Journal of Computational Physics 326 (2016) 544–568

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

The Virtual Environment for Reactor Applications (VERA): Design and architecture ☆

John A. Turner [a,*], Kevin Clarno [a], Matt Sieger [a], Roscoe Bartlett [a], Benjamin Collins [a], Roger Pawlowski [b], Rodney Schmidt [b], Randall Summers [b]

[a] Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
[b] Sandia National Laboratories, Albuquerque, NM 87185, USA

ARTICLE INFO

Article history:
Received 18 December 2015
Received in revised form 15 August 2016
Accepted 2 September 2016
Available online 8 September 2016

Keywords:
Nuclear reactor
Modeling
Simulation
Coupled physics
Multiphysics
Core simulator
Neutronics
Fluid flow
Thermal-hydraulics

ABSTRACT

VERA, the Virtual Environment for Reactor Applications, is the system of physics capabilities being developed and deployed by the Consortium for Advanced Simulation of Light Water Reactors (CASL). CASL was established for the modeling and simulation of commercial nuclear reactors. VERA consists of integrating and interfacing software together with a suite of physics components adapted and/or refactored to simulate relevant physical phenomena in a coupled manner. VERA also includes the software development environment and computational infrastructure needed for these components to be effectively used. We describe the architecture of VERA from both software and numerical perspectives, along with the goals and constraints that drove major design decisions, and their implications. We explain why VERA is an environment rather than a framework or toolkit, why these distinctions are relevant (particularly for coupled physics applications), and provide an overview of results that demonstrate the use of VERA tools for a variety of challenging applications within the nuclear industry.

1. Introduction

The Consortium for Advanced Simulation of Light Water Reactors (CASL) is developing a collection of modeling and simulation (M&S) tools known as VERA, the Virtual Environment for Reactor Applications. VERA provides a software environment for the development and deployment of tools for analysis of operating light-water nuclear reactors, including:

- *Nuclear-specific input and output* to define, model, and analyze the performance of a reactor during nominal operation and anticipated operational transients;

* Corresponding author.
E-mail address: turnerja@ornl.gov (J.A. Turner).

http://dx.doi.org/10.1016/j.jcp.2016.09.003
0021-9991/

- *Nuclear-specific software* such as reactor neutronics, sub-channel thermal-hydraulics, and nuclear fuel performance, and cladding-coolant chemistry; and
- *Nuclear-specific physics integration* enabling coupling across physics domains while utilizing best-in-class algorithms for each domain to provide efficient nuclear-specific solutions.

VERA is neither a general-purpose numerical library such as Trilinos [1] or PETSc [2], a domain-specific framework such as OpenFOAM,[1] libMesh [3] or Deal.II [4], nor a monolithic application like those traditionally used in the nuclear industry such as COBRA-TF, RELAP5, or MELCOR. Instead, VERA combines capabilities such as these to enable development of nuclear-specific solutions, and as such provides an extensible platform on which nuclear reactor analysis applications can be built for decades to come.

VERA comprises four elements: 1) physics components, 2) numerical tools for solving coupled-physics problems, 3) drivers that execute the individual and coupled components, and 4) the infrastructure to develop high-quality software in a collaborative environment. In Section 2, the primary use cases and physics requirements are described. Section 3 describes constraints, defines terminology such as "software environment", outlines the coupling strategy and physics components in VERA, as well as the general infrastructure required for collaboratively developing high-quality software from diverse, distributed institutions. This infrastructure includes a build/test system and strategy for input/output/restart. Finally, Section 4 provides specific application examples and Section 5 describes future directions for VERA.

Using VERA, CASL has developed a set of coupled applications for analysis of challenge problems of particular relevance to its industry partners, and these tools have been deployed and are being used for real-world applications [5–11].

2. Requirements

2.1. Challenge problems

A primary goal of CASL is to develop an understanding of several key "Challenge Problems" for the nuclear industry. The CASL Challenge Problems are described in detail in [12], and each requires a specific combination of physics, spatial and temporal resolution, and level of coupling/feedback between components. For example, the Pellet–Clad Interaction (PCI) Challenge Problem requires two tools: a scoping tool and a high-fidelity tool. The scoping tool is used to model every fuel pin in the reactor using lower fidelity two-dimensional models during an operating cycle and determine the operational procedures that would simultaneously maximize power generation and minimize the risk of fuel cladding failure. The high-fidelity tool includes a three-dimensional nuclear fuel performance code that can predict the detailed behavior of fuel to determine the likelihood and consequences of a fuel failure due to manufacturing defects or operational transients. The high fidelity tool is applied to fuel pins identified by the scoping tool as possible PCI failure candidates.

In each of the primary physics components, two levels of fidelity are required to solve the various challenge problems, but not all levels of fidelity need to be coupled. The following section discusses each of the physics components and their level of fidelity with respect to traditional nuclear industry tools. The VERA core simulator, referred to as VERA-CS, integrates each of the lower-fidelity components to provide a high-resolution (with respect to traditional tools) prediction of the operating history of the entire lifetime of a nuclear reactor over many fuel-loading cycles. This provides the initial and boundary conditions for many of the Challenge Problem high-fidelity analyses.

2.2. Physics components

Simulation of commercial light water reactors (LWRs) requires modeling capabilities for neutronics (generation and transport of neutrons through the reactor core), fuel performance (thermomechanical response of the fuel rods to generated fission energy, fission products, and material property evolution), and thermal-hydraulics (the removal of the energy generated within the core by the flow of water coolant). In addition, models for specific chemical and mechanical phenomena and the capability to represent a variety of geometric features are important for simulation of specific Challenge Problems being addressed by CASL.

Traditional approaches to simulating these phenomena have focused on modeling the relevant physics individually with separate codes, treating coupled phenomena with simplified models or as boundary conditions, and averaging the results over phase space (e.g. space, time and/or neutron energy) to simplify the coupled problem. A major objective of CASL is to achieve tighter integration and coupling of these phenomena within a unified multiphysics environment, with rigorous numerical approaches incorporating key feedback effects.

Table 1 lists in more detail the general needs for each of the relevant phenomena, including foundational physics, specific capabilities and quantities of interest needed to simulate CASL Challenge Problems, and the effects and variables important for coupling with the other phenomena areas. The subsections below discuss the needs for each of these phenomena.

[1] http://openfoam.org/.

Table 1
General needs for simulation of commercial pressurized water reactors (PWRs) [13].

Phenomena	Foundational physics needs	Challenge problem needs	Feedback effects
Neutronics	• Generation of effective neutron cross-sections • Local neutron flux and heat generation • Local isotopic changes due to irradiation and radioactive decay	• Depletion tracking • Discrete grid modeling • Azimuthal fuel rod temperature distributions • Reactor structure doses • Local boron concentration	• Local coolant state • Local fuel temperature • Rod bow • Assembly bow • CRUD deposition
Thermal-hydraulics	• Local axial and cross flow • Local coolant state (pressure, temperature, and phase) • Coolant mixing/turbulence • Inlet coolant mass flow distribution • Heat transfer coefficients	• Analytical departure from nucleate boiling (DNB) predictions and DNB ratio margin • Fluid–structure interaction excitation loads • Pressure drop across components	• Local fuel temperature • CRUD deposition
Fuel performance	• Local fuel temperature • Fuel densification & swelling • Fission gas release • Fuel crack/relocation • Cladding oxidation, hydriding • Clad thermal/irradiation creep • Rod internal pressure • Cladding stress/strain • Pellet/cladding interactions	• Missing pellet surfaces and localized stress/strain • Component and assembly vibration • Grid-to-rod fretting wear • Cladding plasticity	• Local rod power • Local irradiation history • Local coolant heat transfer coefficients • CRUD deposition
Chemical, mechanical, geometric		• Coolant chemistry and CRUD deposition • Clad-grid support gaps • Reactor CRUD inventory • Assembly & rod distortion • Structure oxidation & hydriding • Structure/Joint/nozzle stress & strain	• Local boron concentration • Structure and rod irradiation growth • Rod waterside diameter • Fluid–structure interactions

2.2.1. Neutronics

The heat source in a nuclear reactor is primarily from the fission of actinides in nuclear fuel. The neutrons generated from fission scatter off materials and are captured, leak into the surrounding environment, or cause another fission, causing the actinide to split into fission products. Traditional neutronics uses a multi-step approach to model a local region of the core in two-dimensions with approximate boundary conditions, nominal values from the other physics, and higher neutron energy resolution. The geometric resolution and neutron energy distribution are averaged to produce 'effective' cross sections as a function of the other physics inputs. To provide a full-core solution, the tabulated average cross section of the smeared materials and coarsened neutron energy are used to model the diffusion of neutrons throughout the core. Therefore, an estimate of the fuel pin-resolved solution requires a reconstruction from the local and global solutions.

VERA includes three neutronics solvers: Insilico [14,8], MPACT [15] and Shift [16,17], described in Sections 3.4.2, 3.4.3, and 3.4.4, respectively. Briefly, Insilico and MPACT provide deterministic pin-homogenized and pin-resolved transport capabilities, respectively, each using a multi-group discretization of neutron energy. Approximations inherent in the generation of effective neutron cross sections bound the accuracy of both deterministic solvers. Shift is a continuous-energy Monte Carlo code, which solves an identical neutronics problem without approximations in geometry or neutron energy. Shift provides the ability to quantify the error associated with the deterministic solver and evaluate the radiation distribution throughout the nuclear power plant, rather than just the core. This includes estimating the ex-core detector response, fluence on the reactor vessel and internals, control rod lifetime, in-core instrument response, and fluence-based irradiation assembly bow.

Ultimately, neutronics codes supply the power generation and isotopic depletion and fission product creation as a function of space and time as localized source terms in the solution of the thermomechanics equations for the fuel rods, described in Section 3.4.5. Codes modeling fuel thermomechanics must in turn supply fuel temperatures to the neutronics codes for accurately determining neutron cross-sections in calculating neutron fluxes.

2.2.2. Thermal/hydraulics

Fundamental to simulating reactor behavior is the modeling of the thermal/hydraulic behavior of the coolant. Traditionally, subchannel methods [18] have been used due to their computational efficiency, with CASL electing to utilize COBRA-TF (CTF) [19], which is discussed in more detail in Section 3.4.1. These methods use simplified momentum equations for cross-flow between channels and a variety of models for two-phase (liquid plus vapor) fluid flow. At the opposite end of the spectrum are three-dimensional CFD codes that can more accurately model boundary layer and multidimensional flow effects, including turbulence and mixing, but which are highly computationally intensive and do not yet adequately incorporate multiphase phenomena over all flow regimes. CASL has supported development of the Hydra-TH [20] and Drekar [21] codes and also utilized the commercial STAR-CCM+ code. The range of CASL applications requires both subchannel and CFD approaches.

In addition to coupling across physics areas, an important objective within CASL is to improve the modeling fidelity and resolution for individual physical phenomena. To the extent that variations of fluid conditions and temperatures within a typical subchannel volume are important for certain CASL Challenge Problems, CFD methods must be used, but it is currently not feasible to apply these tools on a core-wide basis. Full core simulations with additional models for chemistry must continue to rely on subchannel methods, which can incorporate information from more localized CFD simulations, such as variations in heat transfer coefficients around the periphery of a fuel rod or in the vicinity of a grid spacer, to improve fidelity at higher resolution.

Thermal/hydraulic behavior of the coolant must be coupled with the thermomechanical behavior of fuel rods, which are typically treated by separate fuel performance codes as described below, thus requiring conjugate heat transfer capabilities. The effects of fluid density and temperature on moderator cross-sections are significant in neutronics simulations, and thus must also be coupled to neutronics codes.

2.2.3. Fuel rod heat transfer and mechanics

Nuclear fission and fission product decay produces heat within the ceramic fuel pellets of a rod; the heat is conducted radially through the fuel, across a helium-filled gap, through the metal cladding, and into the axially-flowing coolant. The fuel and cladding can expand or contract due to thermal and radiation effects, which changes gap thickness and thus fuel temperature. When the actinides fission, some gaseous fission products are produced that build up within the fuel; when the fuel is hot enough some will diffuse out of the pellet and into the gap. This changes the pressure and gas composition in the gap, which can reduce conductivity and increases fuel temperature. In addition the ceramic fuel pellet will densify and then swell due to competing effects from annealing and fission-product swelling. Because the effective neutron cross sections depend strongly on the fuel temperature, there is a strong feedback between the power distribution within the fuel and its temperature. Traditional fuel modeling within the nuclear industry consists of two levels of fidelity: an 'effective' fuel model within the smeared-material core simulator to approximate heat transfer within all of the pins in an assembly in an average sense and a two-dimensional fuel performance code that comprehends the effects described above.

Similarly, VERA includes two levels of fidelity for modeling approximate heat transfer in a core simulator and high-fidelity fuel performance. Bison [22] is a parallel, finite-element fuel performance code that includes all of the physics of a traditional fuel performance code, and is capable of executing in both 2D (r–z & r–Θ) and 3D (x–y–z). Bison is used to predict high-resolution solutions for fuel performance problems that require high-fidelity predictions of intra-rod physics, such as pellet–clad mechanical interaction and cladding integrity evaluations during accident conditions. For modeling nominal operation, lower-fidelity models can be used, such as simple table interpolations of fuel temperature based on the local power and burnup or simple fuel models in COBRA-TF (CTF) [19] that uses heat conduction in each quadrant and axial level of every fuel pin in the core to predict fuel temperature for core physics analyses.

Historically, fuel degradation behavior has been modeled using semi-empirically based constitutive relationships. CASL is conducting fundamental, multiscale investigations that will ultimately lead to improved material property models and constitutive relationships, with improved physical fidelity to enable engineering scale fuel performance predictions. Although the focus has been on normal operation of LWR fuel, once these advanced models have been validated, they can be used to assess the behavior of current and advanced fuels during accident conditions, forming the basis of incorporating models for additional degradation phenomena, such as cladding ballooning.

2.2.4. Chemical, mechanical, and geometric effects

Several of the Challenge Problems selected by CASL require additional models for important chemical and mechanical effects and geometric components beyond the foundational physics addressed by thermal/hydraulics, neutronics, or fuel thermomechanics codes. These specialized models (or codes) have been embedded within either the fuel performance or thermal-hydraulics codes. The Challenge Problems for CRUD[2]-induced power shift (CIPS) and CRUD-induced localized corrosion (CILC) have been addressed through the development of multi-scale clad-coolant chemistry models, which are incorporated in the MAMBA code [23], described in Section 3.4.7, for predicting corrosion and CRUD deposition and erosion. The grid-to-rod fretting (GTRF) challenge problem requires treatment of irradiation growth and creep of fuel rods and spacer grids and of fluid–structure and structure–structure interactions, including wear rates and integration within the fuel performance code.

2.2.5. Core simulator

Because of the high resolution required to address each of the Challenge Problems, approximate initial and boundary conditions cannot be defined with traditional core simulators that rely on assembly-averaged flow profiles and the reconstruction of individual pin powers from lattice and core solutions. Therefore, each of the "low fidelity" capabilities discussed in this subsection have been integrated within VERA to provide an efficient means of predicting, with sufficient resolution, the nominal operating history over many cycles of a nuclear power plant.

The VERA Core Simulator (VERA-CS) provides a pin-resolved, quasi-steady-state prediction of the state of the LWR (temperatures, densities, power, and isotopics) and encompasses the foundational physics described in Sections 2.2.1

[2] CRUD is an acronym for "Corrosion Related Unidentified Deposits".

Table 2
Comparison of VERA-CS with typical industry core simulator methods [13].

Physics area	Typical industry core simulator method	VERA core simulator (VERA-CS)	VERA highest fidelity capability
Neutronics	3-D diffusion (core) 2 energy groups (core) 2-D transport on single assemblies	2D/1D transport 50+ energy groups	3D transport Continuous-energy Monte Carlo
Thermal-hydraulics	Nodal average (1-D)	Subchannel (w/crossflow)	CFD
Fuel performance	Bounding empirically-based	Pin-by-pin (r, z) heat transfer	3D fuel performance
Power distribution	Nodal average with pin-power reconstruction	Explicit pin-by-pin	Explicit pin-by-pin
Target platforms	Workstation (six-core)	<4,000 cores	10,000–300,000 cores
Performance	~110 GF/s[a]	~18 TF/s	~180 TF/s–20 PF/s
Memory	~16 GB	~3 TB	~30 TB–700 TB

[a] Flop/s = floating point operation (flop) per second = (no. of cores) × (clock rate) × (flop/s/cycle).

through 2.2.4. Table 2 provides a comparison of the typical industry core simulator with both VERA-CS and the high-fidelity capability of VERA (exemplified by the Hydra-TH CFD capability described in Section 3.4.6 and the Shift Monte Carlo neutronics capability described in Section 3.4.4).

3. VERA architecture

3.1. Constraints

Due to CASL's nature as a collaborative, multi-institutional program, a number of constraints, both programmatic and technical, influenced the design and implementation of VERA. One of the most significant conditions was the need to leverage existing software. This included codes in broad use throughout the nuclear energy community and industry. These codes are often referred to, somewhat pejoratively, as "legacy" software, and indeed they often lack some aspects of what is now considered standard practice for quality software such as unit testing, robust memory management, and well-defined interfaces. However, they have been extensively validated for problems of interest, sometimes over decades. And it should be noted that in some cases documentation for these codes is superior to more modern codes. In addition, CASL was also working with software (both modern and legacy) that is under active development with funding from other DOE and NRC programs. While CASL benefited from the associated advances, it was also constrained in the substance of changes that could be requested because of competing requirements. CASL also leverages advanced simulation tools that have been developed more recently, for example solver libraries such as PETSc [2] and Trilinos [1], as well as application codes using these libraries.

As a result VERA includes software components that are developed with multiple programming languages (primarily Fortran and C++), user interfaces, build systems (make,[3] Autotools[4] and CMake[5]), and quality assurance practices. More significantly, many of these components were designed to operate in a stand-alone mode or as a driver code, but many VERA use-cases require them to serve as a component in a larger system and be compiled as a library with an API. This sometimes required refactoring to provide separation between initialization and execution phases as well as definition and implementation of interfaces.

3.2. VERA is a software application environment

To better characterize VERA, we provide background on nomenclature related to methods for integrating software components, including "toolkit", "framework", and "environment". Note that these terms are not strictly defined in the software community. The following represents our understanding of these terms based on our experience with scientific and engineering software development efforts.

A toolkit is a collection of software components that can be used individually or combined to perform useful tasks. An ideal toolkit would share a common design philosophy, key features, build system, and even documentation style, standards, and format, but for the most part individual components of a toolkit provide functionality that is usable without requiring a major portion of other components in the toolkit. Toolkits are also designed such that their components can be combined with other software, including components from other toolkits, to create larger toolkits and applications. Common examples of toolkits (also known as libraries), include those for GUI development (Qt,[6] Java's Swing[7] and JavaFX,[8] Windows

[3] https://en.wikipedia.org/w/index.php?title=Make_(software).
[4] https://en.wikipedia.org/w/index.php?title=GNU_build_system.
[5] https://en.wikipedia.org/w/index.php?title=CMake.
[6] https://en.wikipedia.org/w/index.php?title=Qt_(software).
[7] https://en.wikipedia.org/w/index.php?title=Swing_(Java).
[8] https://en.wikipedia.org/w/index.php?title=JavaFX.

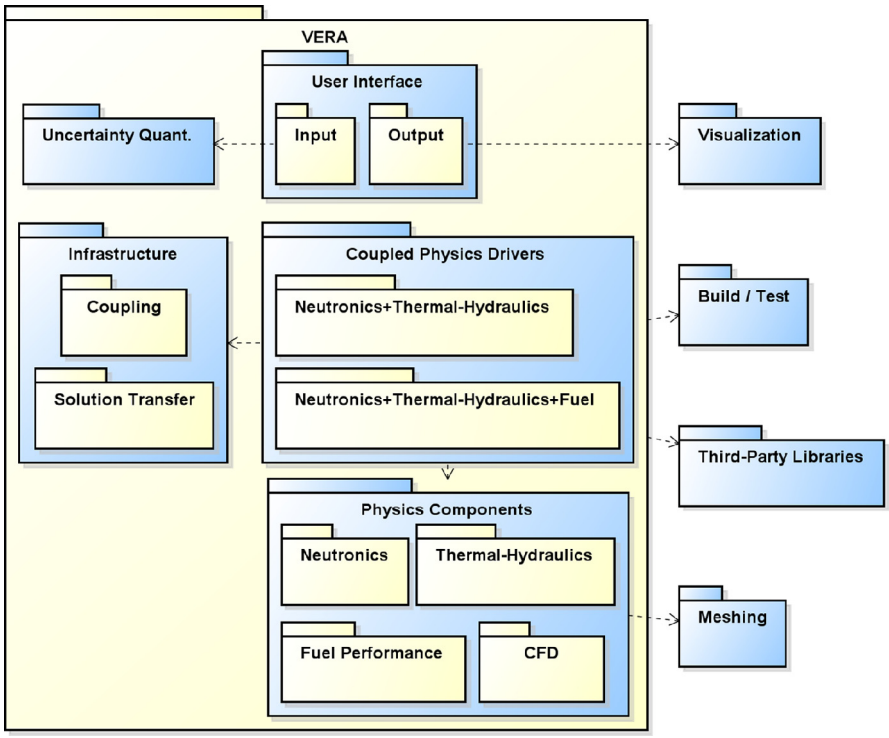


Fig. 1. VERA conceptual/functional package diagram.

Presentation Foundation (WPF),[9] and wxWidgets[10]) and numerical solvers (BLAS [24], LAPACK,[11] PETSc [2], Hypre [25], and Trilinos [1]).

A framework is a more tightly-integrated toolkit. In general, frameworks impose more formality on aspects such as parallel data structures, communication patterns, and programming models, but in return provide more built-in functionality for application developers. A well-designed framework provides a well-defined architecture that ensures components work together coherently. Frameworks can be extremely successful for creating solutions in specific domains for which they are designed. However, applying a framework to an application space for which it was not intended can be extremely difficult, even if source code is available. Examples of frameworks include libMesh [3], deal.II [4], Sierra [26], MOOSE [27], AMP [28], and FEniCS [29].

Note that the term “framework” refers to a spectrum of software systems, with terms such as “lightweight” (“small f frameworks”) and “heavyweight” (“large F Frameworks”) attempting to describe opposite ends of the spectrum. In general, lightweight frameworks impose fewer restrictions on components (and provide correspondingly fewer services) than large “F” Frameworks. Neither is superior in a general sense – both provide tradeoffs in performance, developer effort, etc. that must be weighed for each intended application. However, experience has shown that the most successful frameworks tend to focus on a well-defined domain. Frameworks that become overly-general often suffer from performance challenges, software complexity (leading to fragility, portability and maintenance issues), and other difficulties.

One stark contrast to toolkits is that frameworks are often designed to be the top-level, master application. As a result, they can be difficult to incorporate into larger systems or applications. This is not inherent in the concept of a framework, but is nevertheless a commonly-encountered design weakness.

We use the term “Environment” to refer to a software system that combines the best aspects of both toolkits and frameworks. A well-designed environment provides a collection of functionality, potentially combining multiple toolkits and/or frameworks, within a coherent, integrated system that ensures proper behavior of elements and allows composition of components to create applications as appropriate. Environments are often specific to a particular domain, but general enough to allow creation of multiple applications relevant to that domain. An environment could be considered a generalization of the toolkit concept to full-scale applications, and shares many attributes of lightweight frameworks.

In recent years, a number of excellent surveys of multiphysics approaches and frameworks have been published [30–32]. These, as well as the experience of CASL developers, influenced the strategy for VERA.

[9] https://en.wikipedia.org/w/index.php?title=Windows_Presentation_Foundation.
[10] https://en.wikipedia.org/w/index.php?title=WxWidgets.
[11] http://www.netlib.org/lapack/.

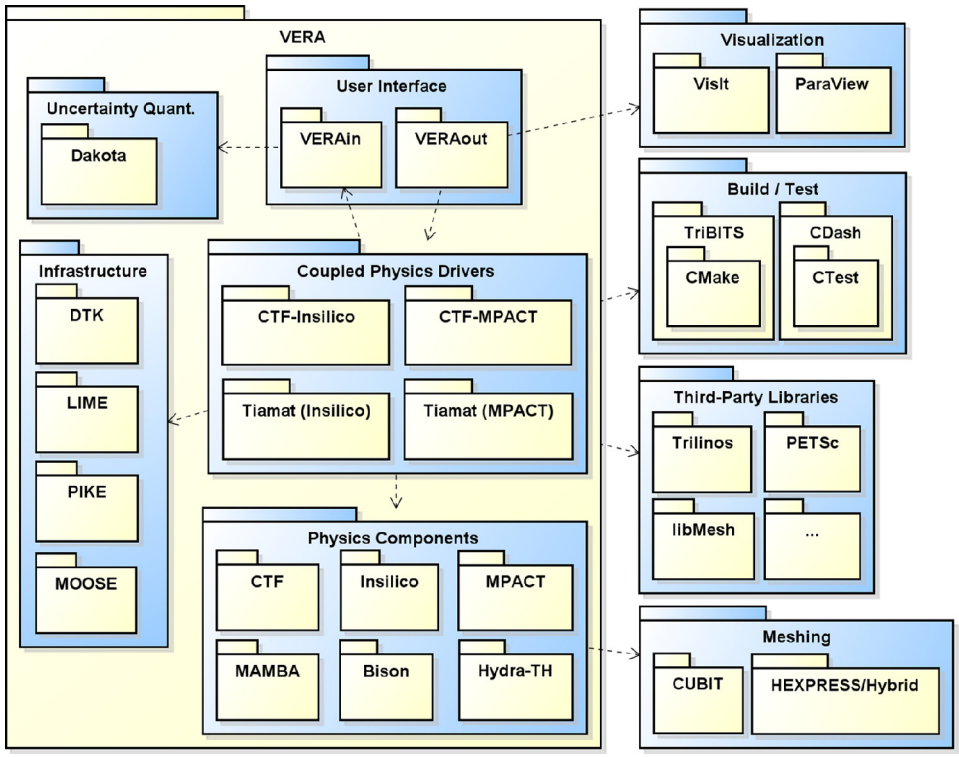


Fig. 2. Conceptual architecture of VERA showing component packages.

One of the first CASL design decisions was that VERA would take the "environment" approach. A high-level view of the architecture of VERA is shown in Fig. 1. The primary advantage to this approach is that in general barriers to integrating existing physics components would be small, since internal data structures should not be affected. However, the effort to integrate components not designed to be composed into larger systems could be significant. In addition, a mechanism for mapping solution data between physics components would be required, and the potential overhead in communicating data between physics components (copying data into and out of data structures) could be significant. Our approach to solution transfer, DataTransferKit (DTK) [33], is described below in Section 3.3.2 as well as in an accompanying article in this issue [34].

3.3. Coupling approach

Because the Challenge Problems required multiple means of coupling, VERA was designed to allow for the development of multiple "driver codes" to solve particular problems. Some coupled-physics problems are (1) well-defined with (2) well-understood numerical methods and (3) leverage similarly-designed software. For these systems, it is appropriate to have a single, problem-specific "driver" code for that particular multi-physics problem. VERA contains a fully-integrated core simulator (VERA-CS, described in Section 4.1) and fuel performance code (Bison, described in Section 3.4.5) that leverage internal mathematical solvers and mechanisms for transferring data between components/physics domains, which categorizes them as a stand-alone Framework for that particular problem. When any of these three constraints do not exist, however, a more light-weight coupling enables the exploration of advanced methods, integration of additional physics components, and simplification of software that was not designed to work together (Fig. 2).

Therefore, CASL developed general-purpose tools for coupling physics: a flexible non-invasive method for solving coupled non-linear systems of equations and a general tool for transferring data between physics components. These tools were used in early versions of the core simulator to scope out appropriate numerical algorithms for coupling and efficient means of data transfer. The generic nature of the tools simplified the transition between neutronics components and facilitated coupling of a fuel performance component with the core simulator.

3.3.1. Numerical solver interfaces

Advanced, parallel mathematics solver libraries (e.g. PETSc, Hypre, Trilinos) contain many tools for solving a set of steady-state or transient, linear or non-linear, systems of equations. These libraries contain advanced solvers that can be used by a driver code to couple systems of equations, but each solver has different requirements and constraints. For instance, some require knowledge of the Jacobian, ability to compute a residual, efficient full-system preconditioner, or an assumption that

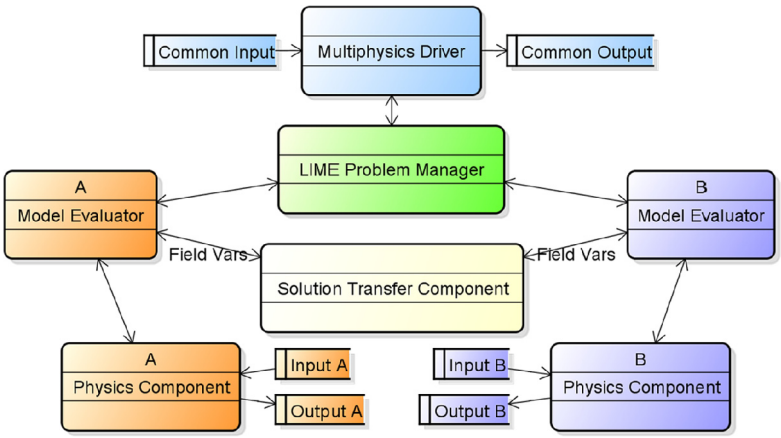


Fig. 3. LIME-based application.

the residual computation is substantially less expensive than solving the problem. When working with existing tools, often these assumptions don't hold or the work to refactor the codes to satisfy these requirements is prohibitively expensive in a viable time frame. Therefore, an existing lightweight environment for multiphysics coupling (LIME) was leveraged and was later redesigned and integrated into Trilinos (PIKE). LIME and PIKE are described briefly in the following sections.

These light-weight frameworks enable each physics component to specify what it can provide (time-dependence, state-point solution, residual, Jacobian, etc.) and what it requires from other physics components to abstract differences between the execution modes and allow for a variety of coupling methods from fixed-point (Picard) iteration to Jacobian-free Newton–Krylov (JFNK) [35]. The interfaces defined here include abstractions for data transfers between codes, allowing for separation of concerns between coupled solution algorithms and the data transfer mechanism.

3.3.1.1. LIME The Lightweight Integrating Multiphysics Environment (LIME [36,37]) was well-suited to the requirements of VERA, and was used as the initial basis for non-invasive (e.g. black-box) coupling of capabilities in VERA. It was specifically designed for integration of existing components that solve different phenomena of a multiphysics problem. LIME provides the key high-level software (written in C++) and interface requirements to enable the assembly of multiple physics codes into a single-executable coupled code multiphysics simulation capability. In order to create a new multiphysics application using LIME, physics codes are required to support a minimal software interface. More invasive coupling requires more interface support for the additional data dependencies (e.g. residuals and Jacobians for JFNK couplings). Depending on the structure and design of the original physics codes, some degree of code refactoring may be required. Fig. 3 illustrates key components of a generic LIME-based coupled-code multiphysics application created from two individual physics codes. Early coupled drivers in CASL were based on LIME [38], but were eventually migrated to use the PIKE package.

3.3.1.2. PIKE PIKE (Physics Integration KErnels) is a newly-released open-source software library for coupling and solving multiphysics applications within Trilinos. An early CASL report [39] outlined plans to address some of the drawbacks and limitations of LIME. That report, as well as other open-source multiphysics coupling libraries such as AMP [28] and MOOSE [27], and especially lessons learned developing coupled applications using LIME, heavily influenced the design and implementation of PIKE. PIKE provides basic interfaces and utilities for performing code-to-code coupling, for simple "black-box" Picard iteration methods for solving a coupled system of equations, including Jacobi and Gauss–Seidel solvers. For more invasive couplings (JFNK) and acceleration techniques such as Anderson acceleration [40], PIKE can be used in conjunction with the Thyra abstraction layers and the NOX non-linear solver packages in Trilinos. See the article in this special issue by Hamilton et al. for detailed comparisons of solution techniques [41]. The principal differentiating features in PIKE are simplified interfaces, minimal upstream dependences, no system-level framework requirements, and an emphasis on user extensibility. Currently, PIKE only relies on the Trilinos tools package Teuchos[12] for memory management, parameter handling, and error handling utilities. One aspect of the code coupling that impacted both PIKE and DTK is that a large number of the codes were designed for parallel high performance computing (HPC) architectures. Each HPC-enabled code used its own unique decomposition method. When coupling both serial and parallel codes, this presented challenges. For example, memory requirements in certain codes were so large that it prohibited instantiating multiple applications on the same node. Therefore a set of tools in PIKE are used to distribute applications across the MPI execution space in both overlapping and non-overlapping modes as appropriate to the coupling.

[12] https://trilinos.org/packages/teuchos/.

3.3.2. Data transfer kit

To facilitate solution transfer between physics components, the DataTransferKit (DTK) [33] was developed. Many software frameworks (MOAB [42], MOOSE [27], Sierra ToolKit [43]) leverage defined data structures, mesh databases, and interpolation libraries which provide general purpose tools for transferring data between physics components. Because VERA allows the use of independently-developed software components, some of which operate on geometrically-defined regions rather than a traditional mesh, it cannot impose strict requirements on defined tools that enable data transfer.

DTK provides multiple algorithms for transferring a solution from one spatially-decomposed physics components to another without requiring the physics component to have any knowledge of the other physics component, its domain decomposition on a parallel architecture, or underlying data structures. DTK can provide both overlapping volume domain transfers and non-conforming surface transfers. Algorithms include basis interpolation, weighted residual common refinement, and mesh free methods including spline interpolation and moving least squares reconstruction. See the article in this special issue by Slattery for more information [34].

In the base case, each physics component registers field variables it can provide (power, flux, temperature, velocity, density, etc.) and the local geometry or mesh on which they will be provided. Similarly, each physics component registers which field variables it requires, along with either the physical locations in space where it needs that data for each local domain or the corresponding target geometry or mesh. Utilizing knowledge of the geometric or mesh data along with the location of the data, DTK generates an optimal MPI communication pattern to transfer field variables between components. DTK contains algorithms for accuracy and constraint enforcement control, although the coupled driver can enforce this as well.

3.4. Physics components selected for VERA

3.4.1. COBRA-TF

COBRA-TF (CTF) includes a rod-bundle subchannel thermal-hydraulics application used to perform transient simulation for the full range of two-phase flow regimes based upon the geometry inherent in current commercial nuclear power reactors [44]. The subchannel approach can be thought of as an extremely coarse-mesh approximation, in which the control volume is of a size equivalent to a single fuel rod and its surrounding coolant. It uses a two-fluid, three-field (i.e. liquid film, liquid drops, and vapor) modeling approach. Both sub-channel and 3D Cartesian forms of nine conservation equations are available for LWR modeling.

In CTF, the conservation equations for each of the three fields and for heat transfer from and within the solid structure in contact with the fluid are solved using a semi-implicit, finite-difference numerical technique on an Eulerian mesh, where time intervals are assumed to be long enough to smooth out the random fluctuations in the multiphase flow, but short enough to preserve any gross flow unsteadiness. The fluid volume is partitioned into a number of computational cells by CTF. The equations are solved using a staggered difference scheme. The phase velocities are obtained at the cell faces, while the state variables – such as pressure, density, enthalpy, and void fraction – are obtained at the cell center. The momentum equations are solved on staggered cells that are centered on the scalar mesh face.

CTF is developed for use with either 3D Cartesian or sub-channel coordinates and features flexible nodalization for both the thermal-hydraulic and the heat-transfer solution. This flexibility allows a fully 3D treatment in geometries amenable to description in a Cartesian coordinate system and the use of the sub-channel approximation for faster calculations when the flow is principally in one direction.

The application is able to handle both hot wall and normal flow regimes maps and it is capable of calculating reverse flow, counter flow, and cross-flow situations. CTF utilizes its own internal methods to calculate the thermo-physical properties of water, and includes simplified conduction models for fuel rod conduction that can be used in place of Bison when a lower-fidelity solution is acceptable.

3.4.2. Insilico

The S_N/SP_N radiation transport code, Insilico, solves the multigroup discrete ordinates (S_N) form of the Boltzmann transport equation for both fixed-source and k-eigenvalue problems. It uses the well-established S_N discretization and solves the transport equation on Cartesian grids using the Koch–Baker–Alcouffe [45] wavefront parallel algorithm. See [8] for details on Insilico, including its use in simulating hot zero power conditions for two Westinghouse nuclear reactor systems.

3.4.3. MPACT

MPACT [15] is a three-dimensional (3-D) whole core transport code capable of generating sub-pin level power distributions. This is accomplished by obtaining the integral transport solutions to the heterogeneous reactor problem in which the actual detailed geometrical configuration of fuel components such as the pellet and cladding is modeled explicitly. The cross section data needed for the neutron transport calculation are obtained directly from a multigroup microscopic cross section library similar to those used in lattice physics codes. Hence MPACT involves neither a priori homogenization nor few-group condensation for the core spatial solution.

The integral transport solution is obtained by means of the method of characteristics (MOC) that employs discrete ray tracing, which increases concurrency and hence potential parallel efficiency. Since the direct application of the 3-D MOC capability in MPACT to 3-D core configuration requires considerable amounts of memory and computing time for practical

reactor applications [46], an alternative approximate 3-D solution method was implemented in MPACT based on a 2D-1D approach which employs planar MOC solutions in the framework of the 3-D coarse mesh finite difference (CMFD) formulation. The axial coupling is resolved by one-dimensional (1-D) spherical harmonic solution and the planar and axial problems are coupled through transverse leakage terms. The use of a lower order 1-D solution in the axial direction is justified by the fact that most heterogeneity in the core occurs in the radial direction rather than the axial. However, it is possible to use one of the higher order methods in MPACT such as the 1-D discrete ordinates kernel if improved solution accuracy is desired.

3.4.4. SHIFT

Shift [16,17] is a radiation transport hybrid code, utilizing the capabilities of both deterministic and Monte Carlo codes. Shift is being developed as a massively parallel code and supports multiple parallel decompositions ranging from full domain replication, full domain decomposition, domain decomposition with overlapping regions, and multiple-set-overlapping domain (MSOD) decomposition. Shift is designed to allow multiple physics and geometric representations in the Monte Carlo transport components without code modification. Thus, for example, multigroup and continuous energy physics can be supported without changing the core transport mechanics; all of the physics-dependent operations are encapsulated in the physics abstraction. Currently, Shift supports KENO and Reactor-Tool-Kit geometry packages and simple multigroup physics. Because Shift is written in the same code base as the Insilico Denovo S_N solver, it can efficiently access the deterministic code mesh generation and solver mechanics to enable integrated hybrid capability [47]. Refer to [17] for details on Shift, including implementation, accuracy, and parallel scaling performance results up to hundreds of thousands of compute cores.

3.4.5. Bison

Bison [22] provides both 2-D r-z and 3-D unstructured-mesh capability to predict fuel rod behavior. Built on MOOSE [27], which leverages the libMesh [3] parallel finite element framework, Bison includes MOOSE-based diffusion and mechanics libraries that support:

- Statics with elasticity, plasticity with strain hardening, creep, large strains, large displacements, and smeared plus explicit cracking;
- Unsteady (transient) heat transfer including conduction, convection and radiation with time and space-dependent internal heat generation;
- 2D axisymmetric, plane strain, and plane stress representations, including contact and friction interactions between pellets and between the pellet and cladding;
- 3D statics and dynamics with contact and friction, and heat transfer; and
- Mixed dimensional coupling (via multipoint constraint equations, etc.), e.g., combined 2D and 3D numerical representations for coupled global (2D) and local effects (3D) modeling.

Bison also includes models specific to nuclear fuel performance applications, including:

- Clad stress, strain, and strain rate;
- Clad oxidation, hydrogen pickup and hydride formation;
- Pellet stress, strain, and strain rate;
- Fission gas release (transient and pseudo-steady-state);
- Pellet densification, swelling and fission product evolution;
- Pellet restructuring and high-burnup rim thickness;
- Pellet cracking and relocation;
- Thermal expansion, including pellet hour-glassing;
- Thermal and irradiation creep;
- Thermal conductivity effects due to clad oxidation and fuel microstructure evolution;
- Material strength and ductility effects due to irradiation, thermal cycling, hydriding, fission product evolution;
- Pellet–cladding gap evolution and local stress due to partial contact; and
- Pellet stack growth and fuel rod growth.

The VERA fuel rod performance subcomponent calculates, on a 3D basis, fuel rod temperature, fuel rod internal pressure, free gas volume, clad integrity and fuel rod waterside diameter. These capabilities allow simulation of power cycling, fuel conditioning and deconditioning, high burnup performance, power uprate scoping studies, and accident performance.

It is important to note that these tools are built around the known performance of existing zirconium-based clad with UO_2 fuel and predictions for other fuel types may not be accurate. Estimates for the global effects of minor modifications to the fuel or clad may be possible; for example, chromia-doped pellets may be simulated with user-supplied models for several of the pellet performance characteristics or steel-based clad may be simulated with similar user-supplied models. Materials such as silicon carbides that do not fit the system paradigm can be simulated but are likely to provide inaccurate results.

Bison is being validated against both accepted industry codes, such as Falcon [48], and against experimental data [49].

3.4.6. Hydra-TH

VERA includes a parallel computational fluid dynamics capability, Hydra-TH, which utilizes a hybrid finite-element/finite-volume incompressible/low-Mach flow Navier–Stokes equation solver [50]. All transport variables are cell-centered and treated with a conservative discretization that includes a high-resolution monotonicity-preserving advection algorithm. The spatial discretization is formally derived using a discontinuous-Galerkin framework that, in the limit, reduces to a locally-conservative finite-volume method. The high-resolution advection algorithm is designed to permit both implicit and explicit advection with the explicit advection targeted primarily at volume-tracking with interface reconstruction. The time-integration methods include backward-Euler and the neutrally-dissipative trapezoidal method.

The solution algorithm used in Hydra-TH is based on a second-order incremental projection algorithm [20]. Projection methods are the most computationally efficient solution method available for solving the time-dependent Navier–Stokes equations. The projection method permits treating the momentum equations in a coupled manner.

In order to address fluid–structure problems, Hydra-TH uses an arbitrary Lagrangian–Eulerian (ALE) formulation [51,52] and provides a mesh-deformation interface that can support multiple mesh smoothing algorithms. The added-mass terms are computed for the structural coupling and can be exported for any structural solver. For explicit coupling, Hydra-TH provides a pressure-stabilized algorithm based on Nitche's variational method that circumvents the stability limitations associated with highly flexible structures and near unity fluid/solid density ratios.

Multiple alternatives for conjugate heat transfer are available in Hydra-TH, including explicit coupling with third-party heat conduction solvers, internal coupling using the existing heat conduction solver or direct integration (with continuous meshing).

3.4.7. MAMBA

The MAMBA code simulates growth of CRUD in the radial direction at the clad-coolant interface of a fuel rod. CRUD is a growth of (primarily) nickel ferrite on the outer surface of cladding due to the presence of metals in the water where the coolant is boiling. The growth of CRUD reduces the heat transfer efficiency across the cladding and raises the temperature of the cladding and fuel. In localized regions, this can lead to corrosion in the cladding and cause CILC (CRUD-induced localized corrosion). In a Pressurized Water Reactor (PWR), which includes boric acid (a neutron poison) in the coolant, CRUD can also include significant quantities of boron. The boron can alter the power distribution and cause CIPS (CRUD-induced power shift).

The primary physics and chemistry associated with CRUD formation currently treated in MAMBA includes:

- Solving for the radial temperature drop across the CRUD layer, including localized heat sinks due to internal (chimney) boiling within the CRUD layer
- The radial growth over time as mass (nickel ferrite) deposits on the surface of the CRUD, which is governed by the time evolving coolant concentration of particulates, surface erosion due to fluid flow, the cladding temperature, and the sub-cooled nucleate boiling rate on the cladding surface
- Time evolving coolant chemistry at the CRUD surface and inside the pores of the CRUD
- Mass transport of the soluble boric acid into the interior of the CRUD due to boiling induced Darcy flow
- Diffusion of soluble boric acid inside the CRUD due to the flow induced concentration gradients within the CRUD layer
- Mass evaporation in the form of steam vapor due to internal (chimney) boiling with the CRUD layer
- Boron hideout (i.e., the precipitation of soluble boric acid into solid lithium tetraborate) within the CRUD layer

3.5. Common input/output/restart

VERA contains several physics capabilities implemented in separate codes. Each was developed with its own input format specification. One of the goals of CASL was to provide a common input file which can be used to drive all of the different physics codes and control how they are coupled [38,53]. One benefit of using a common input is that users only need to understand and be proficient with a single syntax, instead of having to understand multiple inputs for multiple physics codes. Another benefit of using a common input is by ensuring that all codes work from a single set of simulation parameters, including a common geometry description, errors due to inconsistent inputs to different codes can be greatly reduced.

The design requirements for the common input included supporting the ability to:

- Allow users to easily transfer input and output between different computer systems,
- Allow users to easily edit the file on various system architectures and operating systems, including remote computers,
- Provide a format that users can readily read and understand,
- Provide input files compatible with an approved archival format recognized by the Nuclear Regulatory Commission (ASCII,[13] PDF,[14] or TIFF[15]),

[13] https://en.wikipedia.org/w/index.php?title=ASCII.
[14] https://en.wikipedia.org/w/index.php?title=Portable_Document_Format.
[15] https://en.wikipedia.org/w/index.php?title=Tagged_Image_File_Format.

- Allow users to easily compare and "diff" input files which may have been produced on a variety of platforms and operating systems,
- Allow users to archive inputs in standard source code repositories and/or directories with read-only permissions.

Based on these requirements (and others), the common input was chosen to be implemented as a single ASCII input file, which is parsed by an input processor into an intermediate format consumable by the VERA component codes. The ASCII input file uses a free-form format based on a keyword syntax designed by engineers with broad experience with current industry core design tools, so that the input file would be easier for industry users to learn and understand. The intermediate format uses the Extensible Markup Language (XML[16]) and provides a layer of abstraction between the user-visible presentation layer and the component code input interfaces. This can be exploited by future developers to provide graphical user interfaces (GUIs), automated input generators, and other tools that can more easily work with XML rather than free-format ASCII. Until recently, development of a GUI for VERA has not been prioritized since, as noted above, current industry practice is based on ASCII input files, so retaining this practice has removed one of the barriers for early adoption of VERA.

The input file contains a modular description of the physical reactor objects, including fuel assemblies, removable poison assemblies, control rods, non-fuel structures, detectors, etc. By defining each geometric object as a separate block, the objects can be described independently of each other and rely on very little global information. The independent descriptions simplify quality assurance and allow objects defined in one fuel cycle to be re-used in subsequent cycles without worrying about input conflicts. The input file also contains a description of the reactor statepoint including power, flow, and depletion, as well as blocks specifying various simulation options, etc. Additional modules/blocks are used to define modeling options and parameters for each of the physics codes.

The VERA Common Input (VERAIn) is a Perl[17] script which converts the ASCII common input file to the intermediate XML used to drive all of the physics codes in the VERA Core Simulator (VERA-CS). VERA component codes have the option of implementing support for the VERA XML input format directly or providing a preprocessor that can convert the XML into native input (Fig. 4).

In order to translate the user input to input needed for the individual components, a multistep process is used. First, the VERAIn input parser reads the text input file and converts it into an XML file. Some physics components, such as Insilico and MPACT, can read the XML file directly using an in-memory database from readily-available libraries (i.e. the Trilinos Teuchos::ParameterList class). Other components, such as CTF and Bison, require an intermediate step that converts the XML file into the native code input. This process allows the common input file to be used for existing physics codes where making extensive modifications to the native input parser is impractical.

There is one class of input that cannot be automatically generated by the VERA common input. Some physics codes, such as CFD (Hydra-TH), require an unstructured tetrahedral, hexagonal, or polygonal mesh that is usually generated from a CAD file. For these codes, it is necessary for the user to attach an externally generated mesh file and ensure that the mesh file is consistent with the geometry defined in the common input.

VERA common output is a binary HDF5 (hierarchical data format)[18] file commonly used for scientific data storage and I/O. An advantage of the HDF5 format is availability of software tools such as HDFView[19] to view and modify the data, and the HDF5 file is accessible by novice developers and engineers through many software languages such as Fortran, C/C++, Java, MatLab, etc. A standard output format has been defined for all components, and several Fortran and C++ utilities have been developed to generate core analysis edits such as axial power distribution or radial peaking factors, compare multiple files, and perform statistical analyses associated with core design and method validation and benchmarking. In addition, a utility has also been developed to convert the HDF5 file data into a SILO file format suitable for visualization tools such as VisIt [54] or ParaView.[20]

3.6. Unified build/test system

An often undervalued aspect of a large integrated software system such as VERA is the value of a coherent, unified approach to building applications for various platforms and support for extensive automated testing. After considering existing options, CASL developed the Tribal Build, Integrate, and Test System (TriBITS) [55,56], a set of macros and functions that extends the current de facto standard build system, CMake. Initially a part of Trilinos, TriBITS is now available standalone in order to facilitate more widespread use.

These extensions improve support for large software systems comprising multiple packages and third-party libraries, facilitate management of distributed development, and provide a flexible automated testing framework (Fig. 5).

TriBITS enables CASL developers to work in a range of development and sync models. The primary repository for CASL development and testing of each repository exists at ORNL behind the laboratory firewall. A single checkin test script is

[16] https://en.wikipedia.org/wiki/XML.
[17] https://www.perl.org/.
[18] http://www.hdfgroup.org/HDF5/.
[19] https://www.hdfgroup.org/products/java/hdfview/.
[20] http://www.paraview.org/.

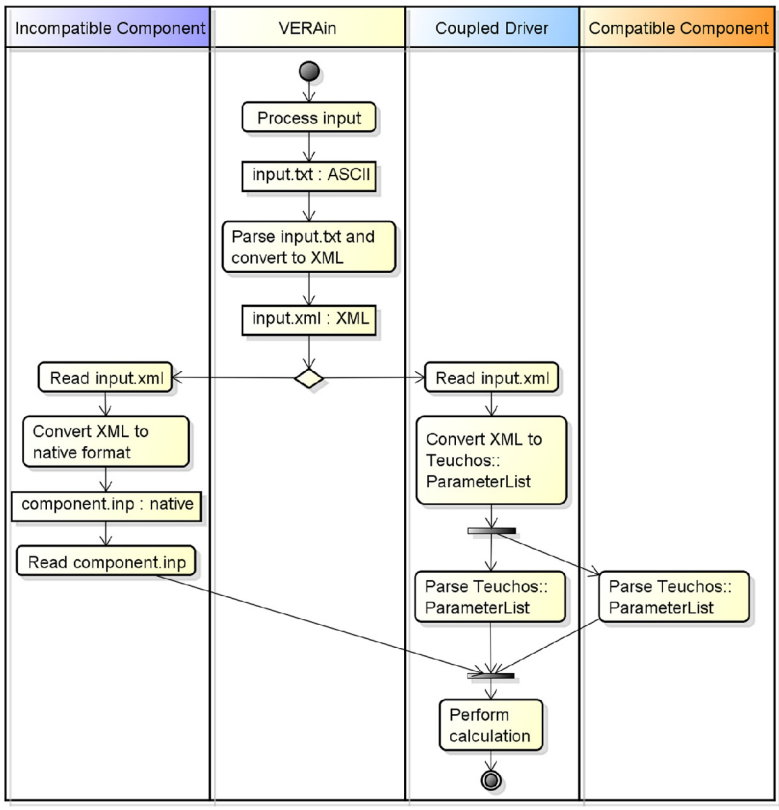


Fig. 4. Flow of data from the VERA Common Input ASCII file to component codes.

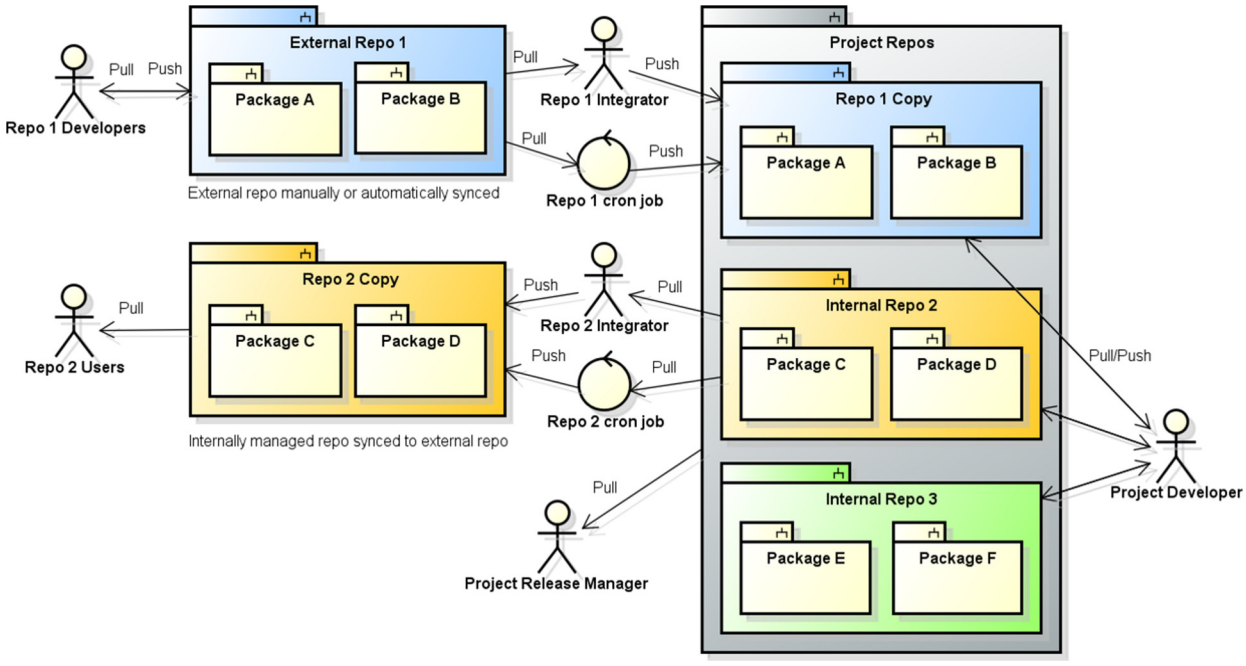


Fig. 5. Managing distributed development.

provided so that developers can automatically test their commits before pushing to the set of primary CASL VERA repositories. The set of primary repositories contains a large suite of unit, nightly, and weekly tests to ensure the performance of individual and coupled codes.

Many of the open source tools (Trilinos, TriBITS, MOOSE, libMesh, DTK) are semi-continuously or regularly synchronized (using the checkin test script) with their primary open-source repository (on github, software.sandia.gov, etc.), so developers can work from a clone of any repository and tests are executed continuously at ORNL for CASL. Several of the export-controlled codes (Insilico, Bison) have a primary repository at their home institution and a nightly script updates the CASL repository. MPACT has no single primary repository because it is jointly developed by ORNL and the University of Michigan. Therefore, a continuous synchronization process runs to ensure the ORNL and Michigan development repositories are in sync and a nightly sync process updates the CASL repositories.

Throughout development, these different models of development have changed and the unified infrastructure allows any given repo to shift between different integration models at different periods of time. Integration of different repositories should be done independently if possible. Errors in one component should not stop pushes of other components. Non-backward compatible changes to upstream repositories require coordinated development and combined pushing to project/master.

Leveraging this infrastructure, we use a hierarchical approach to testing, with fast-running tests for continuous integration, automated nightly and weekly tests to protect against regression, and longer-running verification and validation tests that can be performed as needed (e.g. prior to a release). Tests must pass 100% of the time; broken tests must be addressed immediately or disabled, documented, and flagged for attention. For all tests, results are automatically compared with accepted metrics (solution values, iteration counts) with no human interaction, and in some cases performance metrics are included (CPU time and/or memory usage).

4. Application examples

As noted in Section 3.3, the application codes in VERA can be either fully-integrated, stand-alone drivers that use their own internal numerical solvers and data transfers or coupled using the general purpose capabilities of DTK and PIKE.

The timeline of key coupled-physics development applications provides a clear picture of the support that the VERA infrastructure provides for the migration from exploratory research of a base capability to mature, production quality software (VERA-CS), while still supporting the R&D required for incorporating additional physics to solve key challenge problems.

(1) Insilico + Drekar CFD (2011)
- Initial coupling of Insilico with Drekar CFD [57,21] using DTK and LIME
- Computational cost of CFD deemed too costly for full core nominal operation

(2) VERA-CS: Insilico + CTF (2012)
- Coupling of Insilico with CTF using original Insilico-DTK/LIME interfaces.
- 2012–2013: Extension from single fuel pin to assembly to full-core.
- 2012–2014: Exploration of advanced coupling approaches using LIME/PIKE/AMP/Thyra/NOX.
- Insilico deemed too computationally expensive for industry use.

(3) Tiamat: Insilico + CTF + Bison (2013-4)
- Initial exploration of coupling core simulator with fuel performance code for Pellet–Clad Interaction (PCI) Challenge Problem.
- PIKE developed due to LIME design issues and lack of key features.
- Coupled using DTK and PIKE.

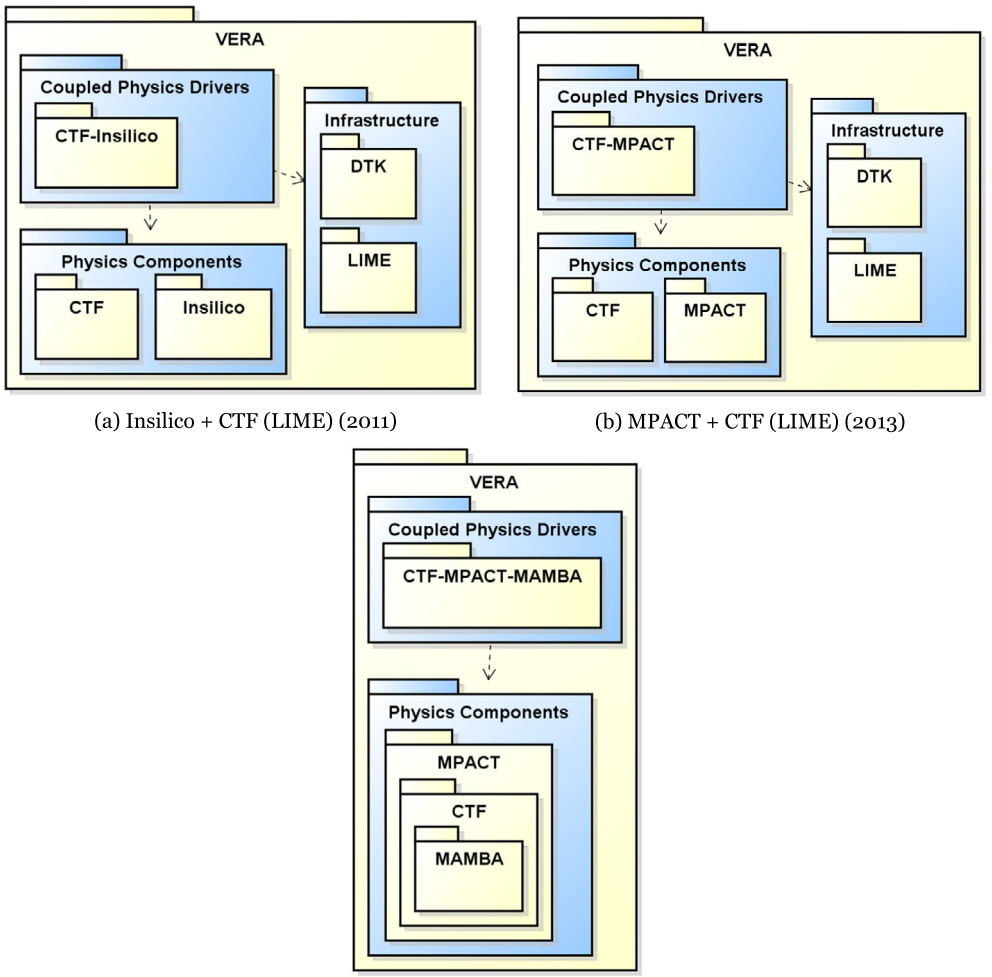
(4) VERA-CS: MPACT/CTF (2014-5)
- Coupling of MPACT with CTF using CTF & Insilico interfaces to DTK & LIME.
- Replaced with direct coupling of CTF within MPACT without DTK/LIME.
- Completes development of production core simulator for Challenge Problems.
- Integration of Acceleration methods into solvers (PIKE/NOX.

(5) Tiamat: VERA-CS + Bison (2015)
- Replacement of Insilico + CTF with MPACT + CTF then VERA-CS.
- Enables use of both optimized VERA-CS and DTK/PIKE for R&D of coupling.

(6) VERA-CS: MPACT/CTF + MAMBA (2015)
- Direct integration of MAMBA as a sub-grid model within VERA-CS; no need for DTK/LIME.
- Multi-cycle capability supporting the CIPS Challenge Problem.

The use of DTK interfaces allowed for a smooth transition from CFD (1) to subchannel thermal-hydraulics (2), with minimal additional work inside Insilico. The use of LIME/PIKE allowed for the exploration of a wide variety of non-linear solution strategies and preconditioners for coupling within VERA-CS (2). The transition in VERA-CS from Insilico (2) to MPACT (4) was relatively smooth because the interfaces to LIME/DTK were easily replicated inside MPACT. CTF did not require any change and the support for internal drivers enabled a simple transition from external coupling through DTK to direct coupling of similar Fortran codes. The persistence of the DTK/PIKE interfaces throughout development of VERA-CS and Tiamat (1–6) simplified exploration of advanced time-integration and coupling strategies required for PCI evaluations.

In this section, we provide detail of two of these coupled applications and examples of their use.



(a) Insilico + CTF (LIME) (2011)

(b) MPACT + CTF (LIME) (2013)

(c) MPACT/CTF + MAMBA (no LIME) (2015)

Fig. 6. Package diagrams showing evolution of the VERA Core Simulator (VERA-CS).

4.1. VERA-CS

As discussed in the introduction of this section, the codes, coupling, and capability of the VERA Core Simulator has evolved. However, the infrastructure has enabled this transition by establishing a single input and output format and a unified build-configure-test infrastructure. Fig. 6(a) shows the original VERA-CS, which utilized a LIME-based driver[21] to solve Insilico and CTF, coupled with DTK, to model a full-core at full power for a single state in time. MPACT incorporated depletion (quasi-static time-dependence in neutronics), but was easily coupled with CTF using a new driver application code,[22] shown in Fig. 6(b). Fig. 6(c) demonstrates the final production version of VERA-CS, which utilizes an internal-coupling of MPACT with CTF along with an internal coupling of CTF with MAMBA.

The development of VERA-CS was guided by a progression of problems to assess the capability, accuracy, and performance of the core simulator [58–61]. This progression enabled subject matter experts and managers to easily track and document the software and data development progress. Upon completion of the progression, a suite of validation problems were defined and modeled to quantify the accuracy of VERA-CS when applied to operating nuclear power plants.

The capability in VERA-CS to simulate operating reactors has been demonstrated through three different demonstrations. The first is the simulation of BEAVRS benchmark [62], which provides two cycles of operating history for a reactor with coolant boron concentrations and detailed neutron flux distributions. The simulation of cycle 1 of this benchmark [63] shows good comparison with measured data and demonstrates the ability for VERA-CS to simulate operating reactors. Continued work on this benchmark is planned.

[21] VRIPSScobra_denovo_coupled.exe.
[22] VRIPSScobra_mpact_coupled.exe.

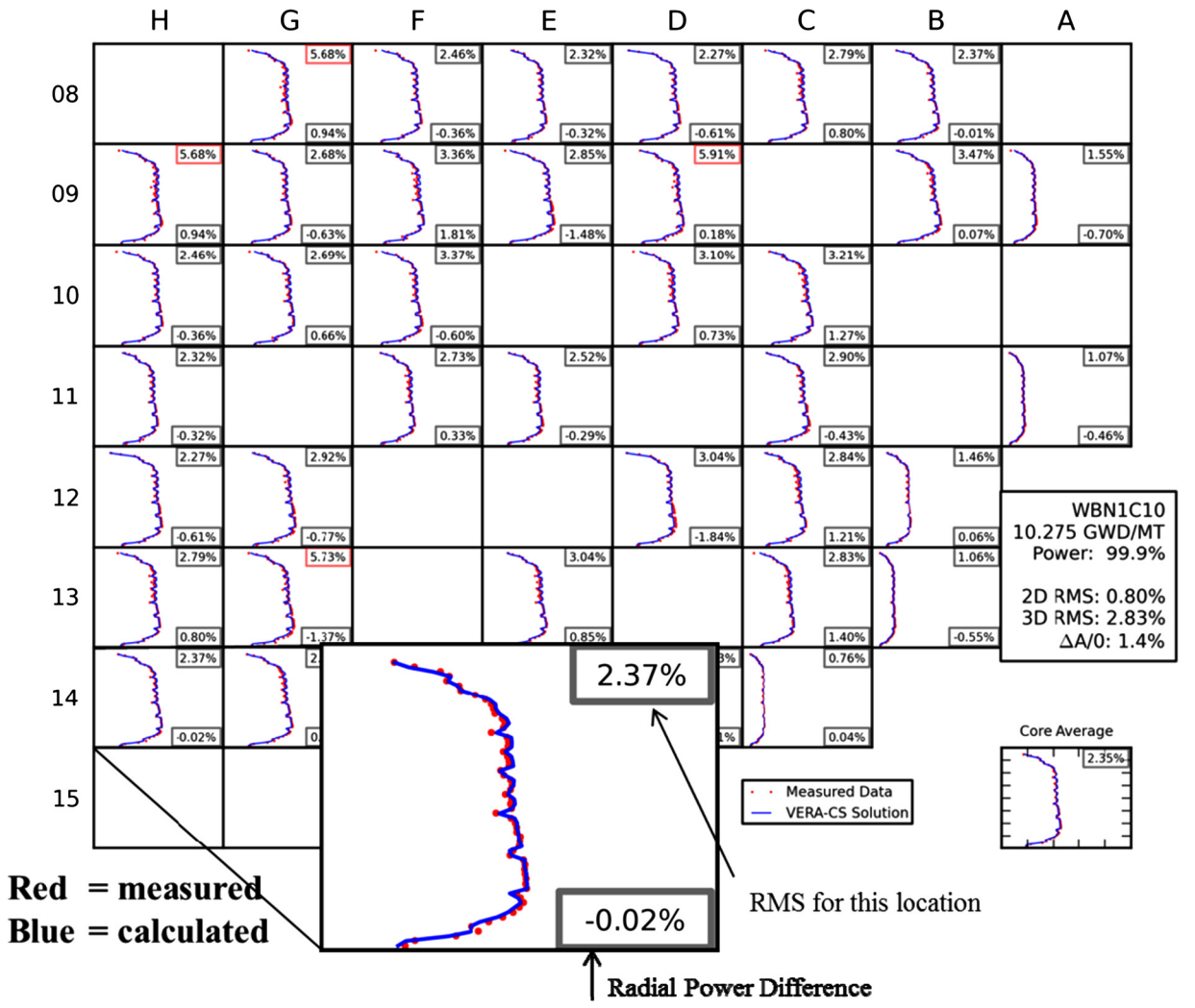


Fig. 7. Neutron Flux Distribution Map for Watts Bar Unit 1 Cycle 10.

The major demonstration of the VERA-CS capability is the simulation of 20 years of operating history of the Watts Bar Nuclear Power Station Unit 1 with 12 core loading cycles [64,65]. For each cycle, coolant boron concentration and neutron flux distributions are compared. Fig. 7 shows a comparison of the measured and predicted flux distribution for the instrumented location of each assembly in the south-east quadrant of the reactor's core.

This distribution is representative of the comparison throughout all 12 cycles simulated. The average radial RMS over all cycles is 1.9% and the average RMS over all data points is 4.1%. One of the largest drivers of the elevated 3D RMS is significant CRUD build up that occurred during cycle 7 which caused a significant axial power shift (CIPS).

The last demonstration of the capability of VERA-CS is to accurately model Watts Bar Cycle 7 which had CIPS [65,66]. The inclusion of the MAMBA coolant chemistry code allows the CRUD to be built up on the surface of the fuel. With the inclusion of MAMBA, the VERA-CS solution was able to reduce the errors observed in the flux maps and compare better with measured data. Fig. 8 shows the flux map difference in the detector signal with and without the inclusion of MAMBA (left) and a comparison of the axial power shape (right).

VERA-CS has been compared to measured operating data for multiple cycles from several reactors [59,67–69]. VERA-CS provides the user with pin-wise and intra-pin solutions that are not available with the current tools available to the nuclear industry. A standard VERA-CS solution provides the user the ability to extract detailed power (Fig. 9a), coolant density (b), total energy generated (c), along with more than 500 individual isotope spacial number densities (d) all from an easy-to-use VERA Common Input specification [38].

4.2. *Tiamat: VERA-CS + Bison*

Tiamat [70,71] was developed to analyze the PCI Challenge Problem by coupling the high-fidelity core simulator with the high-fidelity fuel performance code (Bison). Like VERA-CS, Tiamat has gone through an evolution in design and capability

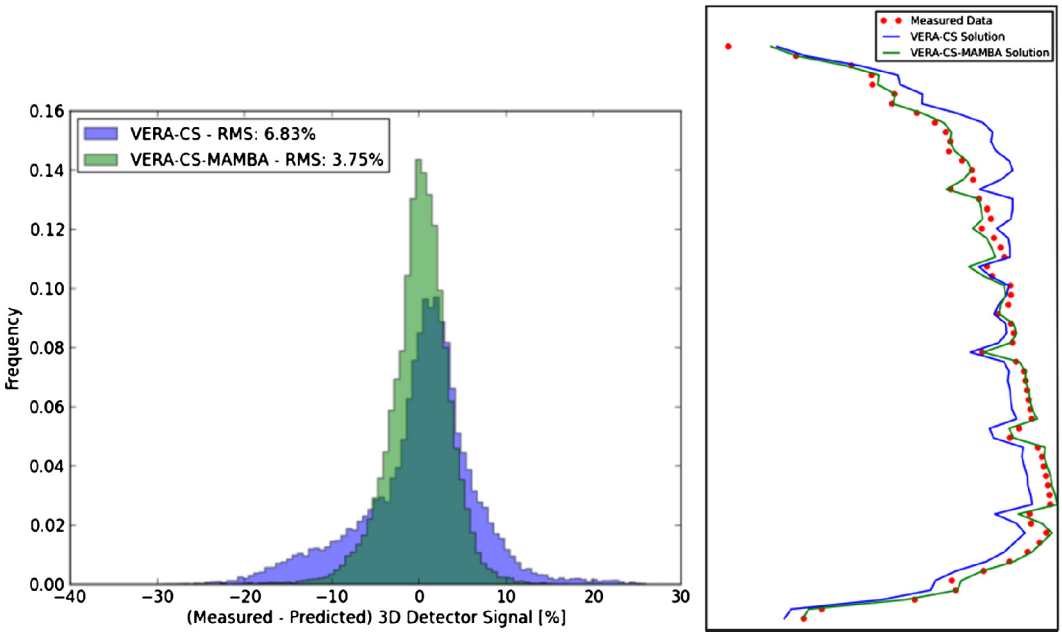


Fig. 8. Distribution of flux map errors in Watts Bar Cycle 7 with and without MAMBA(left) and Axial flux shape with and without MAMBA compared to measured data (right).

throughout development. Fig. 10 shows a) the initial coupling with Insilico, b) the integration of MPACT with initial support for time-dependent neutronics, and c) the integration with VERA-CS, eliminating the need for external coupling with CTF.

As discussed in detail in [71], the time-integration required for coupling a core simulator with a fuel performance code is quite complex because the physics operates at significantly different time scales and the quantities of interest for PCI require each pin to have an accurate Bison solution. The core simulator approximates the fuel and coolant to be in thermal and flow equilibrium throughout the lifetime of the reactor. Similarly, the neutronics is not driven by time-dependent neutron flux distributions, but the slow isotopic decay and transmutation of materials due to neutron fission and capture. Therefore, a separation of variables is performed such that most all of the components in the core simulator are modeled as in equilibrium and the depletion/decay solver uses a predictor–corrector algorithm over long time steps.

However, fuel performance calculations for PCI must resolve small time scales as the fuel and cladding come into mechanical contact at any given axial level of the fuel. Because there are ∼50,000 fuel pins in a reactor and each one approaches mechanical contact at a different point in time, it is computationally burdensome to require all pins utilize a common time step.

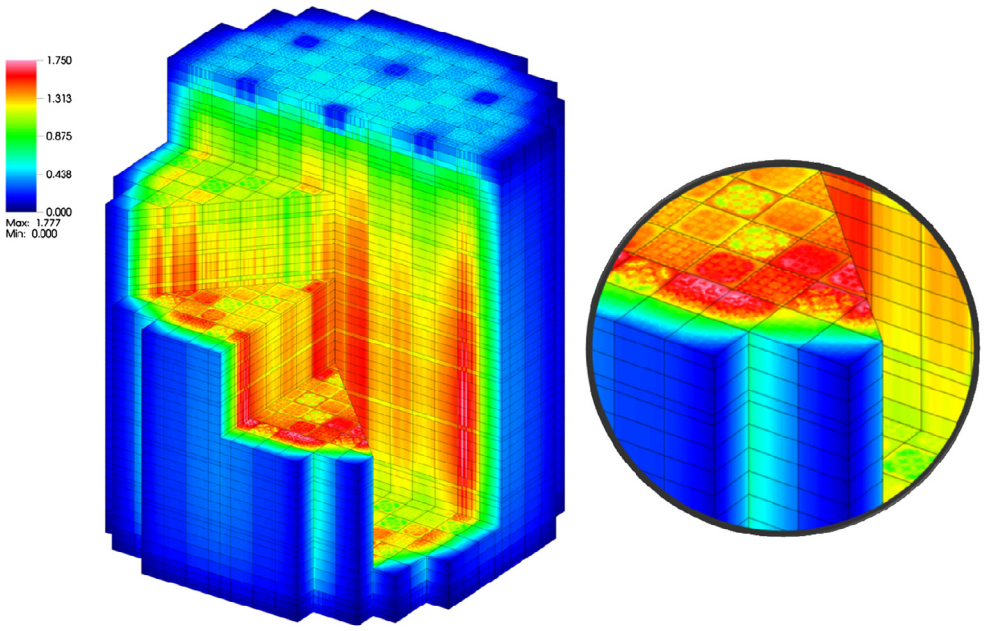
In addition, limitations in the current CASL version of Bison prevent a restart (without file I/O) after a series of time-steps. Therefore PIKE was used to enable a complex time-integration scheme, shown in Fig. 11, which includes a VERA-CS estimate of the hot full power (HFP) conditions (Fig. 13) in parallel with a ramping of Bison from cold-zero power to hot-zero power (HZP), shown in Fig. 14. Once both are complete, Bison can ramp from HZP to the estimated HFP to approach, with adaptive time stepping, a fuel performance solution at the HFP state (Fig. 12). Finally, the full, coupled system is solved using PIKE and the DTK transfers with a steady-state VERA-CS solution and a small time-step in every Bison pin.

The Tiamat coupled driver provides a demonstration capability for scoping potential fuel pin failures by assessing key PCI failure metrics such as peak hoop stress and maximum pin temperature over the operational lifetime. Test problem results for a 5-assembly cross run to HFP is shown in Fig. 15. Future work will scale this driver to full core, multi-cycle analyses.

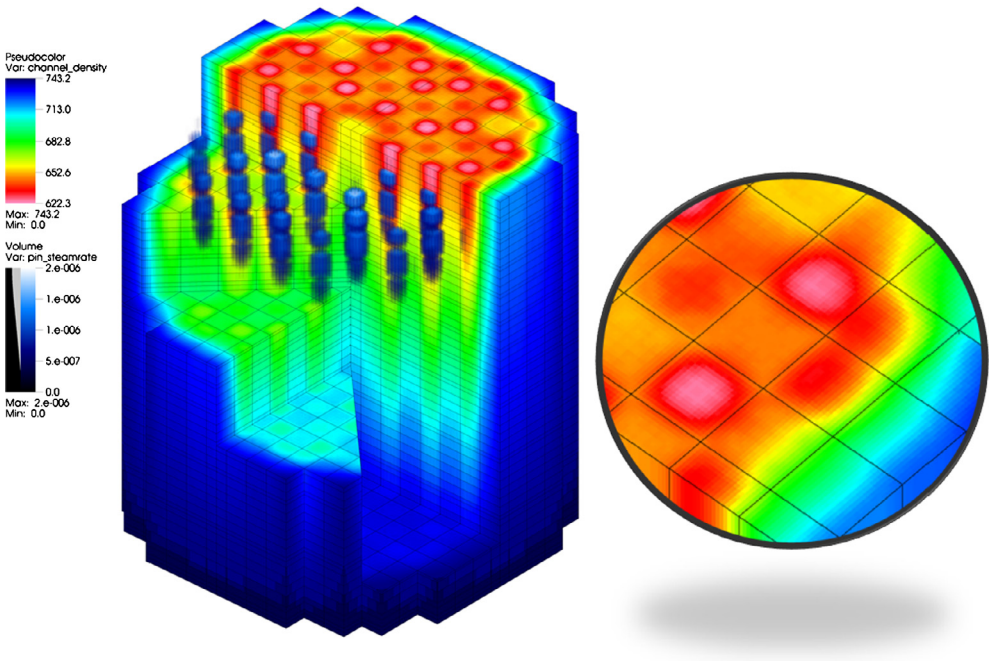
5. Lessons learned and recommendations

The design philosophy of the VERA environment has been demonstrated to be a flexible and powerful multiphysics coupling methodology. By leveraging both toolkits and frameworks, integrating validated and proven physics applications, and developing new physics components and software infrastructure as needed, a significant advance in nuclear reactor simulation has been realized in a fairly short time period. The VERA infrastructure and core components are stable, the developed technology is currently in use with industry partners, and components of the software are available for download both as open source and through the Radiation Safety Information Computational Center (RSICC).[23]

[23] https://rsicc.ornl.gov/ – note that availability through RSICC is currently by request – VERA is not yet listed in the catalog.



(a) VERA-CS Fuel Rod Power Distribution



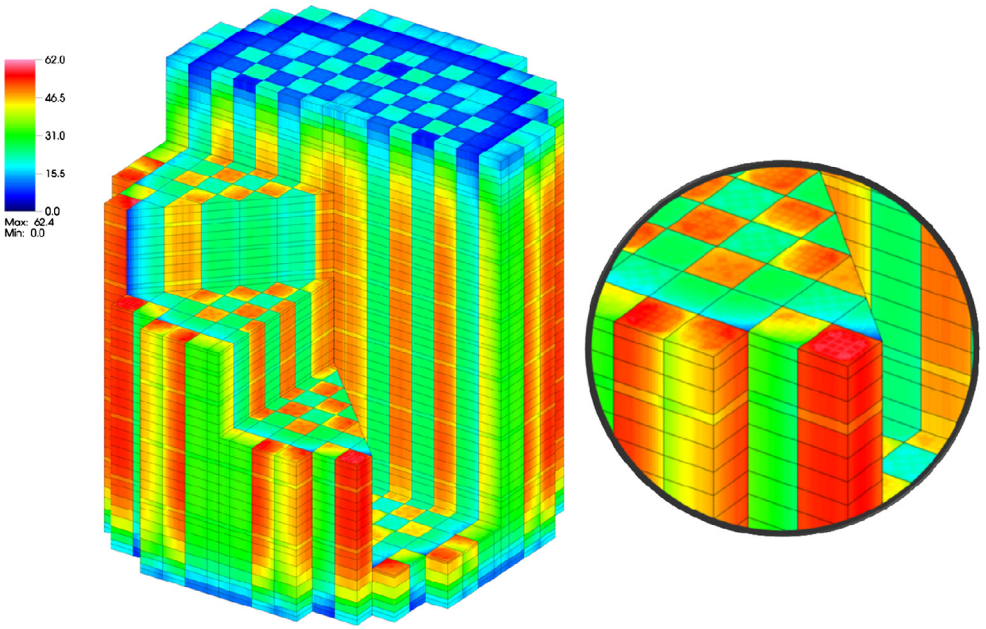
(b) VERA-CS 3D Channel Coolant Densities (kg/m3) and Fuel Rod Steaming Rates (kg/s)

Fig. 9. VERA-CS Simulation Results for Watts Bar Cycle 11.

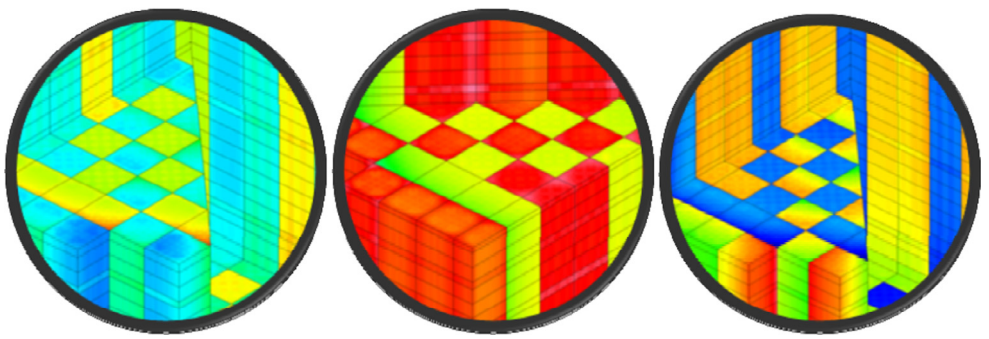
In the process of developing VERA we encountered numerous challenges. The lessons learned in overcoming these challenges can be grouped into two general categories – technical and project management – and we list these in the following two sections, followed by a final thought that includes both technical and management aspects.

5.1. Technical

Technical lessons learned arose from addressing challenges in physics, numerics and software development. Most of those included here are in the latter two categories, while physics challenges are discussed in other articles and reports specific to those components.



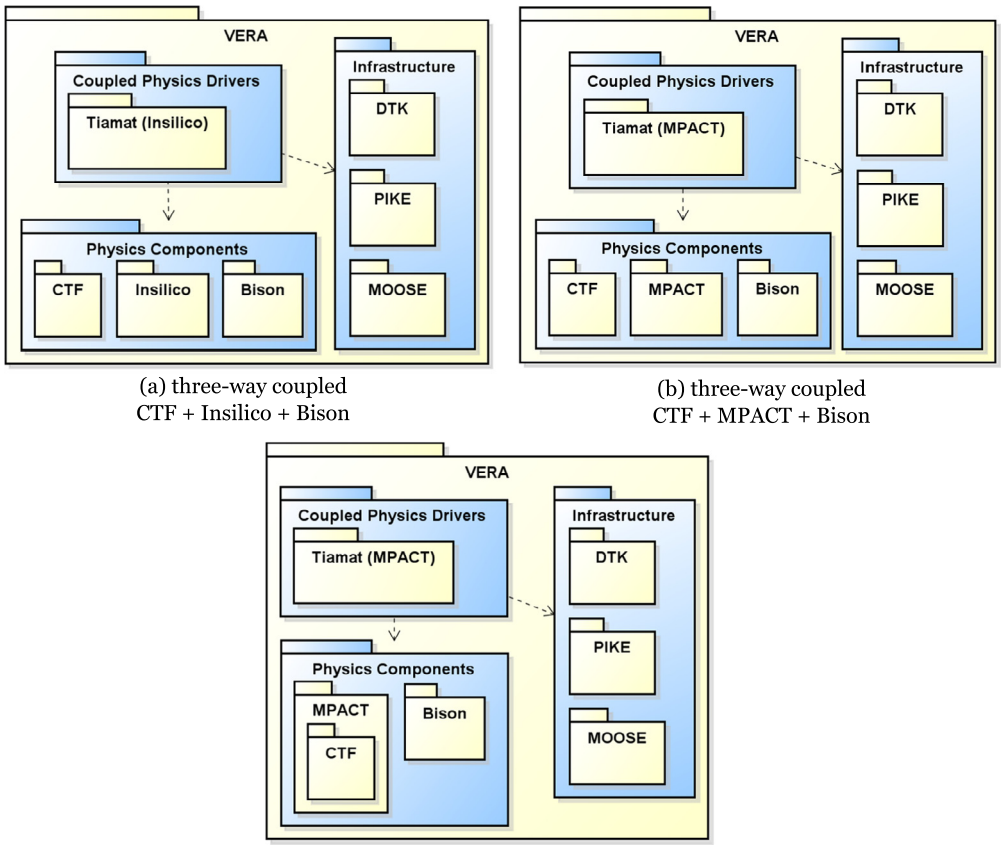
(c) VERA-CS Fuel Rod Exposure Distribution (GWd/MTU)



(d) VERA-CS Fuel Isotope Distribution for U-235, Pu-239, and Pu-241

Fig. 9. (*continued*)

- First and foremost, we found no single ideal solution for complex multi-institutional computational physics applications. No single toolkit, framework, environment, or code provided the optimal foundation on which to build VERA. In addition, many codes assume they are the "master" application, and as a result are not designed to be integrated into a larger software system. Build system, third-party library dependency, namespace, memory management, and MPI communicator conflicts must be resolved. The effort required for these activities is often underestimated and unappreciated.
- Challenge problems defined by subject matter experts (partners in the nuclear industry, in the case of CASL) are essential for defining scope and requirements, and for conveying impact to stakeholders and the community [12]. However, in general they are too complex to be used to drive development. For that, we used "progression problems".
- Progression problems defined by advanced early-adopter users have been critical to success [58]. This well-defined set of progressively more demanding test problems was of tremendous value in both driving development and measuring progress. In CASL, the lack of CFD progression problems led to some challenges that possibly could have been avoided.
- A robust software infrastructure that supports build, test, and continuous integration for multiple distributed software repositories is critical to success, requiring investment that can easily be underestimated. As described in Section 3.6, CASL adopted a common build system that extends widely-used standards [55], as well as a philosophy of continuous integration that enables developers to catch and fix issues and conflicts as early as possible.
- Standardizing input and output forces communication and alignment between component development teams, as well as infrastructure developers [38]. It is also essential in enabling early expert users to begin applying capability to problems of interest. In fact, it is often sufficient, allowing additional "user experience" development effort (e.g. graphical user interfaces) to be deferred until capabilities have advanced to the point that there is a clear need.



(a) three-way coupled CTF + Insilico + Bison

(b) three-way coupled CTF + MPACT + Bison

(c) one-way coupled VERA-CS (CTF+MPACT) + Bison

Fig. 10. Package diagrams showing the evolution of Tiamat.

- Deciding when to develop new software (whether infrastructure or components) and when to leverage existing software can be a complex decision with significant impact. A clear process for assessing existing software and for estimating the cost of new development is essential.
- We knew at the outset that adopting the environment approach would require the ability to efficiently map solution fields between physics components. As a result, DTK [34] was a critical piece of technology, enabling integration and coupling of components and providing a foundational capability that can now be leveraged by other coupled physics programs and applications.

5.2. Project management

Although less technical in nature, the lessons learned managing a large multi-institutional code development effort are significant, and hence of potential interest for similar programs.

- Agile project management is possible, but challenging. CASL's use of a hybrid model using a milestone-driven process for planning and tracking along with an agile approach to development resulted in tensions that declined over the course of the project as a healthy balance was achieved [72].
- The environment approach lowers the bar to entry. As a result, multiple options can co-exist, with the "best" options surviving. Pursue multiple paths as long as possible, since single options are restrictive and risky. Budget realities sometimes preclude multiple paths, but it must be recognized that premature decisions can have lasting repercussions.
- The development team must develop a shared approach to software quality processes, and the entire team must embrace and follow them. Provide training as needed, ideally as a team. Allowing those who will not embrace team practices to continue as team members can introduce substantial conflict and hamper other efforts.
- As described in Section 3.6, a robust hierarchy of testing is central to the CASL software development process. It is not enough to create an infrastructure that supports testing – the development team must create the tests.
- CASL required a geographically-dispersed team of developers with a wide range of expertise, motivations, and priorities to become productive very quickly. This was enabled by extensive use of videoconferencing and virtual collaboration tools. At the outset, regular physical meetings and social activities to support team building were essential, but over

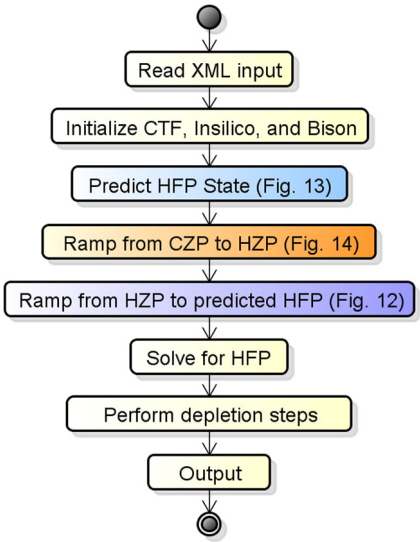


Fig. 11. High-level activity diagram for Tiamat.

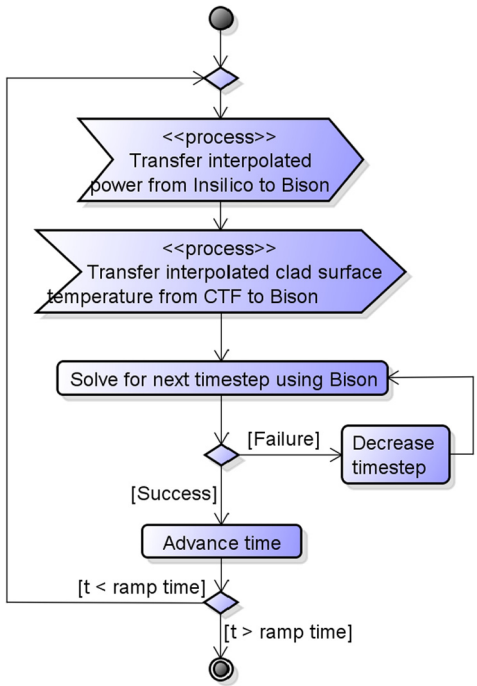


Fig. 12. Activity diagram for Tiamat ramp from HZP to predicted HFP conditions.

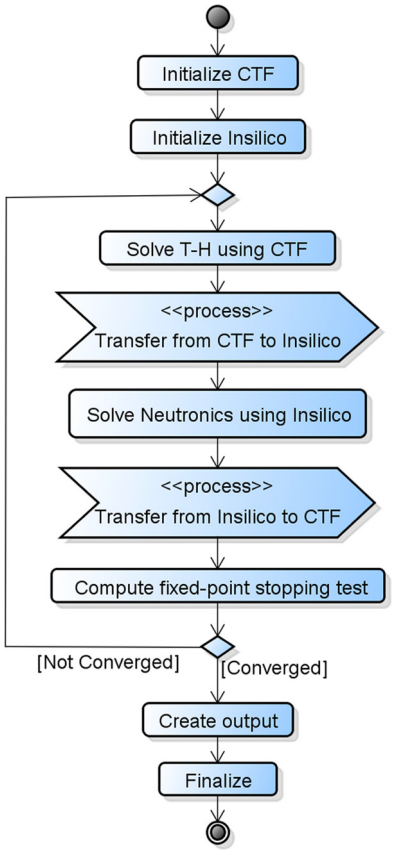


Fig. 13. Activity diagram of Tiamat procedure to predict Hot Full Power (HFP) conditions.

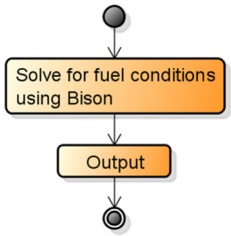
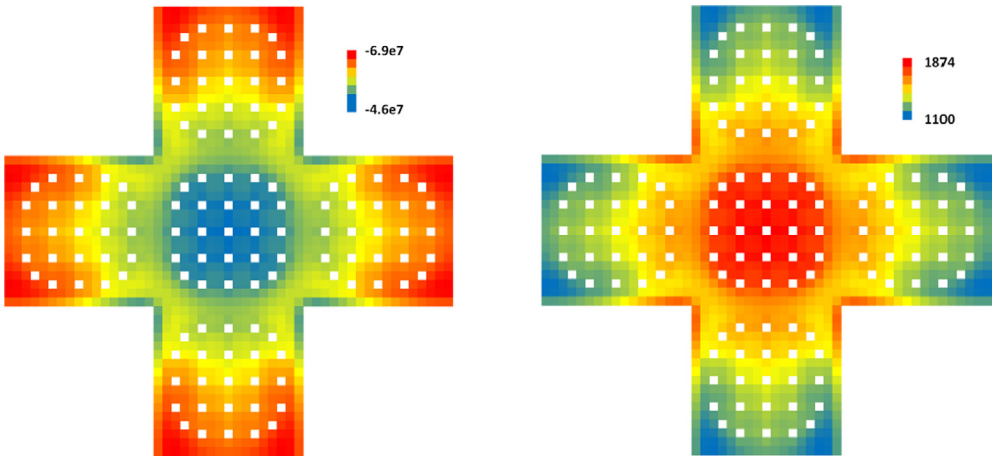


Fig. 14. Activity diagram for Tiamat ramp from Cold Zero Power (CZP) to Hot Zero Power (HZP) conditions.



(a) peak clad hoop stress (b) peak centerline fuel temperature (K)

Fig. 15. Key metrics for assessing pellet–clad interaction (PCI) failures.

time virtual meetings became the dominant means of collaboration. The success of this approach in CASL validates both the "Hub" concept of collaborative teams focused on industry-defined challenges and CASL's "virtual one-roof" implementation [73].

- Since CASL leverages software components under active development for other programs and applications, coordination and prioritization of development efforts can be challenging. Additional challenges arise due to the multi-institutional nature of CASL. An integrated leadership team, with representatives from all institutions, can help to mitigate these challenges.
- Similarly, intellectual property agreements and software licensing must be addressed early and aggressively. Require as much openness as possible (subject to proprietary concerns and export control restrictions), and identify and resolve, or eliminate, incompatibilities as early as possible. This includes both codes with license constraints as well as development teams unwilling to share ownership.
- Given an appropriate development process, universities can both deliver production quality software and perform research. Key contributions to CASL such as DTK, MPACT, and CTF were developed largely by university partners.

5.3. *A final thought on software environments and capability evolution*

After describing the philosophy behind the design of VERA and the coupling approach in Section 3 we provided two application examples in Section 4. In each of these examples an overview of the evolution of the required capability was provided. It is important to note that in each case approaches and software used in early versions were modified or even discarded in later versions. For example:

- VERA-CS was initially based on LIME and DTK, but eventually used direct coupling.
- VERA-CS initially used Insilico for neutronics, but now relies on MPACT.
- Tiamat uses PIKE rather than LIME for coupling.
- As with VERA-CS, Tiamat initially used Insilico for neutronics and later transitioned to MPACT.

In some cases these aspects of evolution were based on availability of physics capabilities and in some cases computational performance. The lesson in these examples is that while the specifics of a particular coupled capability are typically quite problem-dependent, the use of the environment approach, with general coupling capabilities such as LIME, PIKE, or AMP, along with data transfer tools such as DTK, allows development of prototypes able to deliver near-term capability as well as the flexibility to evolve. In some cases the initial implementation meets all requirements, while in other cases the implementation may prove to be overly general, resulting in needless complexity, or may prove to have inadequate performance. In still other cases insights gained during the initial implementation lead to application-specific performance improvements.

One example is the use of DTK for VERA-CS. While DTK was useful in quickly implementing a parallel coupled CTF + MPACT capability, since both components use structured meshes with a simple known parallel decomposition, performance improvements were achieved by bypassing DTK and implementing direct transfers. And of course DTK remains necessary for transfers between components using unstructured meshes.

The lesson is that a flexible approach, leveraging general coupling tools such as PIKE and DTK, allows rapid development of capability that can be deployed to both users and researchers. This initial capability can then be assessed for efficiency, accuracy, and flexibility and incrementally improved based on program priorities.

6. Next steps

Future work will extend the coupled driver and components to additional reactor types such as boiling water reactors (BWRs) and small modular reactors (SMRs). In addition, we will continue to improve the accuracy and efficiency of individual components and coupled applications, with particular emphasis on improving performance on next-generation many-core and hybrid accelerator-based architectures.

CASL VERA is not merely a capability demonstration, but is a tool currently in use by industry to deliver accurate and robust simulations of real operating nuclear reactors. In addition, it provides a platform on which applications for design and analysis of reactors can be developed and deployed for decades to come.

References

[1] M.A. Heroux, R.A. Bartlett, V.E. Howle, R.J. Hoekstra, J.J. Hu, T.G. Kolda, R.B. Lehoucq, K.R. Long, R.P. Pawlowski, E.T. Phipps, A.G. Salinger, H.K. Thornquist, R.S. Tuminaro, J.M. Willenbring, A. Williams, K.S. Stanley, An overview of the Trilinos project, ACM Trans. Math. Softw. 31 (3) (Sep. 2005) 397–423, http://dx.doi.org/10.1145/1089014.1089021.

[2] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Kris Buschelman, Lisandro Dalcin, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Karl Rupp, Barry F. Smith, Stefano Zampini, Hong Zhang, Portable, Extensible Toolkit for Scientific Computation (PETSc), http://www.mcs.anl.gov/petsc/ (accessed September 27, 2015).

[3] B.S. Kirk, J.W. Peterson, R.H. Stogner, G.F. Carey, libMesh: a C++ library for parallel adaptive mesh refinement/coarsening simulations, Eng. Comput. 22 (3–4) (Dec. 2006) 237–254, http://dx.doi.org/10.1007/s00366-006-0049-3.

[4] W. Bangerth, R. Hartmann, G. Kanschat, deal.II – a general-purpose object-oriented finite element library, ACM Trans. Math. Softw. 33 (4) (Aug. 2007) Article 24, http://dx.doi.org/10.1145/1268776.1268779.

[5] J. Turner, Virtual Environment for Reactor Applications (VERA): snapshot 3.1, Milestone report, CASL technical report: CASL-U-2013-0164-000, Oak Ridge National Laboratory, Jul. 2013, http://www.casl.gov/docs/CASL-U-2013-0164-000.pdf.

[6] M. Sieger, VERA release notes ID: 2013 VERA RSICC release (VERA 3.2), Milestone report, CASL technical report: CASL-U-2014-0172-000, ORNL, Nov. 2013, http://www.casl.gov/docs/CASL-U-2014-0172-000.pdf.

[7] VERA 3.3 release notes, CASL-U-2015-0042-000, CASL, Apr. 2015.

[8] S.P. Hamilton, T.M. Evans, G.G. Davidson, S.R. Johnson, T.M. Pandya, A.T. Godfrey, Hot zero power reactor calculations using the Insilico code, J. Comput. Phys. 314 (Jun. 2016) 700–711, http://dx.doi.org/10.1016/j.jcp.2016.03.033.

[9] F. Franceschini, A. Godfrey, J. Kulesza, R. Oelrich, Westinghouse VERA test stand – zero power physics test simulations for the AP1000 PWR, Westinghouse Electric Power, Milestone report, CASL technical report: CASL-U-2014-0012-001, Oak Ridge National Laboratory, Mar. 2014, http://www.casl.gov/docs/CASL-U-2014-0012-001.pdf.

[10] B. Mervin, M. Pytel, D. Hussey, S. Hess, EPRI test stand report: results and feedback from the EPRI test stand, CASL technical report: CASL-U-2014-0121-000-a, Electric Power Research Institute (EPRI), Aug. 2014, http://www.casl.gov/docs/CASL-U-2014-0121-000-a.pdf.

[11] F. Franceschini, M. Kromar, D. Ćalić, A. Godfrey, B.S. Collins, T.M. Evans, J.C. Gehin, Simulation of the NPP KRSKO startup core with CASL core simulator, VERA-CS, Portorož, Slovenia, 01-Sep-2014, http://www.casl.gov/docs/CASL-U-2014-0183-000.pdf.

[12] P.J. Turinsky, D.B. Kothe, Modeling and simulation challenges pursued by the Consortium for Advanced Simulation of Light Water Reactors (CASL), J. Comput. Phys. 313 (May 2016) 367–376, http://dx.doi.org/10.1016/j.jcp.2016.02.043.

[13] Rose Montgomery, VERA tools and workflows, rev. 1, CASL-U-2014-0054-001, CASL, Mar. 2014, http://www.casl.gov/docs/CASL-U-2014-0054-001.pdf.

[14] J.C. Gehin, A.T. Godfrey, F. Franceschini, T.M. Evans, B.S. Collins, S.P. Hamilton, Operational reactor model demonstration with VERA: Watts Bar Unit 1 Cycle 1 zero power physics tests, CASL-U-2013-0105-000, CASL, Jun. 2013.

[15] B. Collins, T.J. Downar, J.C. Gehin, A. Godfrey, A.M. Graham, D. Jabaay, B. Kelley, K.T. Clarno, K.S. Kim, B. Kochunas, E. Larsen, Y. Liu, Z. Liu, W.R. Martin, S. Palmtag, M. Rose, T. Saller, S.G. Stimpson, T. Trahan, J. Wang, W.A. Wieselquist, M.T.H. Young, A. Zhu, MPACT theory manual, version 2.0.0, Milestone report & technical manual, CASL technical report: CASL-U-2015-0078-000, Oak Ridge National Laboratory, Oak, Ridge, TN, Mar. 2015, http://www.casl.gov/docs/CASL-U-2015-0078-000.pdf.

[16] T.M. Pandya, S. Johnson, G.G. Davidson, T.M. Evans, S.P. Hamilton, SHIFT: a massively parallel Monte Carlo radiation transport package, in: Proceedings of the ANS MC2015 – Joint International Conference on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA) and the Monte Carlo (MC) Method, Nashville, Tennessee, 2015, vol. ANS MC2015 CD, http://www.casl.gov/docs/CASL-U-2015-0170-000.pdf.

[17] T.M. Pandya, S.R. Johnson, T.M. Evans, G.G. Davidson, S.P. Hamilton, A.T. Godfrey, Implementation, capabilities, and benchmarking of Shift, a massively parallel Monte Carlo radiation transport code, J. Comput. Phys. 308 (Mar. 2016) 239–272, http://dx.doi.org/10.1016/j.jcp.2015.12.037.

[18] B. Zohuri, N. Fathi, Thermal-Hydraulic Analysis of Nuclear Reactors, vol. 26, Springer International Publishing, Cham, 2015, http://link.springer.com/10.1007/978-3-319-17434-1, 2015, accessed October.

[19] M.N. Avramova, COBRA-TF – a thermal-hydraulic subchannel code for LWRs transient analyses (CTF) users' manual, revision 0, Pennsylvania State University, Mar. 2015, Milestone report & technical manual, CASL technical report: CASL-U-2015-0055-000 http://www.casl.gov/docs/CASL-U-2015-0055-000.pdf.

[20] M.A. Christon, J. Bakosi, B.T. Nadiga, M. Berndt, M.M. Francois, A.K. Stagg, Y. Xia, H. Luo, A hybrid incremental projection method for thermal-hydraulics applications, J. Comput. Phys. 317 (Jul. 2016) 382–404, http://dx.doi.org/10.1016/j.jcp.2016.04.061.

[21] J.N. Shadid, T.M. Smith, E.C. Cyr, T.M. Wildey, R.P. Pawlowski, Stabilized FE simulation of prototype thermal-hydraulics problems with integrated adjoint-based capabilities, J. Comput. Phys. 321 (Sep. 2016) 321–341, http://dx.doi.org/10.1016/j.jcp.2016.04.062.

[22] R.L. Williamson, J.D. Hales, S.R. Novascone, M.R. Tonks, D.R. Gaston, C.J. Permann, D. Andrs, R.C. Martineau, Multidimensional multiphysics simulation of nuclear fuel behavior, J. Nucl. Mater. 423 (1–3) (Apr. 2012) 149–163, http://dx.doi.org/10.1016/j.jnucmat.2012.01.012.

[23] B. Kendrick, V. Petrov, D. Walter, A. Manera, B. Collins, T. Downar, J. Secker, K. Belcourt, CASL multiphysics modeling of crud deposition in PWRs, in: Proceedings of the 2013 LWR Fuel Performance Meeting/Top Fuel, Charlotte, North Carolina, 2013, http://experts.umich.edu/pubDetail.asp?n=Thomas+J+Downar&u_id=587&oe_id=1&o_id=114&id=84902345680.

[24] J.J. Dongarra, J. Du Croz, S. Hammarling, I.S. Duff, A set of level 3 basic linear algebra subprograms, ACM Trans. Math. Softw. 16 (1) (Mar. 1990) 1–17, http://dx.doi.org/10.1145/77626.79170.

[25] R.D. Falgout, J.E. Jones, U.M. Yang, The design and implementation of HYPRE, a library of parallel high performance preconditioners, in: A.M. Bruaset, A. Tveito (Eds.), Numerical Solution of Partial Differential Equations on Parallel Computers, vol. 51, Springer-Verlag, Berlin/Heidelberg, 2006, pp. 267–294, http://link.springer.com/10.1007/3-540-31619-1_8 (accessed September 28, 2015).

[26] J.R. Stewart, H.C. Edwards, The SIERRA framework for developing advanced parallel mechanics applications, in: L.T. Biegler, M. Heinkenschloss, O. Ghattas, B. van Bloemen Waanders (Eds.), Large-Scale PDE-Constrained Optimization, in: LNCS, vol. 30, Springer-Verlag, Berlin/Heidelberg, 2003, pp. 301–315, http://link.springer.com/10.1007/978-3-642-55508-4_18 (accessed September 28, 2015).

[27] D. Gaston, C. Newman, G. Hansen, D. Lebrun-Grandié, MOOSE: a parallel computational framework for coupled systems of nonlinear equations, Nucl. Eng. Des. 239 (10) (Oct. 2009) 1768–1778, http://dx.doi.org/10.1016/j.nucengdes.2009.05.021.
[28] B. Philip, M.A. Berrill, S. Allu, S.P. Hamilton, R.S. Sampath, K.T. Clarno, G.A. Dilts, A parallel multi-domain solution methodology applied to nonlinear thermal transport problems in nuclear fuel pins, J. Comput. Phys. 286 (Apr. 2015) 143–171, http://dx.doi.org/10.1016/j.jcp.2015.01.029.
[29] A. Logg, K.-A. Mardal, G. Wells (Eds.), Automated Solution of Differential Equations by the Finite Element Method, LNCS, vol. 84, Springer Verlag, Berlin/Heidelberg, 2012, http://link.springer.com/10.1007/978-3-642-23099-8 (accessed September 28, 2015).
[30] J.G. Michopoulos, C. Farhat, J. Fish, Modeling and simulation of multiphysics systems, J. Comput. Inf. Sci. Eng. 5 (3) (2005) 198, http://dx.doi.org/10.1115/1.2031269.
[31] A. Gel, S. Pannala, M. Syamlal, T.J. O'Brien, E.S. Gel, Comparison of frameworks for a next-generation multiphase flow solver, MFIX: a group decision-making exercise, Concurr. Comput. Pract. Exp. 19 (5) (Apr. 2007) 609–624, http://dx.doi.org/10.1002/cpe.1085.
[32] D.E. Keyes, L.C. McInnes, C. Woodward, W. Gropp, E. Myra, M. Pernice, J. Bell, J. Brown, A. Clo, J. Connors, E. Constantinescu, D. Estep, K. Evans, C. Farhat, A. Hakim, G. Hammond, G. Hansen, J. Hill, T. Isaac, X. Jiao, K. Jordan, D. Kaushik, E. Kaxiras, A. Koniges, K. Lee, A. Lott, Q. Lu, J. Magerlein, R. Maxwell, M. McCourt, M. Mehl, R. Pawlowski, A.P. Randles, D. Reynolds, B. Riviere, U. Rude, T. Scheibe, J. Shadid, B. Sheehan, M. Shephard, A. Siegel, B. Smith, X. Tang, C. Wilson, B. Wohlmuth, Multiphysics simulations: challenges and opportunities, Int. J. High Perform. Comput. Appl. 27 (1) (Feb. 2013) 4–83, http://dx.doi.org/10.1177/1094342012468181.
[33] S. Slattery, P. Wilson, R. Pawlowski, The data transfer kit: a geometric rendezvous-based tool for multiphysics data transfer, in: Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, Sun Valley, Idaho, 2013, p. 1262, http://www.casl.gov/docs/DTK_slattery_wilson_pawlowski_MC_Proceedings_2013.pdf.
[34] S.R. Slattery, Mesh-free data transfer algorithms for partitioned multiphysics problems: conservation, accuracy, and parallelism, J. Comput. Phys. 307 (Feb. 2016) 164–188, http://dx.doi.org/10.1016/j.jcp.2015.11.055.
[35] D.A. Knoll, D.E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, J. Comput. Phys. 193 (2) (Jan. 2004) 357–397, http://dx.doi.org/10.1016/j.jcp.2003.08.010.
[36] R. Schmidt, K. Belcourt, R. Hooper, R. Pawlowski, An introduction to LIME 1.0 and its use in coupling codes for multiphysics simulation, SAND2011-8524, Sandia National Laboratories, Nov. 2011.
[37] R. Pawlowski, R.A. Bartlett, K. Belcourt, R. Hooper, R. Schmidt, A theory manual for multiphysics code coupling in LIME, version 1.0, SAND2011-2195, Sandia National Laboratories, Mar. 2011.
[38] R. Schmidt, K. Belcourt, R. Hooper, R. Pawlowski, K. Clarno, S. Simunovic, S. Slattery, J. Turner, S. Palmtag, An approach for coupled-code multiphysics core simulations from a common input, Ann. Nucl. Energy 84 (Oct. 2015) 140–152, http://dx.doi.org/10.1016/j.anucene.2014.11.015.
[39] R. Bartlett, LIME 2.0 architecture and design: version 0.5 (rev. 1), Milestone report, CASL technical report: CASL-U-2011-0167-000, Oak Ridge National Laboratory, Sep. 2011, http://www.casl.gov/docs/CASL-U-2011-0167-000.pdf.
[40] D.G. Anderson, Iterative procedures for nonlinear integral equations, J. ACM 12 (4) (Oct. 1965) 547–560, http://dx.doi.org/10.1145/321296.321305.
[41] S. Hamilton, M. Berrill, K. Clarno, R. Pawlowski, A. Toth, C.T. Kelley, T. Evans, B. Philip, An assessment of coupling algorithms for nuclear reactor core physics simulations, J. Comput. Phys. 311 (Apr. 2016) 241–257, http://dx.doi.org/10.1016/j.jcp.2016.02.012.
[42] T.J. Tautges, C. Ernst, C. Stimpson, R.J. Meyers, K. Merkley, MOAB: a mesh-oriented database, SAND2004-1592, 970174, http://www.osti.gov/servlets/purl/970174-xH9bKi/, Apr. 2004 (accessed October 4, 2015).
[43] D.G. Baur, H.C. Edwards, W.K. Cochran, A.B. Williams, G.D. Sjaardema, Sierra Toolkit computational mesh conceptual model, SAND2010-1192, 976950, http://www.osti.gov/servlets/purl/976950-nRPOKM/, Mar. 2010 (accessed October 4, 2015).
[44] D. Cuervo, M. Avramova, K. Ivanov, R. Miró, Evaluation and enhancement of COBRA-TF efficiency for LWR calculations, Ann. Nucl. Energy 33 (9) (Jun. 2006) 837–847, http://dx.doi.org/10.1016/j.anucene.2006.03.011.
[45] R. Baker, K. Koch, An S_n algorithm for the massively parallel CM-200 computer, Nucl. Sci. Eng. (Mar. 1998) 312–320, http://dx.doi.org/10.13182/NSE98-1.
[46] B. Kochunas, T. Downar, Y. Liu, Parallel 3-D method of characteristics in MPACT, in: Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, Sun Valley, Idaho, 2013, p. 54, http://www.osti.gov/scitech/biblio/22212693.
[47] N.C. Sly, B.T. Mervin, S.W. Mosher, T.M. Evans, J.C. Wagner, G.I. Maldonado, Verification of the Shift Monte Carlo Code with the C5G7 Reactor Benchmark, American Nuclear Society, Inc., 555 N. Kensington Avenue, La Grange Park, Illinois 60526 (United States), Jul. 2012, http://www.osti.gov/scitech/biblio/22105896 (accessed September 27, 2015).
[48] Fuel Analysis and Licensing Code: Falcon MOD01: Volume 1: Theoretical and Numerical Bases, EPRI, Palo Alto, 2004.
[49] R.O. Montgomery, Peregrine: initial verification and benchmark evaluations against halden fuel rod data and falcon, L2:MPO. P5.03, CASL, Aug. 2012.
[50] M. Christon, Hydra-TH Theory Manual, Los Alamos National Laboratory, 2011.
[51] W.K. Liu, T. Belytschko, H. Chang, An arbitrary Lagrangian–Eulerian finite element method for path-dependent materials, Comput. Methods Appl. Mech. Eng. 58 (2) (Oct. 1986) 227–245, http://dx.doi.org/10.1016/0045-7825(86)90097-6.
[52] D.J. Benson, An efficient, accurate, simple ALE method for nonlinear finite element programs, Comput. Methods Appl. Mech. Eng. 72 (3) (Mar. 1989) 305–350, http://dx.doi.org/10.1016/0045-7825(89)90003-0.
[53] S. Palmtag, A. Godfrey, User manual for the VERA input processor, CASL-U-2014–0014–002, CASL, Feb. 2015.
[54] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, C. Harrison, G.H. Weber, H. Krishnan, T. Fogal, A. Sanderson, C. Garth, E.W. Bethel, D. Camp, O. Rubel, M. Durant, J.M. Favre, P. Navratil, VisIt: an end-user tool for visualizing and analyzing very large data, in: E. Wes Bethel, Hank Childs, Charles Hansen (Eds.), High Performance Visualization–Enabling Extreme-Scale Scientific Insight, CRC Press, 2012, pp. 357–372.
[55] R.A. Bartlett, M.A. Heroux, J.M. Willenbring, Overview of the TriBITS lifecycle model: a Lean/Agile software lifecycle model for research-based computational science and engineering software, 2012, pp. 1–8, http://dx.doi.org/10.1109/eScience.2012.6404448.
[56] R. Bartlett, TriBITS developer guide and reference, Milestone report, CASL technical report: CASL-U-2014-0075-b, Oak Ridge National Laboratory, Oak Ridge, TN, Mar. 2014, http://www.casl.gov/docs/CASL-U-2014-0075-000-b.pdf.
[57] T.M. Smith, J.N. Shadid, R. Pawlowski, E.C. Cyr, P. Weber, Reactor core sub-assembly simulations using a stabilized finite element method, in: Proceedings of the 14th International Topical Meeting on Nuclear Reactor Thermal Hydraulics, Toronto, Ontario, Canada, 2011, http://www.casl.gov/docs/NURETH14-500.pdf.
[58] A. Godfrey, VERA core physics benchmark progression problem specifications, revision 4, CASL-U-2012-0131-004, CASL, Aug. 2014, http://www.casl.gov/docs/CASL-U-2012-0131-004.pdf.
[59] J. Gehin, A. Godfrey, F. Franceschini, T. Evans, B. Collins, S. Hamilton, Operational reactor model demonstration with VERA: Watts Bar Unit 1 Cycle 1 zero power physics test, Milestone report, CASL technical report: CASL-U-2013-0105-001, Oak Ridge National Laboratory, Westinghouse, University of Michigan, Oak Ridge National Laboratory, Oak Ridge, TN, Aug. 2013, http://www.casl.gov/docs/CASL-U-2013-0105-001_final.pdf.
[60] A. Godfrey, MPACT testing and benchmarking results, Milestone report, CASL technical report: CASL-U-2014-0045-000, Oak Ridge National Laboratory, Mar. 2014, http://www.casl.gov/docs/CASL-U-2014-0045-000.pdf.
[61] B. Kochunas, A. Graham, D. Jabaay, S. Stimpson, T. Downar, B. Collins, K.S. Kim, W. Wieselquist, R. Salko, Operational reactor depletion analysis capability, CASL-U-2014-0189-000, CASL, Sep. 2014.
[62] N. Horelik, B. Herman, B. Forget, K. Smith, Benchmark for Evaluation and Validation of Reactor Simulations (BEAVRS), in: Proc. Int. Conf. Mathematics and Computational Methods Applied to Nuclear, Sci. & Eng., Sun, Valley, Idaho, 2013.

[63] B.S. Collins, A. Godfrey, Analysis of the BEAVRS benchmark using VERA-CS, in: Proceeding of the ANS MC2015 – Joint International Conference on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA) and the Monte Carlo (MC) Method, Nashville, Tennessee, 2015, vol. (ANS MC paper 302_1), http://www.casl.gov/docs/CASL-U-2015-0183-000.pdf.
[64] A.T. Godfrey, B.S. Collins, K.S. Kim, R. Lee, J. Powers, R. Salko, S.G. Stimpson, W.A. Wieselquist, R. Montgomery, R. Montgomery, B. Kochunas, D.R. Jabaay, N. Capps, J. Secker, VERA benchmarking results for Watts Bar Nuclear Plant, Milestone report, CASL technical report: CASL-U-2015-0206-000, Jun. 2015, http://www.casl.gov/docs/CASL-U-2015-0206-000.pdf.
[65] A. Godfrey, B. Collins, K.S. Kim, J. Powers, R. Salko, S. Stimpson, W. Wieselquist, K. Clarno, J. Gehin, VERA benchmarking results for Watts Bar Nuclear Plant Unit 1 Cycles 1–12, presented at the PHYSOR 2016, Sun Valley, ID, 2016.
[66] B. Collins, R. Salko, S. Stimpson, K. Clarno, A. Godfrey, S. Palmtag, J. Secker, B. Kendrick, R. Montgomery, Simulation of CRUD induced power shift using the VERA core simulator and MAMBA, presented at the PHYSOR 2016, Sun Valley, ID, 2016.
[67] F. Franceschini, A. Godfrey, J. Gehin, AP1000® PWR reactor physics analysis with VERA-CS and KENO, part I: zero power physics tests, in: Proceedings of PHYSOR 2014 International Conference, Kyoto, Japan, 2014, http://www.casl.gov/docs/CASL-U-2015-0005-000.pdf.
[68] F. Franceschini, A. Godfrey, J. Gehin, AP1000® PWR reactor physics analysis with VERA-CS and KENO, part II: power distribution, J. Nucl. Sci. Technol. JNST Spec. Issue, no. special, Physor 2014 papers, Oct. 2014 http://www.casl.gov/docs/CASL-U-2015-0006-000.pdf.
[69] F. Franceschini, M. Kromar, D. Ćalić, A. Godfrey, B.S. Collins, T.M. Evans, Simulation of the NPP KRSKO startup core with CASL core simulator, VERA-CS, in: Proceedings of the International Conference Nuclear Energy for New Europe, NENE, Portorož, Slovenia, 2014, http://www.casl.gov/docs/CASL-U-2014-0182-000.pdf.
[70] K.T. Clarno, R.P. Pawlowski, R.O. Montgomery, T.M. Evans, B.S. Collins, B. Kochunas, D. Gaston, J.A. Turner, High-fidelity modeling of pellet–clad interaction using the CASL Virtual Environment for Reactor Applications, in: Joint Intl. Conf. on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA), and the Monte Carlo (MC) Method, 2015.
[71] R.P. Pawlowski, K.T. Clarno, R.O. Montgomery, R. Salko, T.M. Evans, J.A. Turner, D. Gaston, Design of a high-fidelity core simulator for analysis of pellet–clad interaction, in: Joint Intl. Conf. on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA), and the Monte Carlo (MC) Method, Nashville, TN, 2015.
[72] R. Wysocki, Effective Project Management: Traditional, Agile, Extreme, 7th ed., John Wiley & Sons, Inc., 2013, http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118729161.html (accessed July 29, 2016).
[73] A. Lewis, G. Weigand, Virtual Office Community and Computing (VOCC), Designing an energy science Hub collaboration system, in: Human Centered Design: Proceedings of Second International Conference, held as part of HCI International 2011, vol. 6776, HCD 2011, Orlando, Florida, Orlando, FL, USA, July 9–14, 2011, 2011, pp. 425–434.