ORNL/TM-2020/1872
Revision 1

# Coupling of CTF and TRANSFORM using the Functional Mock-Up Interface

December | 2020

William Gurecky
Dane de Wet
M. Scott Greenwood
Bob Salko
Dave Pointer

IES
Integrated Energy Systems

ORNL/TM-2020/1872, Rev. 1

# Coupling of CTF and TRANSFORM using the Functional Mock-Up Interface

William Gurecky
Dane de Wet
M. Scott Greenwood
Bob Salko
Dave Pointer

**December 2020**

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE US DEPARTMENT OF ENERGY

**ORNL/TM-2020/1872, Rev. 1**

Nuclear Energy and Fuel Cycle Division

# COUPLING OF CTF AND TRANSFORM USING THE FUNCTIONAL MOCK-UP INTERFACE

William Gurecky
Dane de Wet
M. Scott Greenwood
Bob Salko
Dave Pointer

Date Published: December 2020

Prepared by
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, TN 37831-6283
managed by
UT-Battelle, LLC
for the
US DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22725

## CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## ACRONYMS

| | |
|---|---|
| API | application programming interface |
| BSD | Berkeley software distribution |
| BWR | boiling water reactor |
| CASL | Consortium for the Advanced Simulation of Light Water Reactors |
| CRUD | Chalk River unidentified deposit |
| CTF | COBRA-TF |
| DOE | US Department of Energy |
| ECI | exterior communication interface |
| FMI | Functional Mock-Up Interface |
| FMU | Functional Mock-Up Unit |
| GPM | gallons per minute |
| HPC | high-performance computing |
| IES | integrated energy system |
| LOCA | loss of coolant accident |
| LWR | light water reactor |
| MAMBA | MPO Advanced Model for Boron Analysis |
| MCCARD | Monte Carlo Code for Advanced Reactor Design and Analysis |
| MCMC | Markov Chain Monte Carlo |
| MOC | method of characteristics |
| MPACT | material protection, accounting, and control technologies |
| MSR | molten salt reactor |
| MSRE | Molten Salt Reactor Experiment |
| MWth | megawatt thermal |
| NEAMS | Nuclear Energy Advanced Modeling and Simulation |
| ODE | ordinary differential equation |
| ORIGEN | Oak Ridge Isotope Generation |
| ORNL | Oak Ridge National Laboratory |
| PWR | pressurized water reactor |
| RAVEN | Risk Analysis Virtual Environment |
| SCALE | Standardized Computer Analyses for Licensing Evaluation |
| TH | thermal hydraulic |
| TRACE | TRAC/RELAP Advanced Computational Engine |
| TRANSFORM | Transient Simulation Framework of Reconfigurable Models |
| UQ | uncertainty quantification |
| VERA | Virtual Environment for Reactor Applications |
| XML | extensible markup language |

## ABSTRACT

Integrated energy systems (IES) combine power-generation with thermally activated technologies. In this coupling of technology, there will arise a natural mismatch in the level of detail and timescales which need to be evaluated for proper design and operation based upon the underlying physics of the individual processes. Given the intimate coupling desired from these IESs, there will need to be a robust method which enables simulation and analysis of coupled high-low fidelity simulations for some critical scenarios. This report summarizes work to couple a nuclear industry standard nuclear core sub-channel tool with a dynamic system level modeling tool using an industry standard modeling interface.

An in-memory coupling between the sub-channel thermal hydraulics code COBRA-TF (CTF), which is included in the Virtual Environment for Reactor Applications (VERA), and the systems code Transient Simulation Framework of Reconfigurable Models (TRANSFORM) was developed in this work. Data exchange is accomplished by using the Functional Mock-Up Interface (FMI), an open standard for coupling dynamic systems models together. The FMI-based coupling necessitated the development of a novel FORTRAN wrapper for communicating with Functional Mock-Up Units. This wrapper will facilitate future FMI-based code couplings with other FORTRAN-based programs. CTF-TRANSFORM is exercised on a simplified Molten Salt Reactor Experiment (MSRE) model in steady-state and transient configurations. The coupled CTF-TRANSFORM model is shown to predict core temperature deltas similar to those available in historical MSRE operational data and is shown to be robust to fast power transients. The coupling paves the way to performing sensitivity and design studies impossible with VERA/CTF alone, in which secondary and tertiary loop operating conditions and design parameters may be perturbed and their impact on the core operating conditions studied. This is a necessary capability for modeling integrated energy systems where nuclear reactors could act as a power-source for coupled thermally activated technologies like industrial heating and energy storage.

## 1. INTRODUCTION

In recent years, the US Department of Energy (DOE) has helped modernize and advance the modeling and simulation of current reactors while simultaneously supporting the capability to develop and license the next generation of reactor technologies. Over the last 10 years this has been accomplished in part by the Consortium for the Advanced Simulation of Light Water Reactors (CASL), as well as the more recently formed Nuclear Energy Advanced Modeling and Simulation (NEAMS) program. These programs have produced tools capable of simulating many aspects of reactor design and operation in coupled simulations that can include reactor physics, fuel performance, chemistry, thermal fluids, and other phenomena with varying degrees of fidelity. The tools were often developed with a mindset for coupling to other tools within the programs, providing improved capabilities for modeling complex integral phenomena. With the development of the NEAMS Workbench, these codes are more integrated than ever before, marking a significant achievement by DOE in advancing the modeling and simulation capabilities of nuclear systems [2]. Recent interest in integrated energy systems proposes to combine power generation technologies with coupled thermally activated technologies to improve energy efficiency, reduce emissions, and improve grid reliability and security [3]. In order to model these systems coupled with a nuclear reactor, it would be necessary to demonstrate the impact of changes in operating parameters of coupled systems on the behavior and safety of the reactor core. The coupling between TRANSFORM and CTF demonstrated in this work provides this capability.

To further enhance the capabilities and applications of these tools, this work demonstrates how the Functional Mock-Up Interface (FMI),"a free standard that defines a container and an interface to exchange dynamic models," can be used to facilitate coupling between high-fidelity tools and lower-fidelity models to expand the use of these tools to more applications while providing users with improved flexibility in addressing complex coupled problems [4]. The motivation for pursuing this coupling interface is to make it possible to characterize the propagation of uncertainty in coupled reactor systems by using a detailed core representation. This coupling standard also allows for broader applications. The FMI standard has been implemented in over 100 tools used by academia and various industries for studying a spectrum of different systems [5]. It provides the capability of transferring and coupling models between tools as an exchangeable package known as *Functional Mock-Up Units (FMUs)* that can be shared with or without their own solvers. The FMUs can be exported as black boxes or with full access to the underlying model, making it possible to better control the sharing and distribution of information between collaborators and allowing vendors to share transient models of relevant components with their clients. This paper demonstrates how the tools developed within the NEAMS and Virtual Environment for Reactor Applications (VERA) Users' Group programs could use this coupling standard to import FMU models from a wide array of systems into coupled simulations.

The current effort only focuses on importing FMUs into a coupled simulation, and it is not intended to export the COBRA-TF (CTF) models or models of CASL or NEAMS tools as FMUs. This is done by incorporating the FMI coupling with the CTF code to perform steady-state and transient simulations between a sub-channel thermal-hydraulic CTF model of the Molten Salt Reactor Experiment (MSRE) core and a 1D model of the external piping, heat exchangers, pumps, and heat removal developed by using the Transient Simulation Framework of Reconfigurable Models (TRANSFORM) library of Modelica models. Ultimately, this will be used to study the propagation of uncertainty throughout the coupled systems via a detailed core representation, making it possible to better understand how uncertainties in properties in one region of the reactor affect the performance of other regions. The MSRE was chosen for the demonstration in this work partly because the uncertainties in the design calculations led to reduced power performance, and similar mistakes should be mitigated for future designs [6]. This is only one demonstration of many that a coupling with these tools could facilitate.

There are many reasons why such a coupling would be useful if tools within the NEAMS workbench supported the FMI standard. For example, during reactor core design, it might be useful to have a mechanical model of the control rod mechanism that can be tested for various tuneable parameters to achieve the desired performance. The neutronics code might not be suited for modeling mechanical systems related to the control rod mechanisms, and adding such a capability would be beyond the scope of interest for a typical user. This coupling interface would allow users to quickly perform coupled simulations without requiring any additional code development or complex custom coupling schemes. There are also many cases in which it would be useful to include control logic similar to that in the instrumentation and controls of a nuclear system, making it possible to incorporate the effects of reactor control on the progression of transient coupled simulations. More simple examples include incorporating a vendor's component model into an overall model of a system or incorporating a black box model for the behavior of a component without giving proprietary dimensions. This could be a model of a heat exchanger, pump, or other components within nuclear systems. This paper does not attempt to provide an exhaustive list of applications to demonstrate. Rather, the focus of this work was to provide a simple demonstration of how the FMI coupling provides a capability that is needed for the design analysis of future reactor designs. If similar efforts were employed with the other tools within the NEAMS and VERA Users' Group programs, then this would provide the capability of coupling powerful high-fidelity models with lower fidelity models of important connected systems, control schemes, or related components that would be more difficult to incorporate otherwise.

## 1.1 VERA/CTF BACKGROUND

VERA is a high-fidelity multiphysics nuclear reactor core simulator developed by CASL based at Oak Ridge National Laboratory (ORNL) [7]. There are several tools within the VERA environment that allow important phenomena to be simulated. MPACT is the deterministic neutron transport code used in VERA based on the 2D/1D scheme in which pin-resolved 2D method of characteristics (MOC) transport sweeps over radial planes are axially coupled via diffusion-like transport approximations to obtain the transverse leakage between planes [8] [9]. The ORIGEN depletion code developed in the SCALE software suite is coupled with MPACT to model the fuel transmutation and decay [10]. VERA employs CTF, a modernized version of the COBRA-TF sub-channel code, for thermal hydraulics (TH) [11]. Optional models for coolant chemistry, cladding corrosion, and the formation of CRUD are provided by the MAMBA software package [12]. With these codes coupled together, VERA produces a pin-resolved model of a nuclear reactor core with millions of unique depletion regions, each tracking over 200 isotopes. A standard VERA coupled solve performs 51 group MOC transport with full-core TH feedback provided at a pin-resolved resolution by CTF. VERA typically requires a high-performance computing (HPC) cluster with approximately 1,000 cores or greater to simulate a pressurized water reactor (PWR) full-core model. VERA primarily targets multicycle simulations in which a reactor is simulated over the course of many months or years by stepping through a series of depletion state points that each require fixed-point iterations to solve the steady-state, coupled neutronic, and TH system.

To date, VERA has been benchmarked against a variety of codes, including comparisons with Monte Carlo transport results from KENO, MC21 [13], and deterministic results from SCALE-Polaris [14]. A variety of burnup benchmarks against SERPENT and MCCARD have also been performed [15]. In VERA development, rigorous solution verifications for the MPACT and CTF codes were conducted to ensure that the computational methods were implemented correctly and documented in validation and verification reports [16] [17]. Additionally, VERA has been employed in a predictive mode. VERA was used to perform a blind prediction of the startup power and flow distribution for the Watts Bar Unit 2 nuclear plant before the unit was brought online in late 2016. The VERA-predicted core behavior matched the measured in-core flux distribution, critical boron, rod bank worths, and reactivity coefficients with high levels of precision [18].

VERA was developed to address key challenge problems identified by CASL industry partners, such as Pellet Cladding Interaction, vessel fluence, and CRUD-Induced Power Shift, which require tight coupling between neutronics, TH, and core chemistry solvers to resolve. Initial work to extend VERA to molten salt reactors (MSRs) to capture neutron precursor drift and volatile and nonvolatile species transport in CTF was completed [19]. Additionally, VERA contains a separate coupled MPACT/CTF transient capability to simulate rod ejection events [20].

A standard VERA simulation does not include components that are outside the reactor core vessel. For certain applications, including transient accident scenarios, LOCA, or changes in secondary and tertiary loop operating conditions (or load follow), is of interest to couple a systems model of the external core loop components. This work investigated coupling VERA/CTF to the TRANSFORM systems code to combine the high-fidelity in-core physics models provided by VERA and the mechanistic models of the primary, secondary, and tertiary loop components provided by TRANSFORM. By performing this coupling, it is possible to propagate the uncertainty of parameters in the region of the reactor system to the detailed core representation to assess impacts on potential performance.

## 1.2 FUNCTIONAL MOCK-UP INTERFACE BACKGROUND

In this work, the CTF code was coupled with TRANSFORM by using a coupling standard for dynamic models called *FMI*. FMI is an open standard developed by the Modelica Association Project to facilitate information exchange between models of dynamic system components so that, if all models conform the the FMI standard, it is simple to connect them together and interchange constituent components of a larger multicomponent system [4]. The FMI standard is widely adopted in industries in which dynamic system models are prevalent so that models developed at different institutions can be used together if all models conform to the same interface. The FMI standard is maintained by the Modelica Association Project FMI and is distributed under the 2-Clause BSD license. The current stable release of the FMI standard is version 2.0 at the time of writing. Therefore, the tools developed in this work target this standard.

An FMU is a container for a system of ordinary differential equations (ODEs) that holds an Extensible Markup Language (XML) file describing all system variables and parameters, as well as a library of C functions that define the system variable time derivatives and how to interact with the system state. The FMU presents an interface that conforms to the FMI standard. Typically, the FMU is compiled as a shared object library and bundled with the XML file into a zip archive for distribution. This zipped archive format is what most FMU wrappers expect, including the tools developed for FMU import in CTF/VERA in this work. However, a zip file format is not strictly required. Only the shared object and XML file are required. Details on the XML file format for FMU models can be found in the FMI white paper specification documentation [21].

FMUs come in two variations: Model Exchange and Co-Simulation. Figure 1 shows that a Model Exchange FMU contains routines to define model state (i.e., system variables), parameters, and time derivatives of each system variable of interest. In a Model Exchange FMU, an external tool provides the ODE stepper. A Co-Simulation FMU also contains its own internal ODE solver and an additional routine called $doStep$. The $doStep$ routine steps the FMU forward by a user-provided time step size via the FMU internal ODE solver, which could leverage any type of explicit or implicit method, depending on the stiffness of the model ODEs.

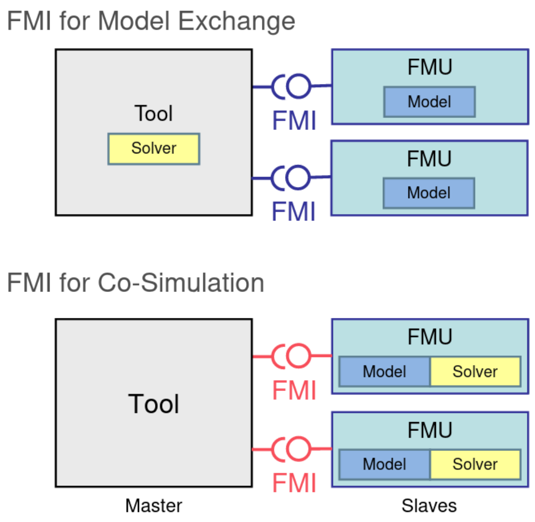


**Figure 1. Overview of FMI interface, Model Exchange, and Co-Simulation FMUs [1].**

The TRANSFORM model used in this work is in the Co-Simulation FMU category because it contains its own ODE stepper. The TRANSFORM system model, which comprises constituent models written in the Modelica scripting language, is translated to C and compiled by the Dymola® software from Dassault Systemes to produce the requisite shared object library and XML model description file for use with VERA/CTF. The FMU contains the C functions provided in source and/or binary form. The FMUs with the binary form can only be run on the platform for which they are compiled [4]. It is possible to have binary forms for different platforms within the same FMU ZIP file, depending on the tool used to export the FMU [4].

## 1.3 TRANSFORM BACKGROUND

TRANSFORM is a library of Modelica models that can be used to model TH energy systems and other multiphysics systems [22]. The models are all written in the Modelica language, which is a "non-proprietary, object-oriented, equation based language to conveniently model complex physical systems" [23]. The Modelica language has been used to model many types of systems, including mechanical, electrical, hydraulic, thermal, and control systems, among others [23]. The TRANSFORM library is a collection of models, thermophysical properties, and formulations that allow for a flexible modeling approach of nuclear systems and components. It has been used to model nuclear reactor phenomena such as system-level TH [24; 25], tritium transport [26], delayed neutron precursor drift [27], mass transport and accountancy [28], and integrated energy systems [29; 30; 31]. The TRANSFORM library of models is open-source and can be easily augmented by users to fit the needs of a given situation to model mechanical systems, thermal-hydraulic systems, or any system that can be reasonably described by differential, algebraic, and discrete-time equations [4]. These models can then be exported as FMUs for use with FMI-compatible tools. In this work, a model of the MSRE piping, pumps, heat exchangers, and heat rejection developed by using the TRANSFORM library was chosen for coupling to the CTF model of the MSRE core as a demonstration.

## 1.4 MSRE BACKGROUND

The MSRE was a fluoride-salt-fueled, graphite-moderated reactor designed, built, and operated at ORNL in the 1960s [6]. It is the longest operating MSR to date, and it provides most of the current understanding about the practical aspects of operating this class of reactors. The reactor operated at a thermal power of approximately 7.4 MWth with a primary flow rate of 1,200 gpm and a secondary flow rate of around 850 gpm [6]. It comprised a primary loop that contained the circulating fuel salt, a secondary loop that contained a nonfuel bearing salt, and an air duct for air-cooled heat removal. The primary loop consisted of the reactor core that contained a graphite moderator assembly, a fuel salt circulating pump, a primary shell-and-tube heat exchanger, the connecting piping, and the supporting and connected systems required for operation. The secondary loop contained a coolant salt pump, the tube-side of the shell-and-tube primary heat exchanger, an air-cooled radiator, and the connective piping. The heat was removed from the system through the forced circulation of air across the radiator tube bank with the air ultimately being rejected through a stack to the surrounding atmosphere.

The model of the primary piping, secondary loop, and heat rejection was created by using the TRANSFORM library of Modelica models (Figure 2), and the reactor core was modeled by using the CTF code. The CTF core model was coupled to the model of the external piping and heat removal by using the functional mock-up interface and the FORTRAN FMU wrapper that was developed as part of this work. The model used in this work has not been validated against experimental measurements. The focus of the work was on demonstrating the coupling of CTF and TRANSFORM, and the potential to couple similar codes to any number of models that can be exported as functional mock-up units.

# 2. CTF-TRANSFORM COUPLING IMPLEMENTATION

A coupling between the TRACE systems code and CTF was developed in prior work [32]. This previously developed coupling was used as a starting point for this work and was extended for use in coupling TRANSFORM with CTF. In the prior development, the Exterior Communications Interface (ECI) was used to facilitate information exchange between CTF and TRACE. In the same vein, the Futility FORTRAN FMU wrapper, which is discussed in Section 2.1, was developed and used in the present work to facilitate information exchange between CTF and TRANSFORM. The existing CTF systems coupling capability was extended according to the hierarchy provided in Figure 3. The boundary condition exchange pattern at the core inlet and outlet was inherited from this previous TRACE coupling development.

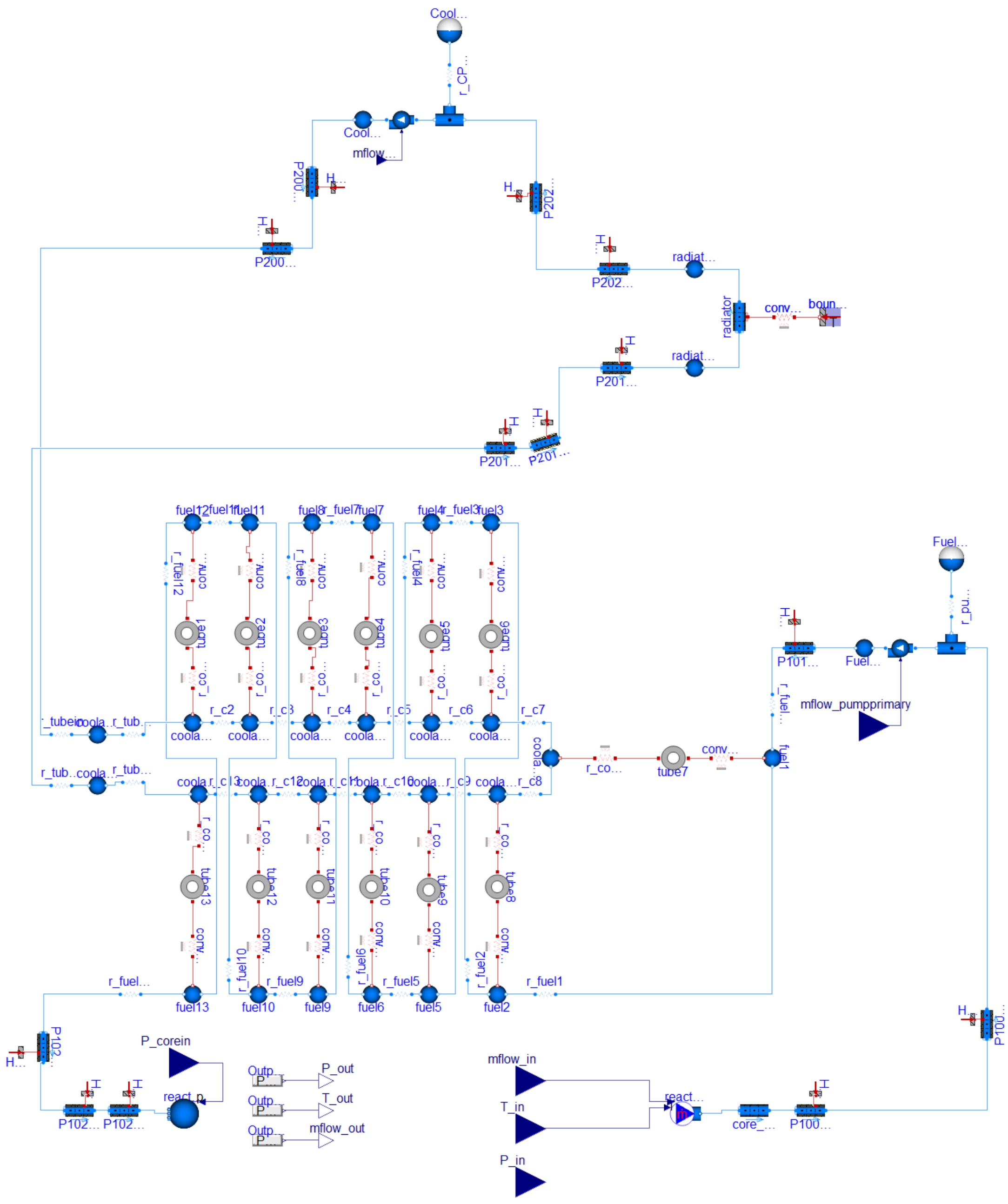


**Figure 2. TRANSFORM nodalization of the MSRE piping and heat removal.**

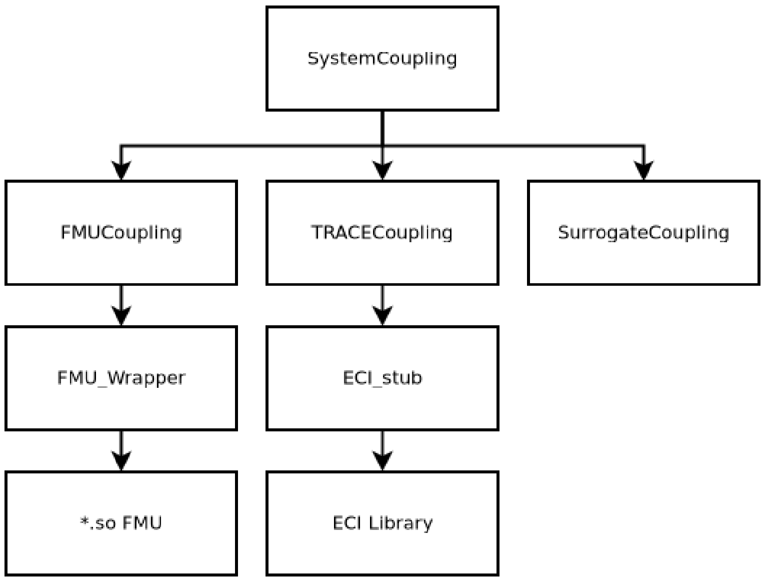


**Figure 3. Representation of the *SystemCoupling* hierarchy in CTF.**

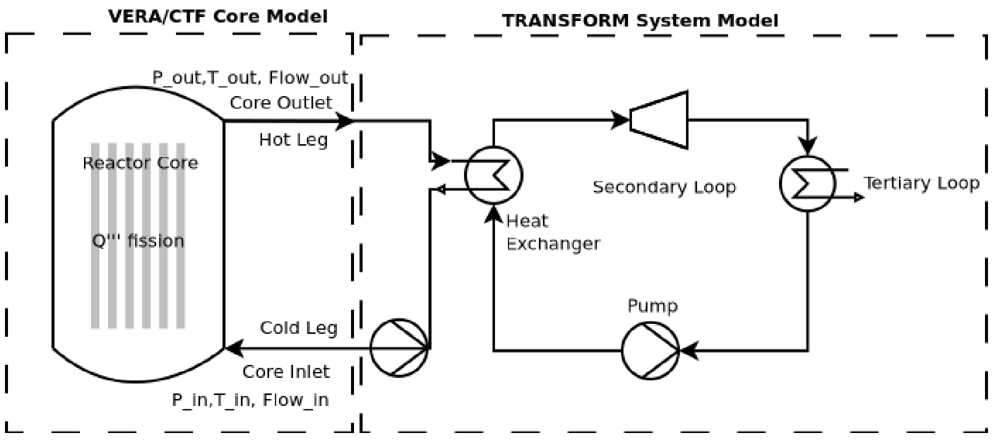


**Figure 4. Example coupled VERA/CTF core model and TRANSFORM FMU system model for a PWR.**

At a high level, relatively few boundary conditions are required in a coupling between a nuclear reactor core model and a systems code. In this work, TRANSFORM is the systems code of interest, and CTF is the core TH simulation package. At the time of writing, the TRANSFORM code does not capture spatial variations of the flow field at the inlet and exit of the core, and thus the exchange of the core inlet and outlet average temperature, pressure, and flow rate values are the only data passed between the two codes. Table 1 describes the current information exchanged between VERA/CTF and TRANSFORM at the core inlet and exit. Figure 5 shows the information exchanged between CTF and TRANSFORM.

| Variable | Unit | Note | Direction |
|---|---|---|---|
| $T_{in}$ | K | Core inlet temperature | VERA/CTF ← TRANSFORM |
| $P_{in}$ | Pa | Core inlet static pressure | VERA/CTF → TRANSFORM |
| $\dot{m}_{in}$ | kg/s | Core inlet flow rate | VERA/CTF ← TRANSFORM |
| $T_{out}$ | K | Core outlet temperature | VERA/CTF → TRANSFORM |
| $P_{out}$ | Pa | Core outlet static pressure | VERA/CTF ← TRANSFORM |
| $\dot{m}_{out}$ | kg/s | Core outlet flow rate | VERA/CTF → TRANSFORM |

**Table 1. Boundary conditions exchanged at core inlet and outlet interface.**

An in-memory coupling between CTF and TRANSFORM in the FORTRAN language was pursued instead of a Python script-style file coupling. Because these are FORTRAN packages, tighter integration with CTF and VERA is possible, future extensions to the the interface are simpler, and optimal performance will be ensured. Existing interfaces in the CTF package were extended to use the FMU wrapper to perform the boundary condition exchanges between TRANSFORM and CTF. Additionally, the FORTRAN FMU wrapper developed as part of this work was included in the open-source Futility library and—to the authors' knowledge—represents a first-of-its kind FMU interface in the FORTRAN language. Comparable wrappers exist in Python (FMPy [33]) and C++ (FMIKit [34]), which inspired this project. This tool provides a simplified means to use FMUs inside existing FORTRAN programs, of which there are many in the nuclear engineering space.

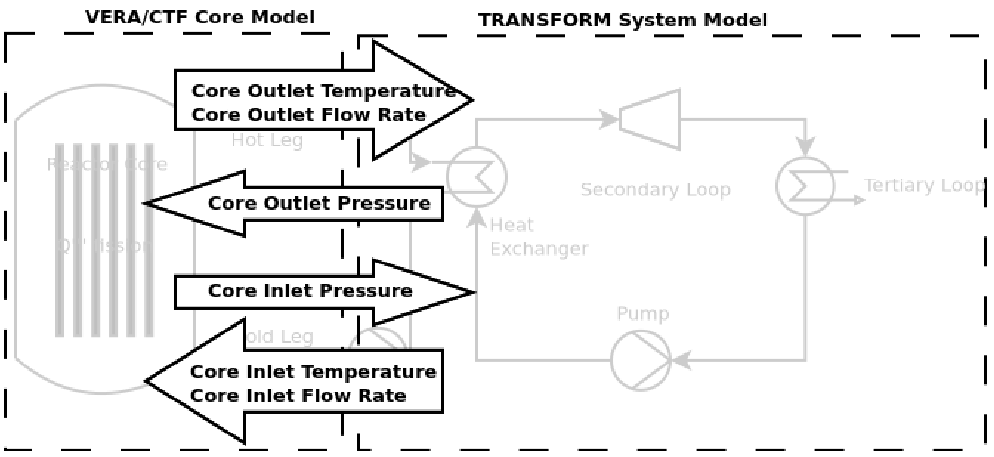


**Figure 5. Example data exchange between CTF and a TRANSFORM FMU system model for a PWR.**

CTF is provided an initial outlet pressure, inlet temperature, and inlet flow rate with these initial values specified via user input. CTF solves for the flow and temperature distribution in the core, including an estimate for the pressure loss over the core. CTF passes the estimated inlet core static pressure to the TRANSFORM model. Additionally, CTF passes the outlet core temperature and flow rate to TRANSFORM. This is done to set up a pressure outlet boundary condition and a mass flow inlet boundary condition in the TRANSFORM primary loop model. TRANSFORM computes the temperature drop across the primary piping and heat exchanger and accounts for any mass loss due to a leak. It then returns the core inlet temperature and core inlet flow rate back to CTF.

## 2.1 FORTRAN FMU WRAPPER

An extensible FORTRAN interface to FMUs was developed and incorporated into the Futility software library [35] as part of this work. Futility is an open-source FORTRAN library jointly developed at ORNL and the University of Michigan to support the rapid development of computational nuclear engineering tools targeting HPC systems. An existing open-source C++ implementation of an FMU interface served as the starting point for this project [34]. The ISO_C_BINDINGS module in FORTRAN was used to generate a thin FORTRAN adapter of the C++ FMU interface. Finally, a set of high-level functions was developed in a pure FORTRAN module, dubbed the FMU_Wrapper, that simplifies some aspects of interacting with an FMU so that users who are not intimately familiar with the FMI standard can import and work with an FMU in a FORTRAN program.

The Futility FMU_Wrapper is a general FMU interface and is not limited to only interfacing with TRANSFORM. Since the boundary condition exchange between TRANSFORM and CTF is performed via the FMU_Wrapper, the CTF system coupling could be extended to couple any compliant FMU to CTF in the future. Additional documentation of the methods available in the FMU_Wrapper is provided in Appendix A.

## 2.2 STEADY-STATE CTF-TRANSFORM COUPLING STRATEGY

To conduct a steady-state calculation, the coupled CTF-TRANSFORM model performs alternating pseudo-transient calculations of the core and system code. These alternating calculations form the outer iteration scheme shown in Figure 6. This is a fixed-point iteration scheme that resolves the interdependency between the in-core CTF model and the system model provided by TRANSFORM. Equation 1 illustrates the fixed point problem.

$$\mathbf{x}_{i+1} = \mathcal{F}_{FMU}(\mathcal{G}_{CTF}(\mathbf{x}_i, \theta_{CTF}), \theta_{FMU}), \tag{1}$$

where $i$ is the outer iteration number, and $\mathbf{x}$ is the vector of steady-state core inlet temperature, core exit temperature, system flow rate, and pressure: $\mathbf{x} = \{T_{in}, P_{in}, \dot{m}_{in}, T_{out}, P_{out}, \dot{m}_{out}\}$. $\theta_{CTF}$ are the CTF model parameters, and $\theta_{FMU}$ are the TRANSFORM FMU model parameters. The parameters comprise reactor geometry, material properties, and model constants. CTF is executed in a pseudo-transient mode, denoted by the function $\mathcal{G}_{CTF}$, which runs the CTF model to steady-state given the current core inlet/outlet state vector as boundary conditions, $\mathbf{x}_i$. Using the updated core outlet temperature and flow rate from CTF, the TRANSFORM FMU model is also run in a pseudo-transient mode denoted by $\mathcal{F}_{FMU}$, and it returns an updated vector $\hat{\mathbf{x}}_{i+1}$ that includes updated core inlet conditions. These updated inlet values are then used by CTF in the next outer iteration. The outer iteration procedure is shown in Algorithm 1.

**Algorithm 1**
Solution strategy for steady-state CTF-FMU coupling.

```
 1: Initialization
 2: (1) Set maximum number of outer iterations, $N$.
 3: (2) Set under relaxation factors, $\omega \in (0, 1]$. Default $\omega = 1$.
 4: (3) Set outer loop convergence tolerance. Default $\varepsilon_{T_{in}} \approxeq 0.25K$.
 5: (4) Supply initial guess for $\mathbf{x}_0 = \{T_{0,in}, \dot{m}_{0,in}, P_{0,out}, ...\}$
 6: (5) Initialize CTF and FMU from input
 7: for Outer step: i in $\{0, ..N\}$ do
 8:     Execute a pseudo-transient CTF computation, given: $\mathbf{x}_i$:
 9:     $\tilde{\mathbf{x}}_{i+1} \leftarrow \mathcal{G}_{CTF}(\mathbf{x}_i, \theta_{CTF})$
10:     Execute a pseudo-transient FMU computation:
11:     $\hat{\mathbf{x}}_{i+1} \leftarrow \mathcal{F}_{FMU}(\tilde{\mathbf{x}}_{i+1}, \theta_{FMU})$
12:     Update the state vector with under-relaxation
13:     $\mathbf{x}_{i+1} = \omega\hat{\mathbf{x}}_{i+1} + (1 - \omega)\mathbf{x}_i$
14:     if $|\mathbf{x}_{i+1} - \mathbf{x}_i| < \varepsilon$ then
15:         Break
16:     end if
17: end for
```

One pseudo-transient calculation involves fixing the boundary conditions as constant and then allowing the code to evolve the solution over time until the flow rates and temperatures no longer appreciably change, thus arriving at a steady-state system. The pseudo-transient CTF solve can be replaced with a steady-state CTF solve without any change to the outer iteration scheme. In a coupled setting, the boundary conditions depend on the other code, requiring that the two codes to be iterated until convergence is achieved. Convergence is achieved when the core inlet and outlet temperatures do not change within a user-set tolerance between outer iterations. This tolerance must be set to be small (i.e., on the order of $\varepsilon_{T_{in}} = 0.1[K]$) to ensure convergence to the steady-state value. Since the convergence is currently judged from iteration-to-iteration differences, it is possible to proclaim convergence before the steady-state value is truly reached if this tolerance is too high. Future work could add additional, more sophisticated convergence checks.

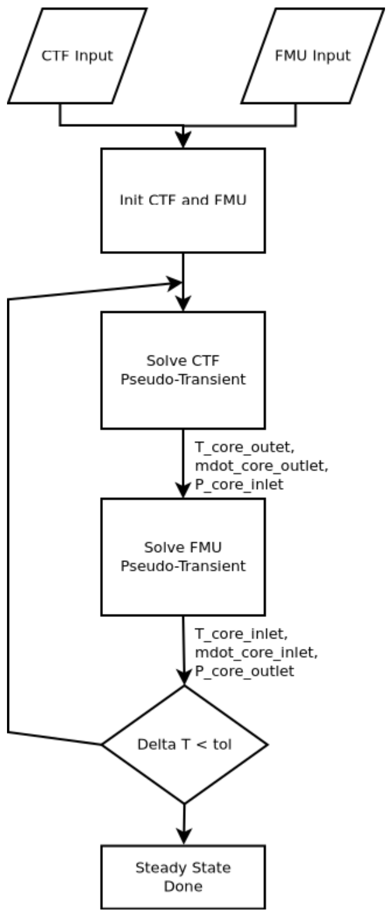


**Figure 6. Steady-state CTF-TRANSFORM coupling flowchart.**

## 2.3 TRANSIENT COUPLING STRATEGY

A transient coupled CTF-TRANSFORM calculation begins by specifying initial conditions and then running a steady-state calculation by the methods described in Section 2.2. After the initial system state is set up, the transient time marching scheme begins. The current scheme does not support repeating the same time step, also known as *solution rewind*, which is useful for a dynamic time step size scheme. However, if the time steps are taken to be sufficiently small, stability issues can be avoided. As an additional precaution against unstable behavior, a time step size can be specified to the TRANSFORM model, $dt_{fmu}$, which is smaller than the CTF outer time step size, $dt_{ctf}$. CTF computes the CTF outer time step size internally to satisfy the core Courant-Friedrichs-Lewy condition and could change as a function of flow rate through the core. If this TRANSFORM time step size is smaller than the outer CTF time step size, then the TRANSFORM model will be sub-stepped, as shown in Figure 7.

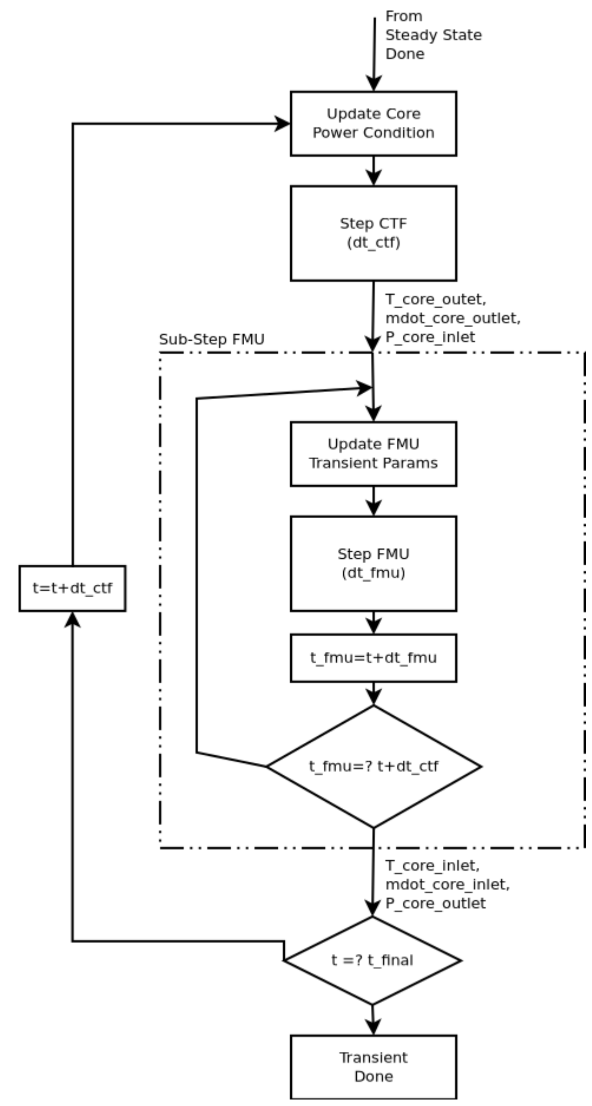

From Steady State Done
Update Core Power Condition
Step CTF (dt_ctf)
T_core_outet, mdot_core_outlet, P_core_inlet
Sub-Step FMU
Update FMU Transient Params
Step FMU (dt_fmu)
t_fmu=t+dt_fmu
t_fmu=? t+dt_ctf
t=t+dt_ctf
T_core_inlet, mdot_core_inlet, P_core_outlet
t =? t_final
Transient Done

**Figure 7. Transient CTF-TRANSFORM coupling flowchart.**

## 2.4 CTF-TRANSFORM USER INPUTS

A coupled CTF-TRANSFORM model requires:

1. a CTF input deck,
2. a TRANSFORM FMU zip archive extracted to a directory, and
3. a TRANSFORM parameter XML coupling specification file named **fmu_params.xml**.

This XML coupling specification file is different than the XML file in the TRANSFORM FMU. The role of the TRANSFORM parameter XML file is twofold. First, it specifies the names of system variables used at the core inlet and outlet for boundary condition exchange. This is done for flexibility of FMU variable naming at FMU export time. Second, this XML file specifies parameters, such as insulation R-values, flow resistances, heat exchanger properties, or any other parameters exposed by the TRANSFORM FMU. This file must be named **fmu_params.xml** and be present in the run directory, which is typically the same directory that contains the CTF input file. This parameter file is important for interfacing with external sensitivity and uncertainty quantification (UQ) packages, such as RAVEN or DAKOTA, because it will be written by these sampling tools in the process of performing a sensitivity study.

An example XML coupling specification file is shown as follows.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ParameterList name="CASEID">
    <Parameter name="FMU_dt_max" type="double" value="0.1"/>
    <Parameter name="ulax_T_corein" type="double" value="1.0"/>
    <Parameter name="toltemp" type="double" value="0.1"/>
    <ParameterList name="FMU_VAR_INIT">
       <!-- Set FMU Parameters and inital values -->
       <Parameter name="T_in" type="double" value="907.0"/>
       <Parameter name="P_in" type="double" value="101.33e3"/>
       <Parameter name="P_corein" type="double" value="101.33e3"/>
       <Parameter name="mflow_in" type="double" value="170.0"/>
       <Parameter name="mflow_pumpprimary" type="double" value="170.0"/>
       <Parameter name="mflow_secondary" type="double" value="105.745"/>
    </ParameterList>
    <ParameterList name="BC_VAR_NAMES">
       <!-- Parameter name= CTF_name     value= FMU_name -->
       <Parameter name="T_corein" type="string" value="T_out"/>
       <Parameter name="T_coreout" type="string" value="T_in"/>
       <Parameter name="P_corein" type="string" value="P_corein"/>
       <Parameter name="P_coreout" type="string" value="P_in"/>
       <Parameter name="mflow_corein" type="string" value="mflow_out"/>
       <Parameter name="mflow_coreout" type="string" value="mflow_in"/>
    </ParameterList>
    <ParameterList name="FMU_VAR_TRANSIENT">
       <!-- FMU vars that vary as a fn of time -->
       <Parameter name="time" type="Array(double)" value="{0,10,100}"/>
       <Parameter name="mflow_secondary" type="Array(double)" value="{80,90,105.7}"/>
    </ParameterList>
    <ParameterList name="FMU_VAR_LOG">
       <!-- FMU vars to log to file at each time step -->
       <Parameter name="mflow_pumpprimary" type="bool" value="true"/>
       <Parameter name="mflow_secondary" type="bool" value="true"/>
       <Parameter name="T_in" type="bool" value="true"/>
       <Parameter name="T_out" type="bool" value="true"/>
    </ParameterList>
</ParameterList>
```

**Listing 1. XML coupling specification.** fmu_params.xml**.**

The XML coupling specification file contains the following sections.

- CASEID: serves as the root parameter list. Contains settings for the FMU coupling. Optionally, users may supply the relaxation factors used in the steady-state solve, the converge tolerances, and the maximum time step size of the FMU model.
  - Parameter **FMU_dt_max** (optional): maximum time step allowed in the TRANSFORM FMU

  in seconds. Default 0.1 s.

  - Parameter **ulax_T_corein** (optional): relaxation factor for inlet temperature in steady-state coupled runs. Default 1.0.
  - Parameter **ulax_mflow_corein** (optional): relaxation factor for inlet flow rate in steady-state coupled runs. Default 1.0.
  - Parameter **toltemp** (optional): temperature convergence tolerance for steady-state coupled runs in degrees Fahrenheit. Default 0.1°F.

- BC_VAR_NAMES: contains mappings from required CTF boundary conditions to FMU variable names. The mappings are given as < name = “**CTF_BC_NAME**”, value = “**FMU_BC_NAME**” > pairs.
  - Parameter **T_corein** (required): name of FMU variable corresponding to the core inlet temperature in Kelvin.
  - Parameter **T_coreout** (required): name of FMU variable corresponding to the core outlet temperature in Kelvin.
  - Parameter **P_corein** (required): name of FMU variable corresponding to the core inlet pressure in pascals.
  - Parameter **P_coreout** (required): name of FMU variable corresponding to the core outlet pressure in pascals.
  - Parameter **mflow_corein** (required): name of FMU variable corresponding to the core inlet mass flow rate in kilograms per second.
  - Parameter **mflow_coreout** (required): name of FMU variable corresponding to the core outlet mass flow rate in kilograms per second.
- FMU_VAR_INIT: contains initial values and parameter constants supplied to the FMU at initialization time. There are no required inputs in this block; however, it is highly recommended to set the TRANSFORM FMU initial conditions and parameter values here, otherwise the defaults specified at *FMU export* time will be used.
- FMU_VAR_TRANSIENT: contains transient FMU parameters that change as a function of time. Transient FMU parameters are provided as time, value pairs. The “**time**” parameter is required in this block. The table will be linearly interpolated between given points. This block is optional. This block is ignored in steady-state only calculations.
- FMU_VAR_LOG: contains FMU variable names, which are desired in the FMU coupling log file. The FMU variables specified here will be written to a comma-separated values file at each time step. The FMU exporter must ensure that these variable are exposed for reading through the FMI interface. This block is optional.

A standard CTF input file is used to construct the in-core model. Details of the CTF input deck format can be found in the CTF user's manual [36]. One modification is needed to specify where CTF should look for the FMU in the file system. The path at which the FMU TRANSFORM model is unzipped is given to CTF in CARD 1.5. An example CTF GROUP 1 input is shown in Listing 2 with the modification required to point CTF to the TRANSFORM FMU. This file location can be changed to suite user needs.

```
*************************************************************************************
*GROUP 1 - Calculation Variables and Initial Conditions
*************************************************************************************
**NGR
  1
**NGAS IRFC EDMD IMIX ISOL          GINIT NOTRN MESH MAPS IPRP MFLX IBTM PPV  NM14
     1    2    0    3    3    1.70995e+02     0    1    0    4    0    0   7     0
*Card 1.2
**         GTOT           AFLUX          DHFRAC
      170.99460        6.40020        0.99990
*Card 1.3
**         PREF             HIN           HGIN         VFRAC1         VFRAC2
        3.44738     -633.88889 288.4200000      1.0000000      0.9999000
*Card 1.4
**GTP(1)   VFRAC(3)  GTP(2) VFRAC(4)  GTP(3) VFRAC(5)  GTP(4) VFRAC(6)
     air     0.0001
*Card 1.5
     {fmu_unzip_directory}  "/home/user/FMUS/Transform/Transform_V3/FMU_Unzipdir"
*************************************************************************************
```

**Listing 2. CTF GROUP 1 input with FMU path specified.**

The input {**fmu_unzip_directory**} in CARD 1.5 is required for a coupled CTF-TRANSFORM run. The path following this input card must be enclosed in quotation marks. The FMU should be extracted into this directory before CTF is executed because the CTF code does not contain logic to extract the zip archive. This is to eliminate the dependence on the zip file format from CTF and Futility because this extraction operation only needs to be performed once.

## 3. RESULTS

### 3.1 STEADY-STATE MSRE COUPLED SIMULATION

The steady-state solution procedure detailed in Section 2.2 was exercised on a coupled CTF-TRANSFORM MSRE model. A variety of reactor powers and primary loop flow rates were considered to demonstrate that the behavior of the system met expectations (Eq. 2). The power and flow rates are set as inputs, and the resulting temperature changes were recorded. For the values from the coupled simulation to match the values of a similar stand-alone simulation, the fluid thermophysical properties also must match—namely, the specific heat capacity. In all steady-state cases shown here, coupling to the neutronic code, MPACT, was not considered. Instead, a simplified uniform axial power profile was adopted for use in the CTF core model. Nominal flow and power values were extracted from an ORNL report that details the steady-state operating conditions of the MSRE [6]. The steady-state heat balances for various flow rates and powers matched expectations for the coupled simulations. The absolute values of the temperatures depend on the specific experimental configuration and temperature boundary conditions for a given test.

$$\mathbf{\dot{Q} = \dot{m}c_p\Delta T}. \tag{2}$$

The steady-state calculation requires an initial guess for the core inlet temperature to begin the outer iteration loop shown in Section 2.2. The initial guess for the steady-state inlet temperature and mass flow rate for all steady-state calculations conducted in this section was 907 K and 171 kg/s, respectively. Additional TRANSFORM model-specific parameters are provided in Table 2.

**Table 2. CTF-TRANSFORM model constant parameters for the MSRE model used in the steady-state runs.**

| CTF-TRANSFORM parameter | Unit | Value |
|---|---|---|
| Secondary loop flow rate | kg/s | 105.745 |
| N channels | – | 569 |
| Channel area | $m^2$ | $2.875 \times 10^{-4}$ |
| Core height | $m^2$ | 2.032 |

**Table 3. Coupled CTF-TRANSFORM steady-state results for the MSRE model given different core powers and flow rates.**

| Case | Power (MW) | Primary flow (kg/s) | Predicted steady state $T_{in}$ (K) | Predicted steady state $T_{out}$ (K) | $\Delta T$ |
|---|---|---|---|---|---|
| 1 | 7.40 | 171.0 | 893.7 | 911.9 | 18.2 |
| 2 | 8.88 | 171.0 | 978.8 | 1,000.6 | 21.8 |
| 3 | 5.92 | 171.0 | 807.3 | 821.8 | 14.5 |
| 4 | 7.40 | 188.1 | 894.5 | 911.1 | 16.6 |
| 5 | 7.40 | 153.9 | 893.1 | 913.3 | 20.2 |
| 6 | 5.92 | 153.9 | 806.6 | 822.8 | 16.2 |
| 7 | 8.88 | 188.1 | 999.4 | 979.6 | 19.8 |

Neither the CTF core model nor the TRANSFORM model of the primary heat exchanger are calibrated to be precisely representative of the true MSRE facility. The models will require calibration for a given experimental configuration to match the absolute core temperatures for a given test based on the settings for the radiator door position (represented by heat transfer coefficients), boundary temperatures, and other parameters. However, the qualitative trends match specific experimental results. An increase in power resulted in the expected increase in the core inlet and exit temperatures. The predicted temperature change across the core at nominal conditions was 18.2 K. This value is near the value reported in the MSRE operational report, which states an inlet temperature of 907.0 K and a core outlet temperature of 924.8 K, giving a measured $\Delta T = 17.8K$ at nominal operating conditions [6]. This is within the uncertainty of the instrumentation used to measure the temperature during operation. Future work will incorporate the system configuration settings and temperature boundary conditions of a given test to allow for a more direct comparison of the model predictions and the available MSRE historical data.

In the case of nominal MSRE operating conditions, 114 outer iterations were required to achieve a steady-state result within the default tolerance of 0.1°F change in the inlet temperature between iterations. To improve computation times, in future work, the pseudo-transient CTF solve could be replaced with a single steady-state CTF solution update. Additional improvements to the convergence characteristics of the coupled CTF-TRANSFORM codes could be possible in a future study because the current relatively simple picard iteration scheme is employed to solve the coupled problem.

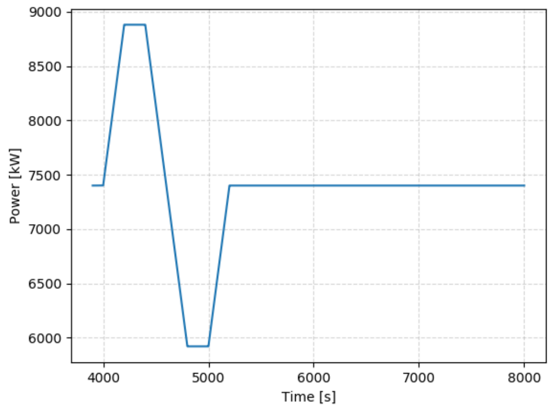
## 3.2 TRANSIENT MSRE

Three transient cases were examined. The first case simulated a slow power ramp to initially test the transient CTF-TRANSFORM capability. This test included rising and falling power profiles to check that the time-dependent system response to mirrored power ramps resulted in symmetric temperature responses. The second case considered fast power pulses with a pseudorandom binary signal to demonstrate that the coupled model is robust to fast changes in the core power and core outlet temperature. Finally, a secondary flow loop transient with a corresponding power reduction due to a resulting SCRAM event was investigated. This final transient illustrates the ability to specify time-varying secondary loop conditions in the coupled model, which would be impossible using a standard stand-alone CTF calculation. This capability also enhances the ability to study the propagation of uncertainties in the external systems to the resulting performance in the detailed representation of the reactor core.

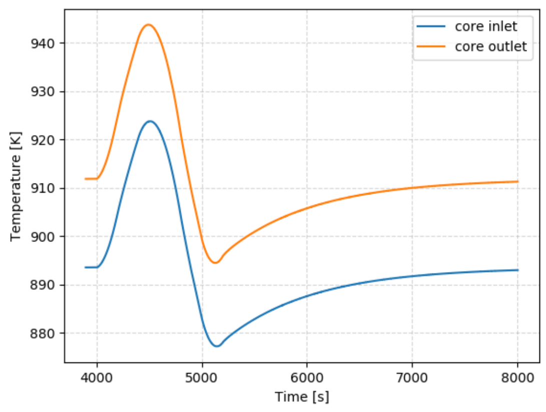
Neutronic coupling was not considered in any of the transient cases, and prescribed axially uniform reactor core powers were used. The transients were initiated after a lead-in period of 4,000 s of simulated reactor time at nominal power and flow conditions to ensure that the system model had reached thermal equilibrium. The MSRE conditions given in Table 2 were used to initialize the model and to run the first 4,000 s lead-in transient.

### 3.2.1 Power Ramp Transient

This case demonstrated the initial CTF-TRANSFORM coupling capability and the overall MSRE system response to a 200 s ramp-up in core power to 8.88 MWth, followed by a 400 s ramp-down in power to 5.92 MWth. Finally, the core was returned to nominal 7.4 MWth operating conditions, and the core inlet and outlet temperatures were allowed to return to their equilibrium values. A second case was executed with a mirrored version of this power ramp transient. This second case was run to examine the temperature response of the system for expected symmetries.



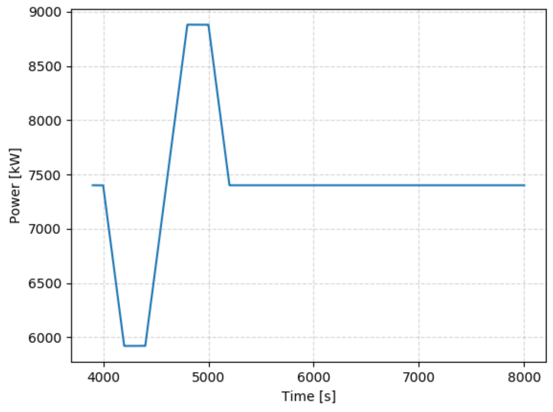
**(a) Power vs. time.**



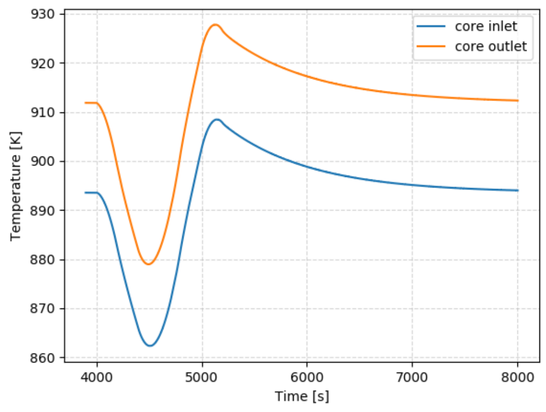
**(b) Core inlet and outlet temperatures.**

**Figure 8. Coupled CTF-TRANSFORM results for a ramped power transient.**

Next, the mirrored power transient was performed.



**(a) Power vs. time.**



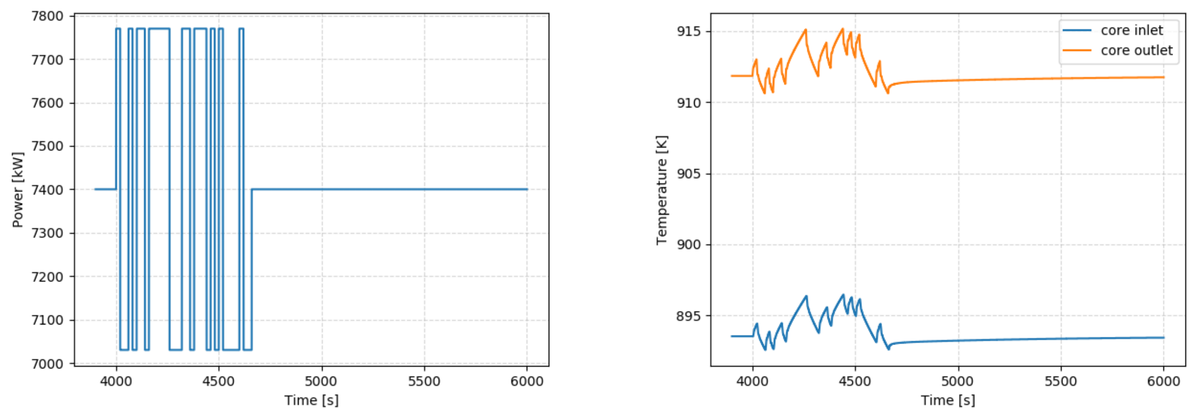
**(b) Core inlet and outlet temperatures.**

**Figure 9. Coupled CTF-TRANSFORM results for a mirrored ramped power transient.**

The results shown in Figures 8 and 9 from the mirrored power transient cases exhibit the expected symmetric time-dependent temperature response. Since the CTF core model is simplified and the TRANSFORM heat transfer and heat loss parameters are not calibrated, the authors do not expect to match the exact inlet and outlet temperatures of the true MSRE facility. This case only was used to test the coupling and demonstrate the qualitative behavior of the coupled models.

### 3.2.2 Pulsed Power Transient

A pseudorandom binary input sequence was used to demonstrate the robustness of the coupling to fast changes in the reactor core power. The input sequence was a 31 bit pseudorandom-binary sequence with a bit-size of 20 s and an amplitude of 5% nominal full power. The perturbation in the reactor power results in perturbations in the reactor outlet temperature that then travel from the core outlet back to the core inlet. The resulting pulses in fluid temperature indicate mixing and residence times for the primary loop and secondary flow circuit, as well as the transient heat transfer on the timescales of the tested input. On this timescale, the results indicated a tight coupling between reactor power and primary fluid temperature, which is expected with a fluid-fueled reactor design. After completing the input sequence, the system returned to the same steady state, indicating that no temperature drift occurred in the coupling. In this case, the TRANSFORM FMU was sub-stepped with a **FMU_dt_max** = 0.02 s.



**(a) Power vs. time.** **(b) Core inlet and outlet temperatures.**

**Figure 10. Coupled CTF-TRANSFORM results for a pulsed power transient.**

### 3.2.3 Secondary Flow and Power Coastdown Transient

This case investigated a reduction in the secondary loop flow rate. This case demonstrates the ability to use the coupled TRANSFORM model to simulate changes in the secondary—and potentially tertiary—loop parameters as a function of time. This showcases a scenario that the CTF core model could not simulate alone.

A ramp-down in the secondary loop flow from the nominal value of 105.475 kg/s to a low value of 5 kg/s over an approximately 10 s period was considered. The pump coast-down curve was extracted from an MSRE ORNL technical report that details transient operating characteristics of the experimental facility [37]. A rapid power reduction coincided with the loss of secondary flow to simulate a SCRAM event. The power reduction curve was extracted from an MSRE ORNL technical report [38].

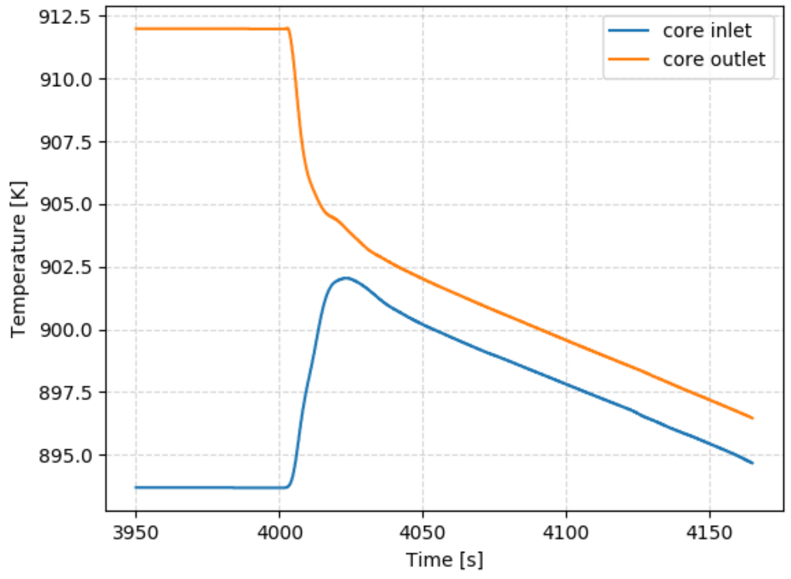
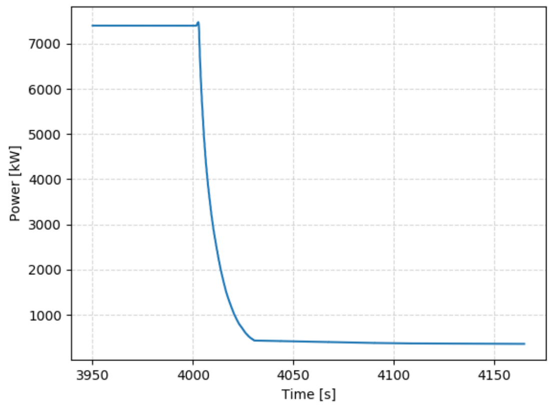
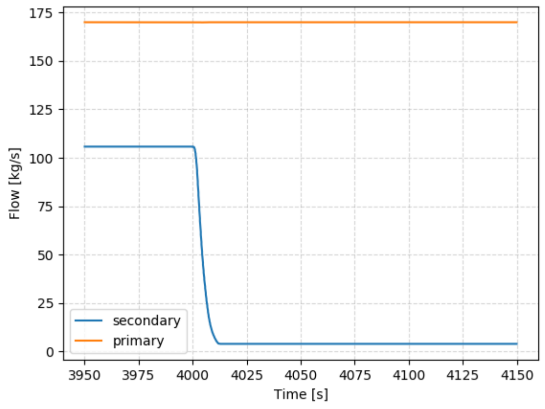


**Figure 12. Coupled CTF-TRANSFORM core temperature results for a secondary flow loss and power coastdown.**



**(a) Power vs. time ramp-down, approximating a SCRAM event.**



**(b) Primary and secondary flow rates vs. time, approximating loss of secondary pump.**

**Figure 11. Coupled CTF-TRANSFORM inputs for a secondary flow loss and power coastdown. Initial secondary flow: 105.745 kg/s. Low secondary flow: 5 kg/s.**

## 4. FUTURE WORK

Future work will focus on performing model calibration, validation, system sensitivity studies, UQ, and reduced-order model construction using the newly developed VERA/CTF-TRANSFORM coupling capability paired with the RAVEN package. Future work avenues are available to improve the steady-state convergence and thus reduce overall run times of the coupled model by (1) replacing the pseudo-transient CTF solve with a steady-state CTF solve or (2) moving from a Picard iteration to an Anderson acceleration fixed-point scheme in the steady-state solution procedure. The transient solution procedure could be improved by investigating an implicit time marching scheme in which the time derivative of the coupled state variables is computed at the end of a given time step and used to project the solution forward in time. It is unclear at this time whether the larger time steps made possible by moving to a backward Euler time-stepping scheme would outweigh the computational burden of performing newton iterations required in the implicit step. Additionally, future work targeted at allowing multiple FMUs to be coupled simultaneously with CTF is feasible and would allow for greater flexibility in handling Model Exchange and Co-Simulation FMUs.

Improvements specific to TRANSFORM and the TRANSFORM MSRE model are also possible and may be done in parallel to work improving in the coupling strategy. Updating the inclusion of minor flow losses and form losses in the pipe components of the MSRE model will allow for the modeling of buoyancy-driven flow in the primary and secondary loops. Next, to implement an implicit stepping scheme, such as backward Euler, the Co-Simulation FMU must support solution rewind, which could be implemented in future work. The Fortran FMU wrapper already supports the requisite FMI methods to perform a rewind of the FMU state, so minimal change to the coupling would be required. Finally, TRANSFORM could be extended to PWR and boiling water reactors (BWRs) if the appropriate two-phase models are implemented for the steam generators and condensers, respectively, to provide similar coupled simulation capabilities for the study of LWRs.

### 4.1 RAVEN-DRIVEN UQ OF CTF-TRANSFORM SYSTEM MODELS

RAVEN is a collection of Python modules for performing statistical inference, surrogate model construction, and sensitivity and UQ studies developed at Idaho National Laboratory [39]. Starting with a sensitivity study, leveraging RAVEN will be a scripting exercise that involves perturbing the parameters of interest and observing the system's response in terms of output variables of interest, such as the core inlet temperature, as a function of the aforementioned input parameters. The main idea is to ensure that CTF-TRANSFORM exposes the parameters of interest through a user-facing input deck format that can be generated easily with a Python script. This is accomplished through the CTF-TRANSFORM XML parameter coupling file specified in Section 2.4.

The Bayesian inference capabilities of RAVEN [39] or other packages (e.g., BiPyMc [40], Emcee [41], Dakota [42]) may be used to perform model calibration, given experimental data from the MSRE, such as measured loop temperatures, to determine the optimal model parameters for heat transfer, insulation, and pump flow rates. Additionally, Markov chain Monte Carlo (MCMC) sampling methods implemented in RAVEN or other open-source Python tools may be used to derive optimal values and uncertainty distributions of CTF-TRANSFORM model parameters given the available MSRE loop experimental data.

Finally, the surrogate construction capabilities of RAVEN could be leveraged to create reduced-order models of a system response of interest with uncertainty intervals. The polynomial chaos expansion implementations in RAVEN will be a plausible starting point for performing basic transient system response projections with corresponding uncertainty intervals. These methods first require that the uncertainties associated with the model parameters are known a priori so that they can be propagated to the system response of interest. If a prior MCMC computation was performed, then these should be known and can be used to generate a polynomial chaos expansion model of the time-dependent system output variable of interest.

### 4.2 OVERLAPPING DOMAIN COUPLING

The current CTF-TRANSFORM coupling employs boundary condition exchange at the core inlet and outlet, as shown in Figure 5; however, this is not the only plausible configuration for coupling CTF to a systems code. A domain overlapping scheme could be tested in the future as an alternative to the boundary interface coupling strategy. In domain overlapped coupling, instead of inlet/outlet information being exchanged between the systems code and core model, the integral power and head loss over the core are passed to the systems code. In a domain overlapping coupling, TRANSFORM would have its own internal, simplified core model. In this mode, the systems code still supplies the inlet core temperature and mass flow rate to CTF. Effectively, TRANSFORM simulates the primary loop in its entirety, including the core, and CTF still remains responsible for high-fidelity predictions of in-core phenomena.

### 4.3 MULTIPLE FMU IMPORT INTO VERA/CTF

The current coupling strategy only allows for one FMU to be loaded and coupled with CTF at a time. This coupling can be extended to allow an arbitrary number of FMUs to be loaded and coupled to CTF. Furthermore, the current coupling implementation only supports Co-Simulation FMUs. In the future, it is possible to add a Model-Exchange FMU wrapper that leverages Futility's built-in ODE solvers to evolve a system of externally supplied Model Exchange FMUs through time.

## 5. CONCLUSION

The newly developed CTF-TRANSFORM coupling capability allows the study of coupled system behavior in the reactor core and the supporting primary, secondary, and tertiary loop components. The fidelity of the in-core physics afforded by CTF and MPACT, provided by VERA, is retained with the added benefit of obtaining mechanistically derived, realistic flow rates and temperature drops across the primary heat exchanger due to the TRANSFORM model. In contrast, a standard VERA simulation only concerns the reactor core itself. Rather than using assumed inlet temperature and flow values as is typically done in a stand-alone core simulation, TRANSFORM provides the inlet conditions from a mechanistic model of the full primary and secondary loops with this new capability. Initial opportunities to leverage this work are straightforward. Through this VERA-CTF-TRANSFORM system code coupling, numerical experiments can be performed by quantifying the impacts of primary, secondary, and tertiary loop design or operating parameters on the core. Conversely, the response of external core components can be studied as a function of changes in the core design parameters or power operation conditions. Since TRANSFORM is modular in nature, this coupling can be extended to many reactor types. Possible paths for future development to compose system models of MSR, PWR, and BWR plants are feasible.

This report demonstrates the coupling between CTF and TRANSFORM operating in a steady-state mode and a transient mode for an MSRE model. The steady-state results showed good agreement with the expected qualitative behavior. The temperature change across the core was in line with the measured value, decreases in the primary flow rate lead to increases in the core temperature change, and increases in the power produced the expected increase in the primary loop temperatures. Disagreement with the experimental MSRE historical data for the inlet temperature can be addressed by performing a primary and secondary loop model calibration in follow-on work. The developed coupling capability is well-positioned to interface with external tools, such as RAVEN, to perform model calibration tasks due to a relatively simple XML file parameter specification format.

The transient coupling was exercised for two power-transient scenarios and—finally—a secondary loop flow rate transient. The fast-pulse transient case showed that the coupling method is robust to fast changes in the core power, largely due to the dynamic time step routines already in place inside CTF but also due to the ability to sub-step the TRANSFORM model. The secondary loop flow rate transient demonstrated an important capability developed in this work because this coupling can be used to perform a transient simulation of secondary and tertiary flow loop conditions. This would otherwise be impossible in a stand-alone VERA/CTF model because these components are not considered in a typical VERA calculation.

## 6. ACKNOWLEDGEMENT

This research is supported by the US Department of Energy, Office of Nuclear Energy, Crosscutting Technology Development Integrated Energy Systems program.

# 7. REFERENCES

## References

[1] T. Blochwitz, A. Junghanns, J. Kohler, and M. Krammer. Overview on FMI, SPP, DCP. *13th International Modelica Conference, Regensburg*, 2019.

[2] Robert A. Lefebvre, Brandon R. Langley, Jordan P. Levebvre, and Adam B. Thompson. NEAMS workbench.

[3] Paul LeMar. Integrated Energy Systems(IES) for Buildings: A Market Assessment, September 2002. ORNL/SUB/409200.

[4] T. Blochwitz, Martin Otter, Johan Akesson, Martin Arnold, C. ClauB, Hilding Elmqvist, Markus Friedrich, Andreas Junghanns, Jakob Mauss, Dietmar Neumerkel, Hans Olsson, and Antoine Viel. Functional mockup interface 2.0: The standard for tool independent exchange of simulation models. *9th international Modelica Conference*, Sep 2012.

[5] Modelica Association. Functional mockup interface tools, 2020.

[6] R. H. Guymon. MSRE systems and components performance. April 1973. ORNL-TM-3039.

[7] J. Turner, K. Clarno, B. Bartlett, B. Collins, R. Pawlowski, R. Schmidt, and R. Summers. The Virtual Environment for Reactor Applications (VERA): Design and architecture. *Journal of Computational Physics*, 326, Dec 2016.

[8] B. Collins, B. Kochunas, S. Stimpson, and et. al. MPACT Theory Manual, Nov 2019. CASL-U-2019-1874-001.

[9] B. Collins, B. Kochunas, A. Graham, T. Downar, and A. Godfrey. Stability and Accuracy of 3D Neutron Transport Simulations using the 2D/1D method in MPACT. *Journal of Computational Physics*, 326, Jan 2016.

[10] B. Rearden, M. Jessee, and Eds. SCALE Code System, 2018. Avail. from RSICC as CCC-834. ORNL/TM-2005/39.

[11] R. Salko, M. Avramova, A Wysoki, A. Toptan, J. Hu, N. Porter, T. Blyth, C. Dances, A. Gomez, C. Jernigan, J. Kelly, and A. Abarca. CTF Theory Manual, Nov 2019. CASL-U-2019-1886-001.

[12] B. Collins, J. Galloway, R. Salko, K. Clarno, A. Wysocki, B. Okhuysen, and D. Anderson. Whole Core CRUD-Induced Power Shift Simulations Using VERA. *PHYSOR 2018*, April 2018.

[13] Daniel J. Kelly, Ann E. Kelly, Brian N. Aviles, Andrew T. Godfrey, Robert K. Salko, and Benjamin S. Collins. MC21/CTF and VERA multiphysics solutions to VERA core physics benchmark progression problems 6 and 7. *Nuclear Engineering and Technology*, 49(6):1326 – 1338, 2017. Special Issue on International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering 2017 (MC 2017).

[14] Jessee Matthew, William Wieselquist, M.L. Williams, and Kang Seog Kim. Vera benchmark calculations using the scale-polaris lattice physics code. *Transactions of the American Nuclear Society*, 109:1413–1415, 01 2013.

[15] K. Kim, D. Lee, H. Shim, and A. Pawel. VERA Depletion Benchmarks By CASL VERA, SERPANT and MCCARD with ENDF/B-VII.0 and VII.1. April 2018. Proceeding of the PHYSOR 2018, Cancun, Mexico.

[16] Robert K. Salko, Taylor S. Blyth, Christopher A. Dances, Jeffrey W. Magedanz, Caleb Jernigan, Joeseph Kelly, Aysenur Toptan, Marcus Gergar, Chris Gosdin, Maria Avramova, Scott Palmtag, and Jess C. Gehin. CTF Validation and Verification Manual. May 2016. CASL-U-2016-1113-000.

[17] V. Mousseau and N. Dinh. CASL Verification and Validation Plan. June 2016. CASL-U-2016-1116-000.

[18] Andrew T. Godfrey, Benjamin S. Collins, Cole A. Gentry, Shane G. Stimpson, and John A. Ritchie. Watts Bar Unit 2 Startup Results with VERA, March 2017. CASL-U-2017-1306-000,ORNL/TM-2017/194.

[19] A. Graham, B. Collins., R. Salko, Z. Taylor, and C. Gentry. Development of Molten Salt Reactor Modeling and Simulation Capabilites in VERA. *Proceedings of Top Fuel 2019*, Sep 2019.

[20] Vefa Kucukboyaci, Brendan Kochunas, Thomas Downar, Aaron Wysocki, and Robert Salko. Evaluation of VERA-CS Transient Capability For Analyzing the AP1000®Reactor Control Rod Ejection Accident. *PHYSOR 2018*, April 2018.

[21] Modelica Association. Functional Mock-up Interface for Model Exchange and Co-Simulation, Oct 2019.

[22] Michael S. Greenwood, Richard Hale, Lou Qualls, Sacit Cetiner, David Fugate, Thomas Harrison, and USDOE. Transform - transient simulation framework of reconfigurable models, Sep 2017.

[23] Modelica Association. Modelica language. `https://modelica.org/modelicalanguage`. Accessed: 2020-12-16.

[24] Michael Scott Greenwood, Benjamin R. Betzler, A. Lou Qualls, Junsoo Yoo, and Cristian Rabiti. Demonstration of the advanced dynamic system modeling tool transform in a molten salt reactor application via a model of the molten salt demonstration reactor. *Nuclear Technology*, 206(3), July 2019.

[25] Michael Scott Greenwood and Dane De Wet. Status report on the msre transform model for thermal-hydraulic benchmarking. Sep 2019.

[26] Jordan D. Rader, Michael Scott Greenwood, and Paul W. Humrickhouse. Verification of modelica-based models with analytical solutions for tritium diffusion. *Nuclear Technology*, 203(1), March 2018.

[27] Michael Scott Greenwood and Benjamin R. Betzler. Modified point-kinetics model for neutron precursors and fission product behavior for fluid-fueled molten salt reactors. *Nuclear Science and Engineering*, 193(4), Nov 2018.

[28] Michael Scott Greenwood and Jake W. Mcmurray. Prototype demonstration of an integration of a gibbs energy minimizer with transform for molten salt reactor mass accountancy studies. Sep 2020.

[29] Michael Scott Greenwood, Askin Guler Yigitoglu, Jordan D. Rader, Whitney Tharp, Willis Poore III,

Randy Belles, Bob Zhang, Riley Cumberland, and Michael Muhlheim. Integrated energy system investigation for the eastman chemical company, kingsport, tennessee facility. May 2020.

[30] Michael Scott Greenwood, Askin Guler Yigitoglu, and Thomas J. Harrison. Nuclear hybrid energy systems south east regional case progress report. Oct 2018.

[31] Michael Scott Greenwood, Sacit M. Cetiner, Thomas J. Harrison, and David Fugate. A templated approach for multi-physics modeling of hybrid energy systems in modelica. Aug 2017.

[32] A Wysoki, K. Borowiec, and R. Salko. Coupling Interface to Systems Code for Transient Analysis, Sep 2019. CASL-U-2019-1909-000.

[33] Dassault Systemes. FMPy, 2020. [Dev: T. Sommer].

[34] Dassault Systemes. FMIKit-Simulink, 2020. [Dev: T. Sommer].

[35] Futility Development Group. Futility: Fortran utility, 2020. [cd393723, 12-01-2020].

[36] R. Salko, M. Avramova, A Wysoki, A. Toptan, J. Hu, N. Porter, T. Blyth, C. Dances, A. Gomez, C. Jernigan, and J. Kelly. CTF User's Manual, Nov 2019. CASL-U-2019-1885-001.

[37] B. E. Prince, S. J. Ball, J. R. Engel, P. N. Haubenreich, and T. W. Kerlin. Zero-power physics experiments on the molten-salt reactor experiment. Jan 1968. ORNL-4233.

[38] P N Haubenreich, J R Engel, B E Prince, and H C Claiborne. MSRE Design and Operations Report. Part III. Nuclear Analysis. Feb 1964. ORNL-TM-730.

[39] C. Rabiti, A. Alfonsi, J. Cogliati, D. Mandelli, R. Kinoshita, S. Sen, and C. Wang. RAVEN User Manual, 2016. INL/EXT-1534123 rev.4.

[40] W. Gurecky. BiPyMc: Bayesian Inference for PYthon using Markov Chain Monte Carlo, 2018. [065ef0e, 9-01-2020].

[41] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman. emcee: The MCMC Hammer. *PASP*, 125:306–312, 2013.

[42] B. Adams and et. al. Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification and Sensitivity Analysis: Version 6.13 User's Manual, Nov 2020. SAND2020-12495.

# APPENDIX A. FUTILITY FMU WRAPPER

## FUTILITY FMU WRAPPER

The FMU wrapper developed in the Futility open-source package is available online at `https://www.github.com/CASL/Futility`. The wrapper enables users to load and interact with Co-Simulation FMUs that conform to the FMI version 2 standard from a Fortran program. To demonstrate the general utility of the wrapper, a third-party, open-source, BSD 2-Clause licensed, pre-compiled Co-Simulation FMU compiled for the Linux operating system was downloaded from the web at:

`https://github.com/modelica/fmi-cross-check/raw/master/fmus/2.0/cs/linux64/Test-FMUs/0.0.2/BouncingBall`.

The FMI cross-check repository supplies many FMUs that conform the the FMI version 2 standard, which could be used for testing. Pre-compiled FMUs, from Dymola or elsewhere, should be built for the target system of interest. Ideally, the FMU would always be distributed as raw C code so that it can be compiled on any machine to enable true FMU portability. However, the standard TRANSFORM constituent models are expressed in the Modelica scripting language and must first be translated by compatible software. This can lead to portability issues.

An example use of the Futility FMU wrapper to drive a bouncing ball simulation is given in Listing A-1. All important lines are commented. The ball is dropped from an initial height of 1 m with zero starting velocity. The coefficient of restitution was set to 0.7 in this case. Air resistance is ignored.

```
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++!
!                          Futility Development Group                        !
!                            All rights reserved.                            !
!                                                                            !
! Futility is a jointly-maintained, open-source project between the University !
! of Michigan and Oak Ridge National Laboratory.  The copyright and license    !
! can be found in LICENSE.txt in the head directory of this repository.        !
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++!
!> @brief Program to demo basic FMU model interaction
!>
!> To use, Download the example third party FMU from the fmi-cross-check repo
!>
PROGRAM testFMU2_BouncingBall
  USE Strings
  USE IntrType
  USE ParameterLists
  USE FMU_Wrapper
  IMPLICIT NONE

  ! FMU CS Wrapper
  TYPE(FMU2_Slave) :: test_fmu2_cs
  ! FMU Initilization parameterlist
  TYPE(ParamType) :: FMU_params
  ! Path to FMU unzip directory
  CHARACTER(len=256) :: unzipDirectory
  ! ID of fmu used for bookeeping
  INTEGER(SIK) :: fmu_id=1_SIK
  ! Time step size
  REAL(SRK) :: dt=1.0E-3_SRK
  ! Start and end of simulation time
  REAL(SRK) :: timeStart=0.0_SRK
  REAL(SRK) :: timeEnd=2.0_SRK
```

```
! FMU ODE solver tolerance
REAL(SRK) :: tol=1.0E-9_SRK
! Local time storage
REAL(SRK) :: time=0.0_SRK
! Variable to store ball height and velocity
REAL(SRK) :: ball_velocity, ball_height
! Iteration counter
INTEGER(SIK) :: i
! Temporary variable name storage
TYPE(StringType) :: varName

! Example FMU parameter settings
CALL FMU_params%clear()
CALL FMU_params%add('FMU_Wrapper->id',fmu_id)
unzipDirectory='/home/user/exampleFMU/reference_fmu_bouncing_ball'
CALL FMU_params%add('FMU_Wrapper->unzipDirectory', trim(unzipDirectory))

! Initilize the FMU
CALL test_fmu2_cs%init(fmu_id, FMU_params)
CALL test_fmu2_cs%setupExperiment(.TRUE., tol, timeStart, .TRUE., timeEnd)

! Ensure that desired variables are present in the FMU
varName='g'
IF(test_fmu2_cs%isXmlVar(varName)) &
    WRITE(*,*) "FMU variable: ", CHAR(varName), " has causality: ", &
    CHAR(test_fmu2_cs%getCausality(varName))
varName='e'
IF(test_fmu2_cs%isXmlVar(varName)) &
    WRITE(*,*) "FMU variable: ", CHAR(varName), " has causality: ", &
    CHAR(test_fmu2_cs%getCausality(varName))
varName='h'
IF(test_fmu2_cs%isXmlVar(varName)) &
    WRITE(*,*) "FMU variable: ", CHAR(varName), " has causality: ", &
    CHAR(test_fmu2_cs%getCausality(varName))

! set gravity acceleration parameter
CALL test_fmu2_cs%setNamedVariable(StringType('g'), -9.81_SRK)
! set the coefficient of restitution
CALL test_fmu2_cs%setNamedVariable(StringType('e'), 0.7_SRK)

! Set restart point
CALL test_fmu2_cs%setRestart()

! Write output header
WRITE(*,*) "time[s]    height[m]    velocity[m/s]"

! Step the CS FMU model forward in time
DO
  ! Get the variable values of interest from the FMU
  CALL test_fmu2_cs%getNamedVariable(StringType('v'), ball_velocity)
  CALL test_fmu2_cs%getNamedVariable(StringType('h'), ball_height)
  ! Print the result to stdout
  write(*,*) time, ball_height, ball_velocity
  time = time + dt
  IF(time >= timeEnd) EXIT
  ! Step the FMU forward
```

```
    CALL test_fmu2_cs%doStep(dt)
  ENDDO

  ! Clean up
  CALL test_fmu2_cs%clear()
  CALL FMU_params%clear()

ENDPROGRAM testFMU2_BouncingBall
```

**Listing A-1. Fortran FMU wrapper demonstration.**

The resulting predicted bouncing ball trajectory is shown in Figure A-1. The results are comparable with those obtained from similar FMU wrappers, such as those found in MATLAB Simulink (`www.mathworks.com/help/simulink/slref/importing-a-model-exchange-fmu-into-simulink.html`). Figure A-2 shows the predicted bouncing ball velocity using the Futility FMU wrapper.

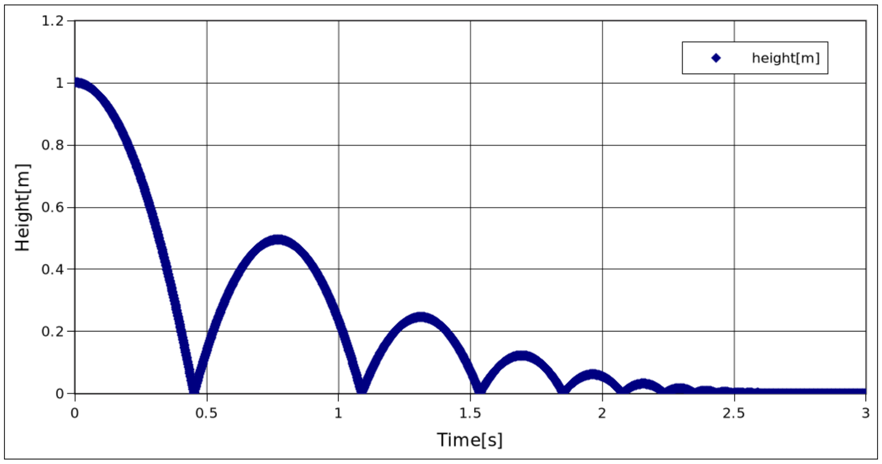


**Figure A-1. Predicted bouncing ball height using the Futility FMU wrapper for a Co-Simulation third-party FMU.**

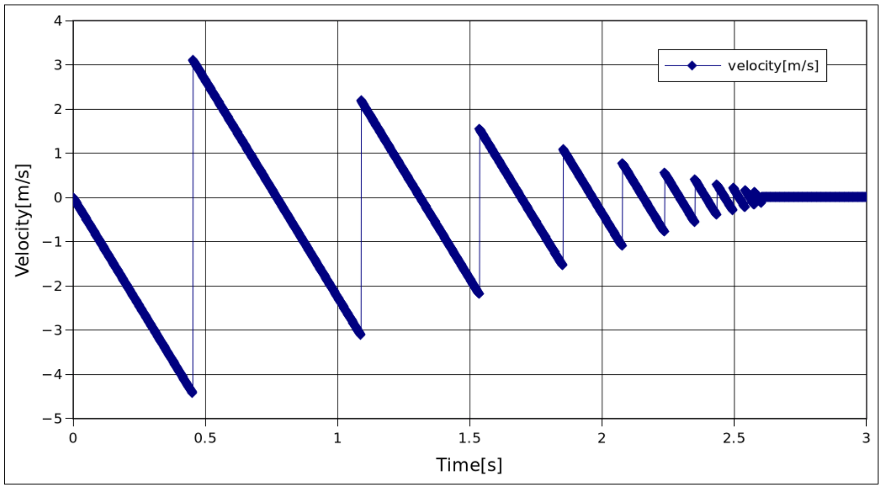


**Figure A-2. Predicted bouncing ball velocity using the Futility FMU wrapper.**

The API for the Co-Simulation FMU wrapper is available at `https://github.com/CASL/Futility`. The FMU generation tool must support solution rewind provided by the Futility FMU wrapper subroutines setRestart() and rewindToRestart(). Special care is required to ensure that the FMU correctly supports the ability to save and reload the system state if the solution rewind capability is desired. Additional information on the use of the FMU wrapper is provided in the example and testing directories of the Futility repository.