

BRISC (Neutronics) Development: Purpose, Plan, Progress, and Opportunities



Kevin Clarno

clarnokt@ornl.gov

Reactor Analysis Group
of the
Nuclear Science and Technology Division

R. Schmidt, L. Humphries, et al

Sandia National Laboratories



OSU-UM-TAMU VHTR Project Review

**College Station, TX
February 24, 2009**

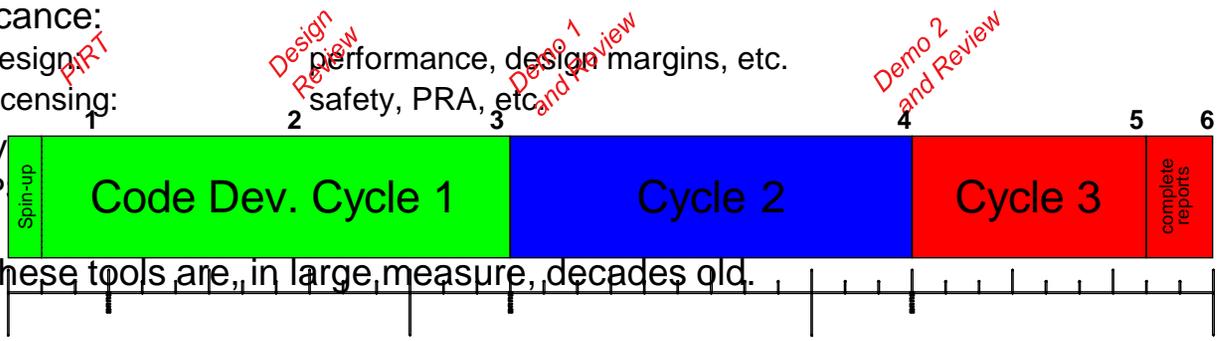
Clarification...

- This presentation is adapted,
 - ✦ with permission from the author (Rod Schmidt),
 - ✦ from a presentation made on Jan. 12-14, 2009 in Idaho Falls, ID
 - ◆ at the INL Workshop on Verification and Validation, Sensitivity Analysis, and Uncertainty Quantification of Next Generation System Safety Analysis Codes
 - ✦ titled, “BRISC Development: Insights and Experience”
 - ✦ which was approved at SNL

- However,
 - ✦ this presentation has not been officially reviewed by the Sandia National Laboratories and
 - ✦ may not accurately reflect their views,
 - ✦ thus should be considered accordingly.

History and Background

- FY2007-2009 Sandia LDRD Project
 - ✦ “Foundational Development of an Advanced Burner Reactor Integrated Safety Code”
- Development Strategy
 - ✦ Three revisions to progressively improve.
- Integrated Performance and Safety Codes
 - ✦ Function:
 - ◆ Treat all important system components and physical processes
 - ◆ During normal, off-normal or hypothetical accident scenarios.
 - ✦ Significance:
 - ◆ Design performance, design margins, etc.
 - ◆ Licensing safety, PRA, etc.
 - ✦ History
 - ◆ IP
 - ◆ These tools are, in large measure, decades old.
 - ✦ Need:
 - ◆ New and better tools are needed, for new reactor types and designs
 - ◆ We can leverage advanced computational tools and techniques
- Why am I here today talking about this?
 - ✦ You (TAMU-UM-OSU, NRC, DOE) may find things we’re doing that you can leverage
 - ✦ We (SNL-ORNL) would love to get your feedback on our vision/path



Project Synopsis

Primary Objective: To develop and demonstrate foundational aspects of a next-generation nuclear reactor safety code (BRISC) that leverages advanced computational technology.

Target Reactor System:
Liquid Sodium Fast Reactor

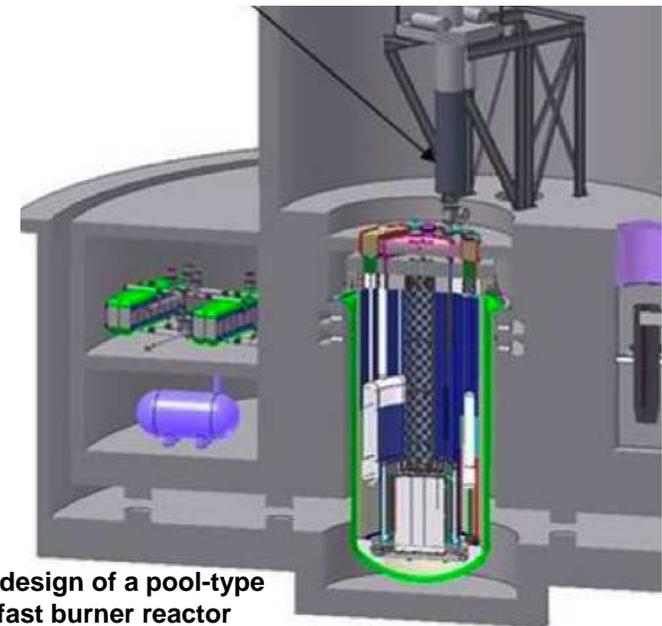
Modest Funding:
~2.5 FTEs for 3 years (FY07-09)

Diverse Team:

3 Institutions: SNL, ORNL, SUNYSB

Many Departments within Sandia: 1400, 1500, 2900, 6700, 8300

Constraint: This project is not intended to produce a complete, validated code that is ready-to-use by the DOE or NRC.



Pre-conceptual design of a pool-type sodium-cooled fast burner reactor

What is “advanced computational technology” and how can it help?

- **Modern software engineering**
 - ✦ Incorporate software from independent developers and languages
 - ✦ Allow efficient portability to many platforms
 - ✦ Efficiently utilize 2009-era moderate-sized clusters
 - ◆ Small clusters today, will be desktops in 7 years
 - ◆ Leadership-class hardware will be moderate in 7 years
- **Advanced computational ‘tools’**
 - ✦ Mathematics, visualization, CAD, meshing, ...
 - ✦ ASC-developed ‘physics’ solvers
- **Improved resolution**
 - ✦ 3D (where necessary) and 1D (when reasonable)
 - ✦ Less ‘effective’ flow and heat transfer
 - ✦ Higher resolution neutronics
- **Leveraging existing software (MELCOR, SCALE)**
 - ✦ Extensive data, quality assurance, and validation
 - ✦ Sufficiently accurate in many situations

Modeling Scope was limited to the Physics of the “Initial Transient” Phase of an Accident*

➤ Physics

✦ Fluid Flow and Heat Transfer

- ◆ 1D and 3D conduction
- ◆ Single phase turbulent flow and convective heat transfer
 - 3D in the vessel
 - 1D in secondary loop

✦ Neutronics

✦ Thermal Mechanics

✦ Material Properties and Equation of State

➤ Reactor System Components

- ✦ All important components of the reactor vessel and secondary coolant loop.

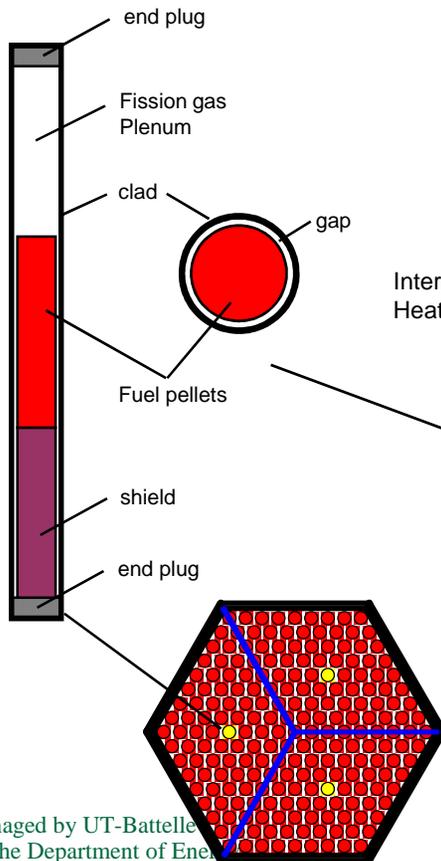
* Severe accident physics is significantly more difficult to address and typically requires more “engineering” or “phenomenological” type modeling approaches.

3-Tiered Multi-Scale Modeling Strategy

Structures and physics whose features are too small for resolution on 3D grid

Fuel-pins and control rods

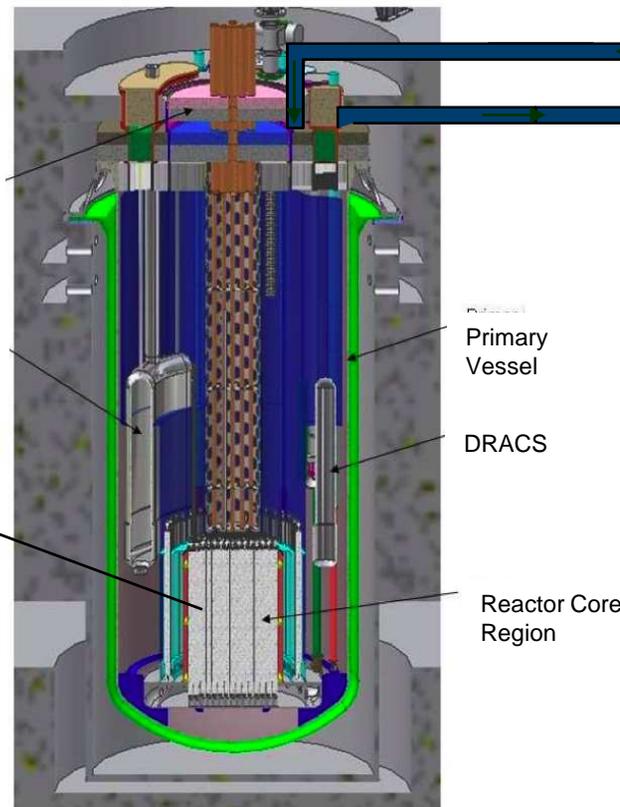
- 0.5 - 10 mm-scale features
- conduction, fission heating ...
- 2D or 3D representative models



"Meso-scale" resolved by 3D grid

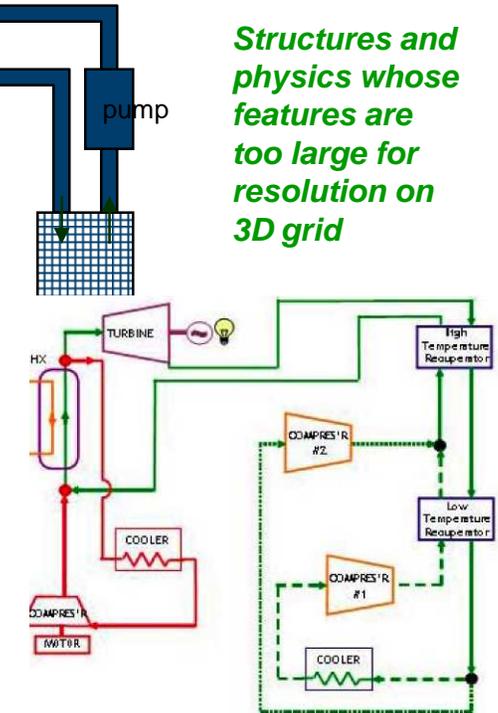
In-vessel Reactor Components

- 10 cm to 10 m scale geomtry
- Neutronics, Turb flow & heat transfer, thermal-mechanics, conduction, ...
- 3D Modeling Framework



Balance of Plant Reactor System Components (& Containment)

- 1 - 50 m scale
- Pipes, pumps, valves, heat exchangers, turbines, rooms,
- 0D MELCOR models
- 3D Fire Modeling with RIO



Structures and physics whose features are too large for resolution on 3D grid

How can/should we couple the codes?

➤ Non-intrusive/black box

- ✦ Communication through file I/O
- ✦ Codes compiled as executables
- ✦ Code developed and maintained independently
- ✦ Con:
 - ◆ File I/O will severely restrict parallelization
 - ◆ Maintaining consistency is difficult
 - ◆ Restricted to simple computational math

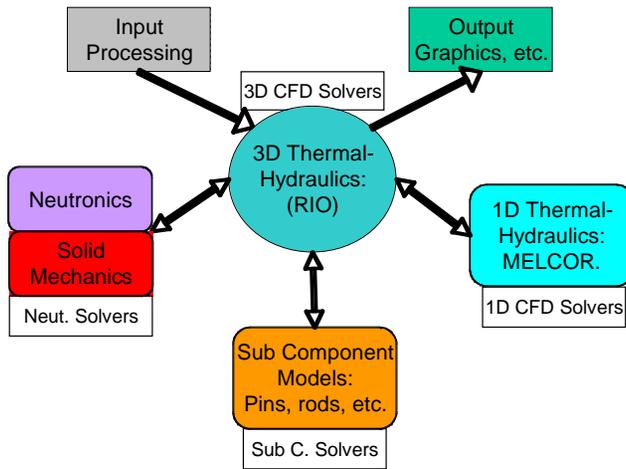
➤ Semi-intrusive

- ✦ Single-driver code that can link with others at compile-time
- ✦ Communication through memory
- ✦ Code relatively easy to develop and maintain independently
- ✦ Allows use of existing codes
- ✦ Con:
 - ◆ Modifications to “interface” with the “driver”
 - ◆ Codes must cleanup after themselves

➤ Fully-intrusive

- ✦ A no-go for this project as it will require nearly discarding existing code

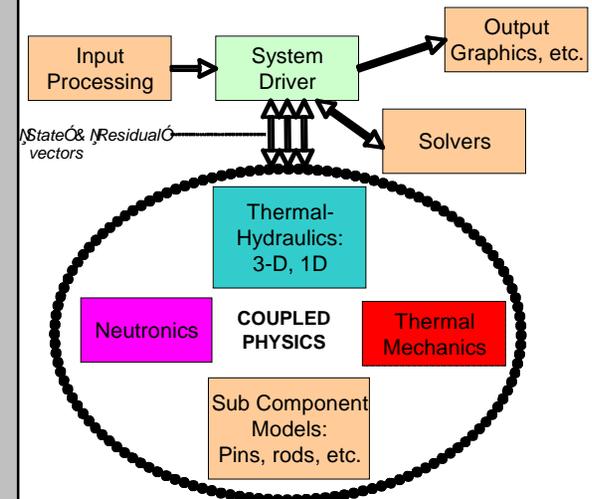
A BRISC Multi-Physics Modeling and Coupling Strategy



A stepping stone

➤ Different codes for different physics

- ✦ Thermal-Fluids
 - ◆ RIO, MELCOR
- ✦ Neutronics
 - ◆ SCALE...
- ✦ Thermal Mechanics
 - ◆ ARIA
- ✦ Fuel pin heat transfer
 - ◆ Simple code, complex data



Our ultimate goal

- Loose coupling in BRISC- α using RIO for overall time integration
- Strong Coupling being developed using JFNK, orchestrated by a Multi-Physics Driver/Solver Code
 - ✦ Designed to accept multiple PhysicsModules (codes).
 - ✦ Primary job of PhysicsModules is to take a complete state vector and return a partial residual.

** JFNK is a Newton method that employs Krylov-based linear solves (eg. CG, GMRES) without requiring formation of the Jacobian matrix.*

BRISC- α : Loose coupling of different codes and models for different physics

- **3D thermal-fluid flow (In-vessel)**
 - ✦ **RIO**, a lightweight, parallel, unstructured mesh FV code
 - ✦ **Brinkmann Forchheimer Equation Set**
 - ◆ Core region treated as a 3D anisotropic non-equilibrium porous media
 - ◆ Inter-pin flow not resolved - Correlations from literature must be applied
 - ◆ Non-core region flow treated with transient RANS type turbulence models
- **1D thermal-fluid flow (Balance of Plant)**
 - ✦ **MELCOR** (modified for Sodium properties and Eq. of State)
 - ◆ Not version 2.1, but version (?)1.67(?)
- **Subgrid-scale pin and control rod modeling**
 - ✦ Simple BRISC-specific code
- **Neutronics**
 - ✦ Simple Point Kinetics model with specified reactivity coefficients
- **Thermal Mechanics**
 - ✦ Not separately modeled - affect included in Neutronics model
- **Miscellaneous in-vessel components (e.g. pumps)**
 - ✦ **Time-dependent, user controlled function calls**

BRISC- α : A RIO-Centric Integrated Code

➤ RIO:

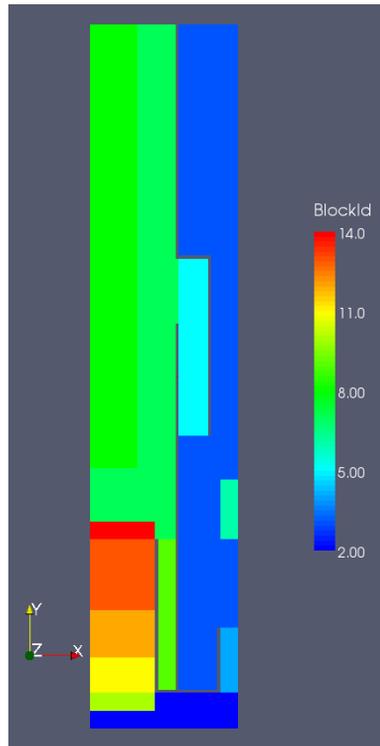
- ✦ A “light-weight” Cell-Centered Finite Volume CFD Code for Unstructured Meshes written by C. Moen (SNL).
- ✦ Treats multicomponent transport for both laminar and turbulent flows.
- ✦ Leverages Sandia Parallel Libraries Trilinos, Zoltan
- ✦ Source code available and written in standard c
- ✦ Wide range of user-written subroutines can be created to customize problem definition

➤ Coupling:

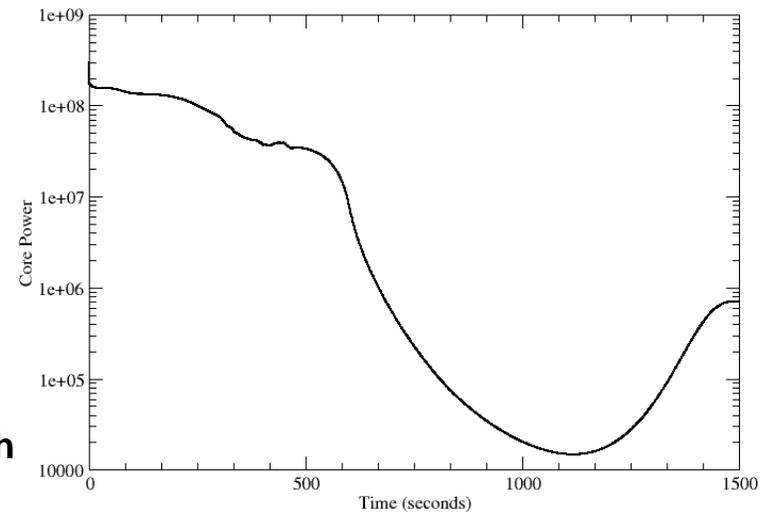
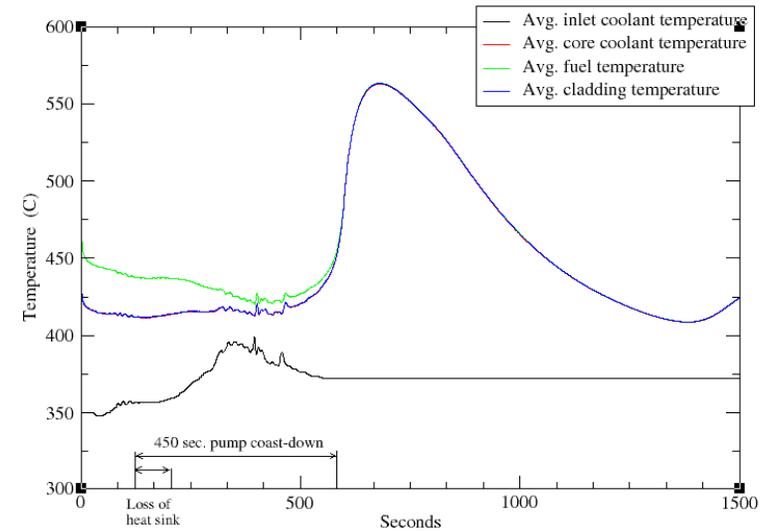
- ✦ Neutronics point-kinetics and subgrid heat transfer models added through user subroutines.
- ✦ Coupling to MELCOR enabled by passing heat flux and temperatures through small files.
- ✦ Minor modifications to source to implement parallel aspects of BRISC specific user subroutine features.

Illustrative 2D Calculation of a Unprotected Loss of Flow Sequence

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.



~ 10000 element mesh
~ 30 min transient



Both 2D and 3D Solid models, and then associated meshes, were created to test BRISC- α

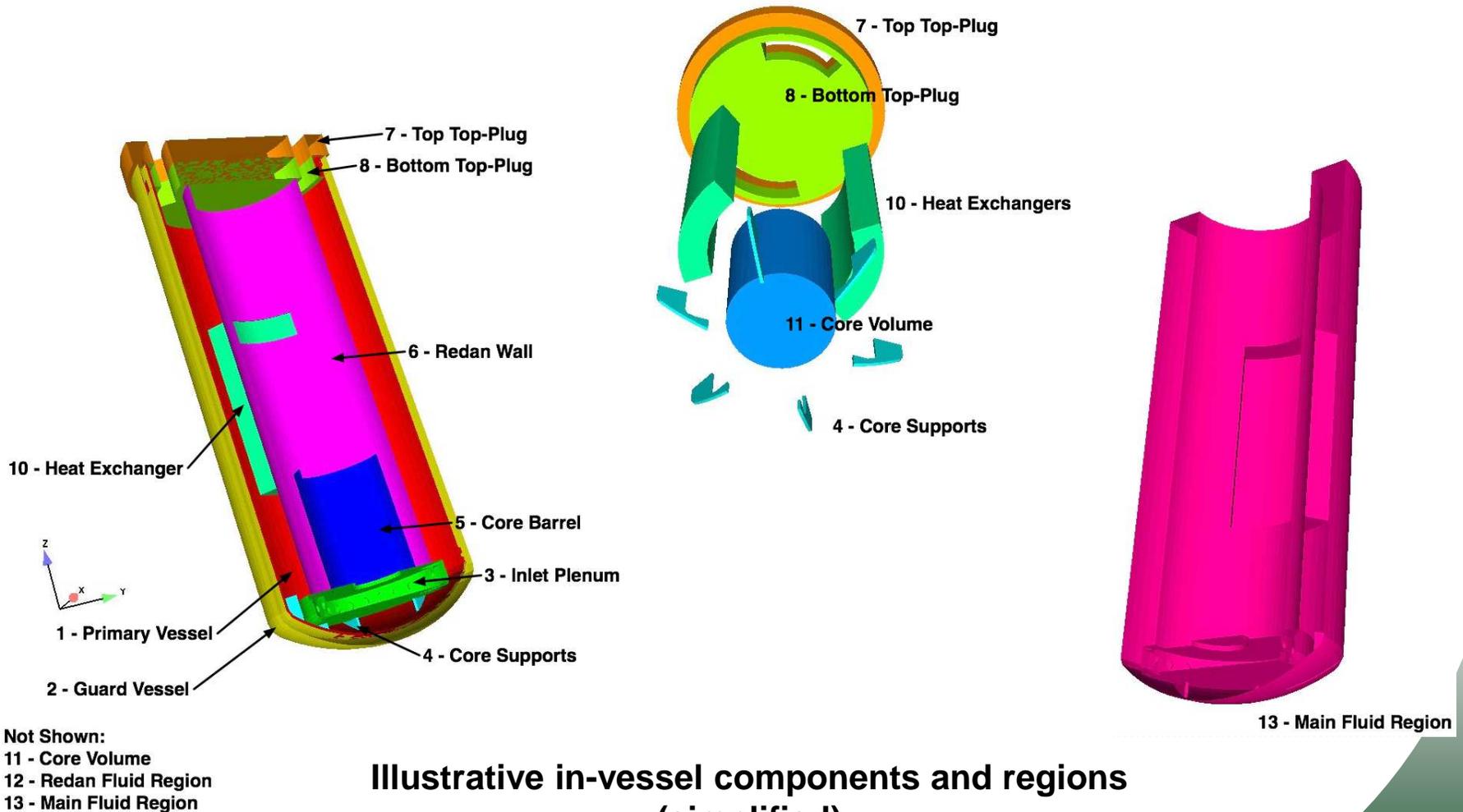
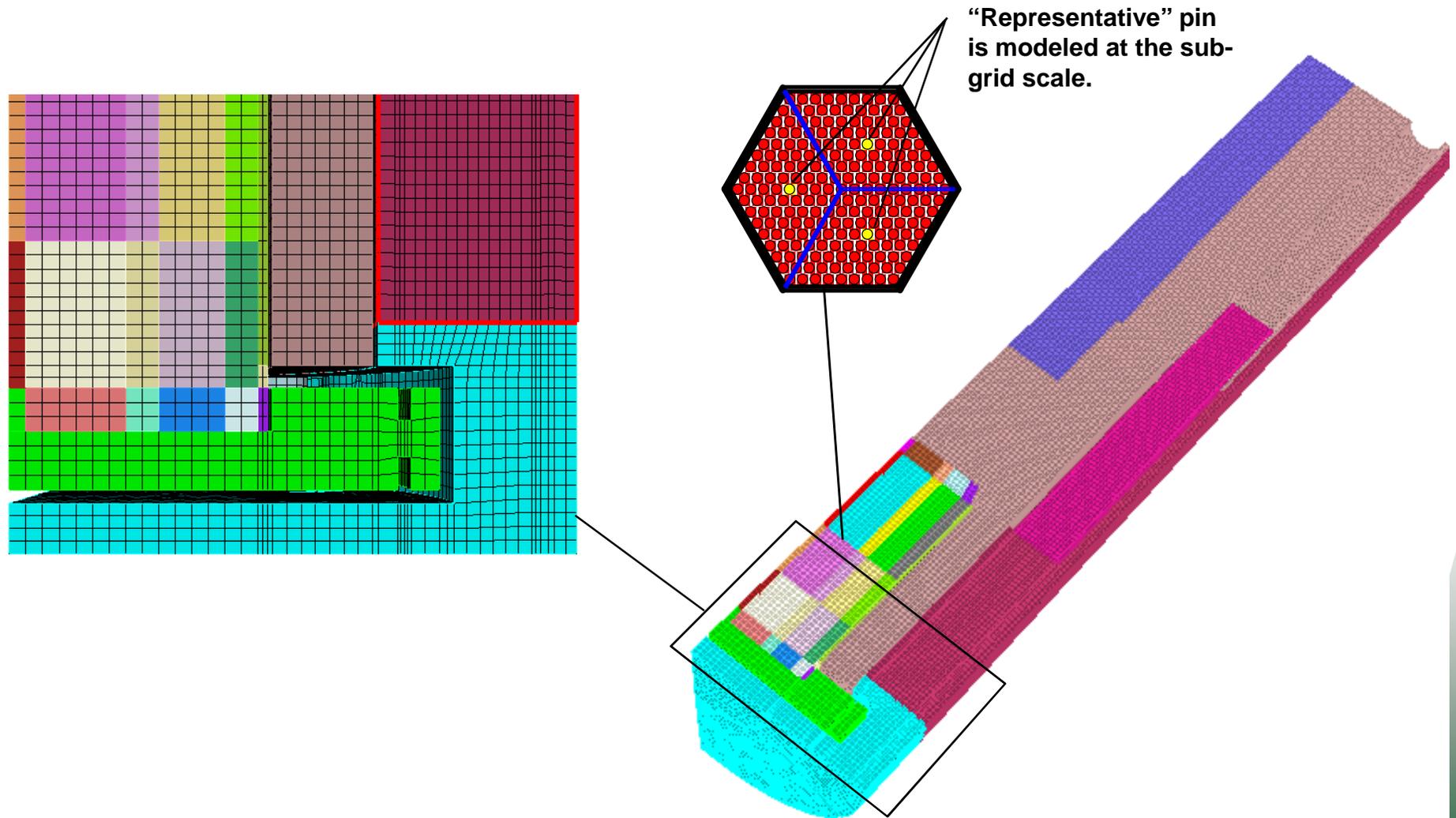
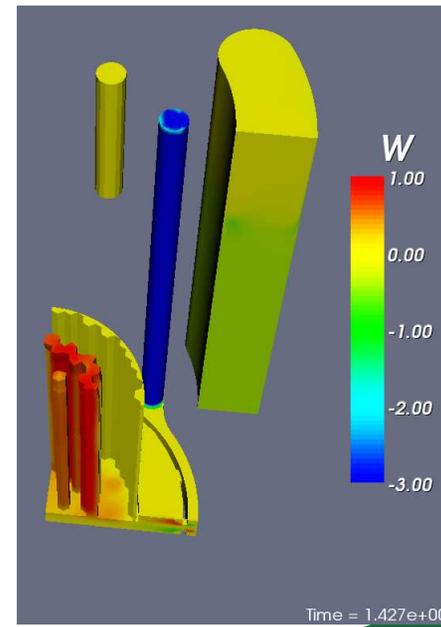
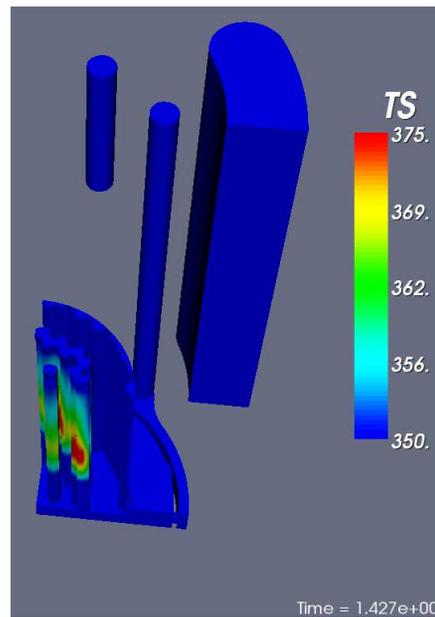
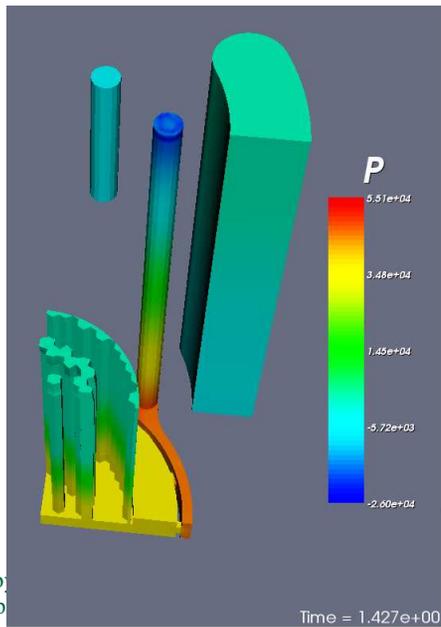
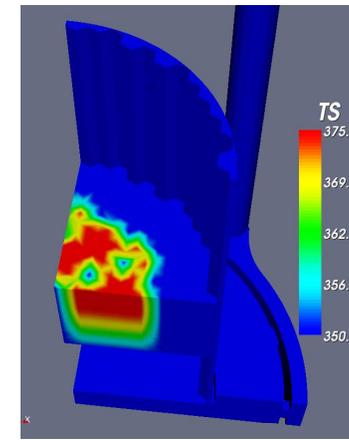
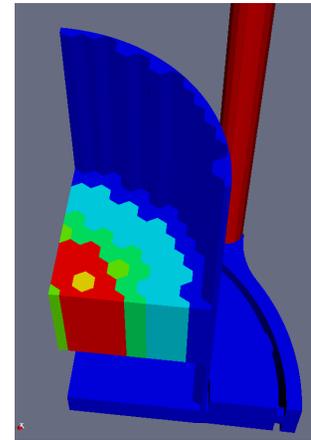
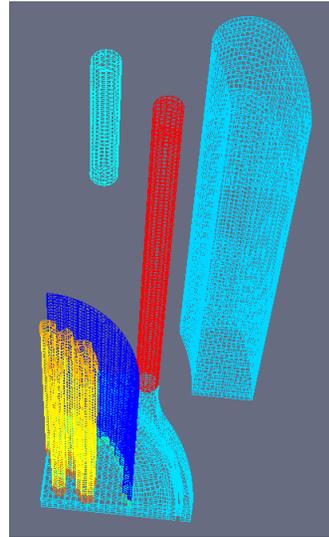
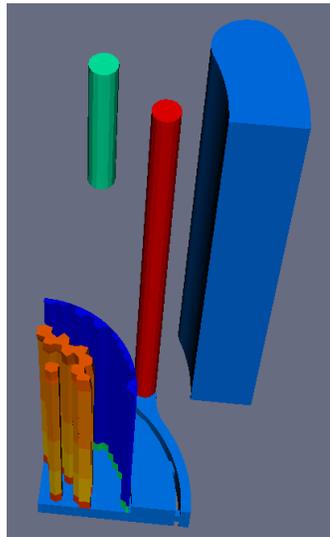


Illustration of a meso-scale 3D mesh used to represent the in-vessel regions



Illustrative Snapshots of 3D Model for Unprotected Loss of Flow Sequence



BRISC- β : Lightweight Multi-scale Multi-Physics Coupling

➤ Overall solution orchestrated by Multi-Physics Driver

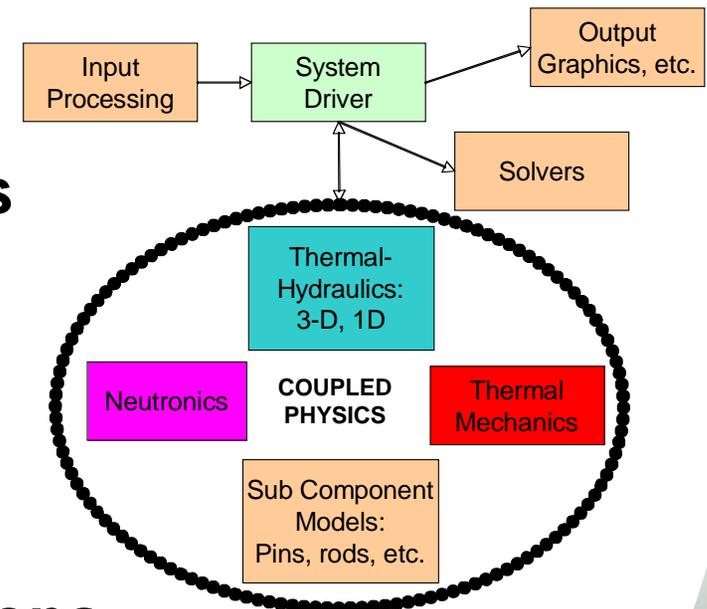
- ✦ Multiple PhysicsModules (codes) interacting through a flexible API.
- ✦ PhysicsModules compute residuals, perform Physics-based preconditioning, . . .
- ✦ **Strong coupling** enabled through JFNK

➤ Different codes for different physics

- ✦ RIO - enhanced with subgrid physics
- ✦ TRITON-RASCAL - ORNL/SNL collaboration
- ✦ MELCOR - balance of plant

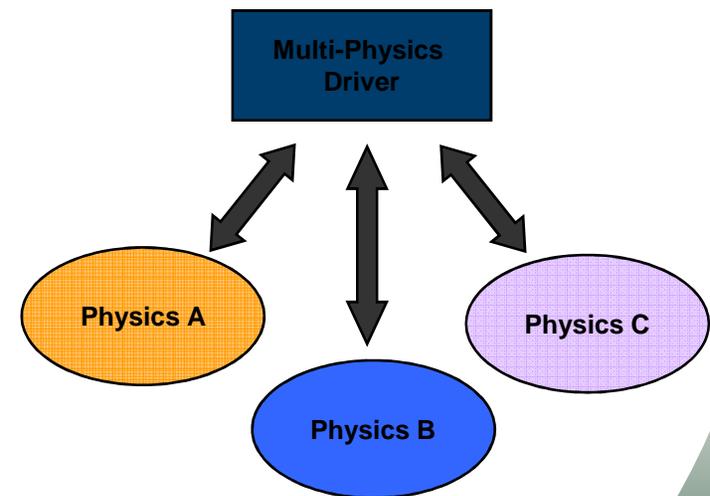
➤ 3D solid modeling and meshing of major reactor components and regions

- ✦ Solid-fluid non-conformal mesh interactions enabled by adapting RocStar utilities from UIUC ASC Center to our needs.



Key Tasks of the Multi-Physics Driver

- Identify and register each physics code
- Set-up global problem and physics-code interfaces
 - ✦ Create global state vector X
 - ✦ Define physics-specific state vectors ($X_A, X_{BA}^*, X_{CA}^*, \dots$)
- Initialize each physics code and negotiate initial time step
- Loop over time
 - ✦ Obtain converged solution (NOX, JFNK)
 - ◆ Request residuals from physics codes
 - ◆ Request physics-based preconditioning
 - ◆ Update state vector
 - ✦ Perform output (each physics code)
 - ✦ Time-step control: Negotiate/calculate based on all the physics
- End simulation



Physics-Code Changes/Additions needed for the Multi-Physics Driver

- Code must be revised so the driver can link to it. (i.e. like a library)
- Code must be organized into several key parts that can be called independently
 - ✦ Initialize: allocate memory, read inputs
 - ✦ Solve: compute solution for a given time step and 'state'
 - ✦ Advance: copy the final 'state' and solution into initial
 - ✦ Modify: change properties set at initialization
 - ✦ Output: print to file (in parallel)
- Additional routines must be added
 - ✦ Register coupling capabilities: "who I depend on"
 - ✦ Pass control variables "I'm converged now"
 - ✦ Compute and pass data for coupling to other physics
 - ✦ Compute residuals
 - ✦ Perform preconditioning
- Code-dependent challenge
 - ✦ For most, these are already there, just not spelled out
 - ✦ For some, it may require reengineering the software

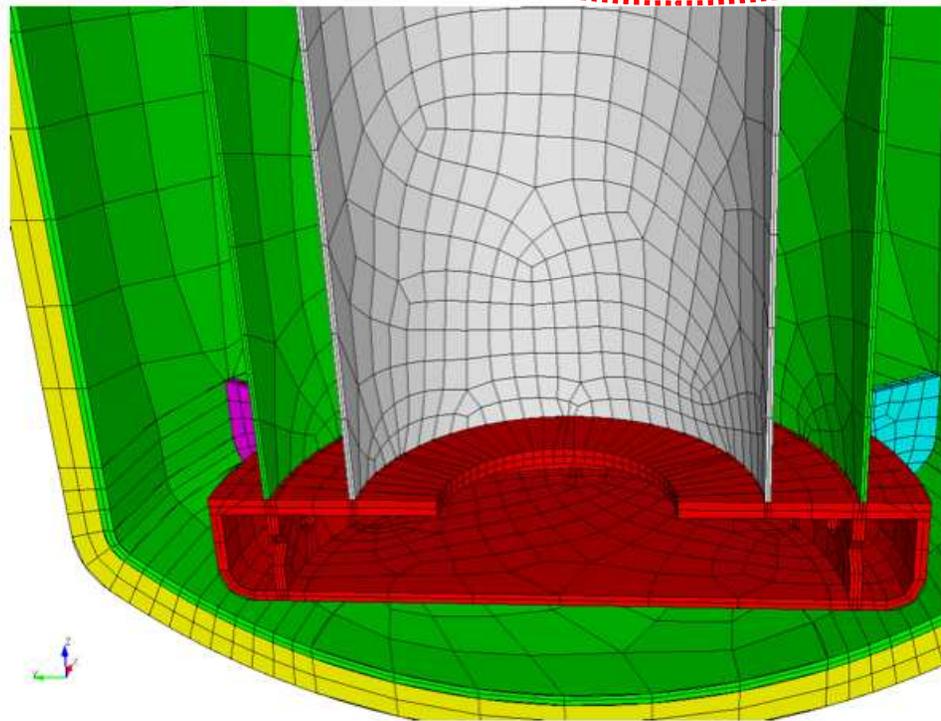
“Physics Codes” in BRISC-β

Physics	Code
3D In-vessel flow and HT	RIO - enhanced
Sub-grid fuel and control rods	Simple 1D
3D In-vessel conduction in solid structures	RIO
Solid-fluid coupling at domain interface	new
Neutronics	Rascal-Scale (2D) Nestle-Scale (3D)
Balance of Plant -	MELCOR

- Each must be written/revised to properly interact with the MP Driver

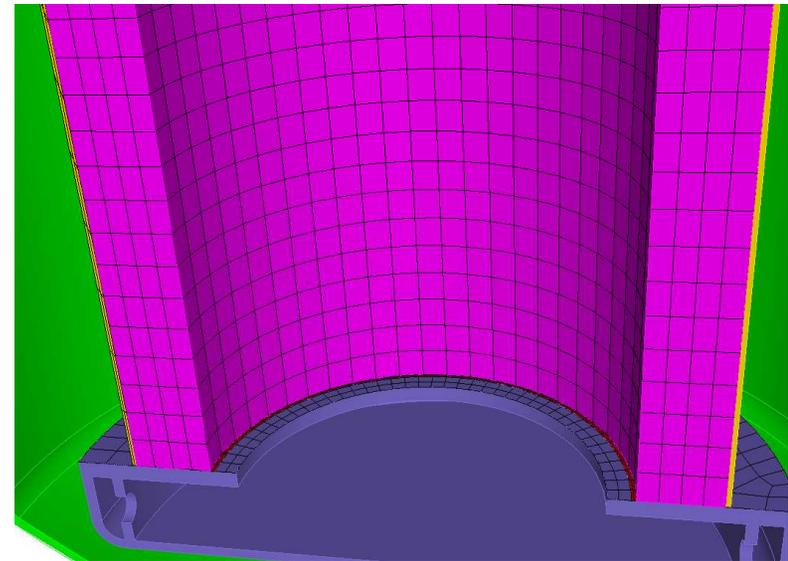
BRISC- β will use separate unstructured meshes for the solid and fluid regions that are not conformal at the interfaces:

Structured vs. **Unstructured**



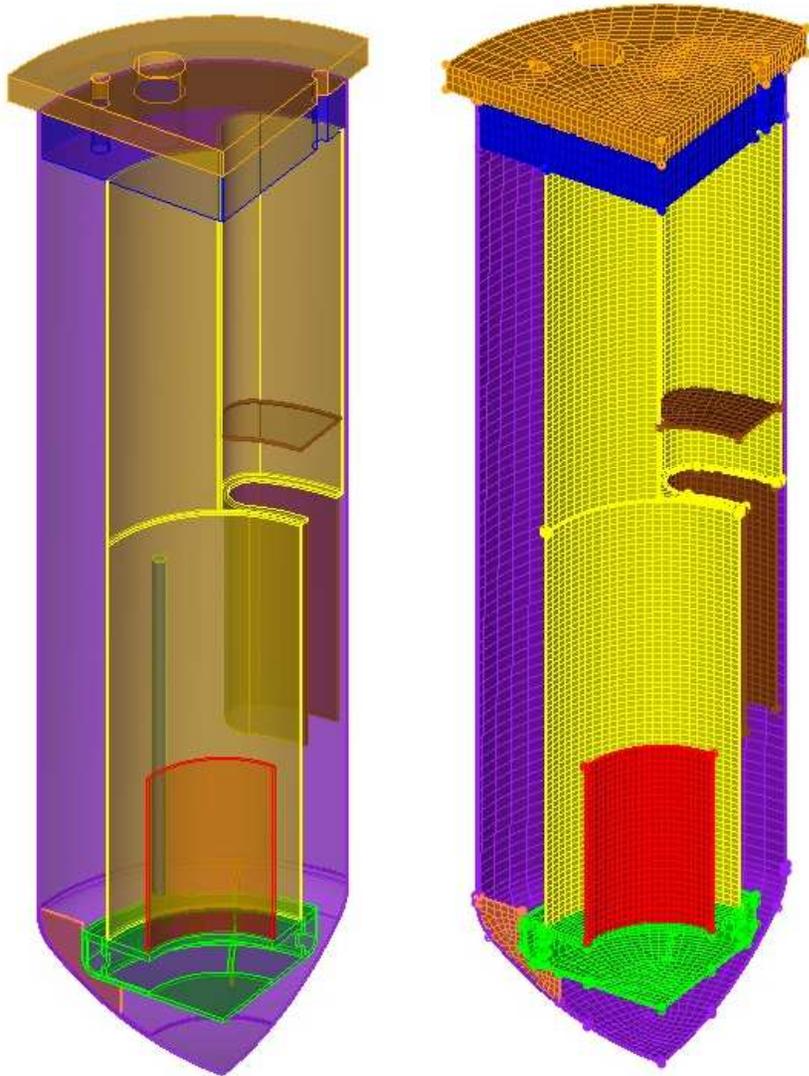
Unstructured mesh of Solid Structure
Regions for Conduction Heat transfer

Conformal vs. **Non-conformal**



Non-conformal mesh of fluid flow
regions

Improved Solid and Fluid region meshes generated (using CUBIT) from solid models created with ProE.



Meso-scale "Solid" region mesh



Meso-scale "fluid" region mesh

Comments on Neutronics

- **BRISC- α :** **“First, get an answer”**
 - ✦ Point-kinetics
 - ✦ Coefficients derived from SCALE and literature

- **BRISC- β :** **“Then, make it better”**
 - ✦ SCALE(TRITON) cross section generation
 - ✦ 2D finite-difference diffusion (RASCAL) for cross-section changes
 - ✦ Point-kinetics for geometric changes
 - ✦ 3D power-distribution provided to the MP driver

- **BRISC- γ :** **“Next, tackle the toughest problem”**
 - ✦ SCALE(TRITON) cross section generation
 - ✦ NESTLE diffusion for cross-section changes
 - ◆ Investigation of NESTLE for geometric changes
 - ◆ Thermo-mechanics may not be ready

- **Future?** **“But always look ahead”**
 - ✦ Model every fuel pin: fine-mesh diffusion or transport
 - ✦ Account for geometric changes: unstructured-mesh solver
 - ✦ Get the initial state right: steady-state depletion

Comments on Status

- **MP Driver development and testing on each physics code is ongoing**
 - ✦ Target completion is Mar. 1
 - ✦ Converted to C++ (from Python)
 - ✦ Can currently drive RIO, MELCOR, and neutronics
 - ✦ Additional interface/control capability needed
- **All major RIO modifications needed are complete**
 - ✦ High-level code restructuring
 - ✦ Compute Residual
 - ✦ Perform Preconditioning
- **Neutronics for BRISC- β is complete**
 - ✦ SCALE(TRITON): Cross-section generation
 - ✦ RASCAL: 2D diffusion on hexahedron (fixed geometry)
 - ✦ Point-Kinetics: To account for geometric changes
- **Code to compute residuals needed for tight-coupling across a non-conformal surface mesh not yet complete.**
- **An illustrative 3D problem is in progress**
 - ✦ Presentation at a SIAM meeting in early March

Opportunities?

- **Can TAMU-UM utilize BRISC software?**
 - ✦ **By this summer, yes.**
 - ◆ Needs ORNL/SNL ‘user-support’ as it’s still developmental
 - ◆ After this summer may be too late (is there follow-on funding?)
 - ✦ **UM/TAMU-student at ORNL during summer 2009 to implement coupling with PARCS/NESTLE?**

- **Can RIO model VHTR?**
 - ✦ **Absolutely (so I’m told)**
 - ◆ What does “can it model” really mean?
 - ✦ **Still requires experiments/CFD to define coefficients**

- **Is this the right path for enhancing existing software at the NRC and/or DOE/NE?**

Example Meso-scale Physics in BRISC- α

3D Convective Flow and HT Model

Governing NS Equations:

$$\frac{\partial}{\partial t}(\rho_f) + \nabla \cdot (\rho_f \bar{u}) = 0$$

$$\frac{\partial}{\partial t}(\rho_f \bar{u}) + \nabla \cdot (\rho_f \bar{u} \bar{u}) = -\nabla P + \nabla \cdot (\mu_f \nabla \bar{u}) + \rho_f \bar{g}$$

$$\frac{\partial}{\partial t}(\rho_f C_f T_f) + \nabla \cdot (\bar{u} \rho_f C_f T_f) = \nabla \cdot (k_f \nabla T_f) + \dot{Q}_f$$

Boundary Conditions:

- No-slip velocity
- Specified temperature
- Specified heat flux (or gradient of T)
- Heat transfer coefficient

Modeling Notes:

- The density, specific heat, and thermal conductivity must be known as functions of temperature for all fluids
- Source terms (energy, mass, momentum) imply coupling to other models.
- The model equations require closure relationships (e.g. B, K, $k_{f,eff}$, μ_{eff} . . .)
- A separate energy equation, not shown, is solved in the “solid” region and the associated sub-component model (e.g. for fuel pins and control rods)

Brinkmann-Forchheimer Model Equations:

$$\frac{\partial}{\partial t}(\phi \rho_f) + \nabla \cdot (\rho_f \bar{v}) + \dot{\rho}'_{src} = 0$$

$$\frac{\partial}{\partial t}(\rho_f \bar{v}) + \nabla \cdot \left(\frac{1}{\phi} \rho_f \bar{v} \bar{v} \right) = \left[-\nabla P - \frac{\phi \mu}{K} \bar{v} - \frac{B \phi}{K^{1/2}} \rho_f |\bar{v}| \bar{v} \right] + \nabla \cdot (\mu_{eff} \nabla \bar{v}) + \phi \rho_f \bar{g} + \frac{\dot{\rho}'_{src} \bar{v}_{src}}{\phi}$$

$$\frac{\partial}{\partial t}(\rho_f \phi C_f T_f) + \nabla \cdot (\bar{v} \rho_f C_f T_f) = \nabla \cdot (\phi k_{f,eff} \nabla T_f) + \phi \dot{Q}_f + \dot{\rho}'_{src} C_{src} T_{src} + A_{fs}(\bar{x}) h_{fs} (T_s - T_f)$$

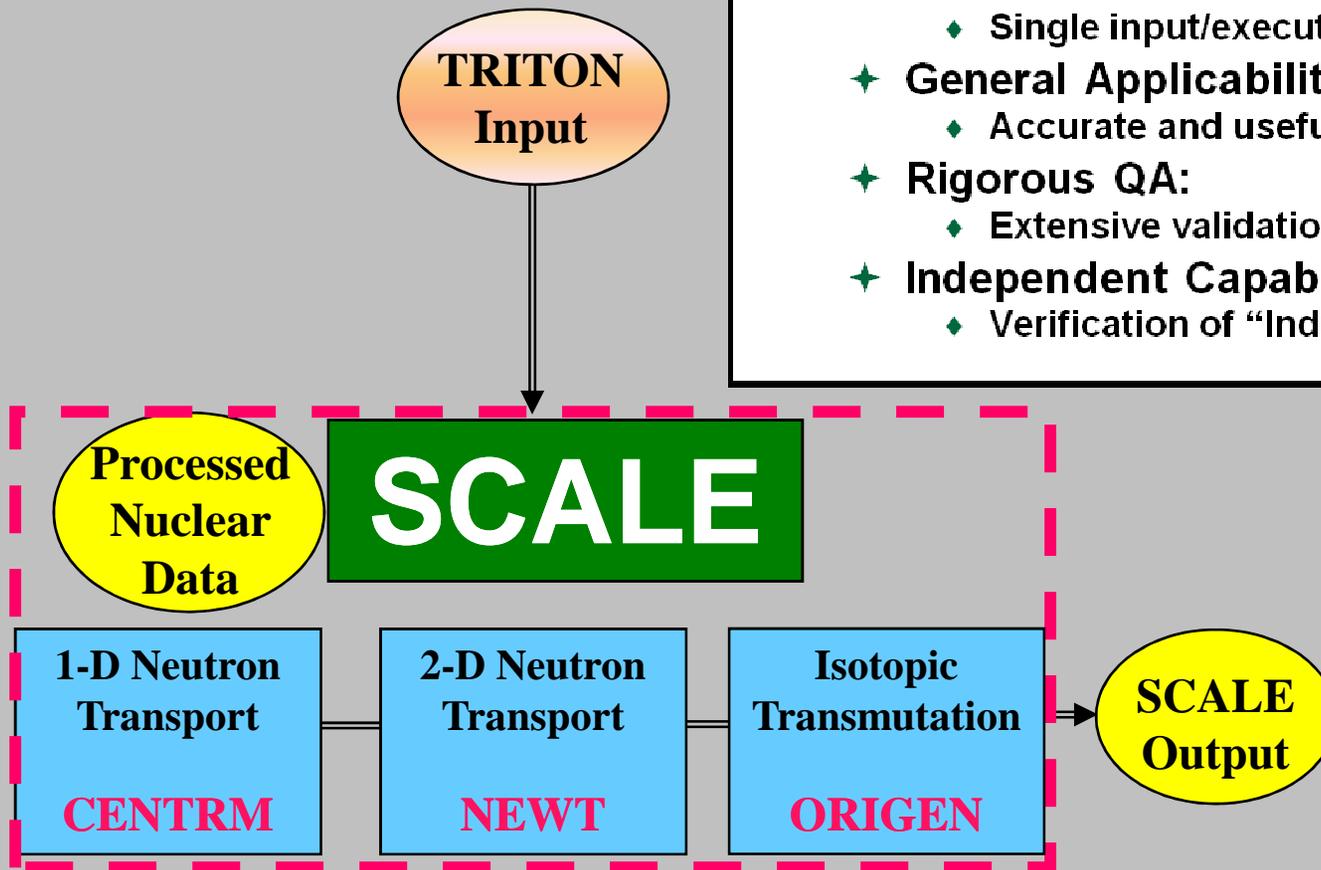
Some comments about using the B.F. Equations

- Variations of this type of approach has been around for many years in a variety of forms. These equations cast them in a very general form.
- Numerous recent papers have appeared in the literature concerning the use of these specific equations. For example:
 - L. Betchen, A. G. Straatman, and B. E. Thompson, “A NonEquilibrium Finite-Volume Model for Conjugate Fluid/Porous/Solid Domains,” *Numerical Heat Transfer, Part A*, 49: 543-565, 2006.
- Standard Friction Factor correlations can be re-written in a form that can be directly implemented.
- Sophisticated correlations developed specifically for wire-wrapped pin bundles, heat exchangers, and so forth are available.
- Improved correlations from DNS simulations can easily be incorporated.

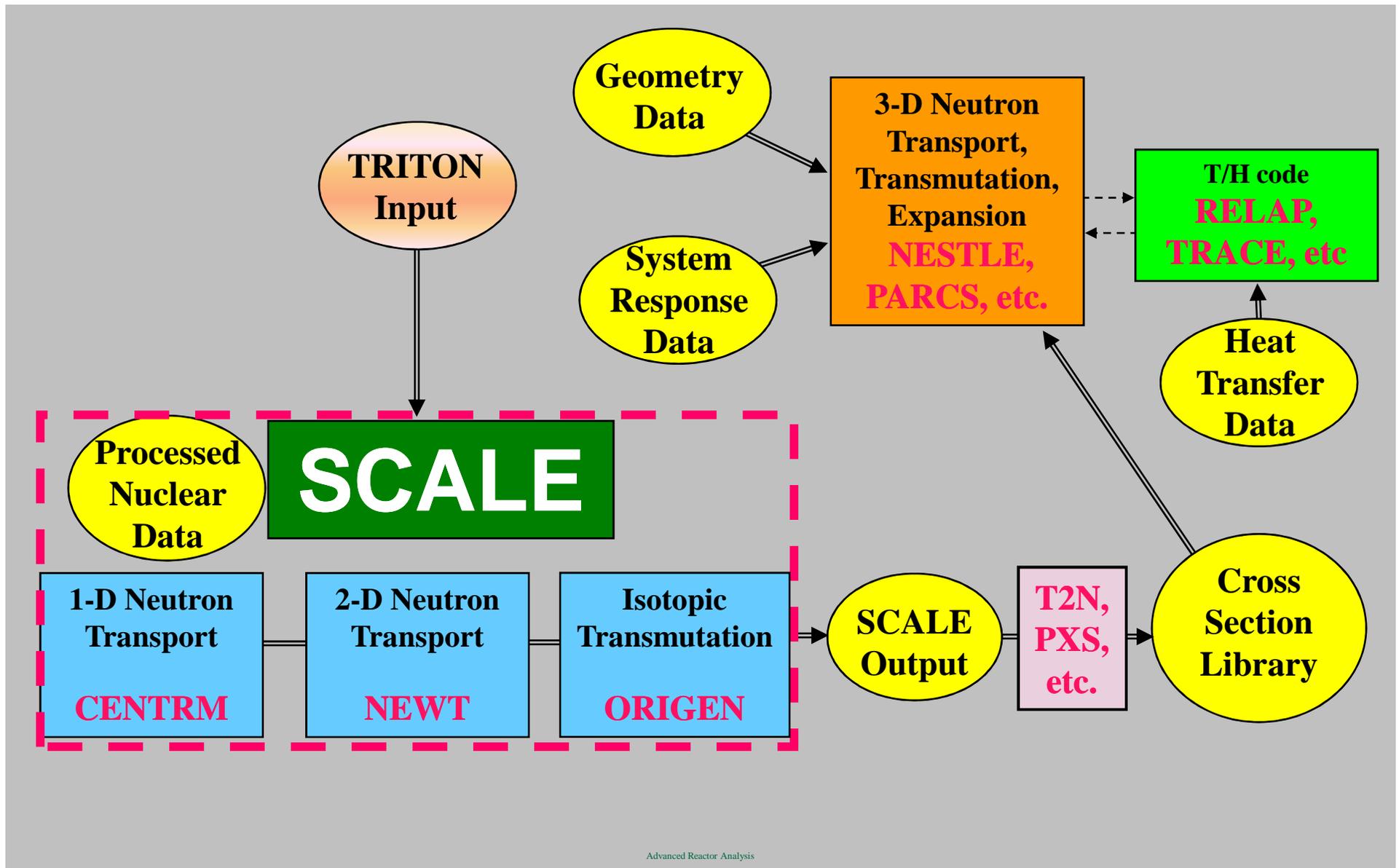
TRITON: “lattice-physics” in SCALE

➤ Secrets of SCALE’s Success

- ✦ **Ease-of-Use:**
 - ◆ Single input/execution/output
- ✦ **General Applicability:**
 - ◆ Accurate and useful for many systems
- ✦ **Rigorous QA:**
 - ◆ Extensive validation for many systems
- ✦ **Independent Capability:**
 - ◆ Verification of “Industry” software

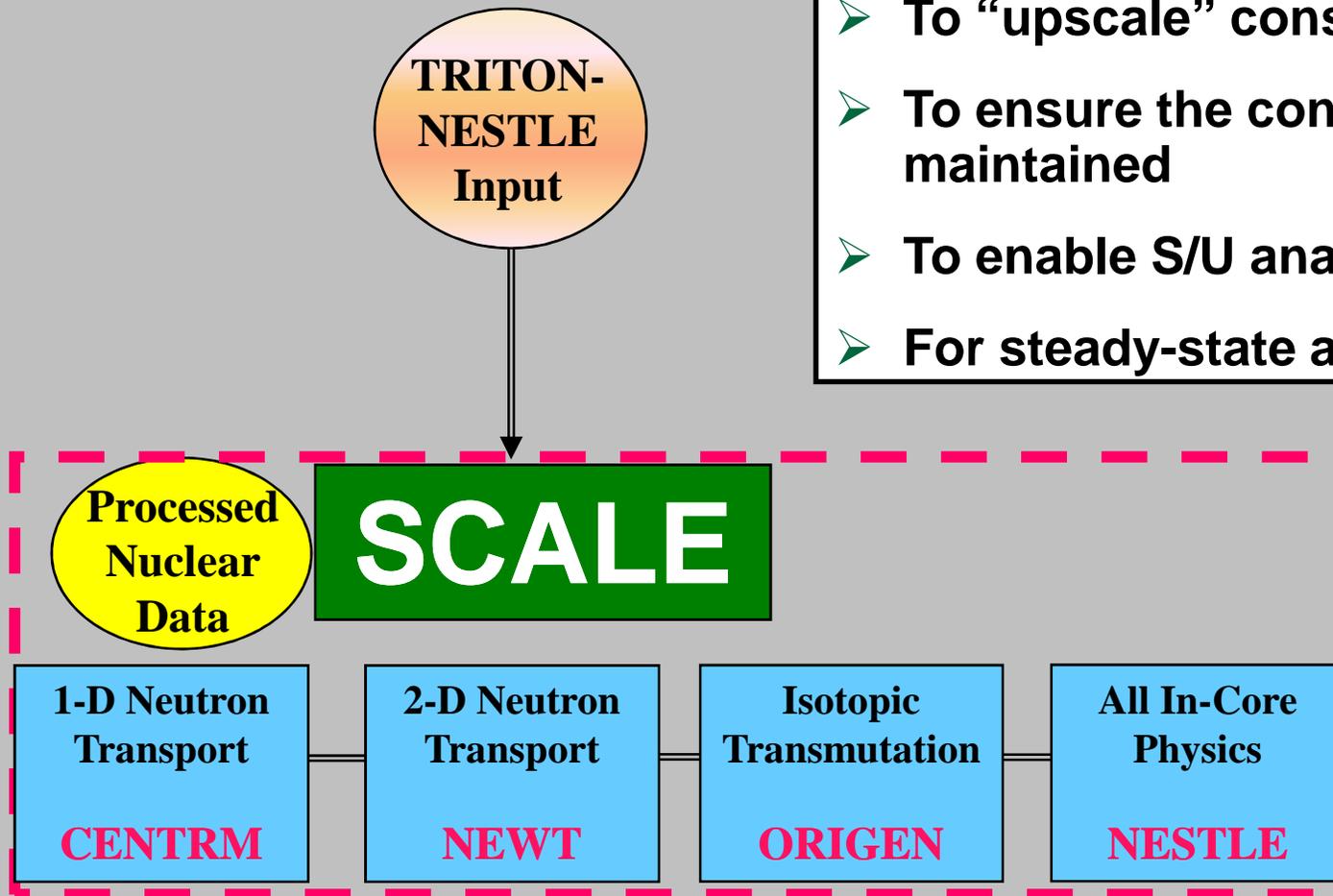


End-to-End reactor analysis with open-source codes is difficult



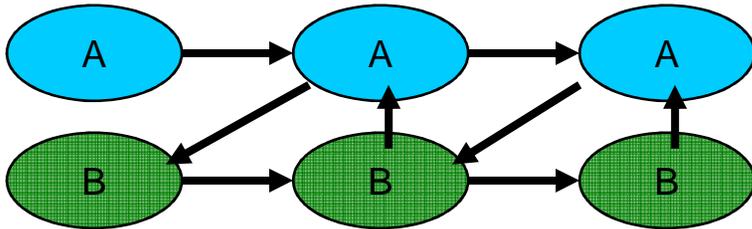
NESTLE is being integrated with SCALE to make the whole process easier

- To “upscale” consistently
- To ensure the consistency is maintained
- To enable S/U analysis
- For steady-state analyses



Why Use JFNK ?

Loose Coupling: Successive Substitution



JFNK can provide Newton-like convergence using loose-coupling information.

Strong (Monolithic) Coupling:
Traditional Newton's Method

While not converged:

$$\mathbf{J}_n \Delta \mathbf{x}_n = -\mathbf{R}_n$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta \mathbf{x}_n$$

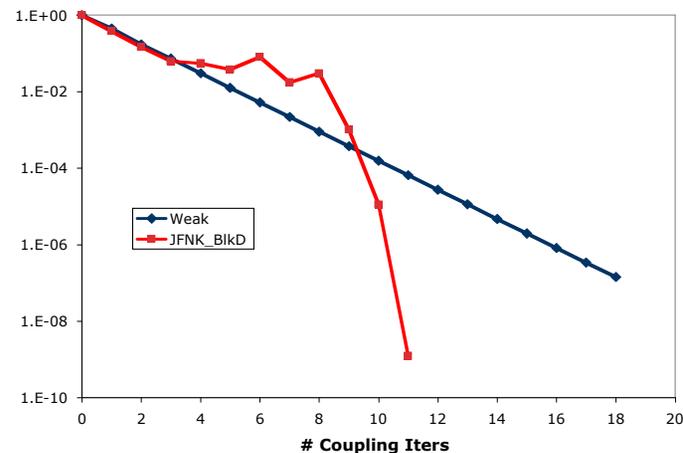
$$\mathbf{J}\mathbf{M}^{-1}\mathbf{p} \approx \frac{\mathbf{R}(\mathbf{x} + \epsilon\mathbf{M}^{-1}\mathbf{p}) - \mathbf{R}(\mathbf{x})}{\epsilon}$$

$$\begin{bmatrix} \mathbf{J}_{AA} & \mathbf{J}_{AB} \\ \mathbf{J}_{BA} & \mathbf{J}_{BB} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x}_A \\ \Delta \mathbf{x}_B \end{bmatrix} = \begin{bmatrix} -\mathbf{R}_A \\ -\mathbf{R}_B \end{bmatrix}$$

\mathbf{M}

Why Use JFNK ? (continued)

- (1) At a minimum it requires no additional information from each code beyond what is required for weak coupling;
- (2) any additional information a code can provide can be used to enrich the performance of JFNK;
- (3) any specialized solution technology embodied in a code can be preserved and leveraged; and
- (4) the nominal order of convergence compared to weak coupling is improved from linear to quadratic.



However: Preconditioning is a key to realizing good convergence