*Transactions*, **SMiRT-22**
**San Francisco, California, USA - August 18-23, 2013**
**Division II**

# PREDICTION OF THE LEAKAGE RATE FOR CRACKED PIPES IN NUCLEAR POWER PLANTS

**Paul T. Williams[1] and Shengjun Yin[2]**

[1] Senior Research Staff, Oak Ridge National Laboratory, Oak Ridge, TN, USA (williamspt@ornl.gov)
[2] Senior Research Staff, Oak Ridge National Laboratory, Oak Ridge, TN, USA (yins@ornl.gov)

## ABSTRACT

Within the Extremely Low Probability of Rupture (xLPR) project, the U.S. Nuclear Regulatory Commission's (USNRC) Office of Nuclear Regulatory Research is working cooperatively with the nuclear industry to develop a new probabilistic, modular-based tool that incorporates a comprehensive assessment methodology for demonstrating compliance with regulatory requirements regarding primary system piping. One of the objectives of the xLPR project is to develop a computer module that can determine estimates of leakage rates from postulated cracks in nuclear power plant piping systems.

In support of the xLPR project, Oak Ridge National Laboratory (ORNL) is developing a thermal-hydraulics code to calculate the critical flow rate of a high-temperature, high-pressure fluid that is "flashing" through a crack in a pipe. The new code is called the **Le**ak **A**nalysis of **P**iping – **O**ak **R**idge (LEAPOR) computer program. LEAPOR implements the empirically-adjusted homogeneous equilibrium model (HEM) developed by Henry and Fauske (1971) with extensions by Abdollahian and Chexhal (1983) and Paul et al. (1994).

One of the primary drivers in the application's development is that LEAPOR must comply with all xLPR project Software Quality Assurance (SQA) requirements including evaluations for correctness, consistency, completeness, accuracy, readability, portability, extensibility, and testability, and be prepared to successfully meet the criteria of all SQA audits. This paper discusses how meeting these requirements informs software architectural design decisions, including the choice of a modular, layered, logical architecture, strict conformance to a selected programming standard, and the implementation of a consistent set of recognized design patterns and good programming practices.

## INTRODUCTION

Within the Extremely Low Probability of Rupture (xLPR) project [Rudland and Harrington (2012)], the U.S. Nuclear Regulatory Commission's (USNRC) Office of Nuclear Regulatory Research is working cooperatively with the nuclear industry to develop a new probabilistic, modular-based computational tool that incorporates a comprehensive assessment methodology for demonstrating compliance with regulatory requirements regarding primary system piping. One of the objectives of the xLPR project is to develop a computer module that can determine estimates of leakage rates from postulated cracks in nuclear power plant piping systems. This leak rate module will be one component working with other modules under the control of a larger computational framework.

In support of the xLPR project, Oak Ridge National Laboratory (ORNL) is developing a thermal-hydraulics code to calculate the critical flow rate of a high-temperature, high-pressure fluid (water) that is "flashing" (adiabatic flow boiling) through a crack in a pipe. Following a naming convention established at ORNL for USNRC-funded computer programs [e.g., the **F**racture **A**nalysis **o**f **V**essels – **O**ak **R**idge (FAVOR) code, Williams, Dickson, and Yin (2007)], the new code is called the **Le**ak **A**nalysis of **P**iping – **O**ak **R**idge (LEAPOR) computer program.

A key driver in the new application development is that LEAPOR must comply with all of the xLPR Project Software Quality Assurance (SQA) requirements, including evaluations for correctness, consistency, completeness, accuracy, readability, portability, maintainability, extensibility, and testability, and be prepared to successfully meet the criteria of all formal SQA audits. As a result of these SQA requirements, careful attention has been paid in the application’s development phase to apply sound architectural design principles and consistent programming practices. Experience has demonstrated that strict conformance to a selected programming language standard plays an important role in meeting the requirements of portability, maintainability, and extensibility.

## THERMAL-HYDRAULICS MODEL

LEAPOR implements in code the empirically-adjusted homogeneous equilibrium model (HEM) developed by Henry and Fauske (1970-1971) with extensions by Abdollahian and Chexhal (1983, 1984) and Paul et al. (1994). The models and assumptions applied in the development of the **S**eepage **Q**uanti-fication of **U**psets **I**n **R**eactor **T**ubes (SQUIRT) code (see Paul et al., 1994) have been followed in this initial version of LEAPOR. The HEM approach considers the two-phase mixture to be a pseudo-fluid which can be described by the same conservation law principles that are valid for a single-phase fluid. In the HEM model, the two phases are assumed to be everywhere in a state of both thermal and mechanical equilibrium. Figure 1 (adapted from Chexhal, et al., (1984)) shows the two-phase flow regimes and the simplified geometry assumed for the model. The LEAPOR code provides the option to model the shape of the cross-sectional flow area by either a rectangle, a diamond, or an ellipse.

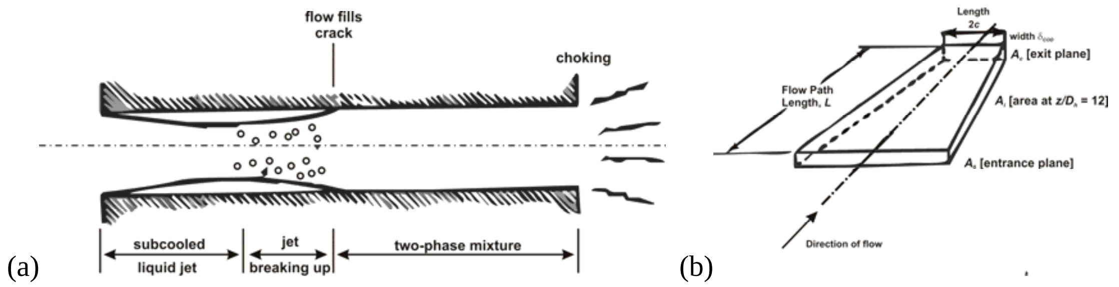

**Figure 1. Two-phase flow through a long narrow crack channel: (a) flow regimes and (b) modeled geometry of the flow path.**

The Henry-Fauske (1970-1971) thermal-hydraulics model consists of a coupled set of nonlinear equations derived from the principle of conservation of momentum, constrained by the conservation of mass, and a force balance based on accounting for mechanisms that contribute to the total pressure drop through the crack flow channel. The constrained conservation of momentum is expressed by the mass flux equation as formulated by Eq. (1),

$$\psi\left(G_c^2, p_c\right) = G_c^2 - \frac{1}{\left[\left(\dfrac{x_c v_{g(c)}}{\gamma_0 p_c}\right) - \left(v_{fg(c)} N \dfrac{dx_E}{dp_c}\right)\right]} = 0 \tag{1}$$

where $G_c$ is the mass flux at the crack exit plane, $x_c$ is the non-equilibrium vapor quality, $v_{g(c)}$ is the specific volume of saturated vapor at the crack exit, $\gamma_0$ is the isentropic expansion coefficient for wet or superheated vapor, $p_c$ is the critical (choking) pressure at the crack exit, $v_{fg(c)}$ is the specific volume change due to vaporization, $x_E$ is the equilibrium vapor quality at the crack exit, and $N$ is an empirical parameter. The non-equilibrium vapor quality at the throat, $x_c$, is modeled as an exponential relaxation process by Eq. (2),

$$x_c = N\, x_E \left\{1 - \exp\left[-B\left(\frac{L}{D_h} - 12\right)\right]\right\} \tag{2}$$

where $B = 0.0523$ is an empirically-derived constant, $L$ is the length of the flow path through the pipe wall, and $D_h$ is the hydraulic diameter of the cross-sectional flow area. The empirical parameter, $N$, is assumed to be linearly dependent on $x_E$ by as described by Eq. (3):

$$N = \begin{cases} 20x_E & \text{for } x_E < 0.05 \\ 1 & \text{for } x_E \geq 0.05 \end{cases} \tag{3}$$

The solution to Eq. (1) is constrained by a force balance which separates the total pressure drop along the flow path into five contributing mechanisms

$$\Omega\left(G_c^2, p_c\right) = \left(p_0 - p_c\right) - \Delta p_e - \Delta p_a - \Delta p_f - \Delta p_k - \Delta p_{aa} = 0 \tag{4}$$

where

| | |
|---|---|
| $p_0 =$ | absolute stagnation pressure of the subcooled liquid water at the crack entrance |
| $p_c =$ | absolute pressure at the crack exit |
| $\Delta p_e =$ | pressure loss due to entrance effects |
| $\Delta p_a =$ | pressure loss due to acceleration by vaporization |
| $\Delta p_f =$ | pressure loss due to frictional effects |
| $\Delta p_k =$ | pressure loss due to tortuosity of flow path (turns and protrusions) |
| $\Delta p_{aa} =$ | pressure loss due to acceleration by changing flow area |

Equations (1) and (4), supported by Eqs. (2)-(3), represent a closed, coupled set of two nonlinear equations with two unknowns, $G_c$ and $p_c$. The solution strategy adopted in LEAPOR is to use the *trust region method* (also known as the *restricted step method*), as implemented in the open-source Nonmonotonic Nonlinear Equation Solver (NNES) library developed by Bain (1993) to iteratively obtain solutions to this nonlinear equation set.

## LOGICAL ARCHITECTURAL DESIGN – DECOMPOSITION INTO LAYERS

As one of a number of modules in the xLPR project, the new LEAPOR code is required to be an SQA auditable application, fulfilling the software work practice requirements of ASME NQA-1-2008 (including Addenda 2009) Quality Assurance Requirements for Nuclear Facility Applications. The code structure must be designed to be *maintainable* for current and future needs. Code maintainability refers to how easily a software system can be changed to add new features, modify existing features, find and fix bugs, and/or improve performance. Additionally, the xLPR Project SQA plan states that "source code components (source code and documentation) must be evaluated for correctness, consistency, completeness, accuracy, readability, and testability." These requirements are driven by the need to provide proven mechanisms for managing increasing and evolving complexity throughout the lifecycle of the application. We assert that by applying the principle of *separation of concerns* through a modular, layered architectural software design the above objectives can be best met.

The *logical architecture* [Larman (2005)] of a computer code describes the large-scale organization of an application into modules, packages, subsystems, software libraries, and layers. It is called a logical architecture because there is no decision required at this stage of development concerning how these elements are connected or deployed. A *layer* is a very coarse-grained grouping of subsystems or modules that has a cohesive responsibility for a major aspect of the software system. In a strictly *layered architecture*, a given layer only calls upon the services of the layer directly below it; however, for the development of LEAPOR, it is more appropriate to adopt the *relaxed layered architecture* [Larman (2005)] approach in which a higher layer can call upon the services of several lower layers. At the logical architecture's high level of abstraction, the layered design structure can be conveniently expressed through the Unified Modeling Language (UML OMG 2007) notation of a package diagram as shown in Figure 2.

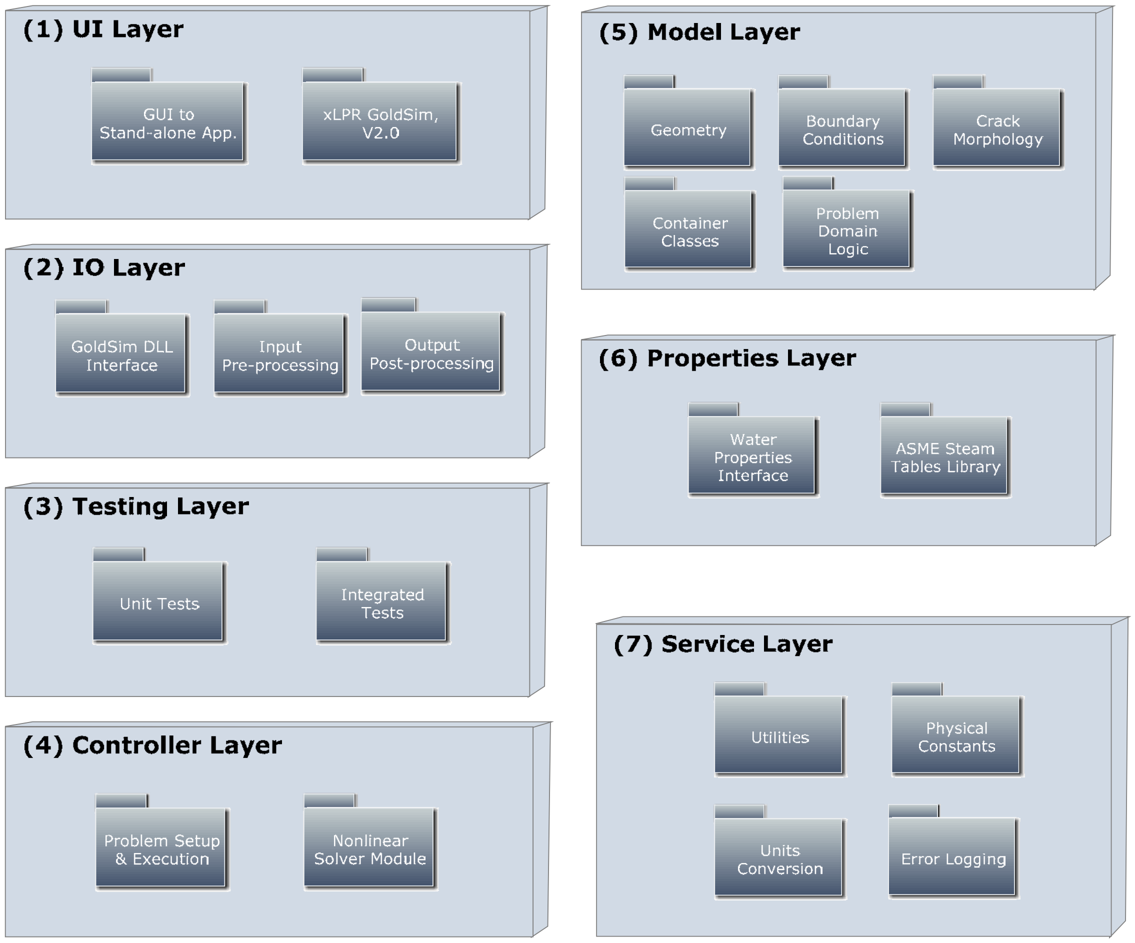

**Figure 2. UML package diagram showing the proposed layered design for LEAPOR.**

### *Description of the Layers in the LEAPOR Architectural Design*

The logical architectural design for LEAPOR is composed of the following seven layers:

1) UI Layer – This user interface layer anticipates two possible modes of user interaction, as a stand-alone program with a console application as a driver or as an external element which is linked directly to the xLPR framework.
2) IO Layer – The IO Layer contains the single point of entry into LEAPOR and has the responsibility for all input data validity checks and any required post-processing operations.
3) Testing Layer – The Testing Layer has the responsibility for executing all unit and integrated tests when directed by the UI Layer through the IO Layer. The results of these tests can be stored in predetermined output files or reported directly back to the UI Layer (again through the IO Layer) as required.
4) Controller Layer – The Controller Layer carries out the actual calculation of the leak rate by setting up the problem, interacting with the Nonlinear Solver Module, and accessing the services of the three layers directly below it.
5) Model Layer – The Model Layer is responsible for all problem-specific data such as pipe and crack geometry, applied boundary conditions, and surface morphology models. This layer also contains model-specific domain logic that constructs the nonlinear equation set to be solved. Employing the object-oriented constructs available in Fortran 2003, LEAPOR's three base classes

are built using the OBJECT design pattern (Rouson et al., 2011), where the geometry, boundary conditions, and surface morphology classes all derive from the same abstract object class. Aggregation is then used to construct a container class that represents the complete state of the leaking crack.

6) Properties Layer – The Properties Layer is responsible for supplying water property data for subcooled liquid, saturated liquid, wet and saturated vapor, and dry (superheated) vapor phases. The ASME IAPWS-IF97 *Steam Properties for Industrial Use* code library provides an implementation of the *International Association of Properties of Water and Steam (IAPWS)* formulation released in 1997, replacing the 1967 IFC formulation. The ASME software library represents a modern, verified, and well-documented resource for the calculation of water and steam properties. Since the ASME library is written in Fortran 77, the implementation in LEAPOR requires it to be wrapped in a Fortran 2003 module to enforce data hiding. These properties are supplied as a service to the rest of the application through a single point of entry interface, based on the FACADE design pattern (Rouson, et al., 2011).
7) Service Layer – The Service Layer is responsible for low-level operations including communications with the operating system, character-string utilities, high-precision stored physical and named constants, units conversions, error logging, etc.

### *Maintainability, Extensibility, and Portability – The Role of Standards Conformance*

To meet the requirements of the xLPR SQA Plan, the new LEAPOR code must exhibit the software design attributes of *maintainability, extensibility*, and *portability*. These attributes can be attained by conforming to a published standard for the programming language used in the module's development and by following the documented set of recommended programming practices that have been established for the xLPR project. All obsolescent features identified by the standard and carried over from previous standards are not allowed in the new coding. These deprecated features are candidates for removal from the language in future standards and could affect future maintainability of the module. Also, the new code should not use any compiler-specific extensions to the selected programming language. The ISO/IEC 1539-1:2004 standard, informally known as Fortran 2003, is the recommended language standard for developing xLPR modules in Fortran, and its object-oriented extensions to previous standards allow a straightforward implementation of the layered architectural design developed for LEAPOR.

There are other areas where published standards can be usefully employed in code development. One area is in units-conversion calculations. All xLPR modules are required to use a consistent set of SI units internally; however, there are instances where some outputs are required to be in U.S. Customary units. The IEEE/ASTM SI 10™ 2010 (2011) *American National Standard for Metric Practice* has been adopted by the xLPR Project as a consistent and reliable source of units conversion data.

Another area where standardization can play a role is in identifying and adopting accepted procedures for calculating required outputs, where, in the absence of the standard, decisions required to complete the calculation may appear arbitrary. As an example, the LEAPOR code is required to return to the xLPR framework an estimate of the volumetric flow rate of a leaking crack. The volumetric flow rate is not a solution variable of the thermal-hydraulics model, but it can be calculated from the mass flux, $G_c$, (which is a primary solution variable) given a specification for the thermodynamic state of the fluid. The ISA-67.03-1982 *Standard for Light Water Reactor Coolant Pressure Boundary Leak Detection*, developed by the Instrument Society of America and cited by reference in USNRC Regulatory Guide 1.45, provides a clear definition of *leakage rate* as a volumetric flow with a specified thermodynamic state requirement, thus removing any potential arbitrariness in the output value.

## VERIFICATION AND VALIDATION

Upon completion of a baseline version of LEAPOR, the next phase of development proceeds to verification and validation of the application. As defined by Williams and Baker (1996), *verification* tests are intended to determine if the mathematics of the selected models have been implemented correctly in the code. *Validation* addresses the question of the validity of the models themselves with respect to the physics that they are intended to simulate. Benchmarking by comparing solutions between independently-developed codes can be employed as one component in a verification study. Comparisons with experimental data typically play an important role in validation studies. The results of both verification (by benchmarking) and validation (by comparison with experimental data) testing are presented below.

Paul et al. (1994) reports the results of a validation study comparing available experimental data with solutions calculated using the SQUIRT code. Figures 3-6 compare the results of both SQUIRT and LEAPOR to experimental data. Figure 3 demonstrates the level of agreement between the two-phase circular pipe flow data of Sozzi and Sutherland (1975) and predictions by SQUIRT and LEAPOR. LEAPOR shows close agreement with both the data and the SQUIRT predictions.

As discussed in NUREG/CR-5128, Paul et al. (1994) also included in their validation study of SQUIRT experimental data obtained on artificially produced slits with known surface roughness and spacing. Figures 4 and 5 show comparisons of the SQUIRT and LEAPOR thermal-hydraulic models to the slit flow data of Amos and Schrock (1983) and Collier, et al. (1984), respectively. Figure 6 presents comparisons of SQUIRT/LEAPOR predictions with the circular pipe flow data of Sozzi and Sutherland (1975) and the slit flow data of Amos and Schrock (1983), Collier, et al. (1984), and Yano et al. (1987). Figures 4-6 demonstrate that agreement with the slit flow data is not as good as that for circular pipe flows where the largest relative errors can be seen at the lowest flow rates. For approximately two-thirds of the data, both the SQUIRT and LEAPOR codes under-predict the measured leakage flow rates for tight slits, possibly due to over-predicting the vapor generation rates for slits. NUREG/CR-5128 suggests that since the geometry is well-defined, the non-equilibrium vapor generation rate may be different for flow through these slits than for flow through long tubes. If less vapor were to be formed in the slit, then the net flow through the slits would be expected to increase. In LEAPOR and SQUIRT, the non- equilibrium vapor generation model is empirically based on data obtained from long circular pipes.

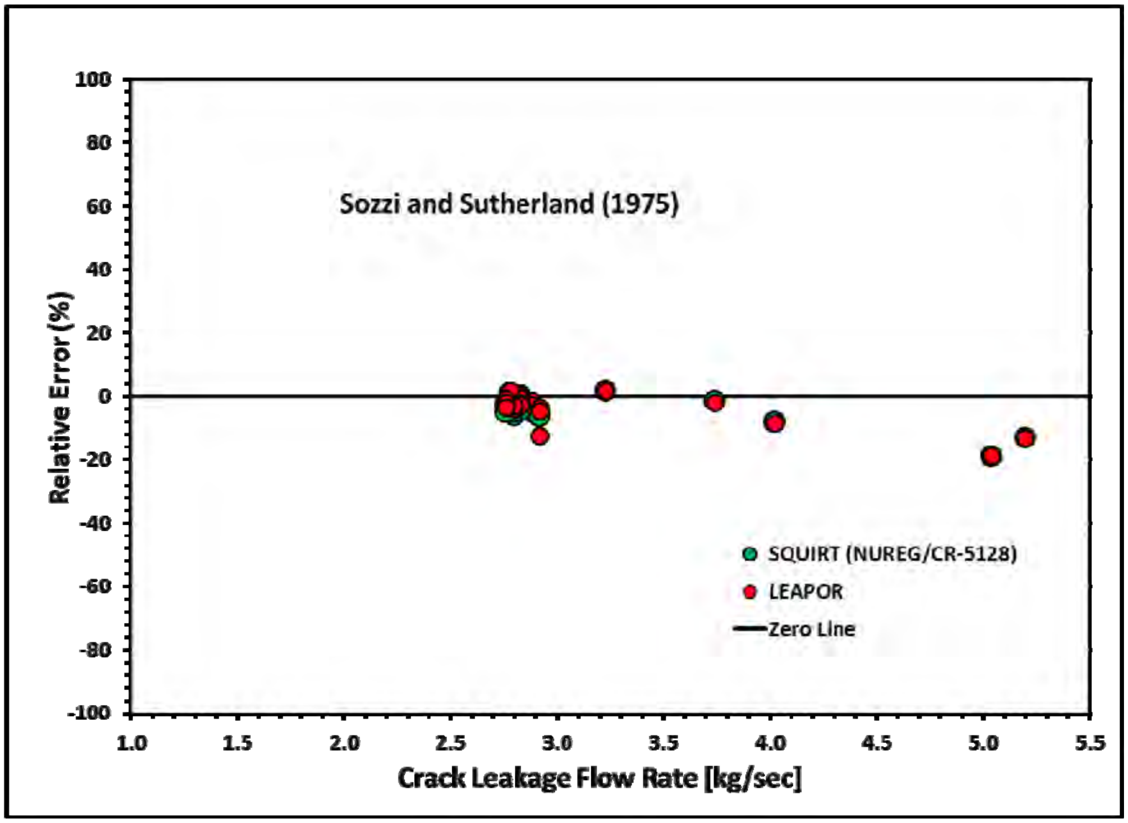

**Figure 3. Comparison of LEAPOR and SQUIRT solutions with the data of Sozzi and Sutherland (1975).**

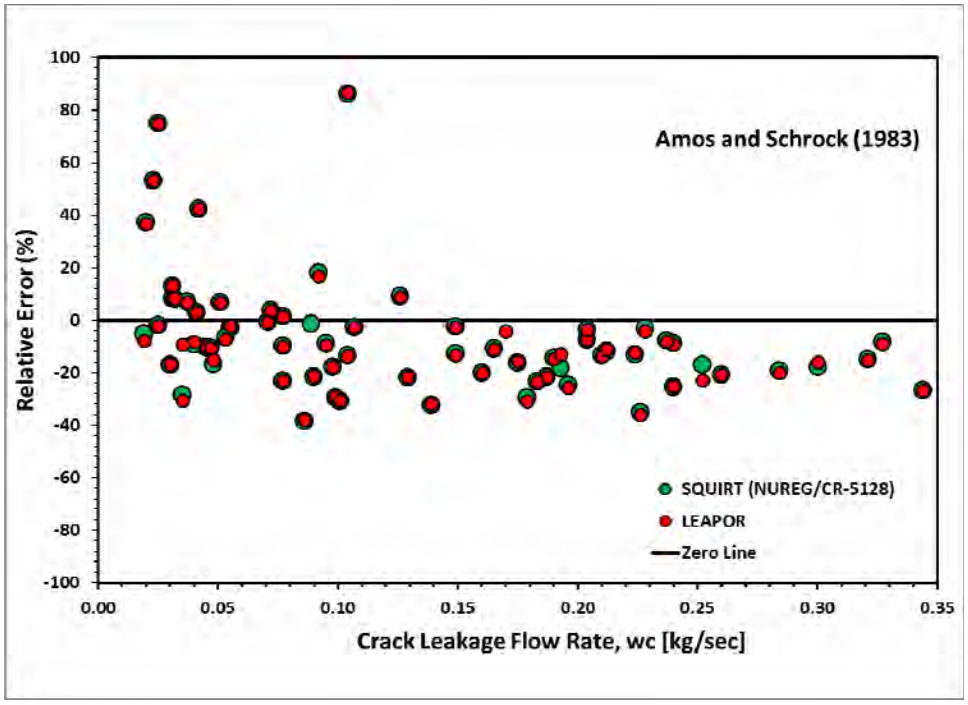

**Figure 4. Comparison of LEAPOR and SQUIRT solutions with the data of Amos and Schrock (1983).**

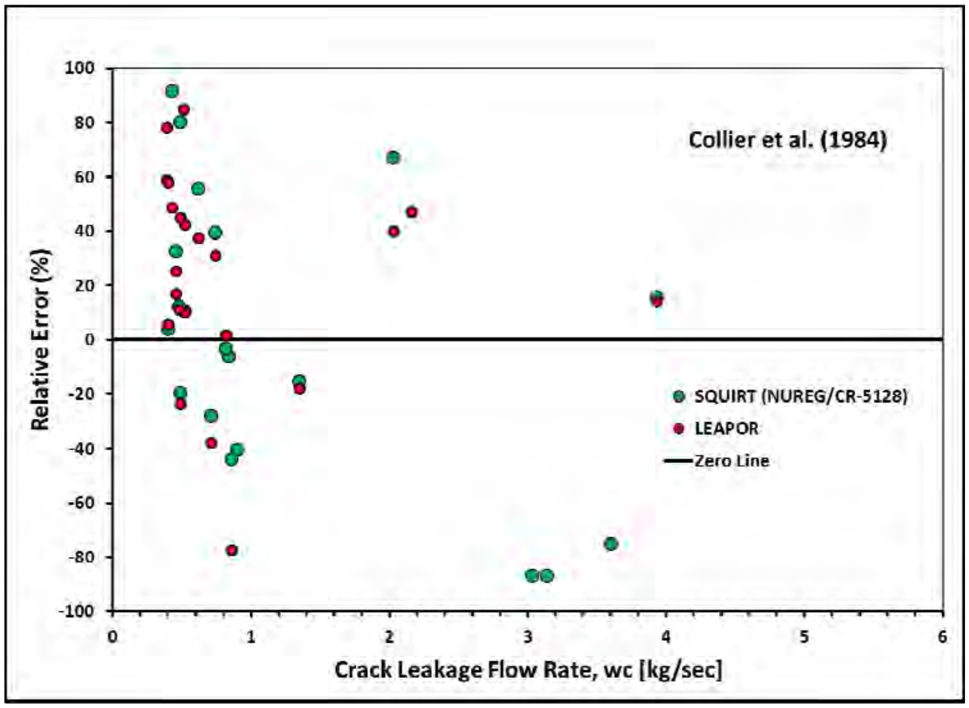

**Figure 5. Comparison of LEAPOR and SQUIRT solutions with the data of Collier et al. (1984).**

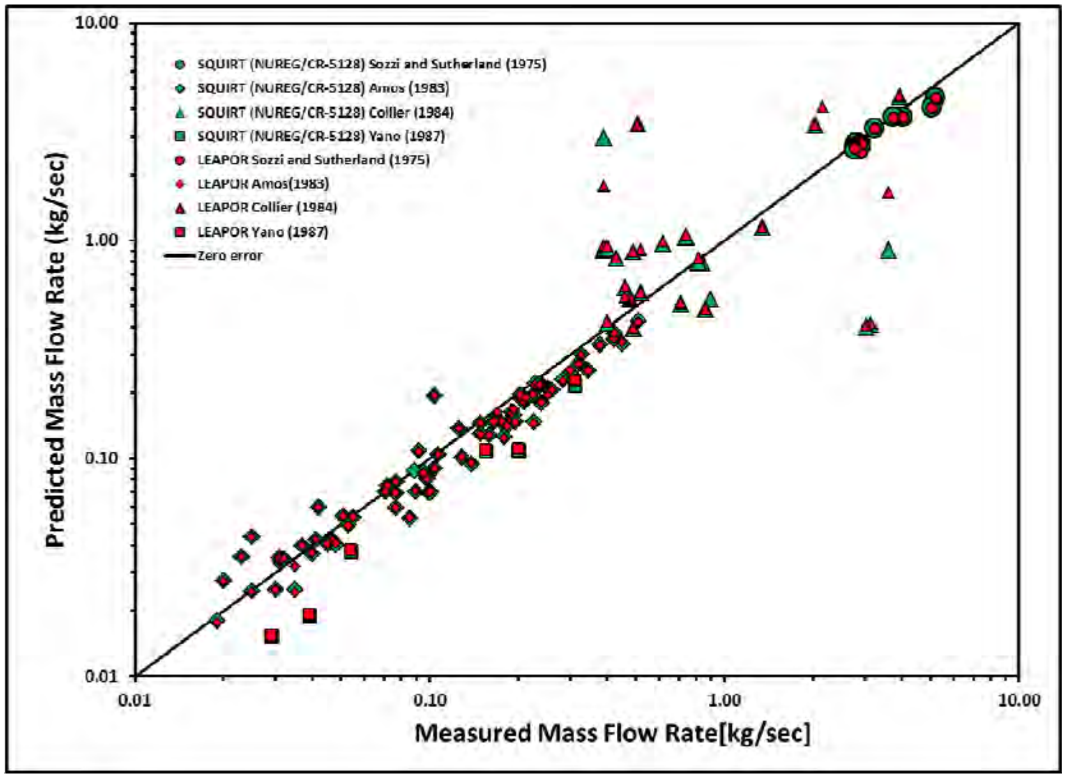

**Figure 6. Comparison of LEAPOR and SQUIRT solutions with the experimental data of Sozzi and Sutherland (1975), Amos and Schrock (1983), Collier et al. (1984), and Yano et al. (1987).**

## SUMMARY AND CONCLUSIONS

In support of the xLPR project, ORNL is developing a thermal-hydraulics code to calculate the critical flow rate of a high-temperature, high-pressure fluid that is "flashing" through a crack in a pipe. The new code is called the **Le**ak **A**nalysis of **P**iping – **O**ak **R**idge (LEAPOR) computer program. LEAPOR implements the same thermal-hydraulics models and assumptions that were employed in the development of the SQUIRT code. The fluid inside the pipe is water in a subcooled state at a high temperature and pressure. LEAPOR utilizes the empirically-adjusted homogeneous equilibrium model (HEM) developed by Henry and Fauske (1971) with extensions by Abdollahian and Chexhal (1983) and Paul et al. (1994).

A key motivation for the development of the new application is that it must comply with all xLPR project SQA requirements including evaluations for correctness, consistency, completeness, accuracy, readability, portability, extensibility, and testability, and be prepared to successfully meet the criteria of all SQA audits. It was determined that the most promising approach to meet these requirements was to apply the basic principles that support a modular software architectural design. The Fortran 2003 standard includes sufficient object-oriented constructs to allow a layered design architecture to be developed and implemented. In addition to LEAPOR's modular design and the implementation of good programming practices, standards conformance is playing an important role in assuring that the requirements of maintainability, portability, and extensibility will be met now and in future versions of the code.

## REFERENCES

Abdollahian, D. and Chexal, B. (1983), "Calculation of Leak Rates Through Cracks in Pipes and Tubes," EPRI Report NP-3395, 1983.

Amos, C., and Schrock, V., (1983) "Critical Discharge of Initially Subcooled Water Through Slits," NUREG/CR-13418.

Bain, R.S. (1993), *Nonmonotonic Nonlinear Equation Solver (NNES) User's Manual*, Leeds University, Leeds, UK.

Chexal, B., Abdollahian, D., and Norris, D. (1984), "Analytical Prediction of Single-Phase and Two-Phase Flow Through Cracks in Pipes and Tubes," *Proc. Heat Transfer – Niagara Falls – AIChE Symposium Series*, 80(236), 19-23.

Collier, R.P., Stulen, F.B., Mayfield, M.E., Pape, D.B., and Scott, P.M., (1984) "Two-Phase Flow Through Intergranular Stress Corrosion Cracks and Resulting Acoustic Emission," EPRI Report No. NP-3450-LD.

Henry, R.E. (1970), "The Two-Phase Critical Discharge of Initially Saturated or Subcooled Liquid," *Nuclear Science and Engineering*, ANS, 41, 336-342

Henry, R.E., Fauske, H.K., and McComas, S.T. (1970) "Two-Phase Critical Flow at Low Qualities, Part I: Experimental," *Nuclear Science and Engineering*, ANS, 41 79-91,

Henry, R.E. and Fauske, H.K. (1970) "Two-Phase Critical Flow at Low Qualities, Part II: Analysis," *Nuclear Science and Engineering*, ANS, 41, 92-98

Henry, R.E. and Fauske, H.K. (1971), "The Two-Phase Critical Flow of One-Component Mixtures in Nozzles, Orifices, and Short Tubes," *Trans. of the ASME Journal of Heat Transfer*, ASME, 95, 179-187.

IEEE/ASTM SI 10™ 2010 (2011) *American National Standard for Metric Practice*, IEEE Standards Association, ASTM International, and American National Standards Institute, New York, NY, USA.

International Standards Organization ISO/IEC 1539-1:2004: *Information Technology – Programming Languages – Fortran – Part 1: Base Language*, International Organization for Standardization, Geneva, Switzerland, 2004.

ISA-67.03-1982 (1982) *Standard for Light Water Reactor Coolant Pressure Boundary Leak Detection*, Instrument Society of America, Research Triangle Park, North Carolina, USA.

Larman, C. (2005) *Applying UML and Patterns – An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3rd. ed.*, Prentice Hall PTR, Upper Saddle River, NJ.

Unified Modeling Language (2007), OMG formal/2007-11-02, Superstructure, version 2.1.2.

Unified Modeling Language (2007), OMG formal/2007-11-04, Infrastructure, version 2.1.2.

Paul, D.D., Ahmad, J., Scott, P.M., Flanigan, L.F., Wilkowski, G.M. (1994) "Evaluation and Refinement of Leak-Rate Estimation Models," NUREG/CR-5128 (BMI-2164), Rev. 1, Battelle, Columbus, OH.

Regulatory Guide 1.45, Revision 1, (2008) "Guidance on Monitoring and Responding to Reactor Coolant System Leakage," U.S. Nuclear Regulatory Commission, Washington, D.C.

Rouson, D., Xia, J., and Xu, X. (2011), *Scientific Software Design – The Object-Oriented Way*, Cambridge University Press, Cambridge, UK.

Rudland, D. and Harrington, C. (2012) *xLPR Pilot Study Report*, NUREG-2110, U.S. Nuclear Regulatory Commission, Washington, D.C.

Sozzi, G.L. and Sutherland, W.A. (1975), "Critical Flow of Saturated and Subcooled Water at High Pressure," NEDO-13418, General Electric Co., San Jose, CA.

Yano, T., Matsushima, E., and Okamoto, A. (1987), "Experimental Study of Leak Flow Through Analytical Slits," SMIRT-9; and "Leak Flow Rate From Through-Wall Crack In Pipe," *ASME/JSME Thermal Engineering Joint Conference*, Honolulu, Hawaii.

Williams, P.T., Dickson, T.L., and Yin, S. (2007) "Fracture Analysis of Vessels – Oak Ridge, FAVOR v04.1, Computer Code: Theory and Implementation of Algorithms, Methods, and Correlations," NUREG/CR-6854 (ORNL/TM-2004/244), Oak Ridge National Laboratory, Oak Ridge, TN.

Williams, P.T. and Baker, A.J. (1996) "Incompressible Computational Fluid Dynamics and the Continuity Constraint Method for the Three-Dimensional Navier-Stokes Equations," special-issue monograph published by *Numerical Heat Transfer, Part B: Fundamentals* 29(2), 137-273.