power Plants – Structural Integrity Assessments Modular – Probabilistic Fracture Mechanics (SIAM-PFM)", ORNL/NRC/LTR-08/197, Oak Ridge National Laboratory, Oak Ridge, TN, October 31, 2008.

4. P.T. Williams, T.L. Dickson and S. Yin, *Fracture Analysis of Vessels – Oak Ridge, FAVOR, v09.1, Computer Code: Theory and Implementation of Algorithms, Methods, and Correlations*, ORNL/TM-2010/05, Oak Ridge National Laboratory, Oak Ridge, TN.

5. T.L. Dickson , P.T. Williams and S. Yin, *Fracture Analysis of Vessels – Oak Ridge, FAVOR, v09.1, Computer Code: User's Guide*, ORNL/TM-2010/04, Oak Ridge National Laboratory, Oak Ridge, TN.

6. D. O. Harris, D. D. Dedhia, and S. C. Lu, *A Probabilistic Fracture Mechanics Code for Piping Reliability Analysis (pcPRAISE code)*, NUREG/CR-5864, Failure Analysis Associates, Inc., Menlo Park, CA, 1992.

7. M. A. Khaleel, F. A. Simonen, H. K. Phan, D. O. Harris, and D. Dedhia, *Fatigue Analysis of Components for 60-Year Plant Life*, NUREG/CR-6674, Pacific Northwest National Laboratory, Richland, WA, 2000.

8. D. Rudland, et al., "Development of Computational Framework and Architecture for Extremely Low Probability of Rupture (xLPR) code," PVP2010-25963, Proceedings of ASME 2010 Pressure Vessel and Piping Conference, July 18-22, 2010, Bellevue, WA.

9. D. Rudland, "Load Module v1.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Alpha Model Dev> Modules>Load_v1.0

10. D. Harris, et al., "Crack Initiation v2.0" https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules> Crack_Initv2.0/.

11. D. Rudland, "kSurf v1.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Alpha Model Dev> Modules> kSurf_v1.0.

12. Anderson, T.L., Thornwald, G., Revelle, D.A., and Lanaud, C., "Stress Intensity Solutions for Surface Cracks and Buried Cracks in Cylinders, Spheres, and Flat Plates," Structural Reliability Technology final report to The Materials Property Council, Inc., March 14, 2000.

13. D. Rudland, "kTWC-circ v1.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Alpha Model Dev> Modules> kTWC_v1.0

14. Anderson, T.L., "Stress Intensity and Crack Opening Area Solutions for Through-wall Cracks in Cylinders, and Spheres," *Structural Reliability Technology final report to The Materials Property Council, Inc*., January 29, 2003.

15. D. Dedhia, Grower v2.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules>Grower_v2.0.

16. D. Dedhia, et al., "Coalesce v2.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules> Coalesce_v2.0

17. 2007 ASME Boiler and Pressure Vessel Code.

18. R. Olson, SC Fail v2.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules>SC_Fail_v2.0.

19. R. Olson, "TWC_Fail v2.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules>TWC_Fail_v2.0.

20. R. Olson, "COD_v2.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules> COD_v2.0.

21. D. Paul, et al., "SQUIRT v1.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules>SQUIRT_v1.0.

22. Henry, Robert, E. "The Two-Phase Critical Discharge of Initially Saturated or Subcooled Liquid" *Nucl. Sci. and Eng*. 41, 336-342 (1970).

23. Henry, Robert E., Fauske, Hans K., Comas, Stuart T. "Two-Phase Critical Flow at Low Qualities Part I: Experimental" *Nucl. Sci. and Eng*. 41, 79-91 (1970).

24. Henry, Robert E., Fauske, Hans K., Comas, Stuart T. "Two-Phase Critical Flow at Low Qualities Part II: Analysis" *Nucl. Sci. and Eng*. 41, 92-98 (1970).

25. Paul, D. D., Ahmad, J., Scott, P. M., et al. "Evaluation and Refinement of Leak-Rate Estimation Models" NUREG/CR-5128. (1994).

26. D. Harris, et al., "Inspection and Repair v2.0 Conceptual Description", https://websps1.battelle.org/nrcnureg/home/xLPR_CM > Beta Model Dev> Modules>ISI_v2.0.

27. D. Rudland, "Program Plan for alpha xLPR Framework Development," Draft, US Nuclear Regulatory Commission, Rockville, MD, October 13, 2009.

28. B. O'Sullivan, *Mercurial: The Definitive Guide*, O'Reilly, June 2009.

29. Trac- Integrated SCM & Project Management; http://trac.edgewall.org/.

30. P. D. Mattie, *xLPR Pilot Study Model Problem Statements*, Sandia National Laboratories, Albuquerque, NM, November 11, 2010.

Appendix A

SIAM-xLPR FRAMEWORK DESIGN

Appendix A. SIAM-xLPR FRAMEWORK DESIGN

As shown in Fig. 36, SIAM-xLPR's logical architecture is based on an object-oriented layered design. The core (Fig. 37) and primary container classes (Fig. 38) are located primarily in the Model Layer. The View Layer contains the graphical user interface, and the actual execution of the probabilistic analysis is under the control of classes defined in the Controller Layer (Fig. 39). Basic services, utilities, and tools are provided in the Service and Utils layers.

Python was chosen as the object-oriented language to develop the SIAM-xLPR framework. Python is an open-source, general-purpose, object-oriented, scripting language that has found extensive application worldwide. Its cross-platform capabilities allow its use on such diverse operating systems as Microsoft Windows, Macintosh OS-X, Unix, and Linux. The following categories represent some of Python's more common applications and capabilities that were considered in its selection for developing the framework:

- **Component Integration** – Python's component integration capabilities allows it to be used as a flexible *glue language* for scripting the behavior of other systems and components. For SIAM code development, it is particularly important that the overall framework program be able to link to and communicate with modules written in a wide variety of programming languages such as Fortran and C.

- **Scientific Computing Libraries Availability –** SIAM-xLPR imports two scientific libraries, NumPy [A1] and SciPy [A2]. NumPy and SciPy are open-source libraries of scientific tools for use with applications written in Python. The NumPy and SciPy libraries provide modules for statistics, optimization, numerical integration, linear algebra, Fourier transforms, signal processing, image processing, ODE solvers, special functions, and tools for integrating C/C++ and Fortran code.

- **Python - Fortran Integration –** The utility *f2py* is a tool that provides an easy connection between Python and Fortran languages. *f2py* creates extension modules from (handwritten or *f2py*-generated) signature files or directly from Fortran sources. The generated extension modules facilitate calling Fortran 77/90/95, Fortran 90/95 modules, and C functions from Python, accessing Fortran 77 COMMON blocks and Fortran 90/95 module data (including allocable arrays) from Python, calling Python functions from Fortran or C (call-backs), automatically handling the difference in the data storage order of multi-dimensional Fortran and *Numerical Python* (NumPy) (i.e. C) arrays. In addition, *f2py* can build the generated extension modules for shared libraries with only one command. *f2py* uses the *NumPy distutils* module from *NumPy* that supports a number of major Fortran compilers. *f2py* generated extension modules depend on *NumPy* which provides a fast multi-dimensional array language facility for Python.

- **User Interface Development** – Python GUI programs can be implemented to be portable and run unchanged on MS Windows, X Windows (on Unix and Linux), and Macintosh computers. The open-source GUI toolkit Qt developed by Nokia Qt Software (Oslo, Norway) with PyQt is utilized for the SIAM-PFM's GUI development. PyQt is a set of Python bindings for Nokia's Qt application framework and runs on all platforms supported by Qt, including Windows, MacOS/X and Linux. There are two sets of bindings: PyQt v4 supports Qt v4; and the older PyQt v3 supports Qt v3 and earlier. The bindings are implemented as a set of Python modules and contain over 300 classes and over 6,000 functions and methods. Like Qt, PyQt v4, is available on all platforms under a variety of licenses including the GNU GPL (v2 and v3) used by SIAM-xLPR.

- **Lines of code:** The Python coverage option in eclipse/PyDev integrated development environment (IDE) was used to calculate the number of lines of Python code for the SIAM-xLPR framework: 28,620. The number of lines of Fortran code are 9,090 (1872 in timeloop and the rest in the modules). These metrics of course don't include comments or blank lines but the actual number of executable statements. They also don't include the number of lines of code in the numerical and PyQt libraries that we are importing into the SIAM-xLPR framework.

Source code file name	**comment lines**	**lines of code**
Load Module		
load_v1.1.f90	298	229
TimeLoop Modules		
Coalescense_v2.2.f90	466	404
COD_v2.1.f90	434	334
crack_init_v2.1.f90	305	444
grower_v2.1.f90	202	132
ISI_v2.1.f90	171	62
kSurf_v1.1.f90	270	410
kTWC_v1.1.f90	347	297
SCFail_v2.1.f90	644	335
SQUIRT_v1.1.f90	988	2705
TimeLoop_v2.1.f90	1036	1872
TWCFail_v2.1.f90	802	828
Total Lines	5665	7823
Post-Processor Modules		
driver_transformers_expectation.f90	5	18
expectation_v1.1.f90	296	544
transformers_v1.1.f90	313	476
Total Lines	614	1038
Grand Total	6577	9090

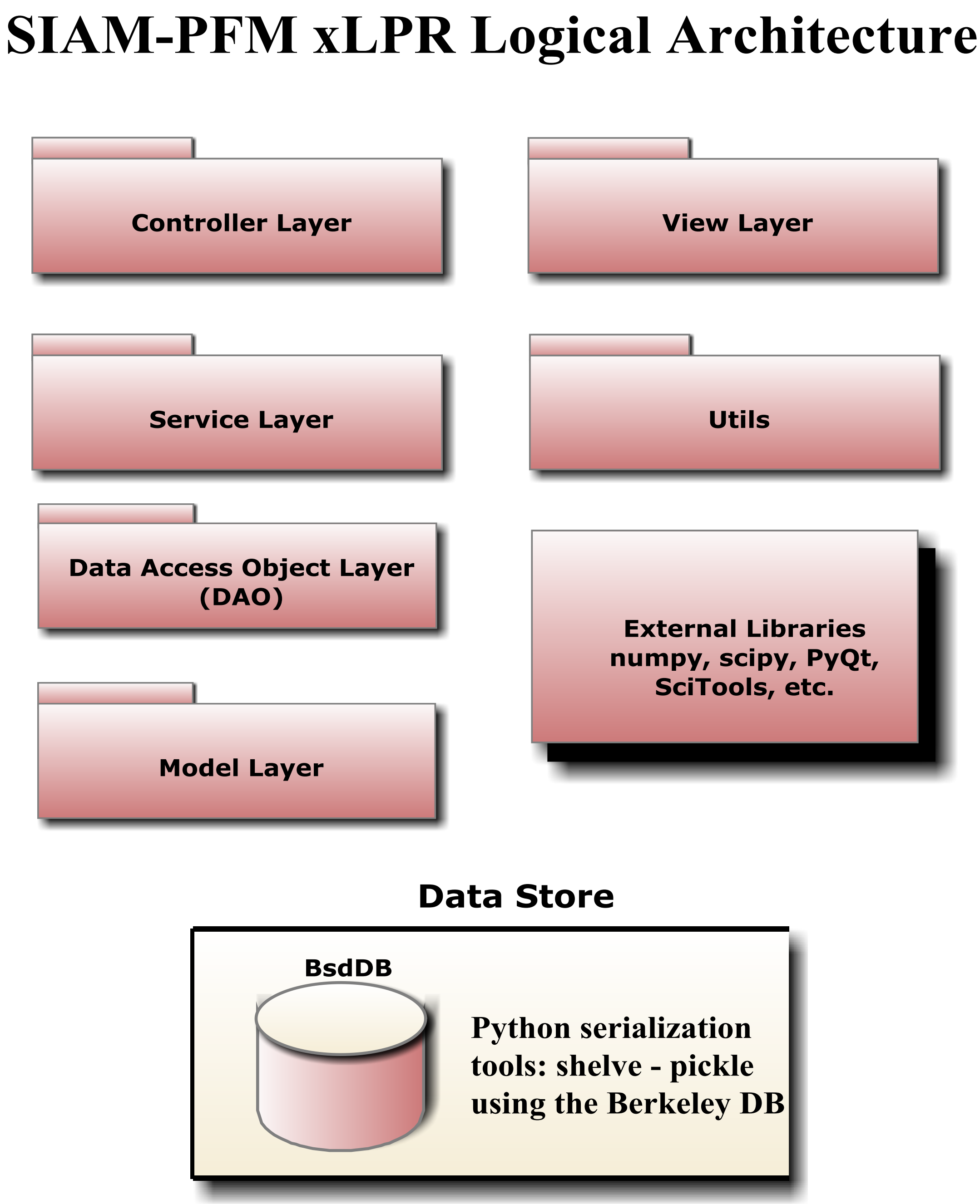

Fig. 36. SIAM-xLPR uses a layered object-oriented architecture.

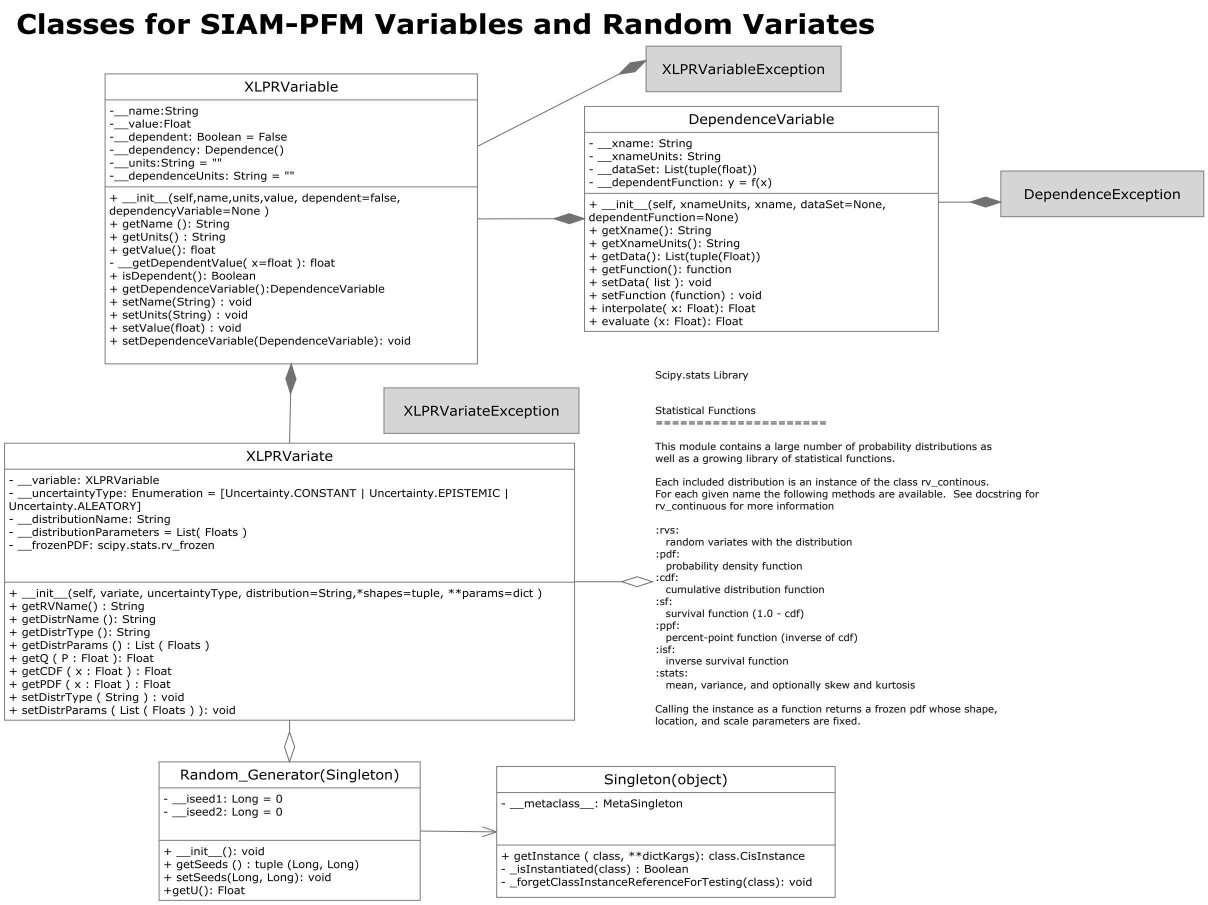

Fig. 37. The SIAM-xLPR core classes provide the basic attributes of "units-awareness" and uncertainty type specification with links to the open-source SciPy statistical library.

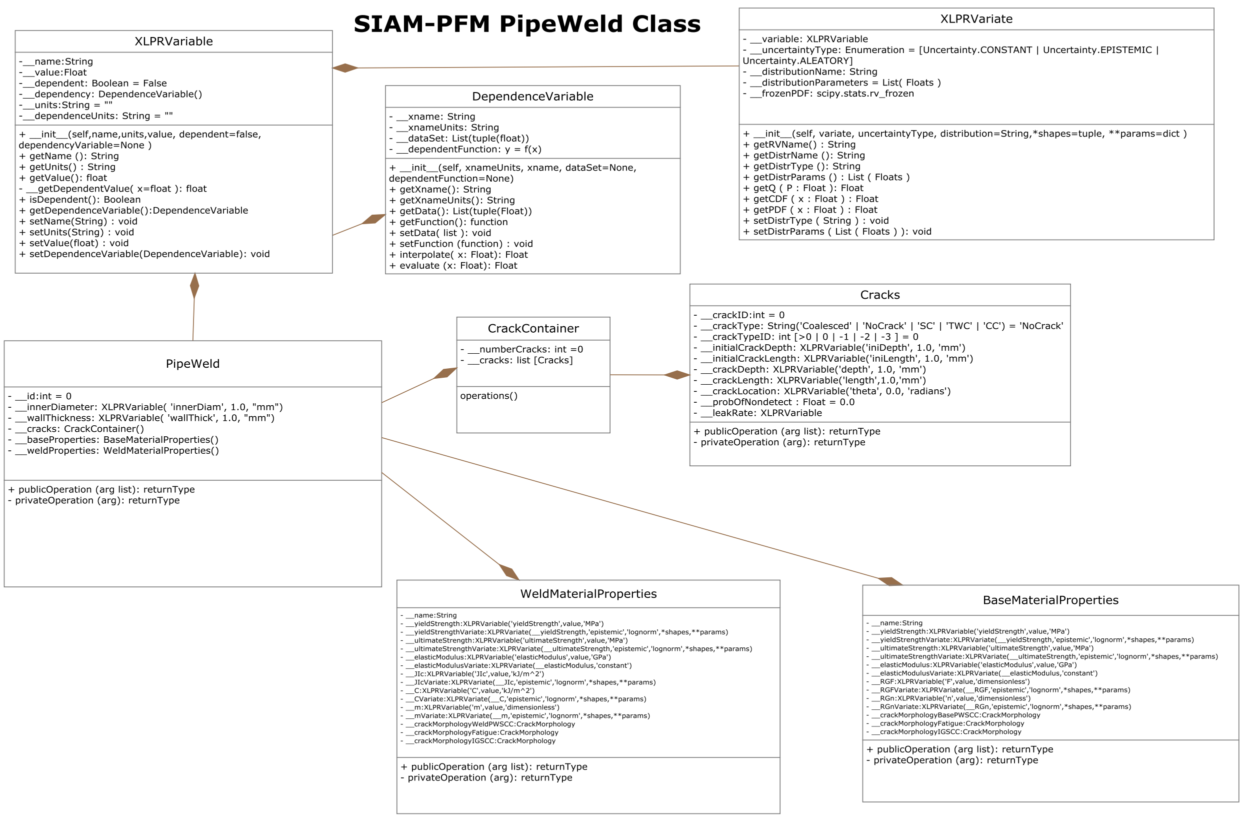

Fig. 38. The SIAM-xLPR container classes use a composition of core and other container classes to create the required objects.

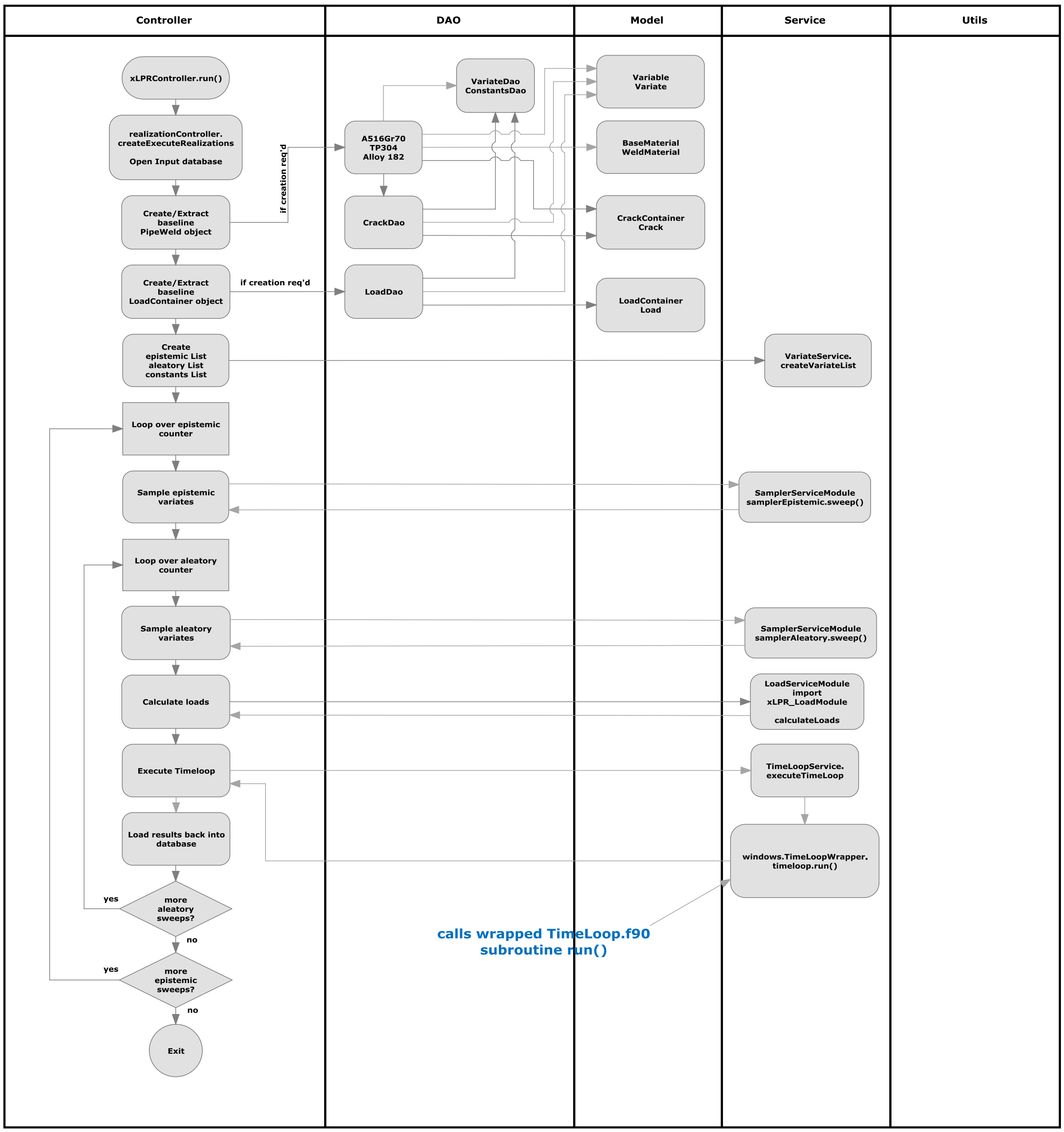

Fig. 39. The SIAM-xLPR layered architecture provides linkages between the framework written in Python to modules written in Fortran.

REFERENCES

A1. Scientific Tools for Python, NumPy - http://NumPy.scipy.org/

A2. Scientific Tools for Python, SciPy - http://docs.scipy.org/doc/

Appendix B

CASE MATRIX

Appendix B. CASE MATRIX

Table 1. Case Matrix

Case Name	Description	Comments	Notes
2.1 Deterministic Runs			
SIAM_v1.01_S2.1_00001_001	Deterministic Problem #1		
SIAM_v1.01_S2.1_00001_002	Deterministic Problem #2		
2.2 Stability Testing			
SIAM_v1.01_S2.2a_05000_001	Stability Testing Sect. 2.2(a)	different seeds 250(e) 20(a)	use baseline seeds
SIAM_v1.01_S2.2a_05000_002	Stability Testing Sect. 2.2(a)	different seeds 250(e) 20(a)	use seeds from end of previous case as initial seeds
SIAM_v1.01_S2.2a_05000_003	Stability Testing Sect. 2.2(a)	different seeds 250(e) 20(a)	use seeds from end of previous case as initial seeds
SIAM_v1.01_S2.2a_10000_004	Stability Testing Sect. 2.2(a)	increased trials 500(e) 20(a)	same seeds as SIAM_v1.01_S2.2a_05000_001
SIAM_v1.01_S2.2a_50000_005	Stability Testing Sect. 2.2(a)	increased trials 1000(e) 50(a)	same seeds as SIAM_v1.01_S2.2a_05000_001
SIAM_v1.01_S2.2a_100000_006	Stability Testing Sect. 2.2(a)	increased trials 1000(e) 100(a)	same seeds as SIAM_v1.01_S2.2a_05000_001
SIAM_v1.01_S2.2b_10000_001	Stability Testing Sect. 2.2(b)	time step = 1 month	repeat of SIAM_v1.01_S2.2a_10000_003
SIAM_v1.01_S2.2b_10000_002	Stability Testing Sect. 2.2(b)	time step = 2 months	
SIAM_v1.01_S2.2b_10000_003	Stability Testing Sect. 2.2(b)	time step = 6 months	
SIAM_v1.01_S2.2b_10000_004	Stability Testing Sect. 2.2(b)	time step = 12 months	
3.1 Probabilistic Base Case			
SIAM_v1.01_S3.1_10000_001	Base case: number of trials 10,000 at 500(e) 20(a)		
SIAM_v1.01_S3.1_50000_002	Base case: number of trials 50,000 at 1000(e) 50(a)		
4. Sensitivity Analysis			
4.1 Effect of Safe End Length			
SIAM_v1.01_S4.1_10000_001	safe end conditions use 10000(e) 1(a)	so_wrs= -16.2 MPA mean and 117 Mpa Stdev Max = 300 Mpa, Min=-300 MPA	
SIAM_v1.01_S4.1_50000_002	safe end conditions use 50000(e) 1(a)	so_wrs= -16.2 MPA mean and 117 Mpa Stdev Max = 300 Mpa, Min=-300 MPA	
		Xc=0.18 mean and 0.036 stdev max=0.5, min=0.1	
4.2 Effect of Stress Mitigation			
SIAM_v1.01_S4.2_10000_001	mitigation at 10 yrs	sig0_wrs_mitigated: N(-344.75,34) -447 < sig0_wrs_mitigated < -242	
SIAM_v1.01_S4.2_10000_002	mitigation at 20 yrs	Xc_mitigated: N(0.38,0.038) 0.26 < Xc_mitigated < 0.5	
SIAM_v1.01_S4.2_10000_003	mitigation at 40 yrs		
SIAM_v1.01_S4.2_10000_004	mitigation at 30 yrs		
SIAM_v1.01_S4.2_50000_005	mitigation at 20 yrs		
SIAM_v1.01_S4.2_50000_006	mitigation at 30 yrs		
SIAM_v1.01_S4.2_50000_007	mitigation at 10 yrs		
4.3 Crack Initiation Model			
SIAM_v1.01_S4.3_10000_001	run with Crack Initiation Method 1	10,000 500(e) 20(a)	
SIAM_v1.01_S4.3_50000_002	run with Crack Initiation Method 1	50,000 1000(e) 50(a)	
4.4 Chemical Mitigation	**Base Case H2 concentration = 25 cc/kg-STP**		
SIAM_v1.01_S4.4_10000_001	H2 concentration = 50 cc/kg-STP		
SIAM_v1.01_S4.4_10000_002	H2 concentration = 80 cc/kg-STP		
5.0 Post-Processing Analysis	**Does not involve re-running SIAM**		
5.1 Leak Detection Capability			
SIAM_v1.01_S5.1_10000_001	0.1 gpm leak detection base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.1_10000_002	1.0 gpm leak detection base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.1_10000_003	10 gpm leak detection base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.1_10000_004	50 gpm leak detection base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.1_10000_005	0.1 gpm leak detection safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
SIAM_v1.01_S5.1_10000_006	1.0 gpm leak detection safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
SIAM_v1.01_S5.1_10000_007	10 gpm leak detection safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
SIAM_v1.01_S5.1_10000_008	50 gpm leak detection safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
5.2 Inspection Schedule	**Inspection Intervals**		
SIAM_v1.01_S5.2_10000_001	30 yrs (years 30) base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.2_10000_002	20 yrs (years 20, 40) base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.2_10000_003	10 yrs (years 10, 20, 30, 40, 50) base case	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.2_10000_004	5 yrs (years 5, 10, 15, 20, 25, 30, 35, 40, 35, 50, 55) b	post-process with SIAM_v1.01_S3.1_10000_001	
SIAM_v1.01_S5.2_10000_005	30 yrs (years 30) safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
SIAM_v1.01_S5.2_10000_006	20 yrs (years 20, 40) safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
SIAM_v1.01_S5.2_10000_007	10 yrs (years 10, 20, 30, 40, 50) safe end case	post-process with SIAM_v1.01_S4.1_10000_001	
SIAM_v1.01_S5.2_10000_008	5 yrs (years 5, 10, 15, 20, 25, 30, 35, 40, 35, 50, 55) s	post-process with SIAM_v1.01_S4.1_10000_001	

NRC FORM 335 (2-89) NRCM 1102, 3201,3202 U.S. NUCLEAR REGULATORY COMMISSION **BIBLIOGRAPHIC DATA SHEET** *(See instructions on the reverse)*	1. REPORT NUMBER (Assigned by NRC, Add Vol., Supp., Rev., and Addendum Numbers, if anv.1
2. TITLE AND SUBTITLE SIAM-xLPR Version 1.0 Framework Report	3. DATE REPORT MONTH | YEAR September 2010 4. FIN OR GRANT NUMBER
5. AUTHOR(S) H. B. Klasky, P. T. Williams, S. Yin, and B. R. Bass	6. TYPE OF REPORT Final, technical 7. PERIOD COVERED *(Inclusive Dates)*

8. PERFORMING ORGANIZATION - NAME AND ADDRESS *(If NRC, provide Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address; if contractor, provide name and mailing address.)*

Probabilistic Pressure Boundary Integrity Safety Assessment (PISA) Program
Oak Ridge National Laboratory
P. O. Box 2008, Mail Stop 6085
Oak Ridge, TN 37831-6085

9. SPONSORING ORGANIZATION - NAME AND ADDRESS *(If NRC, type "Same as above"; if contractor, provide NRC Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address.)*

Division of Engineering Technology
Office of Nuclear Regulatory Research
U. S. Nuclear Regulatory Commission
Washington, DC 2055-0001

10. SUPPLEMENTARY NOTES

1 1. ABSTRACT *(200 words or less)*

This document describes the probabilistic framework developed for the Structural Integrity Assessments Modular – Extremely Low Probability of Rupture (SIAM-xLPR) software tool under development at the Oak Ridge National Laboratory (ORNL), which supports the Nuclear Regulatory Commission's initiative for a new piping system assessment methodology. The new methodology will provide a tool for demonstrating compliance with the regulatory requirement that primary power plants water piping systems exhibit an extremely low probability of rupture (xLPR). SIAM-xLPR is a modular-based assessment tool that incorporates a prototype xLPR model assembled from new and existing fracture mechanics models and software. xLPR represents one of the four subsystems currently installed in the ORNL-developed SIAM Problem Solving Environment. The prototype SIAM-xLPR models and software modules are linked within a probabilistic framework fully developed using open-source languages and software libraries.

12. KEY WORDS/DESCRIPTORS *(List words or phrases that will assist researchers in locating the report.)*	13. AVAILABILITY STATEMENT **unlimited**
probabilistic fracture mechanics, piping rupture, primary water stress corrosion cracking	14. SECURITY CLASSIFICATION *(This Page)* **unclassified** *(This Report)* **unclassified**
	15. NUMBER OF PAGES
	16. PRICE

NRC FORM 335 (2-89)