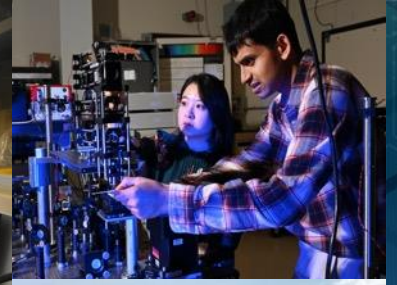




Guillaume Giudicelli
Pierre-Clément Simon
Casey Icenhour
INL/CON-25-06153

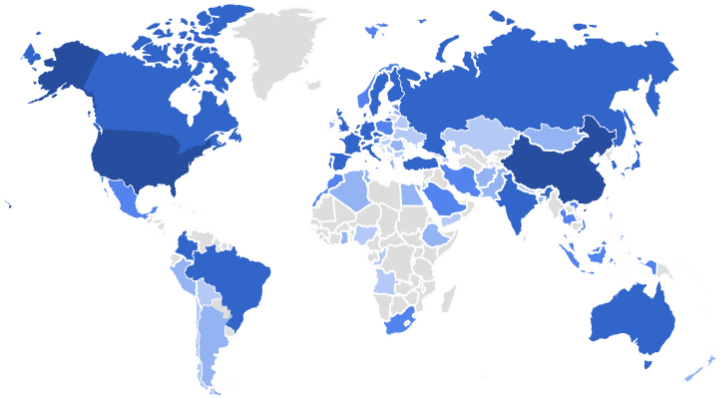
Overview of the capabilities in the Multiphysics Object Oriented Simulation Environment (MOOSE) and recent activities in modeling and simulation for fusion energy systems

Battelle Energy Alliance manages INL for the
U.S. Department of Energy's Office of Nuclear Energy

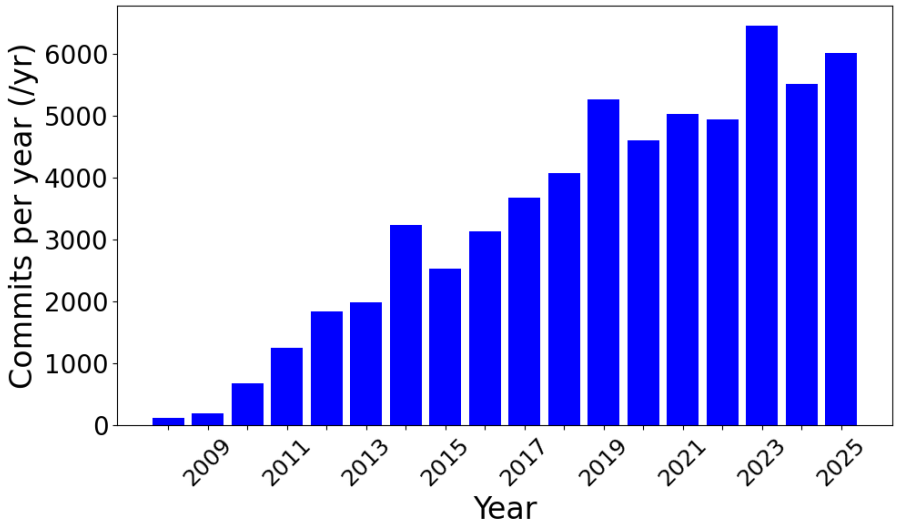
Idaho National Laboratory

MOOSE : Multiphysics Object Oriented Simulation Environment

- Started in 2008 in the INL Laboratory Directed Research and Development program
- Open-sourced in 2014
- Solves Partial Differential Equations, tied to a variety of Physics, using finite element and finite volume discretizations
- Large international user base
- Undergoing strong growth in user base and contributions



COUNTRY	KNOWN USERS
United States	1.2 k
China	366
United Kingdom	194
Germany	174
Japan	129
Hong Kong	94
India	82



Ecosystem of the MOOSE-based applications

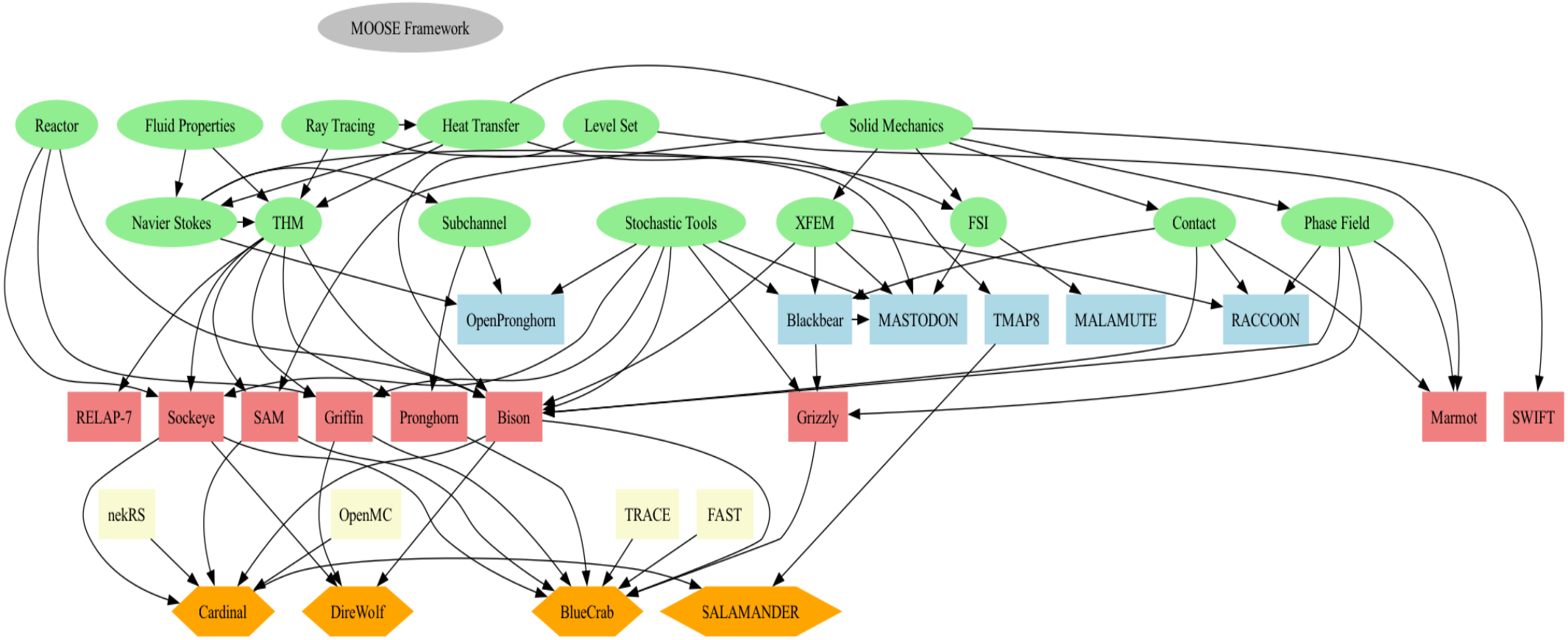
Physics Modules

MOOSE-Based Open-source apps

MOOSE-Based Closed source apps

MOOSE-wrapped External apps

MOOSE Combined applications



MOOSE Modules

Physics modules

- Heat Transfer
- Solid Mechanics
- Contact
- Phase Field
- Porous Flow
- Navier-Stokes (FE & FV)
- Chemical Reactions
- Thermal hydraulics
- Fluid Structure Interaction
- Electromagnetics
- Scalar Transport
- Geochemistry

Numerics modules

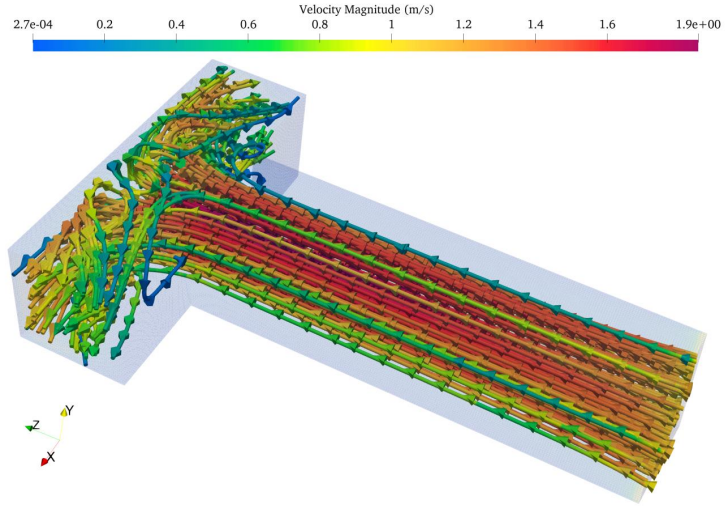
- Reconstructed Discontinuous Galerkin
- eXtended Finite Element (XFEM)
- Level Set
- Functional Expansion Tools
- Stochastic Tools
- Ray Tracing
- External petsc solver
- Optimization

Others / Cross-cutting

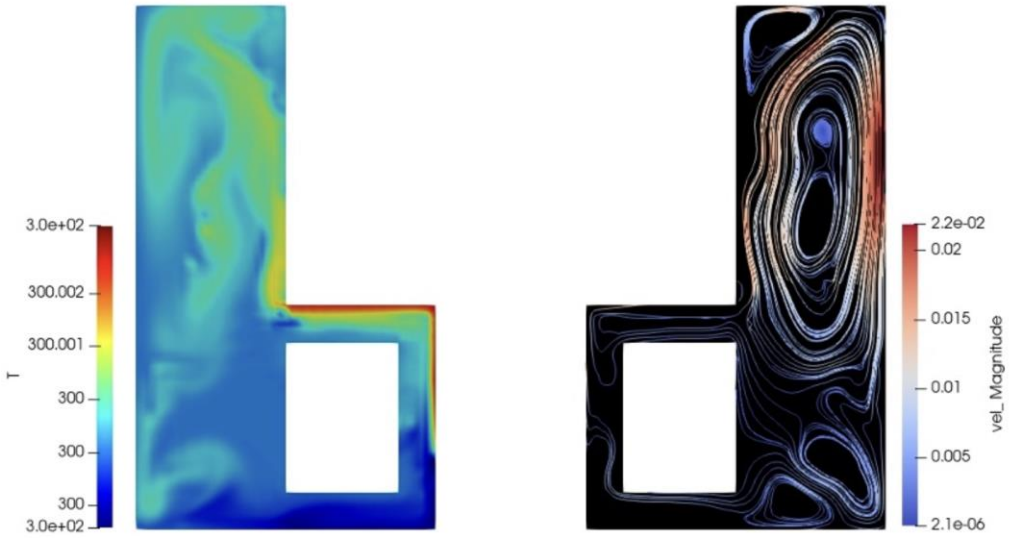
- Reactor
- Fluid properties
- Solid Properties

Navier Stokes module

- A library for the implementation of simulation tools that solve the multi-dimensional Navier-Stokes equations using either the continuous Galerkin finite element (CGFE) or the finite volume (FV) method. The Navier-Stokes equations may be solved with:
  - An incompressible formulation (CGFE & FV)
  - A weakly compressible formulation (FV)
  - A fully compressible formulation (FV)
- K-epsilon turbulence model available, and k-omega-SST to be released
- Basis for open-source CFD code OpenPronghorn, released June 2025



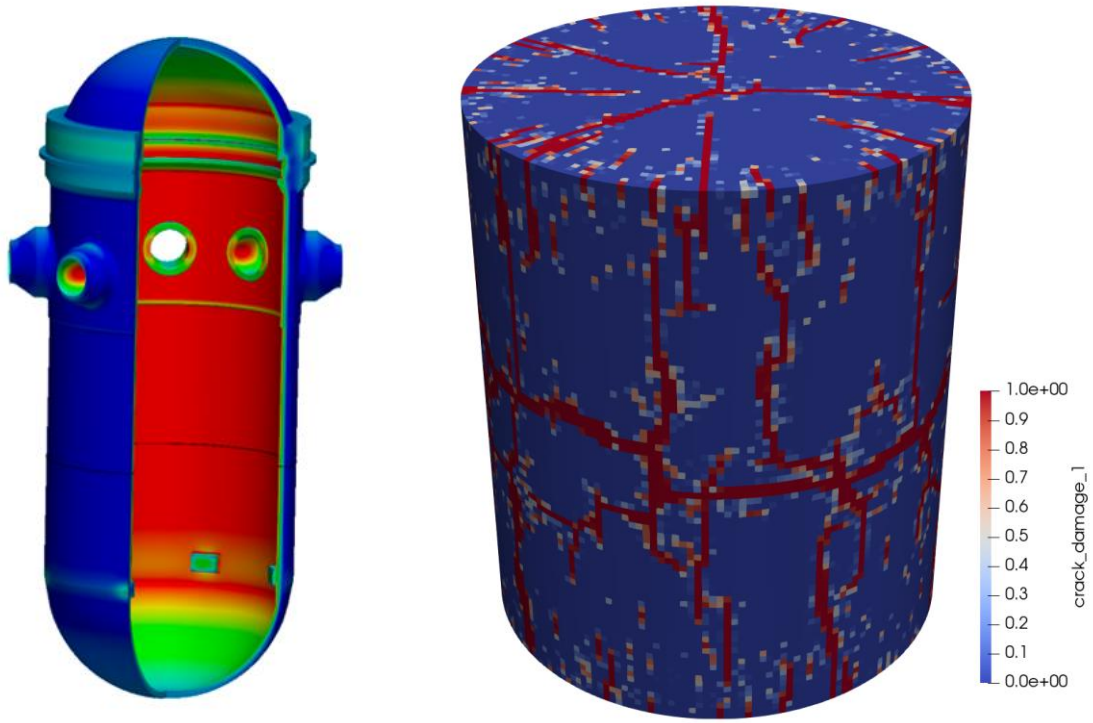
Top: demonstration of the segregated solver on a T-junction
Bottom: velocity and temperature results for Hi-STORM spent fuel cask cooling



Bottom figure from “Development of a MOOSE thermal model of the MPC-32 canister and HI-STORM overpack”, Okyay et al.

Solid Mechanics module

- The Solid Mechanics module is a library of simulation tools that solve continuum mechanics problems. It provides a simple approach for implementing even advanced mechanics models:
  - Plug-n-play design enables users to incorporate the relevant physics for specific and varied simulations
  - Tensor implementation matches mathematical theory
  - Straight-forward procedure for adding new physics
- The solid mechanics system can be used to simulate both linear and finite strain mechanics, including
  - Elasticity and Cosserat elasticity
  - Plasticity and micromechanics plasticity,
  - Creep, and Damage due to cracking and property degradation



Left: Thermo-mechanical stress analysis of a reactor pressure vessel
Right: Simulation of a fractured ceramic nuclear fuel pellet in the MOOSE solid mechanics module—the contours of the damage index, with higher values corresponding to more damage

Optimization module

- Leverages the TAO library for inverse optimization

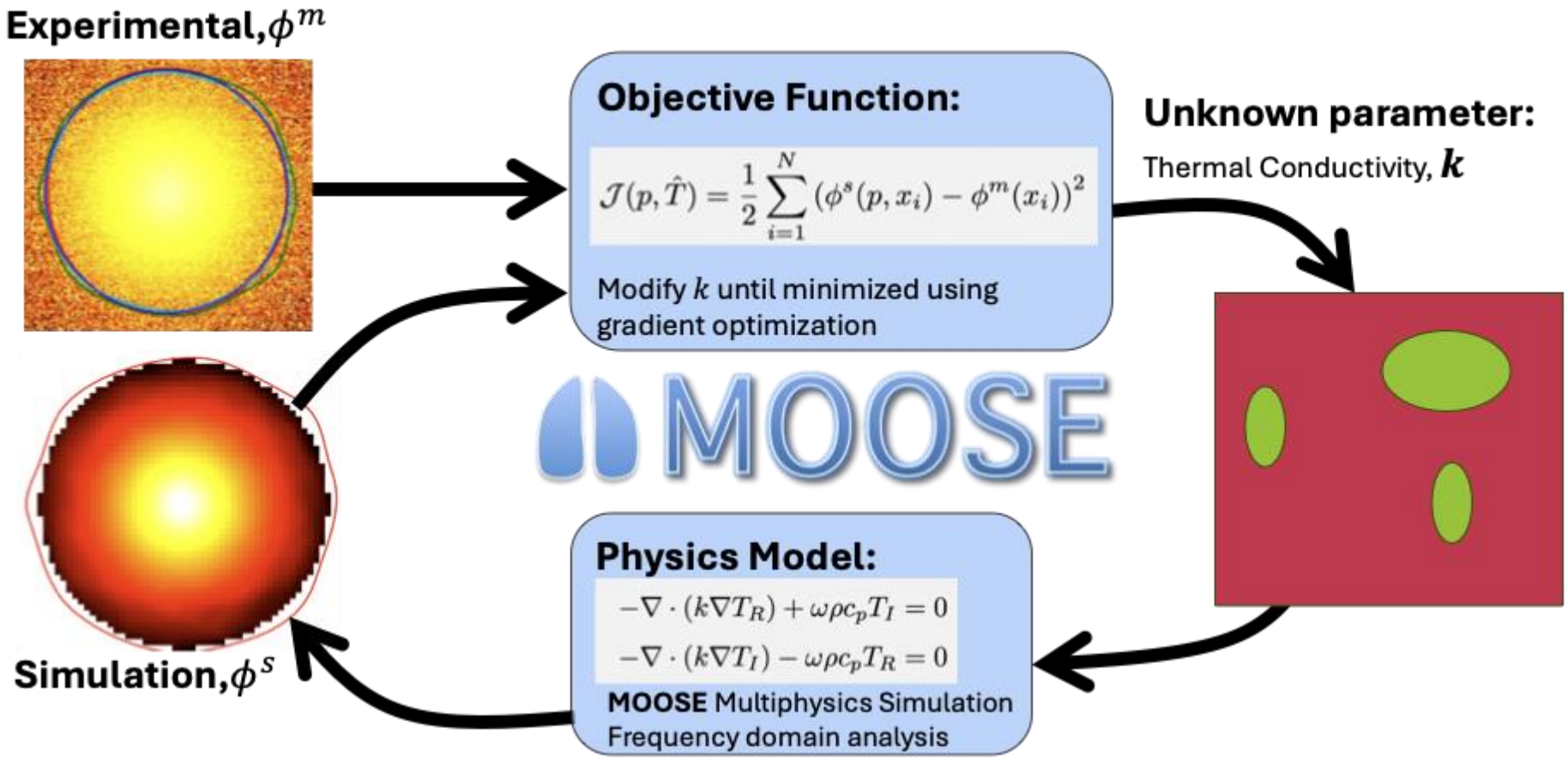
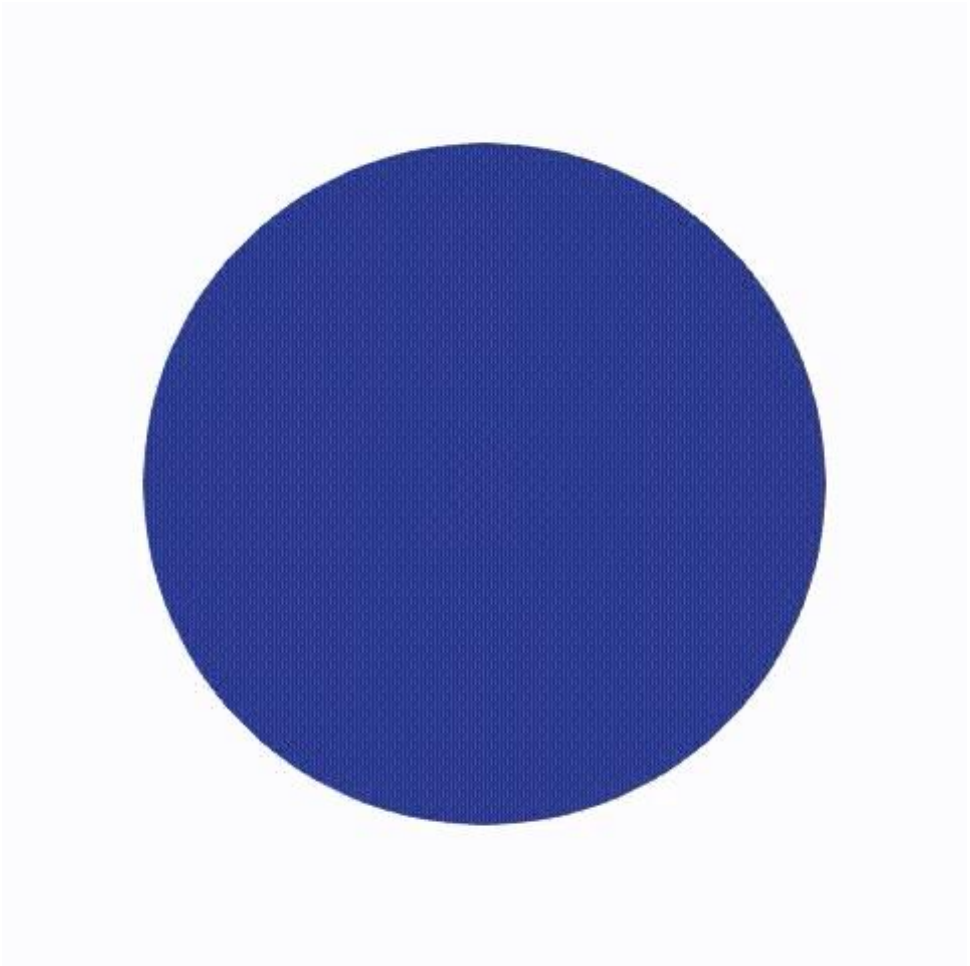


Figure by Lynn Munday (INL)

Electromagnetics module

- Transient and time-harmonic (i.e., single-frequency, steady-state) simulation in 1D and 2D (3D is currently in development)
- Component-wise (scalar variables) and vector field (vector variables) components for the Helmholtz wave form of Maxwell's Equations
- Complex field calculations
- First-order port boundary conditions (scalar and vector forms)
- Verification cases with 2D waveguide and 1D slab



Simulated steady state radiation intensity pattern of a vertically-oriented half-wave dipole antenna driven at 1 GHz.

From “The MOOSE electromagnetics module”, SoftwareX 2024

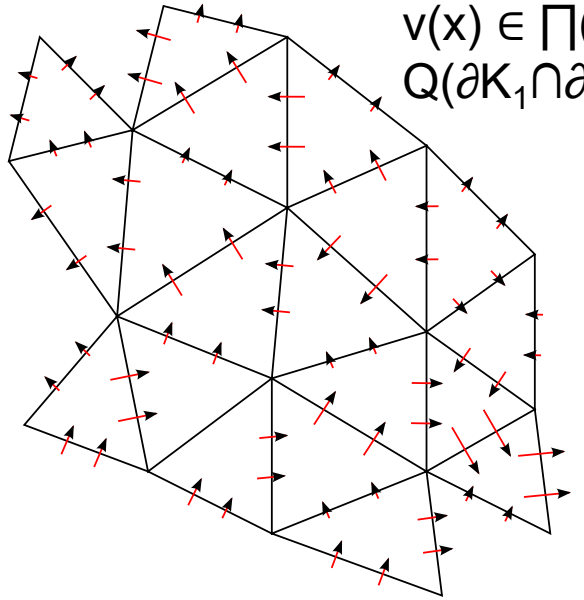
Recent developments : MOOSE framework

- New discretization methods:
  - Finite volumes
  - Side variables
  - Hybridized Discontinuous Galerkin
  - Isogeometric Analysis (IGA)
- Mixed-simulation type restarts
- Arbitrary region meshing : 2D and 3D
- P-refinement
- Constructive Solid Geometry support
- Multiple linear and nonlinear systems
- Standardized Physics syntax
- Functional Mockup Interface (see end of presentation)

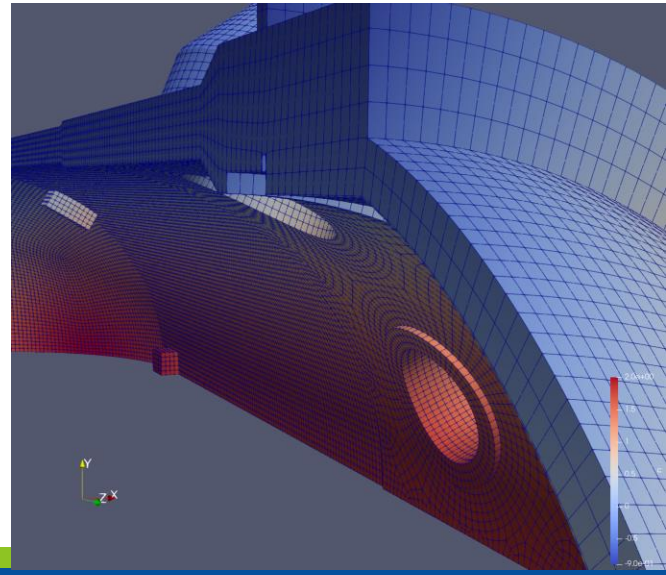
New discretization methods

- Finite volumes
  - Cell-centered variables, with 1st and 2nd order interpolations
- Face variables
  - SIDE_HIERARCHIC: Hierarchical polynomials on element sides, discontinuous at vertices and edges
  - Requires mesh elements with side nodes: EDGE2/3, QUAD8/9, HEX27, TET14, PRISM20 and PYRAMID18
  - Isometric Geometric Analysis (IGA)
  - Solving on higher order geometries
- Hybridized discontinuous Galerkin (HDG)
  - Higher order, conservative, and reduced system size compared to regular DG

Face variables on element side nodes



Reactor Pressure Vessel mesh (body-fit mesh, extraction operator a straight 1-to-1 mapping to NURBS

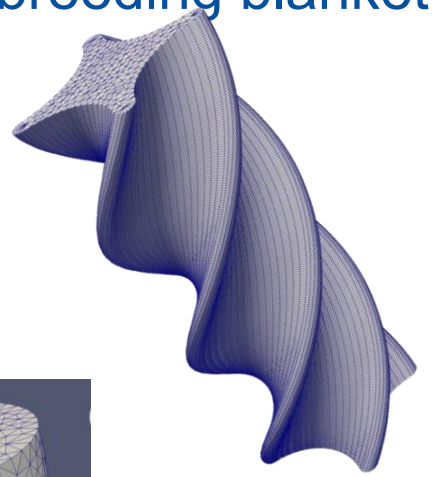


Figures from Roy Stogner

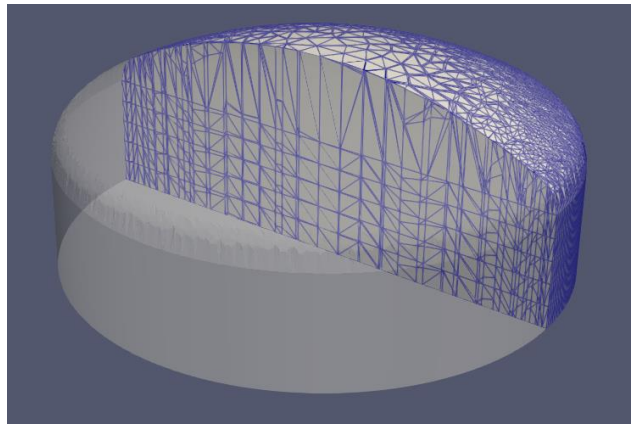
Arbitrary region meshing

Hexagonal element paving of a 2D square

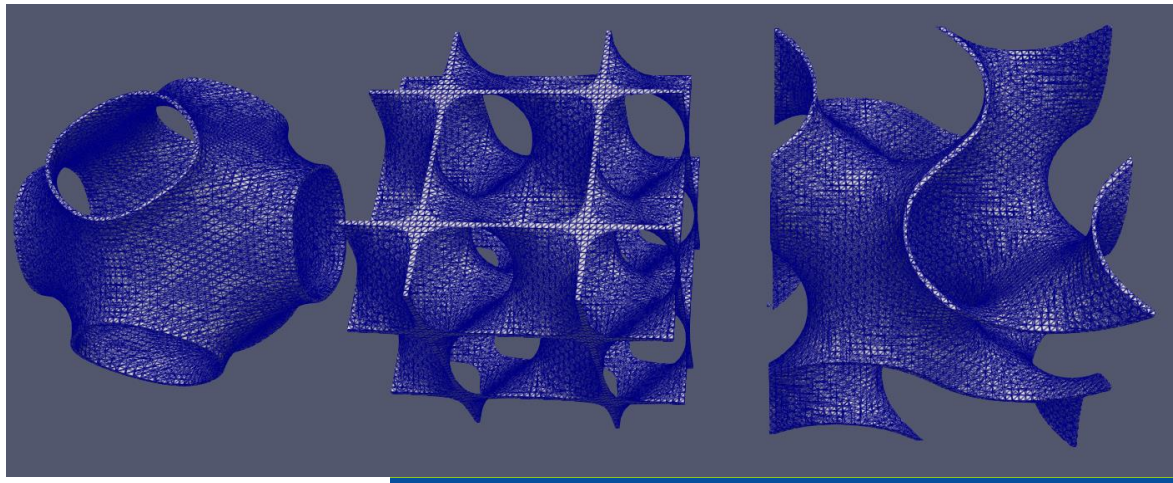
- 2D triangle and 3D polyhedral meshing were added to MOOSE by leveraging Poly2Tri and Netgen respectively
- Polygonal and polyhedral meshes now supported in libMesh
- Used in reactor & breeding blanket meshing



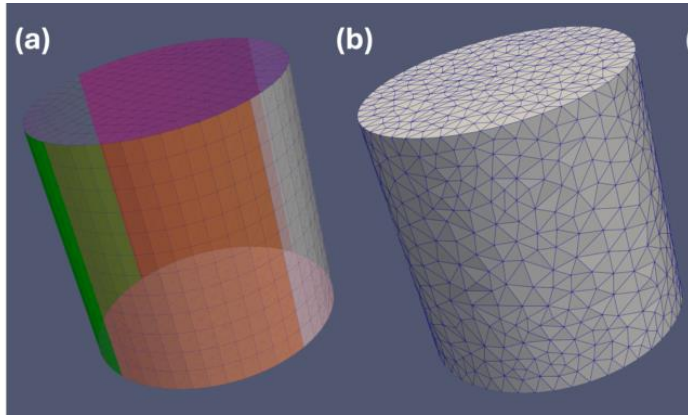
A torispherical dome of a typical containment meshed in MOOSE



Triply periodic minimal surfaces (TPMS) meshed in MOOSE



General surface triangulation

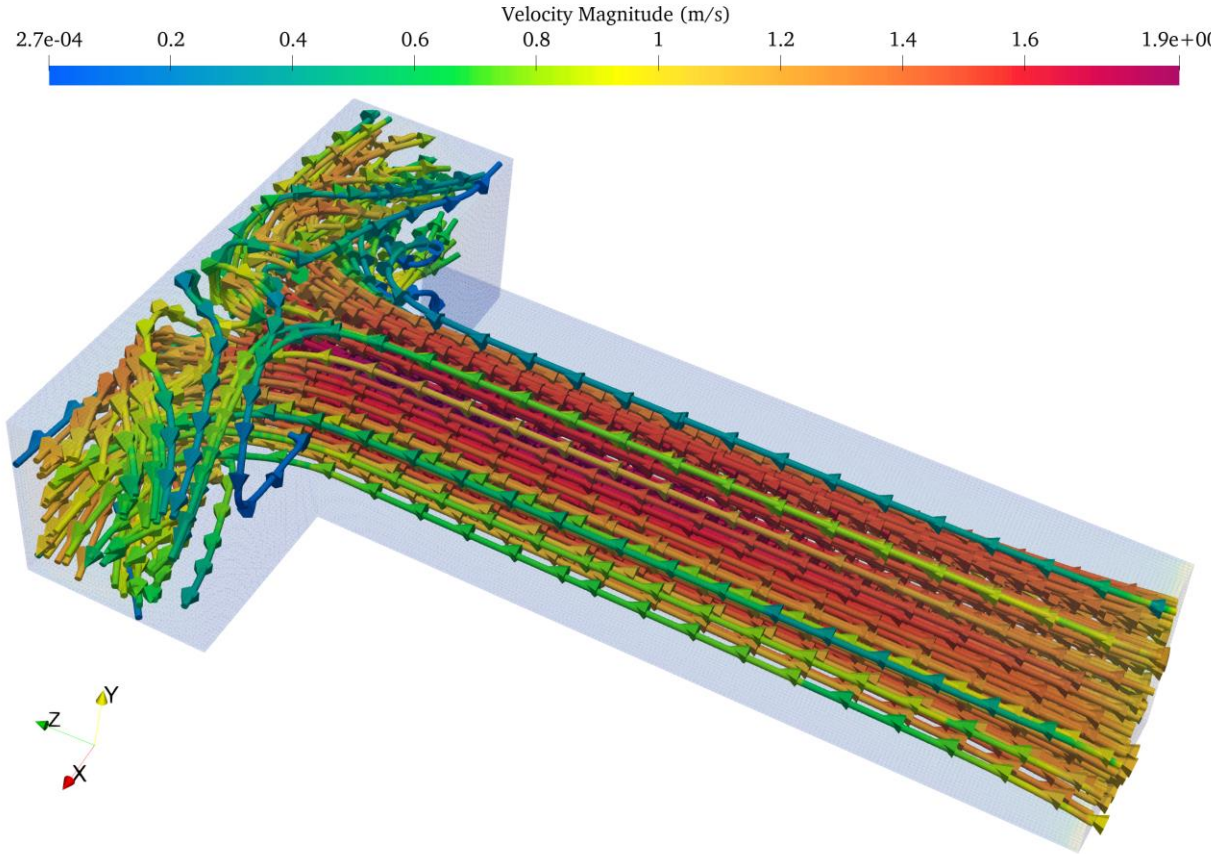


Mesh of the Lightbridge accident-tolerant fuel

Figures from Yinbin Miao and Patrck Behne

More Efficient Solves: Multiple solver Systems in the Same Input

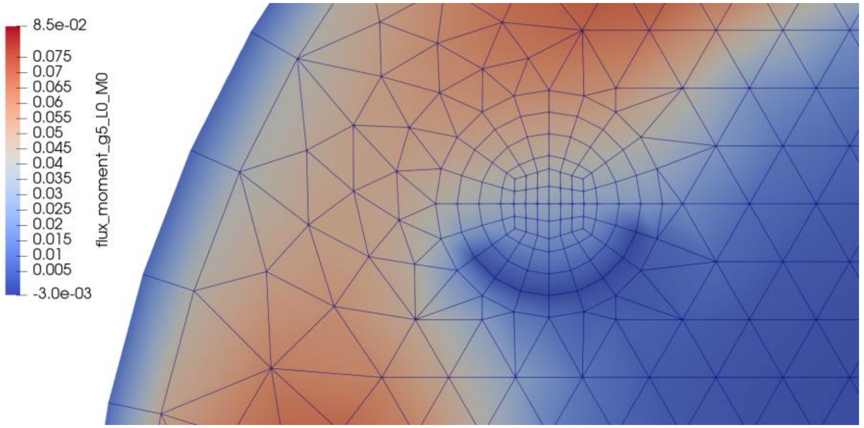
- A monolithic Newton's method can perform poorly when an initial guess is far from the solution
- This has been observed particularly in our TH area when adding required coarse-mesh turbulence models for advanced reactor types such as MSRs
- Some traditional Computational Fluid Dynamics solver algorithms which segregate variables can fare much better
- This has been leveraged by the TH area to create a native MOOSE Semi-Implicit Method for Pressure Linked Equations (SIMPLE) implementation applicable to high Reynolds number flows

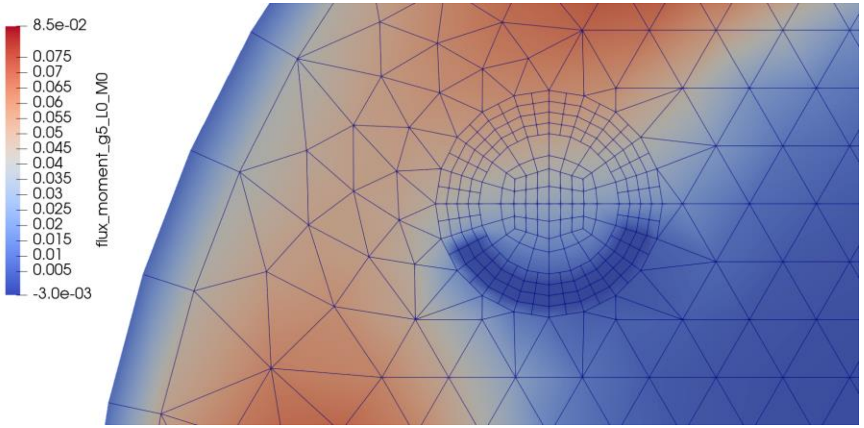


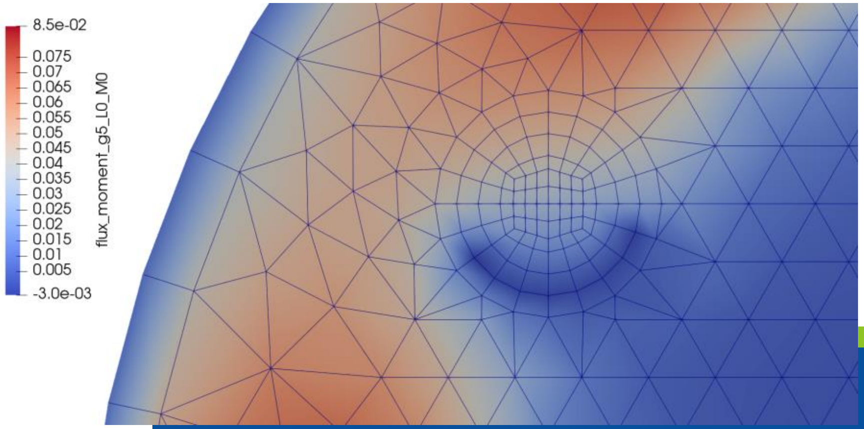
Increased Accuracy from Polynomial Basis Refinement (p-refinement)

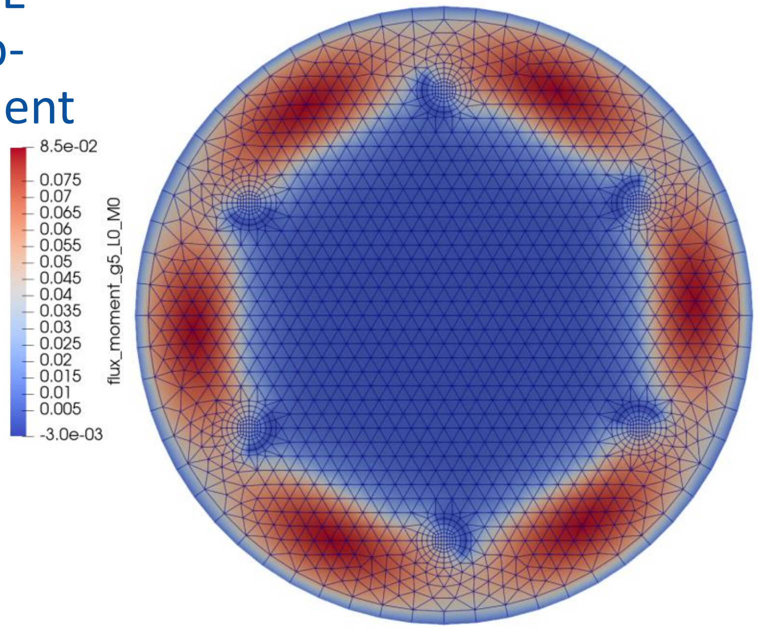
- p-refinement can be used in neutron transport to model neutron attenuation in absorbers without cusping effects
- p-refinement allows for exponential rates of convergence in accuracy
- Along with p-refinement, MOOSE added the capability to disable p-refinement for select finite element families
- Enables arbitrarily high order discontinuous families and low order Lagrange in the same simulation

Control drums in a micro-reactor.
Thermal flux
Base case (top)
H-adaptivity (middle)
P-adaptivity (bottom)
Core (below)





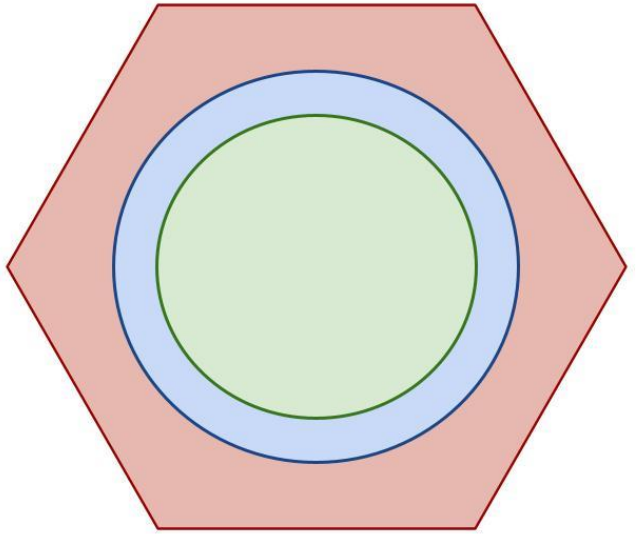




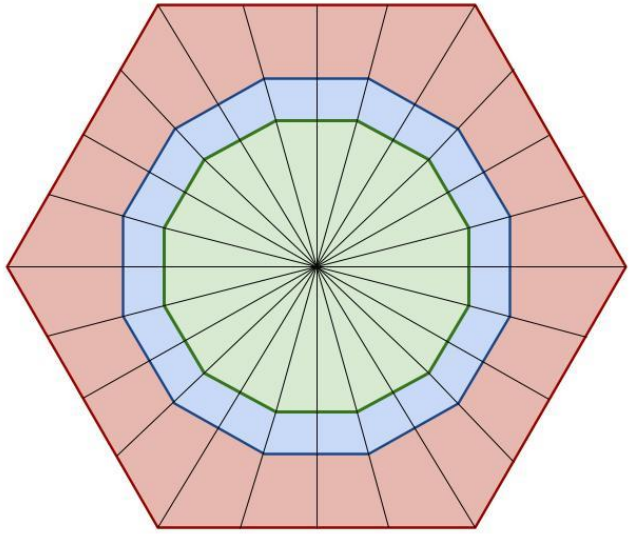
Constructive Solid Geometry (CSG) Support

This support aims to automate the geometry connection between MOOSE/MOOSE-based apps and Monte Carlo (MC) codes for more **reliable and efficient group cross section generation, code-to-code comparisons, generation of reference solutions, and multiphysics workflows**.

- This workflow builds CSG models from the mesh generator data, not the FEM output.
- The output is a *generic* definition which can then be connected to specific MC codes to translate it to code-specific input.



CSG

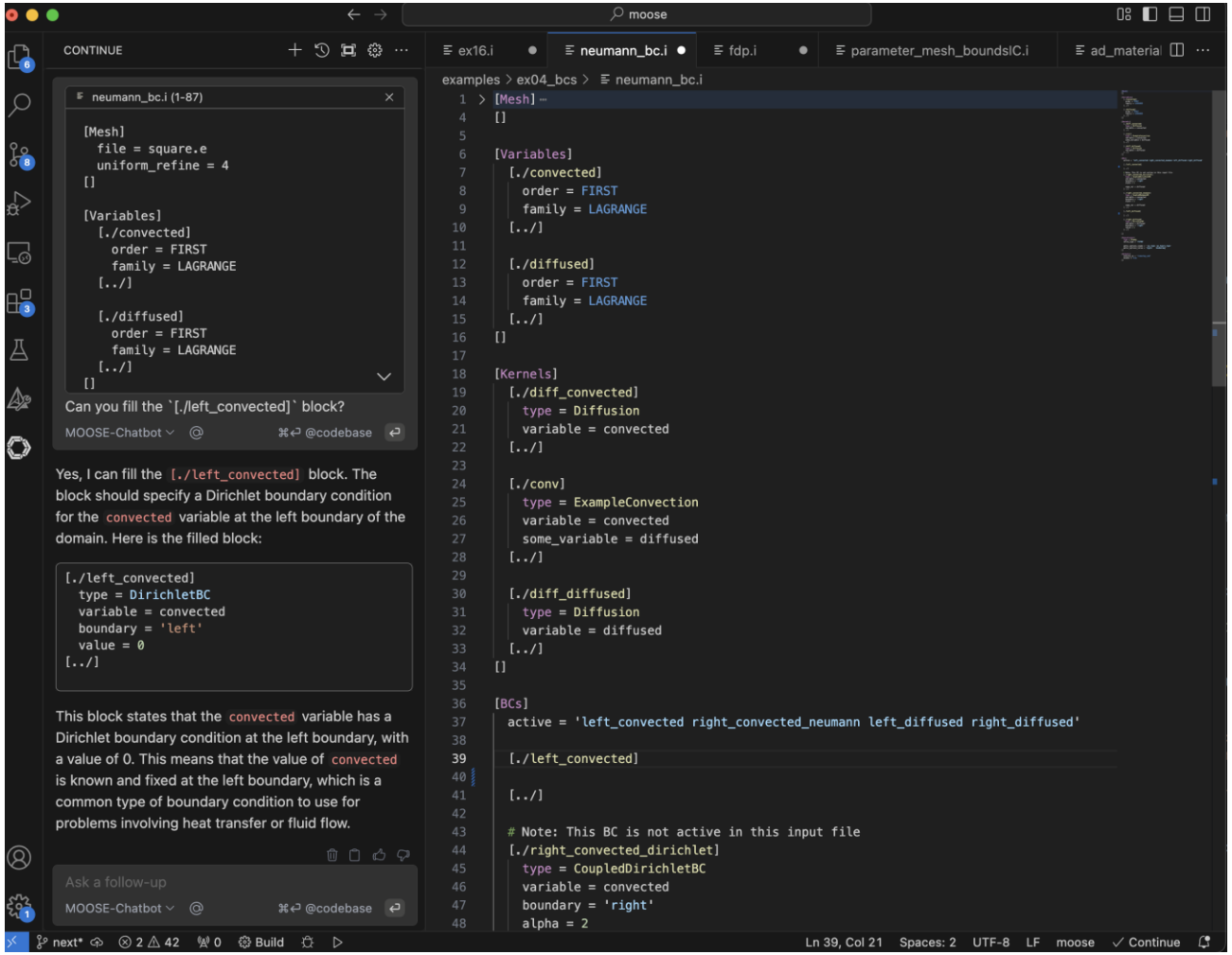


FEM

Slide by K. Kiesling and S. Kumar

Using AI to support research

- Keyword & examples search
- Provide recommendation on complex topics such as preconditioning systems
- Generate a template-type input file from the physics description
- Fine tune the input file via an interactive conversation



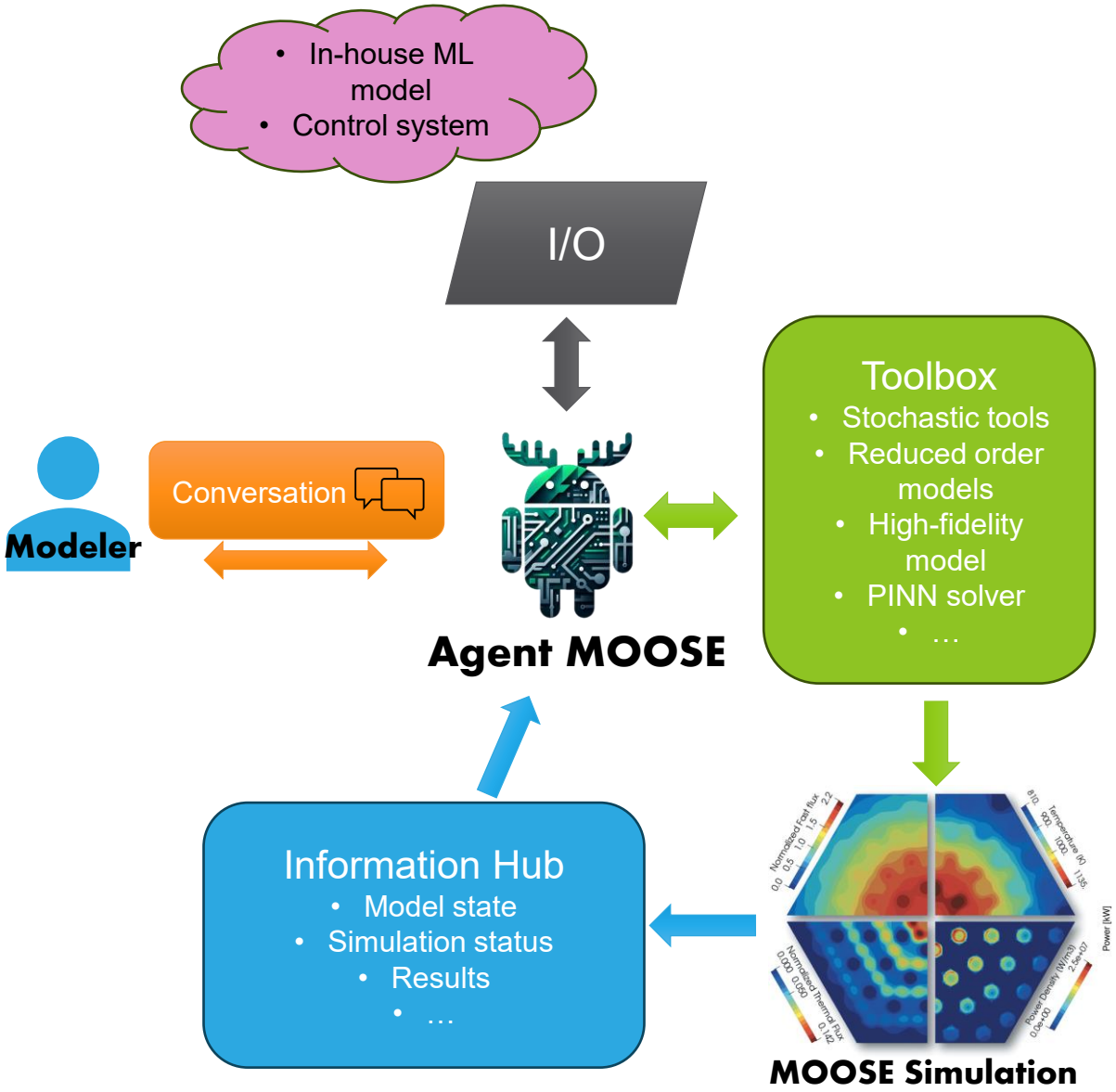
Related work:

1. B. Roziere, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, T. Remez, J. Rapin and others, "Code llama: Open foundation models for code," {arXiv preprint arXiv:2308.12950, 2023.
2. J. A. Prenner and R. Robbes, "Automatic Program Repair with OpenAI's Codex: Evaluating QuixBugs," arXiv preprint arXiv:2111.03922, 2021.

Research by Mengnan Li (INL)

Agent MOOSE will support you

- A live agent babysits ongoing MOOSE simulation
- Provide model state information, simulation status, model results upon user request
- Assistant multiphysics simulations with model fidelity switch, correlation selection, convergence issue, spatial/time resolution changes
- Offer a versatile I/O to Python-based AI/machine learning (ML) model



Research by Mengnan Li (INL)

Graphics processing units (GPU) for distributed computing

CONNECT: Creation Of Next-generation Nuclear Energy Computational Technology

- Meant to connect DOE Office of Science (DOE-SC) post-Exascale Computing Project (ECP) with DOE-NE
- ECP tools best leverage some of the new large-scale computing clusters
- ECP tools which the CONNECT framework team explored:
  - MFEM
  - libCEED
  - Kokkos
- MFEM backend has been added to MOOSE in March 2025 from UKAEA developments

See: Electromagnetic Simulations in MOOSE using the MFEM Finite Element Library 12/11/25, 9:55 AM

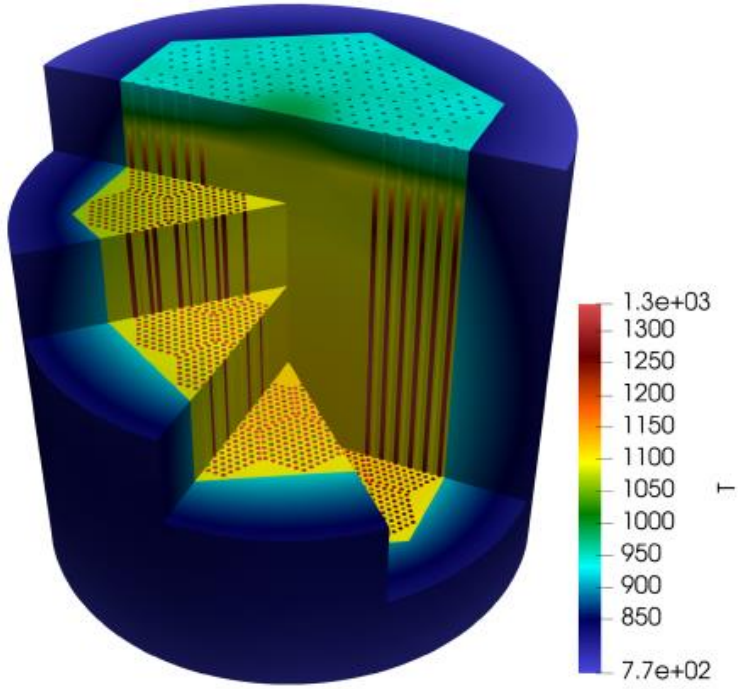
By Alexander Blair (UK Atomic Energy Authority)

Kokkos GPU assembly + solve preliminary performance

- Simplified microreactor heat conduction problem (2.2M elements with 1.98M DoFs)
- ~150 CPU core equivalent for residual computation, and ~100 CPU core equivalent for Jacobian computation (effective values excluding shape function evaluation)

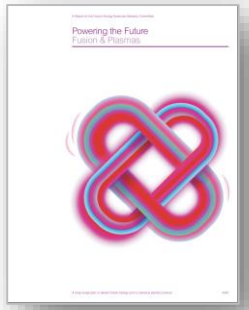
MOOSE	CPU			Kokkos			
PETSc Vector	CPU		CUDA			Kokkos	
PETSc Matrix	CPU	CPU	CUDA	HYPRE	CPU	Kokkos	HYPRE
Number of Nonlinears	5				5		
Number of Linears	20				49		
PETSc Solve (s)	45.87	6.423	8.074	4.120	6.551	21.26	4.109
PCSetUp	13.20	2.814	5.951	1.893	2.842	19.01	1.905
MatConvert	0.970	0.910	4.007	-	0.927	17.10	-
PCApply	30.36	0.449	0.452	0.453	0.450	0.458	0.453
Compute Residual (s)	81.21	0.266	0.269	0.265	0.270	0.268	0.267
Compute	-	0.253	0.255	0.252	0.254	0.252	0.251
Compute Jacobian (s)	150.2	1.872	1.583	1.932	1.895	1.577	1.943
Compute	-	1.104	1.107	1.107	1.118	1.112	1.109
Preallocate	-	0.379	0.405	0.752	0.387	0.342	0.760
Assembly	-	0.385	0.068	0.067	0.386	0.115	0.067

* AMD EPYC 9654 Single Core vs NVIDIA L4



Research by Namjae Choi and Alexander Lindsay (INL)

Deploying MOOSE for nuclear fusion R&D



- Fusion Energy Sciences Advisory Committee (FESAC) report, 2020, “Powering the Future, Fusion & Plasmas.”
  - “An essential component underpinning this effort is a strong theory and computation program, including the advancement of **multiscale, multiphysics theory and modeling capabilities** necessary to predict the complex interactions between numerous **plasma**, **material**, and **engineering processes.**”
- Experimental data is rare and costly to obtain, making design particularly challenging. Predictive computational frameworks need to be an integral part of an accelerated and cost-effective design process by modeling fusion system performance in simulated environments.
- There is a need to support the growth of fusion engineering.

MOOSE is well-suited for this challenge

FESAC, Powering the Future, Fusion & Plasmas, 2020 report

Tritium Migration Analysis Program, version 8

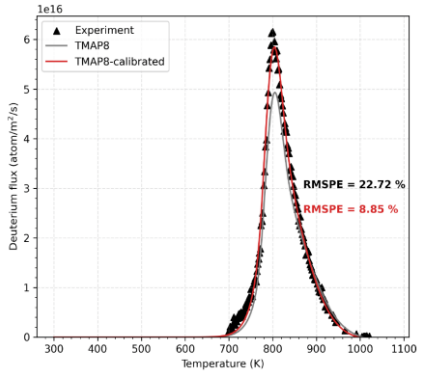
TMAP8

Model Development and V&V

- TMAP4 and TMAP7, although widely used, have significant limitations.
  - TMAP8 enables high fidelity, multi-scale, 3D, multiphysics simulations of tritium transport.
  - TMAP8 is open source, Nuclear Quality Assurance level 1 compliant, expands on TMAP4&7 V&V, offers user support and massively parallel capabilities.

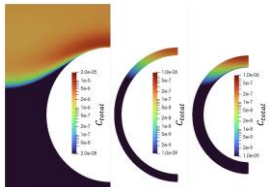
Stochastic tools

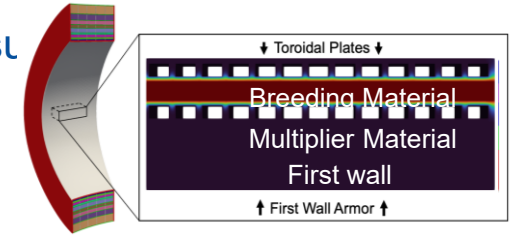
The use of stochastic tools supports model calibration, experimental analysis, and quantification of uncertainty.



High-fidelity simulations in complex geometries

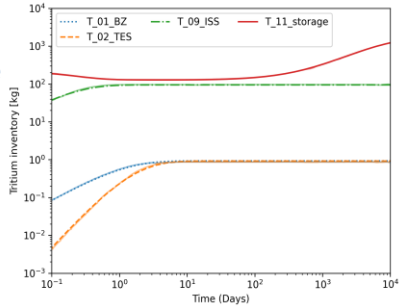
TMAP8 has been used to model complex systems su as a divertor monoblock and a full 3D breeder blanket section.





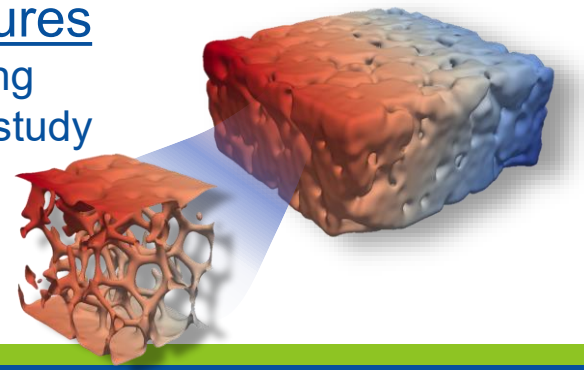
Fuel cycle

TMAP8 support fuel cycle calculations for rapid, full system analysis to track tritium inventory – going beyond TBR. These fuel cycle models can be informed by lower length-scale simulation in a fully integrated manner.



3D pore microstructures

Using phase field modeling capabilities, TMAP8 can study the effect of real and simulated pore microstructure on tritium transport.

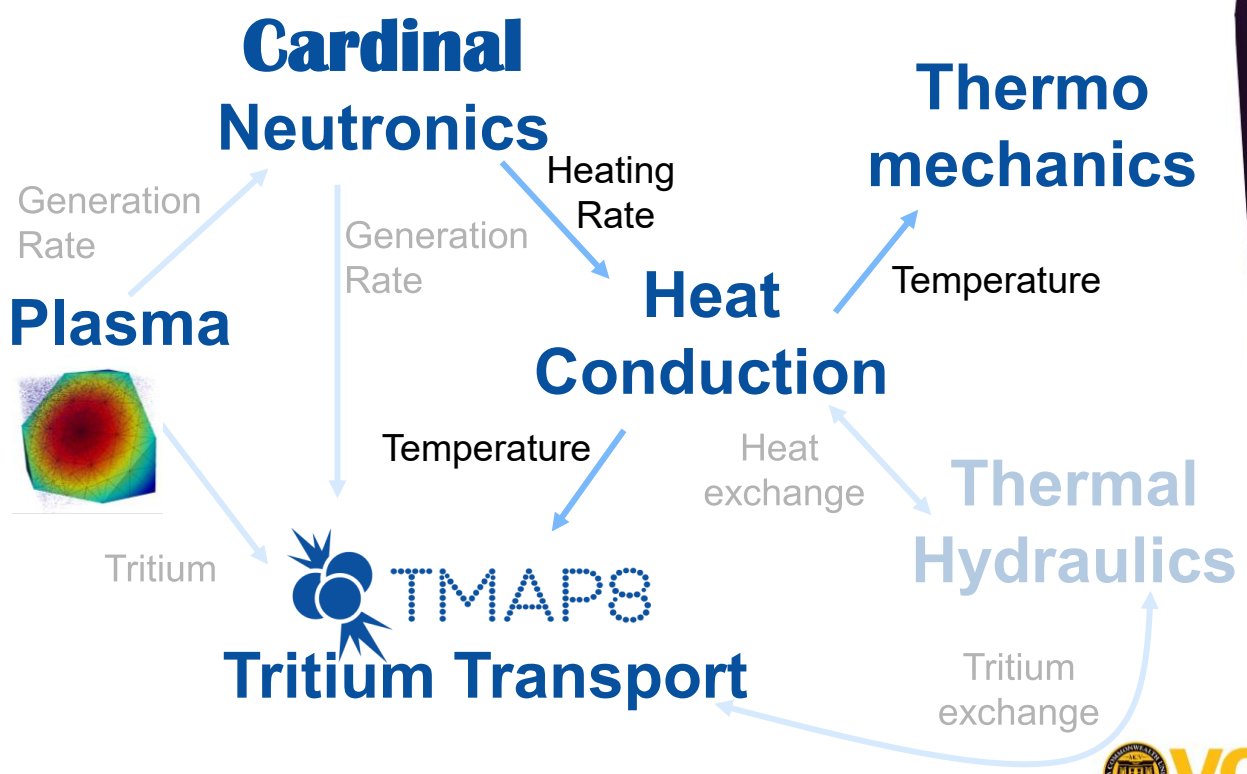


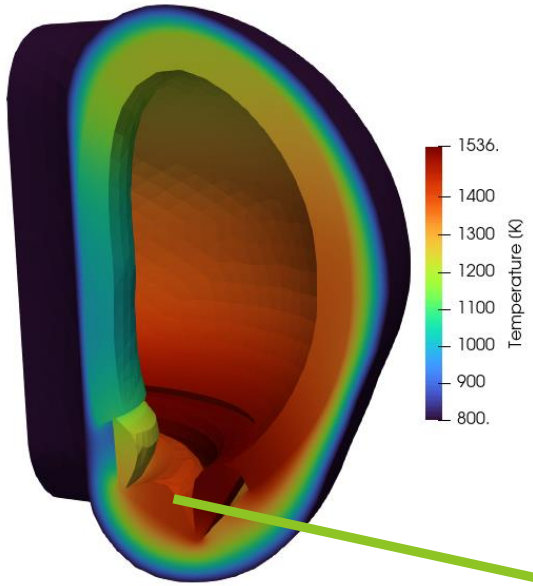
IDAHO NATIONAL LABORATORY

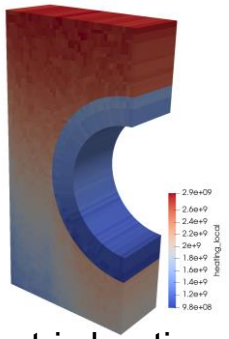
Software for Advanced Large-Scale Analysis of MAgnetic confinement for Numerical Design, Engineering & Research (SALAMANDER)

SALAMANDER

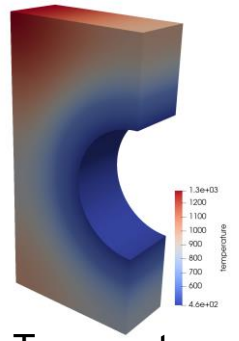
- SALAMANDER couples key physics for blanket systems
  - SALAMANDER enables high-fidelity, multi-scale, 3D, multiphysics simulations
  - SALAMANDER is open source, Nuclear Quality Assurance level 1 compliant, offers user support and massively parallel capabilities.



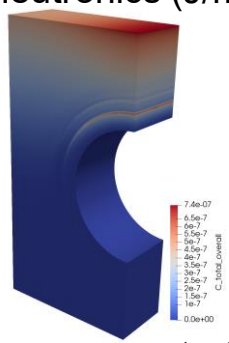




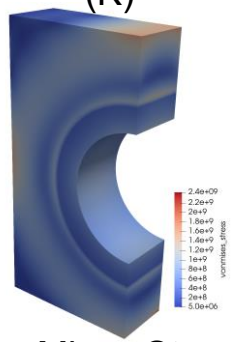
Volumetric heating rate from neutronics (J/m^{3}s)



Temperature (K)



Tritium concentration (atomic fraction)



Von Mises Stress (Pa)

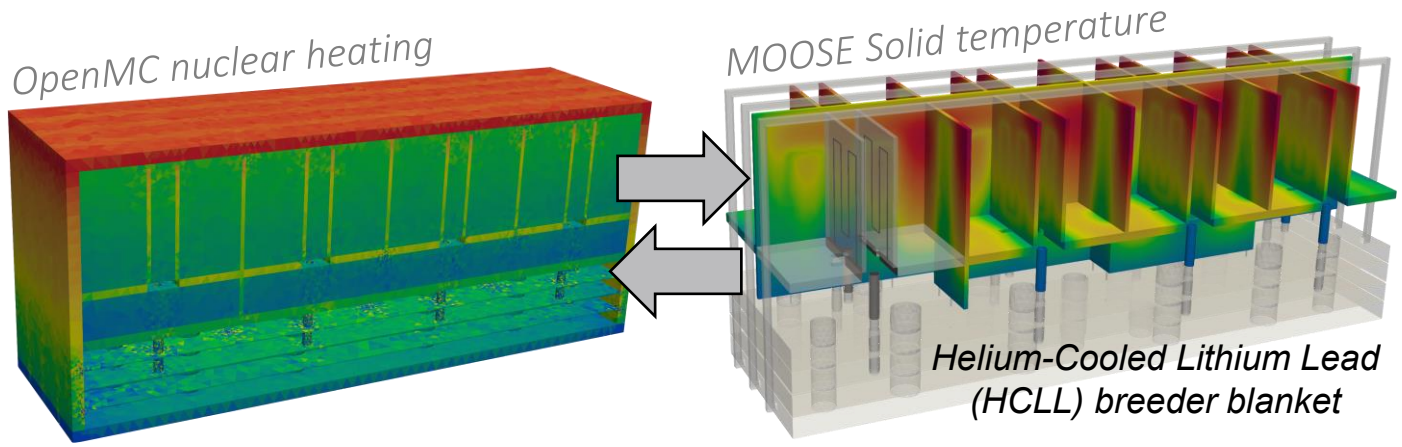
SALAMANDER GitHub repository: https://github.com/idaholab/SALAMANDER. Idaho Falls, April. 30, 2025

IDAHO NATIONAL LABORATORY

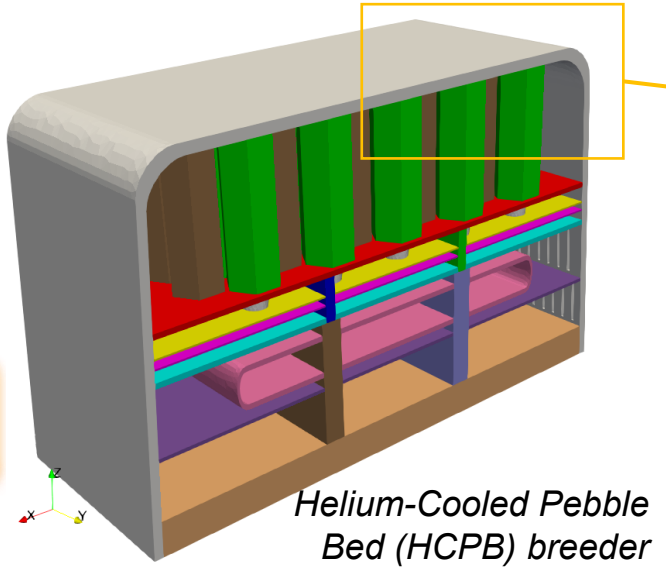
Cardinal: Fusion Multiphysics with AMR

K. Sawatzky et al., "Adaptive Mesh Refinement Applied to Unstructured Mesh Tallies in Cardinal" *Proceedings of ANS* (2025)

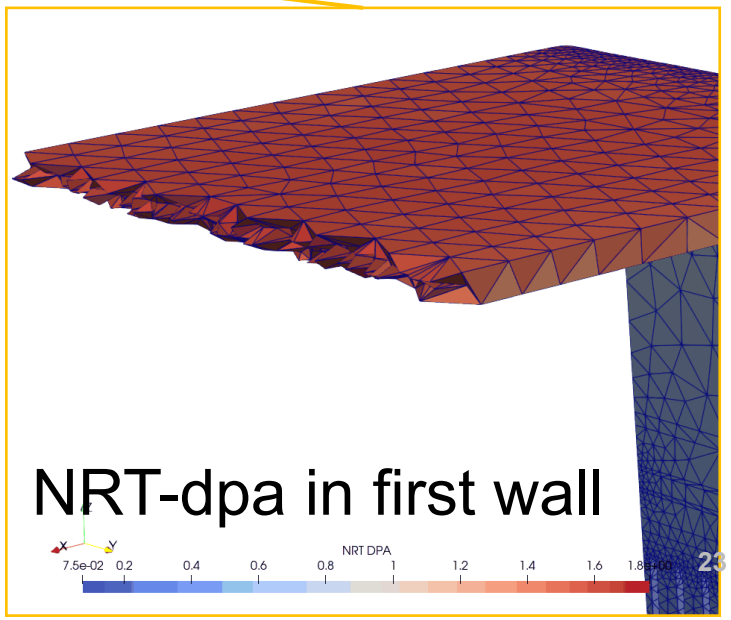
The **resolution** of neutronics fields **drives many multiphysics design metrics** for fusion devices such as magnet heating, material damage, and heating of high-heat-flux components.



Unstructured adaptive mesh refinement (AMR) with Monte Carlo tallies.



Spatial resolution is refined in high-gradient regions.



OpenMC neutron-photon transport

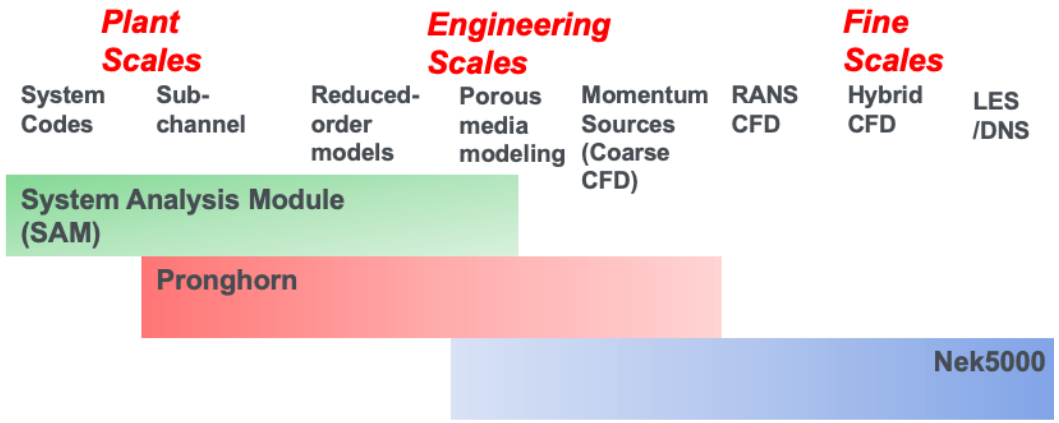
Adapting tally meshes

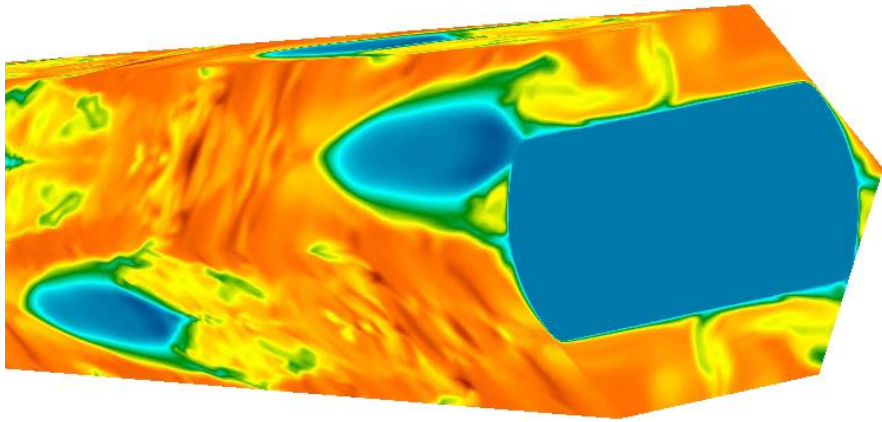
Tallies (nuclear heating, dpa, ...)

MOOSE's Adaptive Mesh Refinement system

NEAMS Thermal-hydraulic Codes overview

- **SAM**
  - Trustworthy and practical plant-level system analysis tool for advanced reactors
  - Advances in software environments and design, numerical methods, and physical models thanks to **MOOSE**.
- **(Open)Pronghorn**
  - Engineering scale environment build on **MOOSE**
  - Coarse CFD, subchannel and distributed resistance
- **Nek5000 / NekRS**
  - Open Source, Spectral element high-fidelity code
  - Proven scalability beyond a million MPI ranks (Gordon Bell prize). Now GPU-capable.





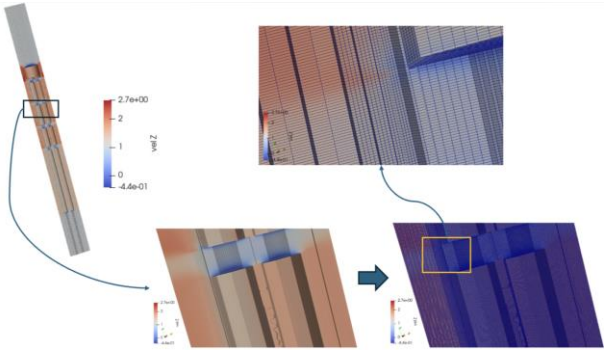
Flow around a twisted pin. (Nek)

See:
- ARC Divertor and Heat Exchanger Thermal Hydraulic Modeling using the Nek5000 CFD Code
12/9/25, 2:55 PM
- Fusion Energy System ARC System Modeling using SAM
12/12/25, 11:45 AM
Pr. Lane Carasik

DOE NEAMS; DOE ART; DOE AFC

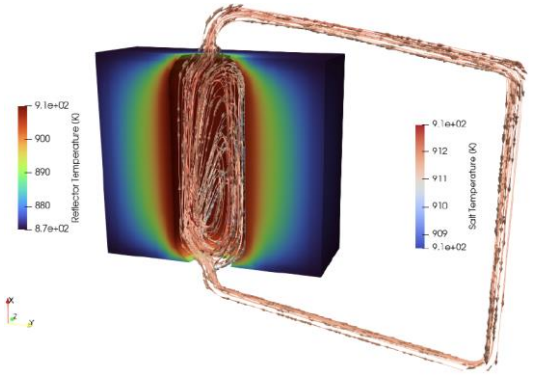
Releasing OpenPronghorn

- **OpenPronghorn** is a growing state-of-the-art computational fluid dynamics (CFD) code supporting nuclear fission and fusion reactor thermal-hydraulics engineering.
- As development progresses, OpenPronghorn will provide:
  - **Validation base:** Incompressible and weakly compressible one- and two-phase flows, with and without heat transfer, for nuclear reactor applications—supporting future commercial-grade dedications.
  - **Porous-media modeling:** Models and correlations for coarse-mesh analysis of advanced reactors.
  - **Turbulence models:** One- and two-phase turbulence modeling, including re-laminarization and buoyancy-driven corrections, to support safety-case modeling for advanced fission and fusion applications.
  - **Phase change & transport:** Aerosol transport and solidification/melting models to support design-basis accident analyses.
  - **Species & electrokinetics:** Species tracking and electrokinetic modeling to support fuel-performance calculations in liquid-fuel systems.

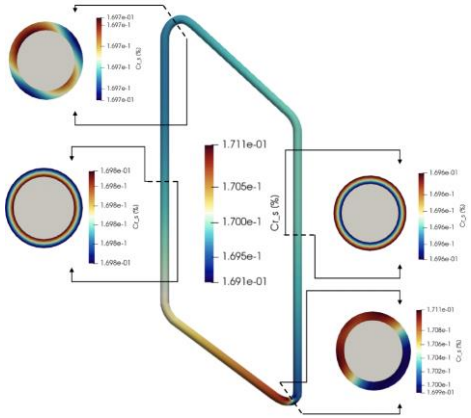


High Fidelity CFD modeling of Accident Tolerant Fuel irradiation experiment using OpenPronghorn

Two-phase flow in pump priming showing flow stream vectors and gaseous phase interface using OpenPronghorn



Conjugate heat transfer in molten salt reactor operation using OpenPronghorn



Corrosion profiles in molten salt loop using OpenPronghorn

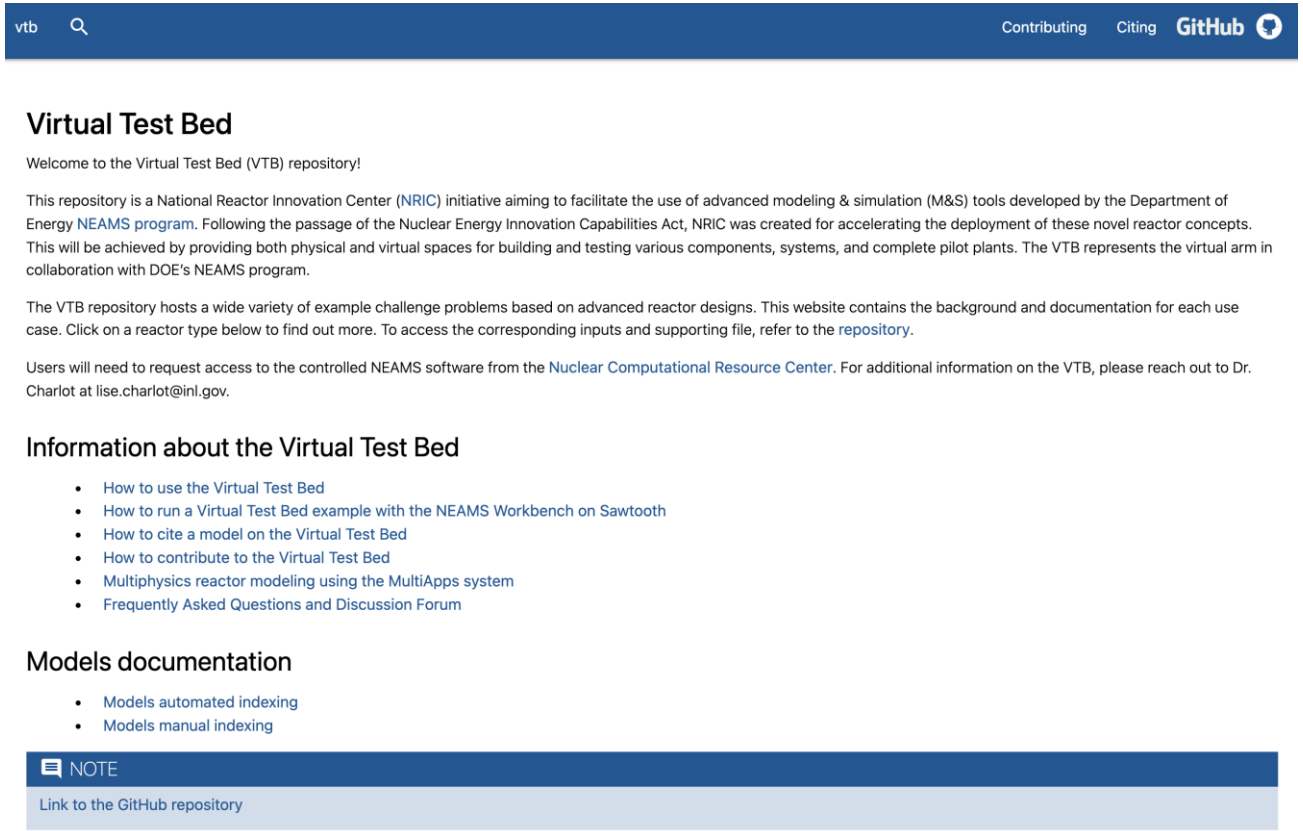
Nuclear Reactor Technology | Reactor Systems and Heat Transport
Mauricio.TanoRetamales@inl.gov; Victor.CoppoLeite@inl.gov; David.Reger@inl.gov
Scientific Computing and Artificial Intelligence | Computational Frameworks
Peter.German@inl.gov; Ramiro.Freile@inl.gov; Guillaume.Giudicelli@inl.gov; Mengnan.Li@inl.gov; Logan.Harbour@inl.gov

National Reactor Innovation Center Virtual Test Bed

https://mooseframework.inl.gov/virtual_test_bed

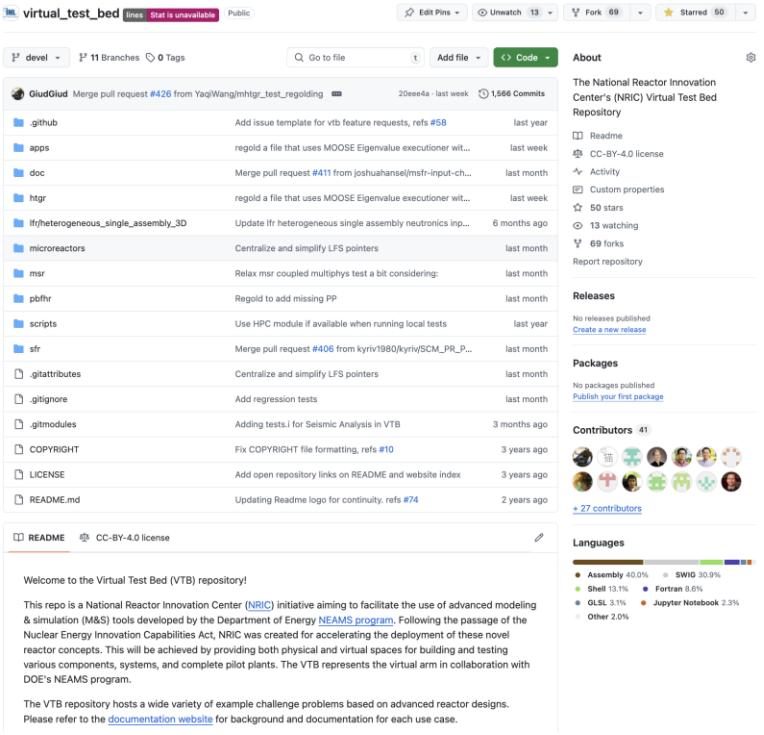
1. Documentation

Detailed explanation of models



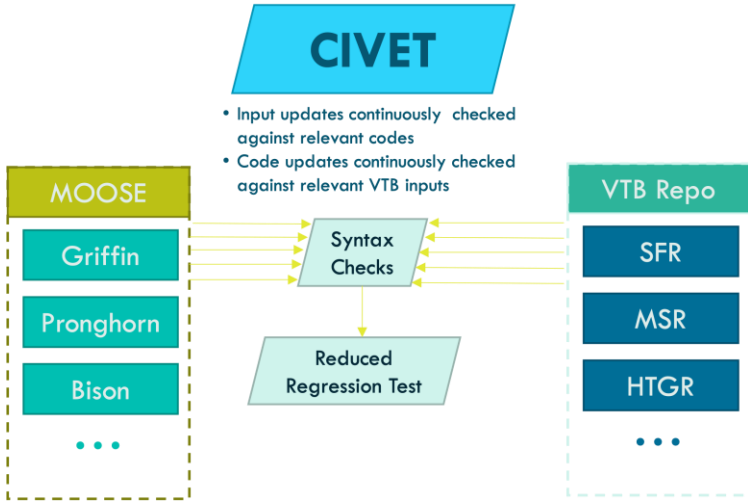
2. GitHub Repo

Hosts the models



3. Integration with Code Development

Continuous testing of models against codes using CIVET



Functional Mockup Interface

Modern engineering simulations involve complex workflow, coupling of different modeling tools. To efficiently communicate between different software packages, we need a standardized interface for information transfer.

MOOSE: A power open-source framework for solving complex multiphysics engineering problems.

FMU/FMI Interface: A standardized tool-independent framework for the exchange and co-simulation of dynamic models in various engineering applications.

Slides kindly provided by Dr. Mengnan Li

Functional Mock-up Interface (FMI)

What is FMI: An open, tool-independent standard for exchanging dynamic simulation models. It is a popular industry tool

Applications: Automotive Manufacturer Co-Simulation, Aerospace Digital Twins & virtual Electronic Control Unit, and Integrated Energy System Optimization & Control

Purpose: Enables interoperability between different simulation software

Advantages: cross-platform, IP protection, simplified integration, open standard evolution & industry adoption

Interface Types:

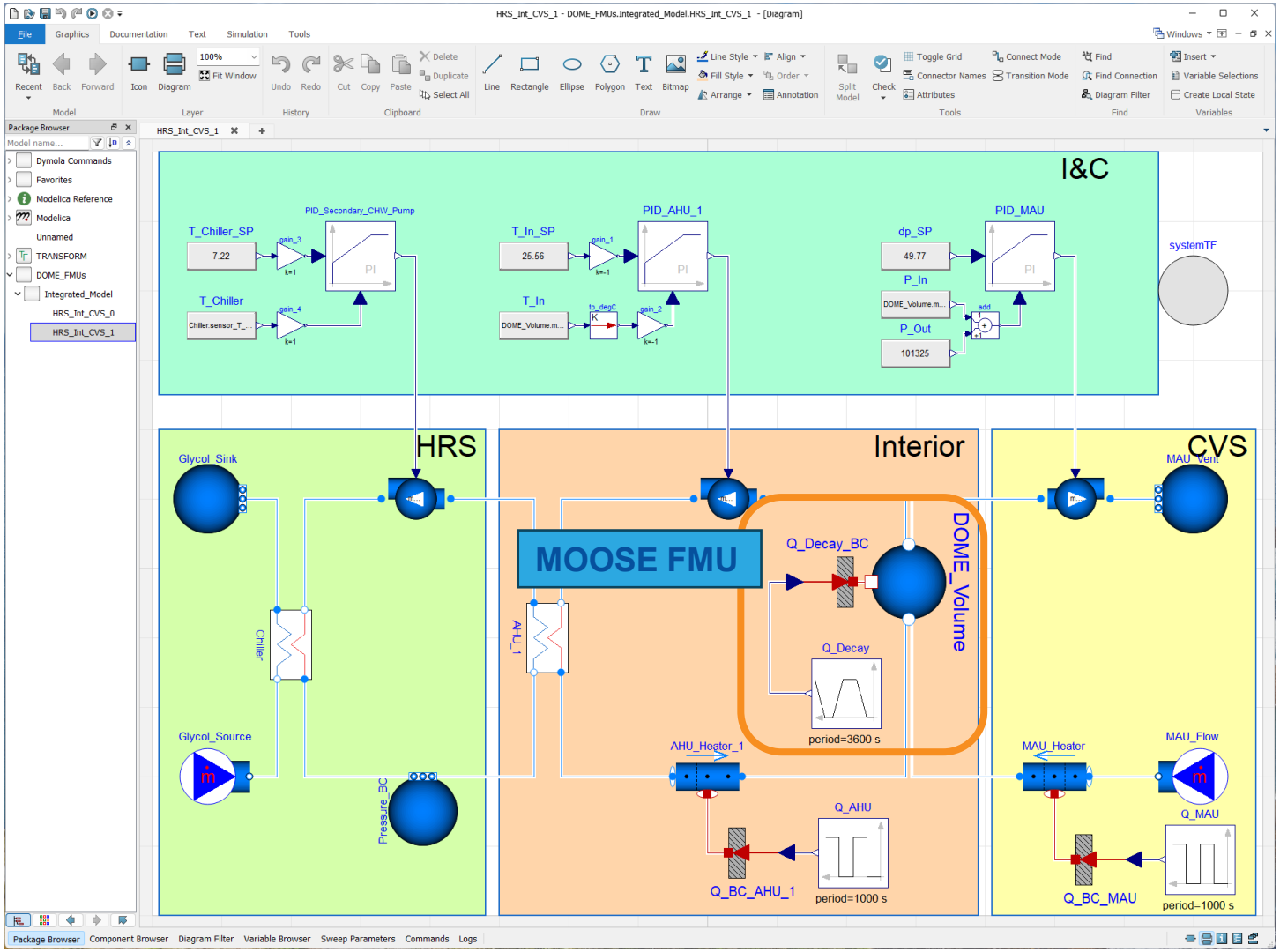
- Model Exchange (ME): The FMU provides equations and algorithms, and the simulation environment provides the solver.
- Co-Simulation (CS): The FMU packages the model with its own numerical solver together. A FMU simulation platform(e.g MathWorks' Simulink, Open Modelica) controls the simulation, coordinating the simulation by synchronizing with the FMU's discrete time steps and exchanging data at defined communication points

FMU Application Example

Modelica controls the simulation

Facility model composed of a mix of FMU & Modelica models

Instrumental provides input for the facility model: forming a digital twin

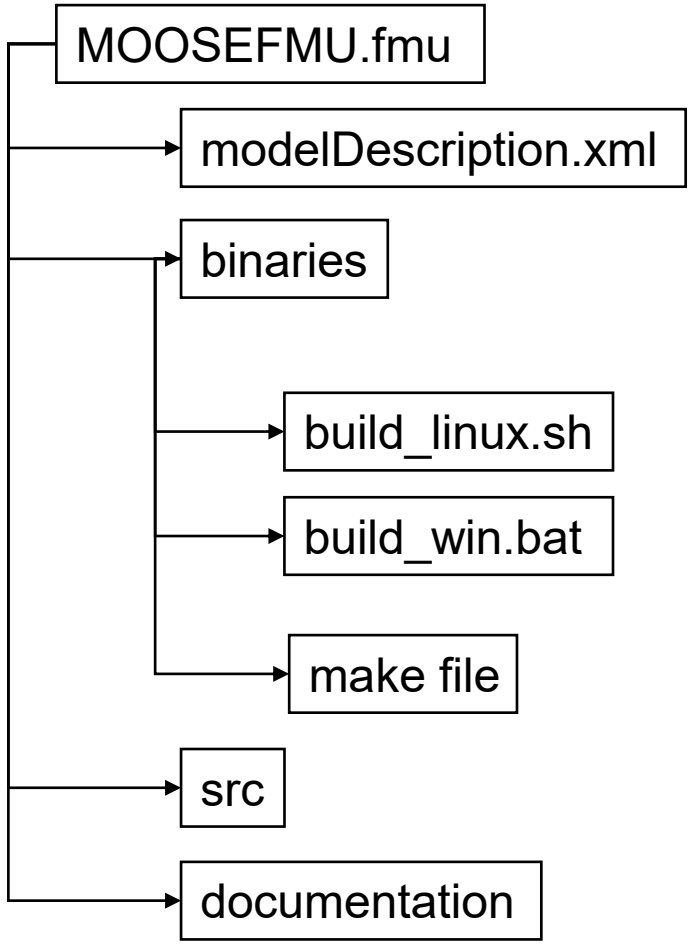


Slide credit: Mengnan Li

Functional Mock-up Unit (FMU)

What is FMU: A ZIP archive defined by the FMI standard, including an XML model description file, compiled binaries or source code to enable standardized model exchange and co-simulation across diverse simulation tools.

- Packaging:
  - ModelDescription.xml: Define interface, variables, parameters, capabiilites
  - Compiled Model Code: implementing the standard FMI functions using FMU API packages
- Other resources



Explore FMI Compatible Interface with MOOSE

1. **Utilize MOOSE Web Server Control**
   - MOOSE provides a web server control interface.
   - It allows modifications to the MOOSE input file using a Python controller.
2. **Develop the FMU Base Class (MOOSE2FMU.py)**
   - Define the fundamental structure for handling MOOSE-FMU coupling.
   - Handle MOOSE executions: Set up MOOSE web server control
   - FMUBase: Provides abstract and concrete methods for FMU interaction.
3. **Build Customized MOOSE FMU based on user needs (MOOSETest.py)**
   - Inherit the MOOSEFMUBase class
   - Initialization: Set default parameters and register variables based on problem needs.
   - XML Representation: Define FMU model structure.
   - Define the input/output: Specify all the variables and parameters the user wants to expose to external codes.
   - Step Function: Define simulation step execution.

MOOSE input file:

```
[Controls/web_server]
type = WebServerControl
port = 12345
execute_on = 'INITIAL TIMESTEP_BEGIN'
[]
```

MOOSE FMU example:

```
result = simulate_fmu(
"MooseTest.fmu",
start_time=t0,
stop_time=t1,
start_values={
'flag': 'TIMESTEP_END',
'moose_executable': '../../../moose_test-opt',
'moose_inputfile': 'fmu_diffusion.i',
'server_name': 'web_server',
'max_retries': 10},

input = input_data,
output = ['time','moose_time', 'diffused']
)

time = result["time"]
dt = result["moose_time"]
diff_u = result["diffused"]
```

Challenges and Complexities

- **Parallelism:**
  - FMI standard lacks direct support for parallel execution within an FMU. Managing MOOSE's internal MPI/OpenMP parallelism within FMU wrapper needs careful design
- **Dependencies & Portability:**
  - MOOSE relies on external libraries (e.g. libmesh, PETSc). These must be packaged correctly within the FMU for portability across different OS/environments
- **Coupling PDE Solvers:**
  - FMI primarily exchanges scalar/vector variables, while MOOSE contains many spatial field variables. Strategies are needed to deal with those quantities (e.g. using postprocessors, stochastic tool)
- **Fidelity & Coupling Schemes:**
  - Supporting different model fidelities and enabling MultiApp adds complexities

Conclusions

- MOOSE is a fast-growing open-source framework for multiphysics and multi-fidelity simulations, with users in a growing number of fields
  - ~500 user-developers on slack
  - ~800 PRs a year
- Applications developed by the community leverage the physics modules to quickly set up simulations for targeted problems
- Fusion device engineering using MOOSE tools is pursued by several institutions
- Support for FMI standard 2.0 has been developed and to be merged in 2026
- Staying in touch:
  - MOOSE Newsletter on website
  - Github **Discussions** for all user questions
  - **moosedevelopers** slack for developers

Other presentations using MOOSE tools

An Integrated Multi-physics Platform for the LIBRTI Facility — 12/9/25, 11:00 AM

Helen Brooks (UKAEA)

Divertor Monoblock Multiphysics Analysis Using the SALAMANDER Code — 12/9/25, 4:40 PM

Lane Carasik (Virginia Commonwealth University)

Bringing together integrated simulation, surrogates, and data to support digital twins for fusion engineering — 12/10/25, 9:55 AM

Casey Icenhour (Idaho National Laboratory)

Electromagnetic Simulations in MOOSE using the MFEM Finite Element Library — 12/11/25, 9:55 AM

By Alexander Blair (UK Atomic Energy Authority)

Software for Advanced Large-scale Analysis of MAgnetic confinement for Numerical Design, Engineering & Research (SALAMANDER) — 12/12/25, 9:30 AM

Pierre-Clément Simon (Idaho National Laboratory)

Fusion Energy System ARC System Modeling using SAM — 12/12/25, 11:45 AM

Lane Carasik (Virginia Commonwealth University)

Idaho National Laboratory

Battelle Energy Alliance manages INL for the U.S. Department of Energy's Office of Nuclear Energy. INL is the nation's center for nuclear energy research and development, and also performs research in each of DOE's strategic goal areas: energy, national security, science and the environment.

Software Quality Assurance

- INL is audited yearly by ASME on the compliance of its quality assurance program to NQA-1
- MOOSE and all the applications developed at INL follows SQA practices defined by the INL SQA Plan 4005
- In practice for MOOSE:
  - Independent code reviews
  - Automated software quality documentation generation
  - Continuous integration achieved through:
    - An extensive test suite involving >50 configurations and 10k tests per config.
    - A dedicated testing cluster (3,500 cores)
    - Continuous integration services offered to other application teams
  - INL is not a qualified supplier, so Commercial Grade Dedication is performed by the end user