INL/EXT-12-27195

Nuclear Hybrid Energy System Modeling: RELAP5 Dynamic Coupling Capabilities

George Mesina
Piyush Sabharwall
Nolan Anderson
Haihua Zhao
Shannon Bragg-Sitton

September 2012

The INL is a U.S. Department of Energy National Laboratory
operated by Battelle Energy Alliance

INL/EXT-12-27195

Nuclear Hybrid Energy System Modeling: RELAP5 Dynamic Coupling Capabilities

George Mesina
Piyush Sabharwall
Nolan Anderson
Haihua Zhao
Shannon Bragg-Sitton

September 2012

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Nuclear Hybrid Energy System Modeling: RELAP5 Dynamic Coupling Capabilities

INL/EXT-12-27195

September 2012

George Mesina 9/27/2012

George Mesina
Nuclear Engineer (Staff Scientist)
Nuclear System Design and Analysis Division

Date

Reviewed by

James Wolf 9/27/12

James Wolf
RELAP5-Program Manager
Nuclear System Design and Analysis Division

Date

EXECUTIVE SUMMARY

Nuclear Hybrid Energy Systems (NHES) could be a key part of the solution to achieving energy security, may provide reliable power availability even with increasing renewable energy penetration into the power grid, and may allow repurposing excess electricity in times of low demand. Incorporating a fission-based power source in a multi-output system (electricity and process heat) can offer significant advantages over carbon-based production sources, such as coal or natural gas, including reduction in atmospheric carbon emissions. In an integrated multi-output system, thermal energy from the nuclear reactor subsystem can be diverted to industrial applications in times of low electricity demand. High temperature, high quality heat from advanced reactor designs might be used for hydrogen production or synthetic fuel production (coal-to-liquid or natural gas-to-liquid processes). Low temperature heat from advanced reactor designs or the current and future fleet of light water reactors (LWRs) might be applied to desalination processes or be stored for later use in low temperature ammonia or hydrocarbon-based thermal cycles.

The NHES research team is currently developing a dynamic simulation of an integrated hybrid energy system. A detailed simulation of proposed NHES architectures will allow initial computational demonstration of a tightly coupled NHES to identify key reactor subsystem requirements, identify candidate reactor technologies for a hybrid system, and identify key challenges to operation of the coupled system. This work will provide a baseline for later coupling of design-specific reactor models through industry collaboration. The modeling capability addressed in this report focuses on the reactor subsystem simulation.

Many computational tools and methods are available for modeling a nuclear reactor, power conversion systems and associated process applications. One method is to develop a single program capable of modeling the entire system. Another is to combine two or more existing computer programs, each modeling a portion of the system for which it is validated, where the programs exchange information to compute the solution. The former method requires a lengthy development and verification and validation process. The latter method leverages industry standard software to reduce development and funding requirements.

Preliminary system models have been developed by Idaho National Laboratory researchers and are currently being enhanced to assess integrated system performance given multiple sources (e.g., nuclear + wind) and multiple applications (i.e., electricity + process heat). Initial efforts to integrate a Fortran-based simulation of a small modular reactor (SMR) with the balance of plant model are being completed in FY12. This initial effort takes advantage of an existing SMR model developed at North Carolina State University to provide initial integrated system simulation for a relatively low cost. These efforts will be reported in a separate project report.

As the system model is enhanced, there is significant interest in adopting an industry standard software package to model the reactor subsystem and other integrated system components. The RELAP5-3D code is both a nuclear industry standard and provides the necessary dynamic coupling technique to connect to other component and subsystem simulations. This approach will allow adoption of standard simulation platforms used by researchers outside the nuclear field, recognizing that nuclear hybrid energy systems represent truly cross-cutting system architectures that bring together researchers from a wide range of industries and experience.

Dynamic code coupling capability with RELAP5-3D© has been developed and exercised for over a decade.[1,2] It is well tested and available for use with the Light Water Reactor Sustainability, Small Modular Reactors, and other projects. This report presents and illustrates some of the key features of the RELAP5-3D code dynamic coupling,[3,4] as well as RELAP5-3D code architecture, capabilities, and verification and validation. This study will inform potential users of the code, of the code's unique capabilities that extend its range of applicability beyond that presently available with any other thermal-hydraulic system code, and introduce the coupling methodology software.

CONTENTS

FIGURES

TABLES

ACRONYMS

ABWR	Advanced Boiling Water Reactor
APWR	Advanced Pressurized Water Reactor
ATR	Advanced Test Reactor
ATWS	Anticipated Transient Without Scram
BWR	Boiling Water Reactor
CASL	Consortium of Advanced Simulation of LWRs
CFD	Computational Fluid Dynamics
DOE	Department of Energy
EP	Execution Path
HPIS	High Pressure Injection System
INEEL	Idaho National Engineering and Environmental Laboratory
INEL	Idaho National Engineering Laboratory
INL	Idaho National Laboratory
IRUG	International RELAP5 User Group
LIME	Lightweight Integrating Multiphysics Environment
LOCA	Loss Of Coolant Accidents
LOFT	Loss Of Fluid Test
LPIS	Low Pressure Injection System
LWR	Light Water Reactor
NGNP	Next Generation Nuclear Plant
NRC	Nuclear Regulatory Commission
PVM	Parallel Virtual Machine
PWR	Pressurized Water Reactor
RIUS	RELAP5 International User Symposium
TH	Thermal Hydraulics
THTF	Thermal Hydraulic Test Facility
V&V	Verification and Validation
VHTR	Very High Temperature Reactor
WD	Working Directory

Nuclear Hybrid Energy System Modeling: RELAP5 Dynamic Coupling Capabilities

1. INTRODUCTION

Many means are available for modeling a nuclear reactor and processing for the hydrogen economy. One method is to develop a single program capable of modeling the entire system. Another is to combine two or more existing computer programs, each modeling a portion of the system for which it is validated, and have them exchange information to compute the solution. The former method requires a lengthy development and verification and validation (V&V) process. The latter method leverages industry standard software to reduce development and funding requirements. The RELAP5-3D code is both an industry standard and provides such a dynamic coupling technique.

Historically, the RELAP5[5] code series began development at Idaho National Engineering Laboratory (INEL) for the U.S. Nuclear Regulatory Commission (NRC) in 1979. The NRC series includes RELAP5/MOD0, RELAP5/MOD1, RELAP5/MOD2, RELAP5/MOD3, whose development began in 1989. The U.S. Department of Energy (DOE) began sponsoring additional RELAP5 development in the early 1980s to meet its own reactor safety assessment needs. The application of RELAP5 to various reactor designs created the need for new modeling capabilities. For example, the Advanced Test Reactor (ATR) required a new critical heat flux correlation applicable to its unique fuel design. Altogether, DOE-sponsored improvements and enhancements have amounted to a multimillion-dollar investment in the code.

At the request of NRC, the DOE enhancements were kept separable from the base code and were always removed before transmission to NRC. RELAP5-3D is an outgrowth of RELAP5/MOD3 created in 1997 by including all prior DOE contributions and continuing its development. The most prominent attributes that distinguish the DOE code from the NRC code are: (1) the fully integrated, multi-dimensional thermal-hydraulic and kinetic modeling capability in the DOE code, and (2) the large library of working fluids. This removes any restrictions on the applicability of the code to the full range of postulated reactor accidents, maneuvers, and operational transients for virtually any kind of reactor.

Because Idaho National Laboratory's (INL) RELAP5 represents state-of-the-art nuclear reactor system safety analysis, and has been for over two decades, the code has continued to evolve. The code's infrastructure has gone from Fortran 66 to Fortran to 95,[13,14] from fixed-memory database to dynamically allocated memory database, [15] from unstructured code to structured programming, [16,17] from sequential to both vectorized[18] and parallelized[19] for shared memory computers, from machine dependent to portable, from Cyber to Cray to Unix workstations to Linux and Windows platforms, and added Graphical User Interfaces.[21] Its physics has gone from five equations to six,[16] has added three-dimensional Thermal Hydraulics (TH) and neutron kinetics, has eliminated some operator-splitting with an implicit coupling of TH and heat conduction,[13] and has added a much more implicit connection between neutronics and TH with a Picard iteration between the INSTANT[23,24] neutronics package and TH. Other numerical method improvements[2] include a second order accurate Godunov Boron tracking scheme, mixture level tracking, fast sparse and Khrylov solvers, perfectly accurate integer time-stepping,[20] and modifications to integrate into the Lightweight Integrating Multiphysics Environment (LIME) for the Consortium of Advanced Simulation of LWRs (CASL). Together with the existing modeling capabilities of RELAP5/MOD3.2, these enhancements make the code a very powerful tool.

RELAP5-3D is distributed worldwide and well-tested by its user community. It is estimated that many tens of millions of dollars have been invested in RELAP5 input model development and input decks by industry, laboratories of many governments, and universities. Graphical User Interfaces such as SNAP have been developed by many external organizations for visualizing RELAP5-3D input and calculations. A system of recording and solving user reported problems is well-developed and a yearly report on problem resolution given to its International RELAP5 User Group (IRUG) at its annual RELAP5 International User Symposium (RIUS).

At the outset of the decision to split the code into NRC and DOE versions in the late 1990s, the Idaho National Engineering and Environmental Laboratory (INEEL) as it was designated from 1997 to 2005 recognized the importance of retaining the pedigree stemming from the extensive validation history of RELAP5/MOD3. To ensure that the code remains true to its validation history (or provides improved results), developmental versions of the code are periodically tested using a subset of cases from the RELAP5/MOD3 validation library. Two such cases are the G.E Level Swell[6] (conducted in the late 1970s provided excellent data for assessing flashing and interphase drag models) and the Thermal Hydraulic Test Facility (THTF) experiments[7] at Oak Ridge National Laboratory (designed to simulate conditions in a Pressurized Water Reactor [PWR] core).

However, no computer program can model everything about every system to the degree needed by every user or situation. Therefore, a means to model more complex systems than just a nuclear reactor has been devised. Domain decomposition is applied to a complex system to divide it into subdomains each of which can be well-modeled by an existing computer program. For example, a Computational Fluid Dynamics (CFD) code might be applied to the lower plenum of a reactor vessel to model complex mixing and thermal gradients, while RELAP5-3D models the rest of the primary and secondary sides of the plant. The programs supply information about the solution on their subdomain to attached subdomains only at the shared boundaries. By exchanging this information as needed, the solution of the complex system is calculated by the coupled codes.

The RELAP5-3D software for this dynamic code coupling is written in Fortran 95 using the Parallel Virtual Machine (PVM) library from Oak Ridge National Laboratory. The program is called the PVM Executive and is conceptually based on previous executive programs that coordinate individual programs coupled in nuclear power plant simulators. To be generic, the PVM Executive defines just over a dozen PVM messages that must be programmed into each participating coupled code. These library calls have already been added to RELAP5-3D, Cobra-Trac, Contain, and other industry standard codes.

Section 2 details some of the RELAP5-3D modeling capabilities. Section 3 covers RELAP5-3D code architecture and processing. Section 4 demonstrates Verification and Validation of RELAP5-3D. Section 5 introduces the PVM Executive and RELAP5-3D dynamic coupling techniques and illustrates it with an example. Section 6 provides a brief introduction to RELAP-7, which is currently being developed. Section 7 summarizes the major points of this document. Section 8 is a Bibliography.

2. RELAP5-3D CAPABILITIES

RELAP5-3D has multi-dimensional thermal-hydraulic and neutron kinetic modeling capabilities. The multi-dimensional component in RELAP5-3D was developed to allow the user to accurately model the multi-dimensional flow behavior that can be exhibited in any component or region of a nuclear reactor coolant system. There is also two-dimensional conductive and radiative heat transfer capability and modeling of plant trips and control systems. RELAP5-3D allows for the simulation of the full range of reactor transients and postulated accidents, including:

- Trips and controls
- Component models (pumps, valves, separators, branches, etc.)
- Operational transients
- Startup and shutdown
- Maneuvers (e.g., change in power level, starting/tripping pump)
- Small and large break Loss Of Coolant Accidents (LOCA)
- Anticipated Transient Without Scram (ATWS)
- Loss of offsite power
- Loss of feedwater
- Loss of flow
- Light Water Reactors:
  - Pressurized Water Reactor (PWR), Advanced PWR (APWR),
  - Boiling Water Reactor (BWR), Advanced BWR (ABWR), etc.
- Heavy Water Reactors (e.g., CANDU reactor)
- Gas-cooled Reactors:
  - Very High Temperature Reactor (VHTR), Next Generation Nuclear Plant (NGNP)
- Liquid metal cooled reactors
- Molten-salt cooled reactors.

2.1 Hydrodynamic Model

RELAP5-3D is a transient, two-fluid model for flow of a two-phase vapor/gas-liquid mixture that can contain noncondensable components in the vapor/gas phase and/or a soluble component in the liquid phase. The multi-dimensional component in RELAP5-3D was developed to allow the user to more accurately model the multi-dimensional flow behavior that can be exhibited in any component or region of a Light Water Reactor (LWR) system. Typically, this will be the lower plenum, core, upper plenum, and downcomer regions of an LWR. However, the model is general, and is not restricted to use in the reactor vessel. The component defines a one, two, or three-dimensional array of volumes and the internal junctions connecting them. The geometry can be either Cartesian (x, y, z) or cylindrical (r, θ, z). An orthogonal, three-dimensional grid is defined by mesh interval input data in each of the three coordinate directions.

The functionality of the multi-dimensional component has been under testing and refinement since it was first applied to study the K reactor at Savannah River in the early 1990s. A set of 10 verification test cases with closed form solutions are used to demonstrate the correctness of the numerical formulation for the conservation equations.

Recent developments have updated the programming language to FORTRAN 95 and incorporated viscous effects in multi-dimensional hydrodynamic models. Currently, RELAP5-3D contains 28 different working fluids including:

- Light water (e.g., 1967, 1984, and 1995 steam tables)
- Heavy water
- Gases (e.g., helium and carbon dioxide)
- Molten salts (e.g., FLiBe and FLiNaK)
- Liquid metals (e.g., sodium and lead-bismuth eutectic)
- Alternative fluids (e.g., glycerin and ammonia)
- Refrigerants (e.g., R-134a).

Working fluids allow single-phase, two-phase, and supercritical applications.

2.2 Thermal Model

Heat structures provided in RELAP5-3D permit calculation of heat transferred across solid boundaries of hydrodynamic volumes. Modeling capabilities of heat structures are general and include fuel pins or plates with nuclear or electrical heating, heat transfer across steam generator tubes, and heat transfer from pipe and vessel walls. Temperature-dependent and space-dependent thermal conductivities and volumetric heat capacities are provided in tabular or functional form either from built-in or user-supplied data. There is also a radiative/conductive enclosure model, for which the user may supply view/conductance factors.

2.3 Control System

RELAP5-3D allows the user to model a control system typically used in hydrodynamic systems, including other phenomena described by algebraic and ordinary differential equations. Each control system component defines a variable as a specific function of time-advanced quantities, which permits control variables to be developed from components that perform simple, basic operations.

2.4 Reactor Kinetics

There are several options that include a point reactor kinetics model and two multi-dimensional neutron kinetics models[3] (NESTLE and INSTANT). A flexible neutron cross section model and a control rod model have been implemented to allow for the complete modeling of the reactor core. The decay heat model developed as part of the point reactor kinetics model has been modified to compute decay power for point reactor kinetics and multi-dimensional neutron kinetics models.[3]

3. RELAP5-3D ARCHITECTURE AND PROCESSING

RELAP5-3D is coded according to the structured programming paradigm, the most strongly modular method available. Each block of code has only one entry point at the top and one exit point at the bottom. Major steps of the solution algorithm correspond to high-level coding blocks organized as subprograms and substeps as either smaller blocks within subprograms or lower-level subprograms. This creates a top-down structure. Thus the various models and procedures are isolated in separate subroutines. The topmost level structure consists primarily of input and transient/steady-state blocks (RELAP5-3D Manual[2] pp. 2-2).

The input block processes input, checks input data, prepares required data blocks for all program options, and places initial data on various output files.

The transient/steady-state block handles both transient and steady-state options. The steady-state option determines the steady-state conditions if a properly posed steady-state problem is presented. Steady-state is obtained by running an accelerated transient until the time derivatives approach zero. Thus, the steady-state option is very similar to the transient option but contains convergence testing algorithms to determine satisfactory steady-state, divergence from steady-state, or cyclic operation. If the transient technique alone were used, approach to steady-state from an initial condition would be identical to a plant transient from that initial condition. Pressures, densities, and flow distributions would adjust quickly, but thermal effects would occur more slowly. To reduce the transient time required to reach steady-state, the steady-state option artificially accelerates heat conduction by reducing the thermal capacity of the conductors.

The following sections discuss the input and transient/steady-state blocks in more detail.

3.1 Input Processing

RELAP5-3D provides detailed input checking for all system models using three input processing phases. The first phase reads all input data, checks for punctuation and typing errors (such as multiple decimal points and letters in numerical fields), and stores the data keyed by card number such that the data are easily retrieved. A list of the input data is provided, and punctuation errors are noted.

A run can be of three types: NEW, RESTART, or STRIP. STRIP is only for more efficient plotting and is fully explained in the RELAP5-3D manuals, but it is not discussed here. A NEW run begins at time zero whereas a RESTART picks up where a previous run left off. This may be at the final time of a previous run, or at any time when RESTART information was written to its restart file. RESTARTS have many uses, such as performing numerical studies from an intermediate point in a transient, changing a plant configuration, or saving input processing time by starting at the time zero restart.

During the second phase, in a NEW problem, storage is dynamically allocated in Fortran modules to exactly hold the data. Some specific data is of fixed size also goes into modules. Note that if a particular data is not used in the model description, for example trips and controls in the rigid body rotation problem of Figures 1 and 2, then dynamic storage of that type is not created.

During the second phase, if the problem is a RESTART type, restart data is read from specified time in the restart file, then all input data are processed. In RESTART problems, dynamic storage space may be created, deleted, added to, partially deleted, or modified as modeling features and components within models are added, deleted, or modified by subsequent input. Extensive input checking is done, but at this level, checking is limited to new data from the cards being processed. Relationships with other data cannot be checked because the latter may not yet be processed. As an illustration of this level of checking, junction data are checked to determine if they are within the appropriate range (such as positive, nonzero, or between zero and one) and volume connection codes are checked for proper format. No attempt is made at this point to check whether or not referenced volumes exist in the problem until all input data are processed.

The third phase of processing begins after all input data have been processed. Since all data have been placed in fixed or dynamic module storage space during the second phase, complete checking of interrelationships can proceed. Examples of cross-checking are existence of hydrodynamic volumes referenced in junctions and heat structure boundary conditions; entry or existence of material property data specified in heat structures; and validity of variables selected for minor edits, plotting, or used in trips and control systems. As the cross-checking proceeds, cross-linking of the stored data is done so that it need not be repeated at every time step. The initialization required to prepare the model for the start of the transient advancement is done at this level.

Input data editing and diagnostic messages can be generated during the second and/or third phases. Input processing for most models generates output and diagnostic messages during both phases. Thus, input editing for these models appears in two sections.

As errors are detected, various recovery procedures are used so that input processing can be continued and a maximum amount of diagnostic information can be furnished. Recovery procedures include supplying default or benign data, marking the data as erroneous so that other models do not attempt use of the data, or deleting the bad data. The recovery procedures sometimes generate additional diagnostic messages. Often after attempted correction of input, different diagnostic messages appear. These can be due to continued incorrect preparation of data, but the diagnostics may result from the more extensive testing permitted as previous errors are eliminated.

3.2 Transient Setup

After input processing is complete, if the problem has no errors and is not merely an input check, then further preparations are made for transient/steady-state (referred to as simply transient from here forward) processing are conducted. Here, pointers for the many quantities such as time-dependent volumes and junctions and PVM coupling data are set up. Also, memory for temporary storage used throughout the transient are created. Many quantities associated with the linear equation solvers are numbered and their connectivity recorded. Finally, dynamic storage is set aside for the linear equation solvers.

3.3 Transient Processing

Subroutine TRAN is the driver that controls the transient advancement of the solution; see page 2-4 Figures 2.4-2 and 2.4-3 of RELAP5-3D Manual Volume I.[2] Nearly all the execution time is spent in this block. A description of the function of each of the second-level subroutines driven by TRAN follows. Under the control of TRAN, these routines and their subroutine trees perform all the calculations for each advancement of the transient.

The level module (CHKLEV) controls the movement of two-phase mixture levels between volumes.

The trip system module (TRIP) evaluates logical statements. Each trip statement is a simple logical statement that has a true or false result. The decision of what action is needed resides within the components in other modules. For example, valve components are provided that open or close the valve based on trip values; pump components test trip status to determine whether a pump electrical breaker has tripped.

The equation of state boundary volume module (TSTATE) calculates the thermodynamic state of the fluid in each hydrodynamic boundary volume (time-dependent volume). This subroutine also computes velocities for the time-dependent junctions.

The heat structure module (HTADV) advances heat conduction/transfer solutions. It calculates heat transferred across solid boundaries of hydrodynamic volumes.

The hydrodynamics module (HYDRO) advances the hydrodynamic solution.

The reactor kinetics module (RKIN) advances the reactor kinetics of the code. It computes the power behavior in a nuclear reactor using the space-independent or point kinetics approximation, which assumes that power can be separated into space and time functions. It also optionally computes the power using a multi-dimensional nodal kinetics model.

The control system module (CONVAR) provides the capability of simulating control systems typically used in hydrodynamic systems. It consists of several types of control components. Each component defines a control variable as a specific function of time-advanced quantities. The time-advanced quantities include quantities from hydrodynamic volumes, junctions, pumps, valves, heat structures, reactor kinetics, trip quantities, and the control variables themselves. This permits control variables to be developed from components that perform simple, basic operations.

The time step control module (DTSTEP) determines the time step size, controls output editing, and determines whether the transient advancements should be terminated. During program execution, this module displays such information as CPU time, problem time, time step size, and advancement number on the standard output, usually a terminal screen.[2]

4. RELAP5-3D VERIFICATION AND VALIDATION

The multi-dimensional component in RELAP5-3D was developed to allow the user to more accurately model the multi-dimensional flow behavior that can be exhibited in any component or region of a LWR system. Typically, this will be the lower plenum, core, upper plenum, and downcomer regions of an LWR. However, the model is general, and is not restricted to use in the reactor vessel. The component defines a one, two, or three-dimensional array of volumes and the internal junctions connecting them. The geometry can be either Cartesian (x, y, z) or cylindrical (r, θ, z). An orthogonal, three-dimensional grid is defined by mesh interval input data in each of the three coordinate directions. Viscous terms have been added in both coordinate systems, including a rudimentary turbulence model.

4.1 Verification

Verification is a procedure for checking that the code of portion thereof, meets its design specification as recorded in RELAP5-3D Manual Volume 1: Code Structure, System Models, and Solution Methods. The verification has been performed by using conceptual problems that have exact solutions. These types of problems are used to demonstrate that the equations have been correctly coded, and are a precursor to model validation using experimental data. Three such problems are the “Pure Radial Symmetric Flow,” “Rigid Body Rotation,” and R-θ Symmetric Flow” problems.[8] Each of the problem is based on a cylindrical, multidimensional component with eight rings, six sectors, and one axial level. All six sectors are symmetrical, with non-uniform radial spacing. Six time-dependent volumes are attached to the six outer sectors by time-dependent junctions for inlet flow specification. In addition, six time-dependent volumes are attached to the six inner sectors by a multiple junction component.

The Rigid Body Rotation problem represents a hollow cylinder with a symmetric flow pattern in the azimuthal direction. Comparisons between the RELAP5-3D calculated results and the analytic solution for the Rigid Body Rotation problem are shown in Figures 1 and 2. These are reproductions of Figures 3.13-2 and 3.13-3 on pages 3-60 to 3-61 of the *Developmental Assessment Manual*.[8]

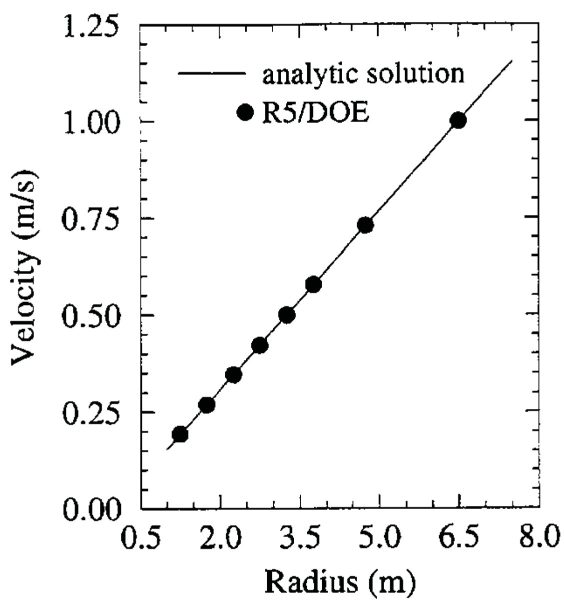

Figure 1. Azimuthal velocity profile for rigid body rotation problem.

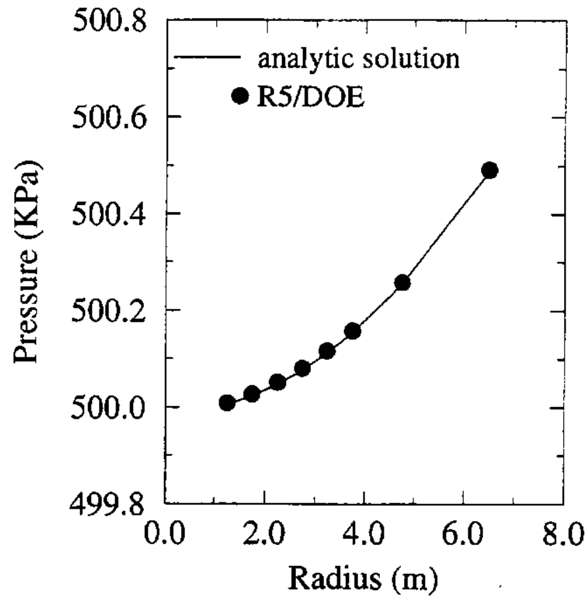

Figure 2. Radial pressure profile for rigid body rotation problem.

The Pure Radial Symmetric Flow problem represents a hollow cylinder with a symmetric flow pattern in the radial direction. This problem tests only the radial momentum terms. Comparisons between the calculated results and the analytic solution for the Pure Radial Symmetric Flow problem are shown in Figures 3 and 4. These are reproductions of Figures 3.12-2 and 3.12-3 on pages 3-56 to 3-57 of the Developmental Assessment Manual[8] Again, the RELAP5-3D results are in agreement with the exact solution.

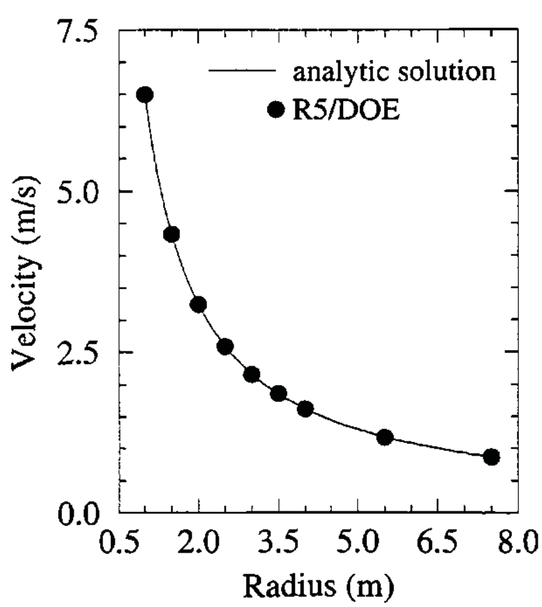

Figure 3. Radial velocity profile for pure radial symmetric flow problem.

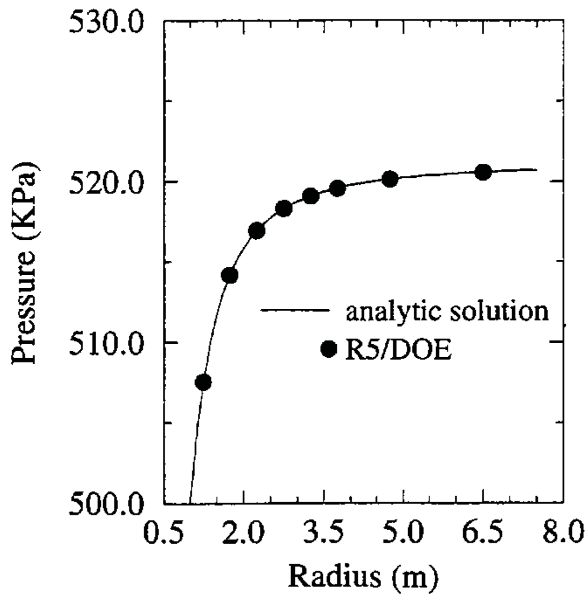

Figure 4. Radial pressure profile for pure radial symmetric flow problem.

The R-θ Symmetric Flow problem represents a hollow cylinder with a symmetric flow pattern in both the radial and azimuthal directions. The calculated and analytic solution results are seen to be in agreement (Figures 5, 6, and 7). These are reproductions of Figures 3.14-2, 3.14-3, and Figure 3.14-4.on pages 3-64 to 3-66 of the Developmental Assessment Manual[8]

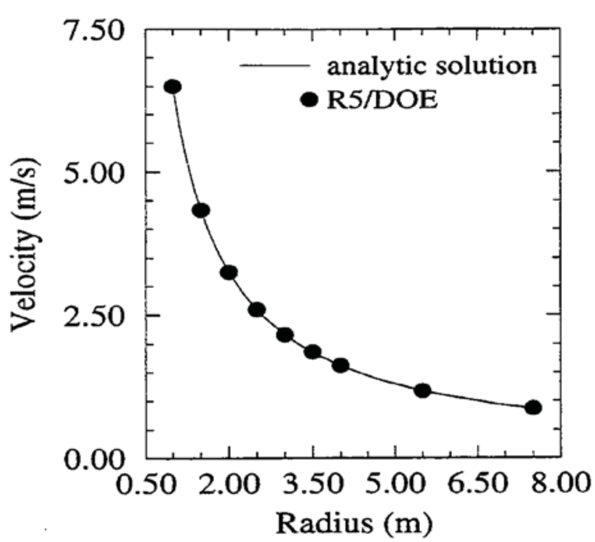

Figure 5. Radial velocity profile for R-θ symmetric flow problem.

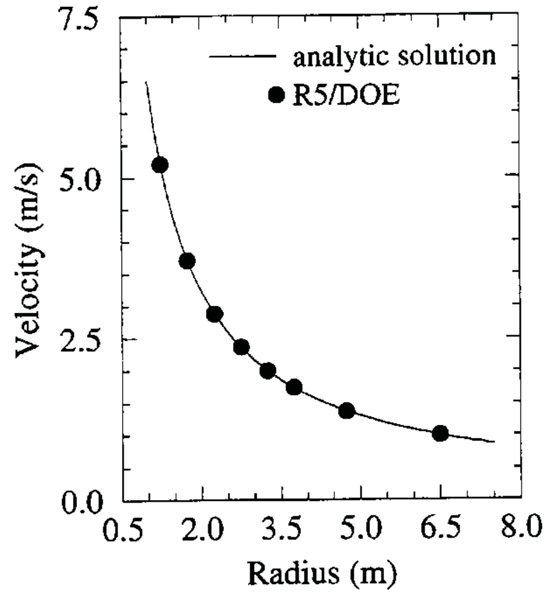

Figure 6. Azimuthal velocity profile for R-θ symmetric flow problem.

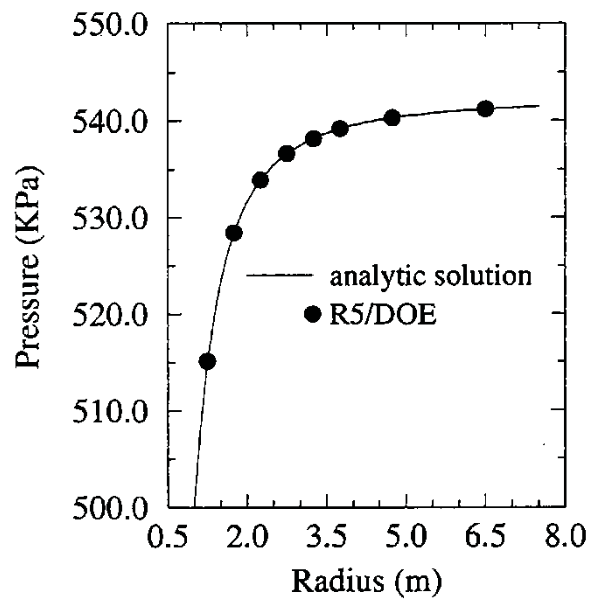

Figure 7. Radial pressure profile for R-θ symmetric flow problem.

These simple test problems verify the correct implementation of the three dimensional continuity and momentum equations. This is a necessary, but certainly not sufficient, test of the three-dimensional model. A number of validation cases using experiments exhibiting multidimensional flow behavior are also carried out and briefly mentioned in the later section.

4.2 Validation

The Loss Of Fluid Test (LOFT)[9] facility was a 50-MW PWR that was designed to simulate the response of a commercial PWR during a LOCA. Test L2-5 simulated a double-ended offset shear of a cold leg. The experiment was selected because it was judged to provide the most challenging test of the multi-dimensional model of all the experiments in the existing developmental assessment test matrix used for RELAP5/MOD3.

The RELAP5/MOD3 model that was used in the developmental assessment of LOFT Test L2-5 is illustrated in Figure 8 (next page). This is a reproduction of Figure 5.5-2.[9] The model contains 131 volumes, 142 junctions, and 77 heat structures.

Steady-state calculations were performed for LOFT L2-5 with both the one-dimensional and three-dimensional models. Table 1 shows that the results of the steady-state calculations were in excellent agreement with the measurements.

Table 1. A comparison of calculated and measured initial conditions in LOFT L2-5.

Parameter	Measured Value	One-dimensional Model	Three-dimensional Model
Intact loop			
Mass flow (kg/s)	192.4 ± 7.8	192.4	192.4
Hot leg pressure (MPa)	14.94 ± 0.06	14.92	14.92
Cold leg temperature (K)	556.6 ± 4.0	556.6	556.7
Hot leg temperature (K)	589.7 ± 1.6	590.4	590.5
Pressurizer liquid level (m)	1.14 ± 0.03	1.14	1.14
Average pump speed (rad/s)	131.5 ± 1.2	130.7	130.7
Pump differential pressure (kPa)	73.3 ± 9.2	63.4	63.5
Reactor vessel			
Power (MW)	36.0 ± 1.2	36.0	36.0
Maximum linear heat generation rate (kW/m)	36.0 ± 2.7	34.3	34.3
Maximum fuel centerline temperature (K)	1660 ± 57	1710	1712
Differential pressure (kPa)	28.0 ± 1.4	28.1	28.1
Steam generator secondary side			
Pressure (MPa)	5.85 ± 0.06	5.85	5.86
Mass flow (kg/s)	19.1 ± 0.4	19.1	19.1
Feedwater temperature (K)	482.0 ± 1.2	482.0	482.0
Accumulator			
Pressure (MPa)	4.29 ± 0.06	4.29	4.29
Temperature (K)	303.2 ± 6.1	303.2	303.2
Liquid level above standpipe (m)	1.17 ± 0.01	1.16	1.16

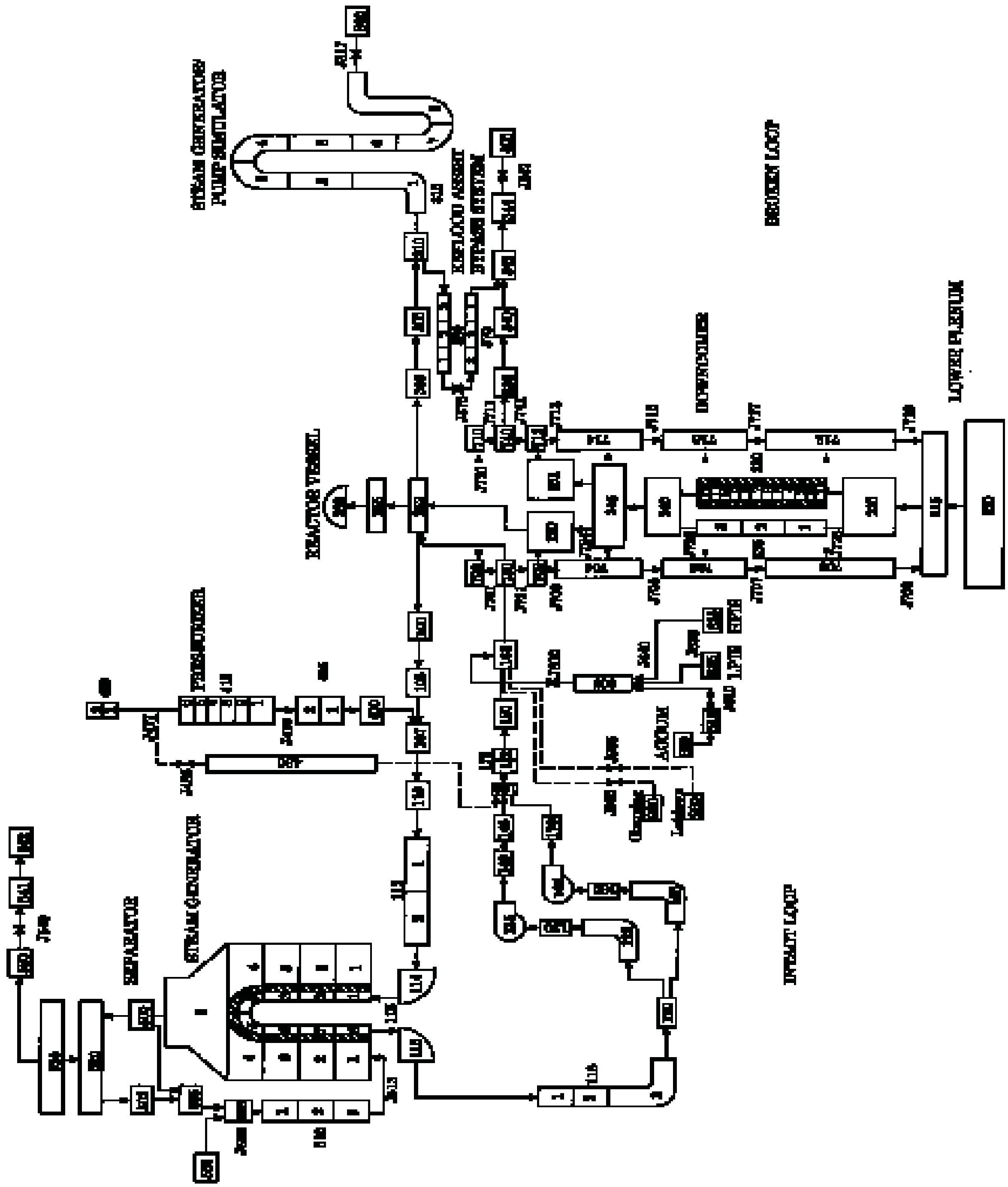

Figure 8. RELAP5/MOD3 nodalization for LOFT L2-5.

The measured sequence of events for LOFT L2-5 is presented in Table 2. The test was initiated at 0.0 s when the quick-opening blowdown valves began to open. A reactor trip signal was generated at 0.02 s on low hot leg pressure, and the reactor scrammed shortly thereafter. The operators tripped the primary coolant pumps at 0.94 s. Flow from the accumulator, High Pressure Injection System (HPIS), and Low Pressure Injection System (LPIS) began at 16.80 s, 23.90 s, and 37.32 s, respectively. The accumulator emptied at 49.6 s. The fuel rod cladding temperature first departed from near the saturation temperature at 0.91 s. The peak cladding temperature occurred at 28.47 s. All of the core cladding had been quenched by 65 s, concluding the interesting portion of the test.

Table 2 also presents the calculated event times with the one-dimensional and three-dimensional RELAP5-3D models. The calculated event times with both the 1D and 3D models were generally in reasonable quantitative agreement with the measured values. One exception was that the calculated peak cladding temperatures with both models occurred near 6 s, compared to about 28 s in the test. As will be shown later, the measured cladding temperatures increased slowly between 5 s and 28 s while the calculated temperatures decreased slowly, so the effect of the difference in timing is not as significant as might be inferred from Table 2. The core cladding also quenched 10 s to 15 s earlier in the calculations than in the test.

Table 2. Calculated and measured sequence of events for LOFT L2-5.

	Time after rupture (s)		
Event	Test	One-dimensional model	Three-dimensional model
Test initiated	0.0	0.0	0.0
Reactor trip signal	0.02 ± 0.01	0.02	0.02
Quick opening blowdown valves fully opened	0.04 ± 0.01	0.04	0.04
Primary coolant pumps tripped	0.94 ± 0.01	1.60[1]	1.60[1]
Subcooled break flow ended (cold leg)	3.4 ± 0.5	3.7	3.9
Steam control valve fully closed	9.38 ± 0.05	9.38	9.38
Pressurizer emptied	15.4 ± 1.0	15.5	15.5
Accumulator injection initiated	16.8 ± 0.1	15.0	14.3
HPIS injection initiated	23.90 ± 0.02	23.90	23.90
Maximum cladding temperature reached	28.47 ± 0.02	6.0	6.3
LPIS injection initiated	37.32 ± 0.02	37.32	37.32
Accumulator emptied	49.6 ± 0.1	50.0	50.5
Core cladding quenched	65 ± 2	49	55

5. RELAP5-3D DYNAMIC COUPLING

Three major methods have been developed to couple RELAP5-3D with other codes for solving more complex problems. Each method has been exercised to couple RELAP5-3D.

The dynamic coupling of RELAP5-3D is primarily achieved via PVM. Recently CD ADAPCO has also added the capability to couple RELAP5-3D to their STAR-CCM+ CFD software.[22] This capability is currently under development, with some existing software bugs. A third method involves writing a Fortran 95 module that stores common information between RELAP5-3D and another Fortran code. The module contains subprograms that move and translate data between the two codes. This method was used to couple INSTANT with RELAP5-3D.[23,24] Whereas the former two methods allow two separate executable codes to couple, the latter produces a single executable when linked and loaded.

For the purpose of this document, only the PVM technique is considered. Other methods should also be investigated for the specific RELAP5-3D application.

A PVM executive program[13] was written in Fortran 95 with PVM Library calls. It is used to couple RELAP5-3D with any number of other computer programs to perform integrated analyses of nuclear power reactor systems and related experimental facilities. The executive program manages all phases of a coupled computation. It starts up and configures a virtual machine, spawns all of the coupled processes, coordinates the time step size between the coupled codes, manages the production of printed and plotable output, and shuts the virtual machine down at the end of the computation. The executive program also monitors the status of the coupled computation, repeating time steps as needed and terminating a coupled computation gracefully if one of the coupled processes is terminated by the computational node on which it is executing.

5.1 Design of the PVM Executive Program

The PVM executive program has five major responsibilities:

1. It configures the virtual machine, starting the PVM daemon process on the computational nodes comprising the virtual machine.
2. It starts up the coupled processes on each computational node.
3. It tells each of the coupled processes what data to send to and what data to receive from the other processes.
4. It manages the time advancements of the coupled computation, coordinating the time step size between coupled codes, monitoring the status of the advancements and directing code backups and time step repeats as necessary.
5. It coordinates the production of printed and plotable output between the coupled codes so that computational results are available from all of the coupled codes at the same simulation times during the computation.

The user supplies the information needed by the PVM executive program in an input file. The input file is divided into four sections (the input needed to accomplish fourth and fifth responsibilities are contained in the same section of the input file). The sections of the input file are delimited by reserved keywords: Virtual, Processes, Messages, Timesteps.

An example is shown in Figure 9 that illustrates the operations of the PVM Executive with a simple model of a pressure vessel. First, the system model is subdivided into subdomains called “core” and (core) “bypass,” shown in dashed lines, and “primary” which is comprised of the rest of the vessel.

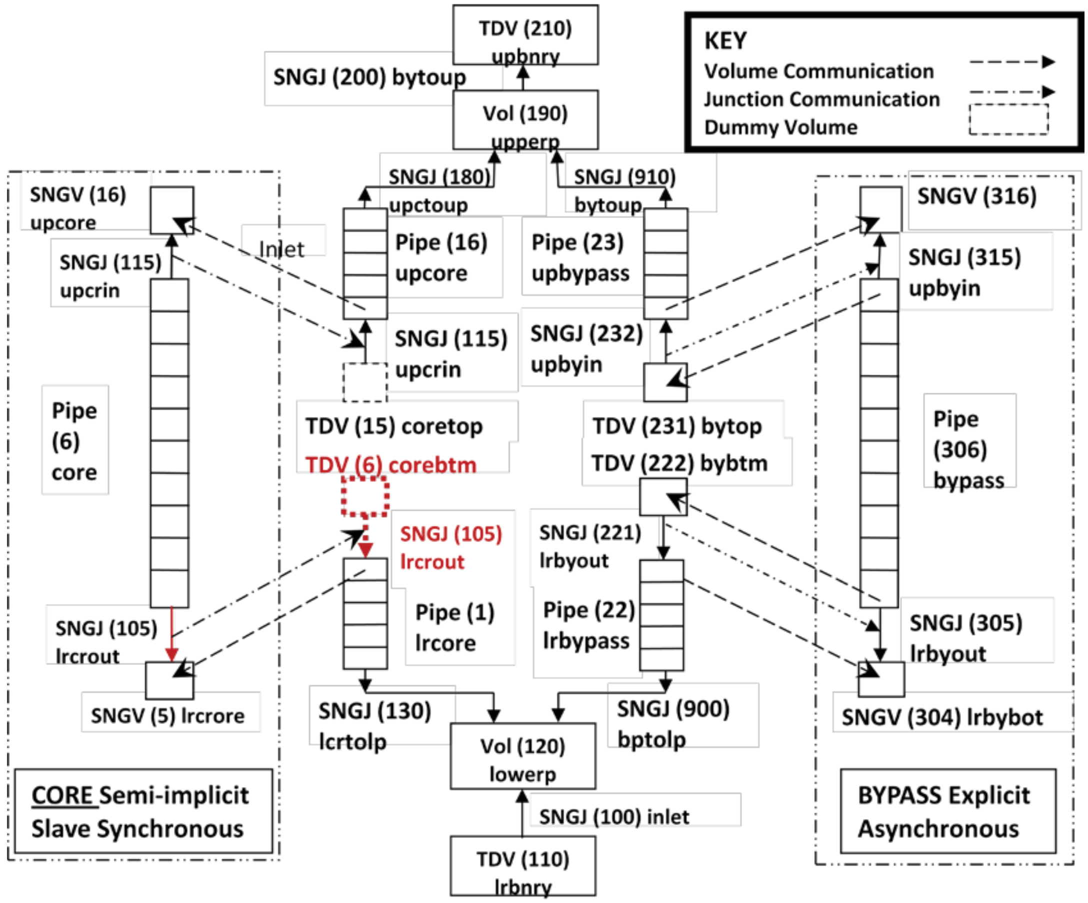

Figure 9. Domain Decomposition of a vessel model into 3 subdomains for 3 coupled codes.

The first or Virtual section of the Executive input file specifies the names of the computational nodes to be used in the virtual machine along with the location of the executable files to be used by that computational node and the location of the input files for the processes to be executed on that computational node (i.e., the working directory). Note that it is possible to run just one code, not coupled to anything, but under the control of the PVM Executive so that there is only one "coupled" process. Using this information, the PVM executive program builds a PVM hostfile for the virtual machine and starts the PVM daemon process on each computational node. For example, to run everything on Eos in Working Directory (WD) ~/relapInput with an Execution Path (EP) in ~/relap, the first section would be:

virtual
eos wd=~/relapInput ep=~/relap

The second or Processes section of the Executive input file specifies each process (i.e., code) to be executed on a computational node, its command line information, and its time-step synchronicity.

Each name of a computational node contained in the first section of the Executive input file becomes a keyword in its second section. One or more coupled processes may be executed on each computational node. The specification of each coupled process contains a unique name for that process as well as any command line parameters that are to be passed to it (names of input files, output file, etc.) as execution begins. The names of the coupled processes are used to distinguish multiple instances of the same executable file being executed in the virtual machine.

In the Processes section, each process is labeled “synchronous” or “asynchronous.” This label indicates whether or not the time step size for the process is determined by the executive program. Synchronous processes, such as processes that are coupled semi-implicitly, need to use the same time step size for each advancement, so their time step size is coordinated by the executive program. Asynchronous processes, such as processes coupled explicitly, need only exchange data at fixed intervals and it does not matter what time step sizes are used to advance, only that they reach the same point in time to exchange data. For example for a RELAP5-3D plant model of the primary side broken into (1) core, (2) core-bypass, and (3) rest of the primary, called simply primary, the specification would be:

processes

eos

core	*synchronous*	*relap5. x -i cj3wayc.i -o cj3wayc.p*
bypass	*asynchronous*	*relap5.x -i cj3wayb.i -o cj3wayb.p*
primary	*synchronous*	*relap5.x -i cj3wayp.i -o cj3wayp.p*

The third or Messages section of the input file specifies the data to be sent to and received from the other coupled processes. Each message uses the unique name of the sending and receiving process along with the specification of the data to be sent or received. The specification of messages occurs in pairs, one for the process sending the data, another for the process receiving the data. The data items to be sent are specified in terms that the sending code uses and vice-versa for the receiving code. This means that the same data item may be specified by a different identifier in the two processes. For example, the sending code may refer to the liquid density using the code variable “rhof” while the receiving code calls it “rholiq”. The data specifications are sent to the several coupled codes as they appear in the third section of the input file. It is the responsibility of the individual coupled codes to understand their data specification.

As mentioned above in the explanation of time-step synchronicity, there are two time step advancement schemes: explicit and semi-implicit. Explicit coupling is simple to implement in new codes, while semi-implicit requires much more theory, algorithm, and programming development. However, explicit coupling has its time step restricted by the sound-speed Courant limit, where as semi-implicit has a material Courant limit, which is larger. Only one semi-implicit coupling is allowed among the processes, the rest must be explicit. Also in semi-implicit coupling, one process is the Master process while the Other is the Slave; this is not to be confused with the PVM Executive which is the Parent of all spawned processes including the Master and Slave.

Continuing the example with a three-way decomposition of the flow region, let the “primary” process be the Master and “core” be the Slave of a semi-implicit coupling. The rules for constructing are in the manual, but an example of this section would be:

messages

semi-implicit

#	*Process*	*Action*	*2nd Proc*	*Data Spec.*	*2nd Data Spec.*
	primary	*sends*	*core*	*lrcore 5*	*upcore 1*
	primary	*receives*	*core*	*lcrout 0*	*upcrin 0*
	core	*sends*	*primary*	*lrcout 0*	*upcrin 0*
	core	*receives*	*primary*	*lrcore 1*	*upcore 1*

explicit

	primary	*sends*	*bypass*	*lrbypass 5*	*upbypass 1*
	primary	*receives*	*bypass*	*bybtm 1*	*bytop 1*
	bypass	*sends*	*primary*	*bypass 1*	*bypass 10*
	bypass	*receives*	*primary*	*lrbybot 1*	*upbytop 1*

The last or Timesteps section of the input file contains data for one of more simulation intervals during the coupled computation. The data for each simulation interval are:

- Its end time
- Its maximum and minimum time step sizes
- Its special control information flag (see the manual)
- Its frequencies of output to the print file, plot file, and restart file
- Its frequency for explicit coupling exchanges (if any).

Frequency is given as an integer. Its interpretation is that the output or explicit exchange occurs at every whole number multiple of the frequency value and the maximum timestep. Thus if the maximum time step is 0.25 and the frequency is 30 for plots and 50 for print output, then information is placed on the plot file every time a process reaches a multiple of 7.5 seconds and on the printed output file every multiple of 12.5 seconds. An example of timestep card section is:

timesteps

# End	*Min dt*	*Max dt*	*Special*	*Plot Freq*	*Print Freq*	*Restart Freq*	*Explicit Exchange*
0.5000	*0.000001*	*0.00625*	*003*	*1*	*20*	*500*	*1*
1.0000	*0.000001*	*0.00625*	*003*	*20*	*20*	*500*	*1*
	0.000001	*0.0125*	*003*	*10*	*50*	*500*	*1*

5.2 Sequence of Events in a Coupled Computation

A coupled computation can be divided into two phases. The first is the input and initialization phase, and the second is the transient simulation phase of the coupled computation.

5.3 Input and Initialization Phase

The PVM executive program is executed by the user in a manner appropriate for the users operating system specifying the input file and the output file for the executive program as command line parameters (default input and output files are also defined).

The executive program reads the Virtual section of its input file, constructs a PVM hostfile, and starts the PVM daemon process on each computational node in the virtual machine. Then the executive program reads the rest of its input, and uses the information in its Processes input section to spawn all coupled processes on each computational node. The coupled processes that are spawned read their respective input files, process the data contained in their input files and then listen to receive messages from the executive process. After the executive process has spawned all of the coupled processes, it sends information from the Message of its input file to each of the spawned processes. The information contains the data specifications for messages to send to and receive from the other coupled processes. Each spawned process proceeds with its own input and initialization after the coupling data specifications have been received from the executive process. The executive process listens to receive a message from each process describing its initialization status and its run status.

Each coupled code sends its initialization status to the executive program at the end of its initialization process. This initialization status may be zero (initialization successful) or one (errors during input and initialization). They also sent the executive program their run status, where zero denotes no transient to be executed because of input or initialization errors or because this run was for input checking only, or one, ready for transient simulation. The coupled computation is terminated if any of the coupled processes return an initialization error or returns a zero run status. The executive program determines the global initialization and run status and broadcasts this status to all of the coupled processes.

5.4 Transient Computation Phase

Assuming that the initialization was successful for all of the coupled processes and that the run status indicated that all coupled processes are ready to perform a transient simulation, the executive program broadcasts an initial set of output control times. This message specifies the next simulation times for the production of printed output, printing of RELAP5-3D minor edit variables, generation of plot data, writing of restart data and the next time explicit coupling data transfers are to be performed. The PVM executive program assumes that each code will produce its own initial printed output, plot data, and restart data automatically. The executive program then coordinates the initial exchange of any explicit coupling data between asynchronously coupled processes. When more than two codes are coupled explicitly, the data exchanges between the codes needs to be coordinated by the executive program.

The data exchange paradigm used in the PVM coupled computation is that all messages received will be followed by an acknowledgement returned to the sender. The sending process waits to receive an acknowledgement before sending the next message. If all of the codes were to send all of their messages and then listen to receive all of their messages, there would be a deadlock condition because all processes would be sending and no processes would be listening for acknowledgements. The executive program broadcasts the PVM identifiers of each of the explicitly coupled processes to all of the explicitly coupled processes one at a time. The process named in the broadcast message sends its data and all of the other explicitly coupled process listen to receive the messages sent by the process named by the executive process. This works like a cash register line, where each customer (process) must wait its turn to exchange money for goods with the cashier (Executive). Each explicit process receives permission to send its data in its turn. This process of coordinating the exchange of explicit coupling data occurs each time the simulation time reaches the time for an explicit exchange of data.

Once any initial explicit coupling data is exchanged, time step advancements may begin. The executive program listens to receive a time step size from each of the synchronously coupled processes. Each synchronically coupled code determines the time step size that it wants to use and sends it to the PVM executive. The executive program receives the several time step sizes, determines a global time step size as the minimum of the time step sizes received from the synchronously coupled processes and broadcasts the global time step size back to the synchronously coupled processes. This message also contains updated edit, print, and plot times so that output may be produced each time step rather than at predetermined intervals.

Output every time step is useful in debugging and this capability existed previously in RELAP5-3D. After the synchronously coupled processes receive the global time step size from the executive program, they proceed with the time step advancement, performing any communication needed with the other coupled processes during the advancement. At the end of the advancement, just before the point of no return, each of the synchronously coupled processes sends its advancement status to the PVM executive program. The point of no return is that point in the computations sequence after which no backup may be performed in order to fix any errors that occurred during the advancement. The executive program listens to receive the advancement status from all of the synchronously coupled processes, determines the global advancement status, and broadcasts the global advancement status to the synchronously coupled processes. Assuming that no errors have occurred during the advancement, the coupled codes and the executive program proceed to the next time step advancement. Time step advancements are performed until the end time for the simulation is reached. The executive program assumes that all of the coupled processes will terminate automatically when the end time is reached. The executive program waits for the coupled code to finish, and then shuts down the virtual machine.

5.5 Error Handling

If any one of the synchronously coupled processes encounters an error during its advancement, it specifies the type of error in its advancement status flag. There are three categories of advancement errors; errors that cause the computation to terminate, errors that cause a backup and time step repeat with a smaller time step size, and errors that cause a backup and time step repeat with the same time step size. RELAP5-3D has 10 types of advancement errors that the code detects and for which there are methods for fixing the errors. If the advancement status flag indicates a code termination, the executive sends a terminate message to all of the coupled processes, waits for confirmation that all of the coupled processes have terminated, and shuts down the virtual machine as if the computation had finished successfully. If the advancement status indicates that any one of the coupled codes wants to perform a backup and a time step repeat, a backup message is sent to all of the coupled processes and the time step is repeated as if the time step had been successful. That is, each process sends its desired time step size to the executive and a new global time step size is chosen.

If the process requesting the time step repeat needs to reduce the time step size, it sends a reduced time step size to the executive program and the global time step size will be reduced per their request. If the type of failure can be fixed using the same time step size, the global time step size will remain the same but the advancement will be repeated from the same starting point. The code requesting the time step repeat must remember the reason for the time step repeat and proceed through a different logic path during the repeated advancement to avoid the error that caused the time step repeat.

The previous discussion assumes that all of the coupled processes continue to execute and do not fail catastrophically. If any of the coupled process fails catastrophically with a divide by zero, floating point overflow, etc., the process will be terminated by the operating system of the computational node on which the process is executing. The executive process monitors the execution status of all of the coupled processes and sends a terminate message to all executing processes if one of them fails catastrophically. It then waits for all of the processes to terminate and then shuts the virtual machine down. The same process of termination occurs if a process exceeds its wait time while waiting for a message or message acknowledgement from another process. The user defines the length of time a process is to wait to receive a message or message acknowledgement from another coupled process. If the wait time is exceeded, the process exceeding the wait time sends a message to the executive program and shuts itself down gracefully. If the executive program receives a time-out message, it broadcasts a terminate message to all of the other coupled processes, waits for them to terminate, and then shuts the virtual machine down as if the coupled computation had terminated normally. The wait time was implemented for the case in which a coupled code might get into a infinite loop where it never sends an expected message, but also never fails catastrophically.

For the user to understand the state of the coupled computation, status messages are written to the terminal from which the PVM executive process was executed at 10 second intervals. These messages are similar to the messages that RELAP5-3D writes to the terminal screen when it executes as an uncoupled process. The messages contain the current simulation time and the current advancement count. Failure messages are also written to the terminal so that the user will understand why the coupled computation terminated. These status messages are also written to the output file of the PVM executive program.

5.6 Verification of PVM Executive Program

The operation of the PVM executive program was verified by executing several test cases under the control of the PVM executive program. These test cases use two instances of RELAP5-3D coupled to each other executing on the same computational node. The results from the execution of the test cases as a single uncoupled process were compared to the results of the execution of the same test cases under the control of the PVM executive program. Examination of the two versions of each test case showed that identical results were obtained. These test cases assume that no initialization or advancement faults occur and in fact the input for these test cases had previously been adjusted so that no faults occurred.

Test cases were developed for each of the 10 types of advancement faults in RELAP5-3D. Two versions of each of the 10 test cases were developed, that is one version of the test case as an uncoupled computation and the other as a coupled version of the test case executed under the control of the PVM executive program. The results from the execution of the two versions of each test case were compared and identical results were obtained. These 10 test cases verify that the PVM executive program recognizes the several types of advancement faults, directs the coupled processes to perform a code backup, and coordinates the repeated attempted advancement.

Finally, the several types of catastrophic failures were simulated by using the “kill” operating system command to manually terminate one of the coupled processes during both the input and initialization phase of a coupled computation and during the transient phase of a coupled computation. The correct messages were written to the terminal and output file for the PVM executive program, the other process in the coupled computation was shut down gracefully, and the virtual machine was shut down as designed. The time-out mechanism was also tested by manually interrupting the execution of one of the coupled processes and observing that the correct time-out messages were sent, that the processes terminated as directed, and the virtual machine was shut down.

6. RELAP-7: THE NEXT GENERATION OF RELAP

INL has a legacy of developing reactor system safety analysis codes spanning several decades with its flagship products—the RELAP3, RELAP4, and RELAP5 codes—widely used worldwide to perform reactor safety analyses. The RELAP-7[25] code is the next generation nuclear reactor system safety analysis code being developed at INL. RELAP-7 is a new project that was started in October 2011. The code is being developed to take advantage of the significant progress that has been made in the development of innovative, fully coupled algorithms for solving the partial differential equations describing strongly coupled and complex phenomena like those in a nuclear power plant. These new algorithms provide a mechanism to remove traditional operator splitting errors, and to allow the coupled equations to be solved implicitly with improved accuracy and efficiency. Modern well-posed two-phase flow models will allow the employment of more accurate temporal and spatial discretization.

The RELAP-7 coding is built on a Multiphysics, Object Oriented Simulation Environment (MOOSE) framework,[26] INL's framework for solving computational engineering problems in a well-planned, managed, and coordinated manner. By leveraging millions of lines of open source software packages such as PETSC (a nonlinear solver developed at Argonne National Laboratory) and LibMesh (a Finite Element Analysis package developed at the University of Texas), MOOSE significantly reduces the expense and time required to develop new applications. Numerical methods, mesh management, and parallel computational controls are all provided within MOOSE.

To summarize, RELAP-7 is being developed as a new generation RELAP series code aiming at enabling a new safety analysis paradigm. The goals of RELAP-7 include maintaining RELAP5 capabilities, and greatly extend its capabilities. The first official version of RELAP-7 is envisioned to be released near the end of 2016 while the first beta version will be released near the end of 2014.

7. CONCLUSIONS AND FUTURE EFFORTS

Preliminary NHES simulation is being accomplished through the integration of a Fortran-based simulation of a small modular reactor (SMR), developed by North Carolina State University, with the balance of plant simulation, developed using the Dymola software and Modelica language at the INL. This initial effort takes advantage of an existing SMR model developed at NCSU to provide initial integrated system simulation for a relatively low cost. As the overall system simulation is enhanced, there is significant interest in adopting an industry standard software package to model the reactor subsystem and other integrated system components. RELAP5-3D is being considered for use in a nuclear hybrid energy system simulation for a variety of reasons:

- RELAP5-3D represents state-of-the-art of nuclear reactor system safety analysis, has an international user group, a huge library of qualified input models, and is available today.
- RELAP5-3D comes equipped with means to couple with other computer programs to solve complex problems via domain decomposition and information exchange at the subdomain boundaries. The communication is carried out through the PVM message passing library.
- A PVM executive program has been developed to control and coordinate the coupled computation of several computational codes for solving a complex system. The design of this executive program has been described along with the sequence of events that occur during a coupled computation.
- Specific PVM send and receive messages are listed in the PVM Executive manual. These messages allow a variety of exchanges between the PVM Executive and the coupled code, as well as directly between the coupled codes. These messages must be programmed as calls to PVM library procedures in every coupled program to enable it to participate in the coupled calculation.
- Verification testing has demonstrated that the PVM executive program performs as designed and that it is capable of initiating a coupled computation, controlling the coupled computation including recognizing and correcting faults in the coupled computation and termination of the computation when it is finished. The operation of the executive program was demonstrated using the RELAP5-3D computer program but the PVM executive program is general enough to be used to couple any number of simulation codes.
- RELAP-7 is being developed as a new generation RELAP series code aiming at enabling new safety analysis paradigm. The goals of RELAP-7 include maintaining RELAP5 capacities, addressing long existing legacy issues in RELAP5-3D, and extend its capabilities.

The RELAP5-3D code is a nuclear industry standard and provides the necessary dynamic coupling technique to connect to other component and subsystem simulations. Coupling of RELAP5-3D to external component and subsystem simulations will allow adoption of standard simulation platforms used by researchers outside the nuclear field, recognizing that nuclear hybrid energy systems represent truly cross-cutting system architectures that bring together researchers from a wide range of industries and experience. Follow-on work will use the current NHES simulation to assess a variety of system architectures to identify key system design criteria and constraints. As the architectures of interest for further study are identified, enhancement of the system simulation through the adoption of RELAP5-3D for the reactor subsystem will be pursued to ensure confidence of potential industry partners in the simulation results as plans are developed for design and build-out of a nuclear hybrid demonstration plant.

8. BIBLIOGRAPHY

1. http://www.inl.gov/relap5
2. RELAP5-3D, 2012, “RELAP5-3D Code Manuals, Volume I,” Idaho National Laboratory, INEEL-EXT-98-00834, Revision 4.0.
3. P. J. Turinsky et al., “NESTLE: A Few-Group Neutron Diffusion Equation Solver Utilizing the Nodal Expansion Method for Eigenvalue, Adjoint, Fixed-Source Steady-State and Transient Problems,” EGG-NRE-11406, Idaho National Engineering Laboratory, June 1994.
4. Weaver, W. L., Tomlinson, E. T., Aumiller, D. L., 2002, “A PVM Executive Program for Use with RELAP5-3D,” *Proceedings of Conference*, pp. 1057-1061, ICONE10-22678, Arlington, VA, 14-18 April, 2002, American Society of Mechanical Engineers
5. RELAP5-3D Wikipedia page. http://en.wikipedia.org/wiki/RELAP5-3D
6. Bayless, P. D., “RELAP5-3D Code Manual Volume III: Developmental Assessment,” pp 4-50 to 4-68 INEEL-EXT-98-00834, Revision 2.4, December 2011. http://www.inl.gov/relap5/r5manuals/ver_2_4/vol3_v2_4.pdf.
7. Bayless, P. D., “RELAP5-3D Code Manual Volume III: Developmental Assessment,” pp 4-73 to 4-81 INEEL-EXT-98-00834, Revision 2.4, December 2011. http://www.inl.gov/relap5/r5manuals/ver_2_4/vol3_v2_4.pdf
8. Bayless, P. D., “RELAP5-3D Code Manual Volume III: Developmental Assessment,” pp 3–54 to 3–66, INEEL-EXT-98-00834, Revision 2.4, December 2011. http://www.inl.gov/relap5/r5manuals/ver_2_4/vol3_v2_4.pdf
9. Bayless, P. D., “RELAP5-3D Code Manual Volume III: Developmental Assessment,” pp 5-85 to 5-138 INEEL-EXT-98-00834, Revision 2.4, December 2011. http://www.inl.gov/relap5/r5manuals/ver_2_4/vol3_v2_4.pdf
10. V. H. Ransom and V. A. Mousseau, “Convergence and Accuracy Expectations for Two-Phase Flow Simulations,” *Canadian Nuclear Society International Conference on Simulation Methods in Nuclear Engineering*, Montreal, Canada, April 18–20, 1990.
11. V. H. Ransom and V. A. Mousseau, “Convergence and Accuracy of the RELAP5 Two-Phase Flow Model,” *International ANS Topical Meeting on the Advances in Mathematics, Computation, and Reactor Physics*, Pittsburgh, PA, April 28–May 2, 1991.
12. Geist, A., et al, *PVM: Parallel Virtual Machine - A Users' Guide and Tutorial for Networked Parallel Computing*, The MIT Press, 55 Hayward Street, Cambridge, MA 02142.
13. RELAP5-3D, 2012, “RELAP5-3D Code Manuals,” Volume II Appendices A, B, and C, Idaho National Laboratory, INEEL-EXT-98-00834, Revision 4.0.
14. G. L. Mesina, “RELAP5-3D in Fortran 90, “RELAP5 International Users Seminar (RIUS), Idaho Falls, ID, Nov 18–20, 2008.
15. G. L. Mesina, “Fortran 95 Project Summary,” RELAP5 International Users Seminar, Park City, UT, Aug 10–13, 2009.
16. Mesina, G.L., Hykes, J., M., and Guillen, D. P., “Streamlining of the RELAP5-3D Code,” *Proceedings of The 12th International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH-12)*, Sheraton Station Square, Pittsburgh, Pennsylvania, U.S.A. September 30–October 4, 2007.

17. D. P. Guillen, G. L. Mesina, J. M. Hykes, “Restructuring RELAP5-3D for Next Generation Nuclear Plant Analysis,” *2006 Transactions of the American Nuclear Society*, Vol. 94, June 2006

18. G. L. Mesina and P. P. Cebull, “Extreme Vectorization in RELAP5-3D,” *Proceedings of the Cray User Group 2004*, Knoxville, TN, USA, May 16–21, 2004.

19. G. L. Mesina, “Architectural Advancements in RELAP5-3D,” Proceedings of the 2005 ANS Winter Meeting, November 12–17, 2005.

20. R. M. Cumberland and G. L. Mesina, “Improvements to the Time Stepping Algorithm of RELAP5-3D,” *The Journal of Undegraduate Research of the DOE Office of Science*, Vol 9, Sep, 2009.

21. G. L. Mesina, “Visualization of RELAP5-3D Best Estimate Code,” Best Estimate Seminar 2004, Washington, D.C, Nov. 14–18, 2004.

22. Vaghetto, R., Hassan, Y. A., “Texas A&M RCCS Experimental FacilityRELAP5-3D Simulations,” *Proc. Of the 2011 RELAP5 International User Seminar*, July 26-28, 2011 Salt Lake City, UT, http://www.inl.gov/relap5/rius/saltlakecity2011/rius_2011_vaghetto_tamu_rccs_sim.pdf.

23. A. Epiney, C. Rabiti, Y. Wang, J. Cogliati, T. Grimmett, P. Palmiotti, “INSTANT/PHISICS — RELAP5 coupling,” *Proc. Of the 2011 RELAP5 International User Seminar*, July 26-28, 2011 Salt Lake City, UT, http://www.inl.gov/relap5/rius/saltlakecity2011/irug_relap-phisics_cr.pptx.

24. C. Rabiti, “PHISICS Reactor Physics Package in RELAP5-3D,” RELAP5-3D Newsletter 2nd Quarter 2012, Idaho National Laboratory, http://www.inl.gov/relap5/news/2012_2q_r5_3d_phisics_in_r5.pdf.

25. D. Andrs, et al., “RELAP-7 Level 2 Milestone Report: Demonstration of a Steady State Single Phase PWR Simulation with RELAP-7,” INL/EXT-12-25924, May, 2012.

26. D. Gaston, et. Al., “MOOSE: A Parallel Computational Framework for Coupled Systems of Nonlinear Equation”, *Nucl. Eng. Design*, 239 (1768-1778), 2009.