Recent progress in Kraken development
(Serpent relevant parts)

V. Valtavirta
VTT Technical Research Centre of Finland Ltd

Serpent UGM 2020 – Organized by TUM

Serpent with external couplings and control

past, present and future

V. Valtavirta
VTT Technical Research Centre of Finland Ltd

Serpent UGM 2020 – Organized by TUM

Contents

- What do I mean: Some definitions
  - Coupling
  - Control
- Why is this topical: Kraken framework
- Where did we come from, where are we now, what’s in the future?

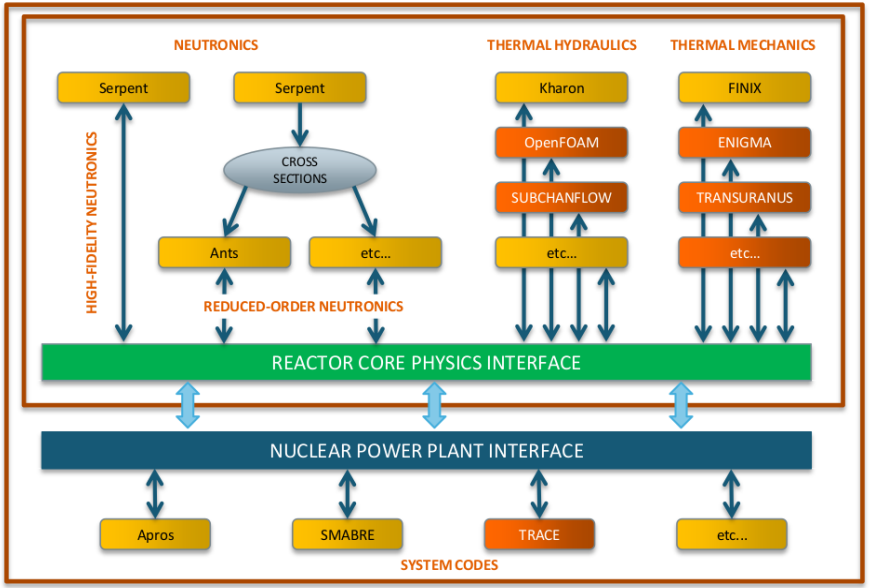
Definitions: Coupling

Working together and exchanging data with other solvers in order to solve a multi-physics problem

Definitions: Control

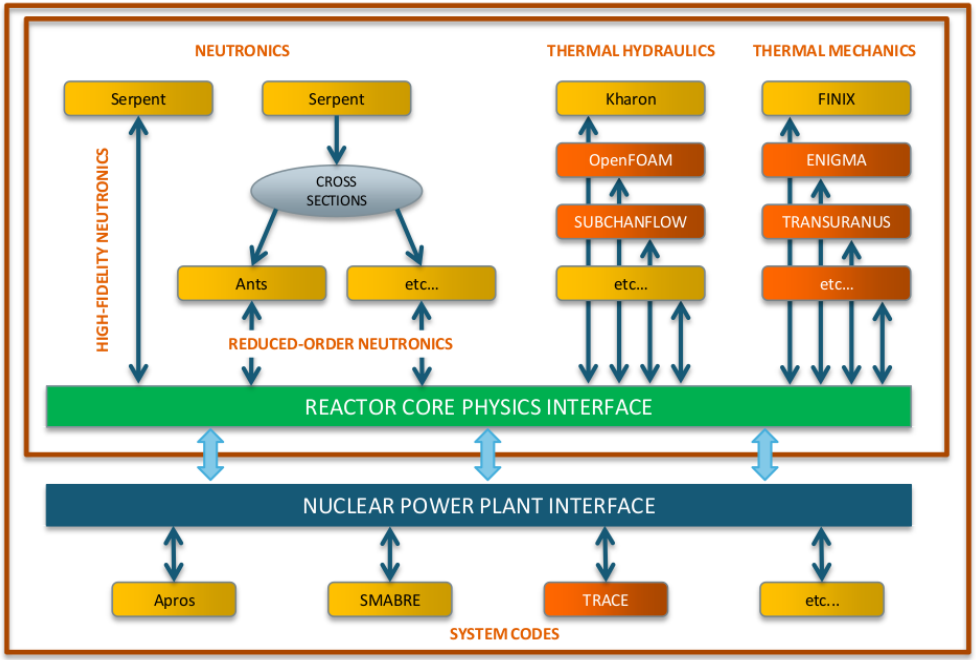
Controlling what Serpent does next during its runtime via an external tool based on some externally evaluated criteria.

The Kraken framework



The Kraken framework

- VTT's new computational reactor analysis framework[1,2,3].
- Built for:
  - Independent deterministic safety analyses.
  - Evaluation of new reactor concepts.
  - Generation of input data for system codes, e.g. Apros.
- Intended to replace our current tools in some years.
- Burnup and transient capable reactor simulator using either Serpent or Ants as the neutronics solver.



A schematic of the planned couplings in the Kraken framework. Yellow boxes indicate solvers developed at VTT while orange ones are potential third party couplings. VTT is a contributor to the OpenFOAM project.

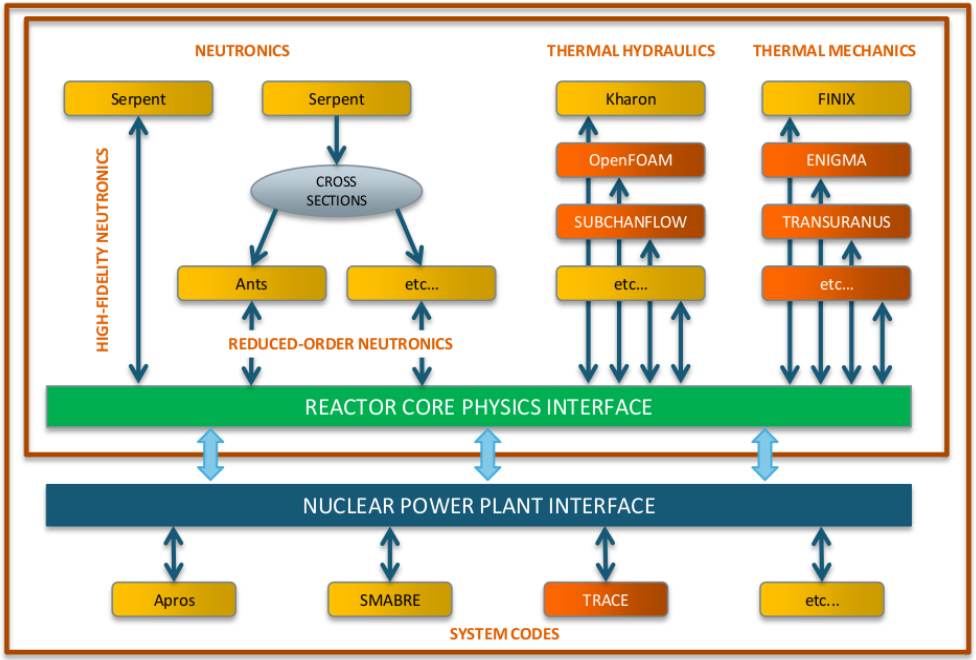
[1] V. Valtavirta *et al*. "Kraken – an Upcoming Finnish Reactor Analysis Framework". *Proc. ANS MC2019*. Portland, OR, USA, Aug. 2019.

[2] V. Valtavirta *et al*. "High Fidelity and Reduced Order Solutions to an SMR-Level Progression Problem with the Kraken Computational Framework". *Proc. PHYSOR2020*. Cambridge, United Kingdom, 2020

[3] V. Valtavirta *et al*. "Validating the Serpent-Ants calculation chain using BEAVRS fresh core HZP data". *Proc. NENE2020*. Portorož, Slovenia, September 7–10, 2020

The Kraken framework

- New requirements for Serpent.
- Serpent needs to be applied for:
  - Operating cycle analyses:
    - Criticality search by boron, control rods, coolant flow rate or power.
    - Evaluation of reactivity coefficients, control bank worths, shutdown marginal etc.
    - Fuel shuffling.
  - Transient analyses:
    - Actuation of control systems based on detector responses.
- Serpent is a particle transport tool, not a reactor simulator.
  - Need to expose Serpent to much greater external control.

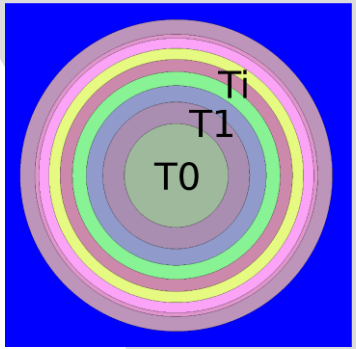


A schematic of the planned couplings in the Kraken framework. Yellow boxes indicate solvers developed at VTT while orange ones are potential third party couplings. VTT is a contributor to the OpenFOAM project.

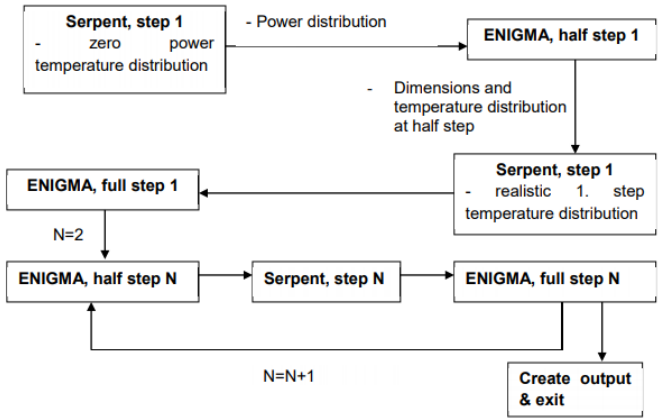
External couplings and control for Serpent
Past, present and future

Humble beginnings

- Direct input-file manipulation and manual re-execution of Serpent.
- No on-the-fly temperature treatment. (huge memory demand for multi-physics)
- No multi-physics interface. (re-write new mat-cards)
- No solution control. (re-execute with re-(pre-)processing of XS etc.)



- Thermosss (2010-) – for Serpent 1
  - Read radially tallied power distribution from Serpent output.
  - Solved radial temperature distribution.
  - Reduced radial temperature distribution into a step profile.
  - Wrote a new Serpent input with new temperatures of each fuel material tmp).
  - Reran Serpent



- SEN (Serpent-ENigma) (2012-) – (Serpent 2.1.5 –)
  - Coupled Serpent with fuel performance code ENIGMA.
  - Each axial layer of fuel rod represented by a separate 2D Serpent model.
  - Burnup capability.

Making progress

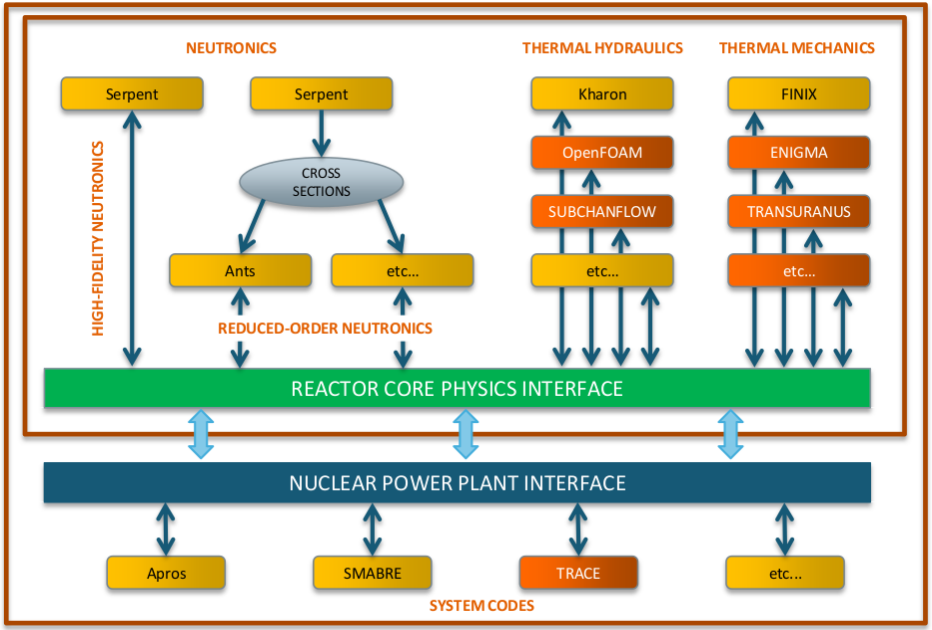
- Much work between 2011 – 2017.
  - Finnish Academy “Numerical Multi-Physics” (NUMPS) project provided major funding for 2012-2016.
- Two major doctoral theses related to Serpent development for coupled multi-physics calculations.
  - T. Viitanen, “Development of a stochastic temperature treatment technique for Monte Carlo neutron tracking” (2015)
  - V. Valtavirta, “Development and applications of multi-physics capabilities in a continuous energy Monte Carlo neutron transport code” (2017)
- The major building blocks were implemented during this period:
  - On-the-fly temperature treatment.
  - On-the-fly density treatment.
  - Multi-physics interfaces.
  - File and POSIX-signal based solution control.
- Last development presentation on the topic from UGM 2017:
  - Ville Valtavirta (VTT) - Coupled calculations with Serpent 2.1.29

Status for Serpent 2.1.31

- Description in the wiki
- Temperatures and densities are separated from material definitions using the multi-physics interface.
  - Allows complex temperature and density distributions from coupled solvers without intermediate processing.
- On-the-fly treatment of both temperature and density.
- Rudimentary control of solution flow via files or POSIX signals.
  - Iterate further / move to next step.
- Coupled calculations possible in
  - Steady state
  - burnup
  - and transient calculations

Coupling approach in the Kraken framework

- The solvers only communicate with the multi-physics driver (via a dedicated socket).
- The solvers provide information on their
  - Input and output fields.
  - Input and output variables
- The output data can be read and the input data can be set via the multi-physics driver.
- The multi-physics driver also tells the solvers what to do next:
  - Recalculate
  - Move to time *t*
  - Finalize and exit



Coupling approach in the Kraken framework

```
# --- Create input

solver_input = CodeInput("", "./Inputs/Serpent.inp")

# --- Create and initialize solver (Serpent)

serpent = Solver("Serpent", "/home/vvvillehe/Kraken/Serpent/sss2",
                 ["-omp", "8", ])
serpent.input = solver_input
serpent.initialize(port_nr)
```

```
# --- System power (input)

power_var = serpent.transferrables["power"]

# --- K-effective (output)

keff_var = serpent.transferrables["IMP_KEFF"]

# --- Assembly power detector (output)

det_var = serpent.transferrables["DET_assembly_power"]
```

- Our Python package Cerberus functions as the multi-physics driver.
- Solvers are first created and initialized.
- After the initialization, the solvers can be executed, their output-variables can be read directly to Python and input-variables can be also set directly from Python (during runtime!).
- This allows for adding complex control and feedback logic to simulations without extensive changes to Serpent source code.

Coupling approach in the Kraken framework

```
% --- Transformations for moving all eight
%     control rod banks

trans u CR12 name CR12  0.0 0.0 0.0
trans u CR13 name CR13  0.0 0.0 0.0
trans u CR21 name CR21  0.0 0.0 0.0
trans u CR22 name CR22  0.0 0.0 0.0
trans u CR23 name CR23  0.0 0.0 0.0
trans u CR32 name CR32  0.0 0.0 0.0
trans u CR33 name CR33  0.0 0.0 0.0
trans u CR43 name CR43  0.0 0.0 0.0
```

```python
# --- Names of control rod trans cards in Serpent input

cr_names = ["CR12", "CR13", "CR21", "CR22",
            "CR23", "CR32", "CR33", "CR43"]

# --- Control rod trans-cards (input)

cr_vars = []

for name in cr_names:
    var = serpent.transferrables[name]

    # --- Set initial height and send to Serpent

    var.value_vec = np.array((0.0, 0.0, 194.920))
    var.communicate()

    # --- Store variable for easy access

    cr_vars.append(var)
```

- Our Python package Cerberus functions as the multi-physics driver.
- Solvers are first created and initialized.
- After the initialization, the solvers can be executed, their output-variables can be read directly to Python and input-variables can be also set directly from Python (during runtime!).
- This allows for adding complex control and feedback logic to simulations without extensive changes to Serpent source code.

Input and output variables for Serpent currently exposed to Cerberus

- Input variables:
  - Normalization (power, powdens, flux, etc.)
  - Neutrons per cycle
  - Active cycles
  - Transformation cards
  - Equilibrium xenon and samarium flags
  - Atomic density iteration flag (iterable or fixed) and value
  - Coolant boron content (via atomic density iteration)

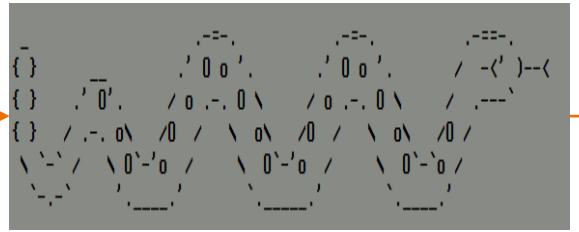
- Input fields:
  - Multi-physics interface temperature and density fields.
  - Xenon and samarium data for depletion zones

Input and output variables for Serpent currently exposed to Cerberus

- Output variables:
  - Current normalization (power, powdens, flux, etc.)
  - Current coolant boron content (via atomic density iteration)
  - All stat variables including detectors:
    - IMP_KEFF, …, TOT_FISSRATE, …, SIX_FF_ETA, …, ANA_THERM_TIME, …, etc. etc.

- Output fields
  - Multi-physics interface power tallies.
  - Xenon and samarium data for depletion zones

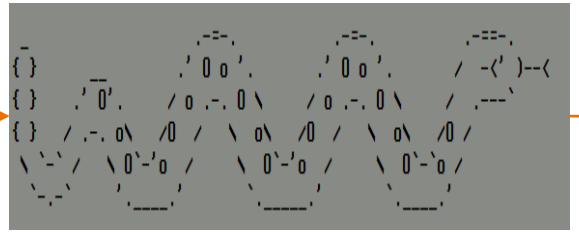
Our current uses for the new capabilities: Coupled multi-physics calculations



Temperature and density fields

Power from interfaces or detectors

Our current uses for the new capabilities: Criticality search (boron)

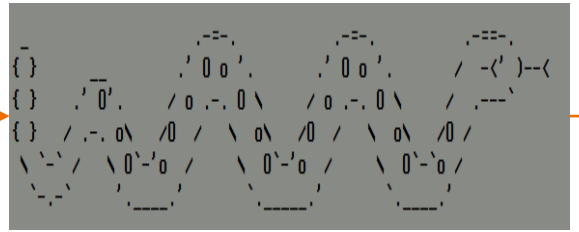


Atomic density iteration factor

k_{eff}

[redacted]

Our current uses for the new capabilities: Criticality search (power)

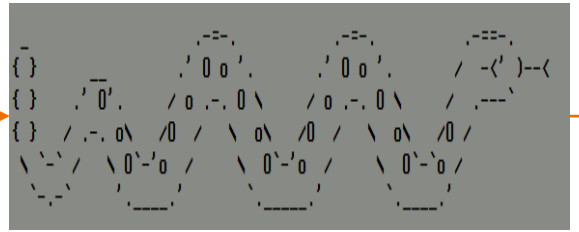


Temperature and density fields, total power for normalization

k_{eff}

power distribution

Our current uses for the new capabilities: Criticality search (control rods)

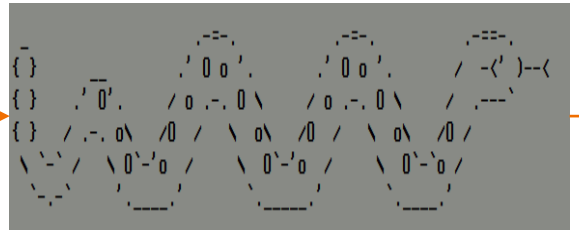


Control bank heights (via transformation)

k_{eff}

[redacted]

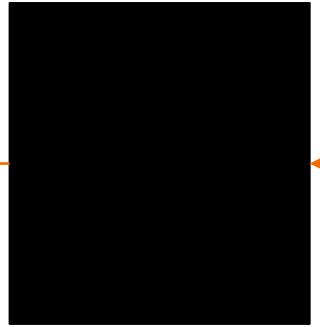
Our current uses for the new capabilities:
Criticality and power shape search (control rods)



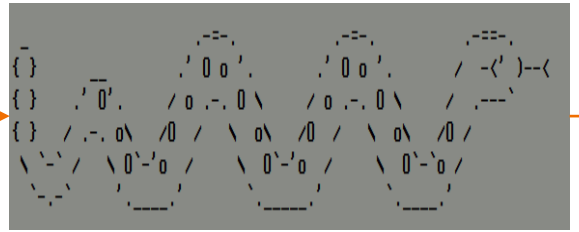
Control bank heights
(via transformation)

k_{eff}

power distribution



Our current uses for the new capabilities:
Reactivity coefficients and control rod worths

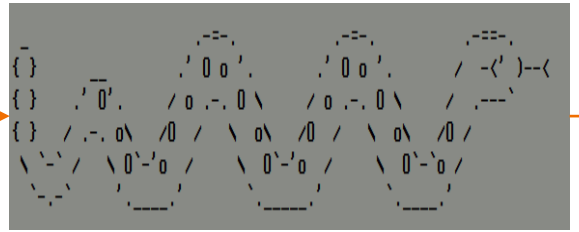


Material temperatures, densities, control bank heights etc.

k_{eff}

[redacted]

Future uses for the capabilities

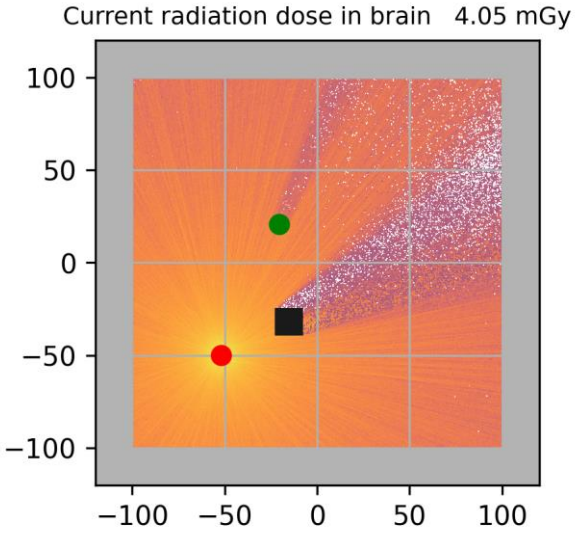


Modifications to Serpent model

Interesting output

Some complex feedback system e.g. neural network

Future uses for the capabilities



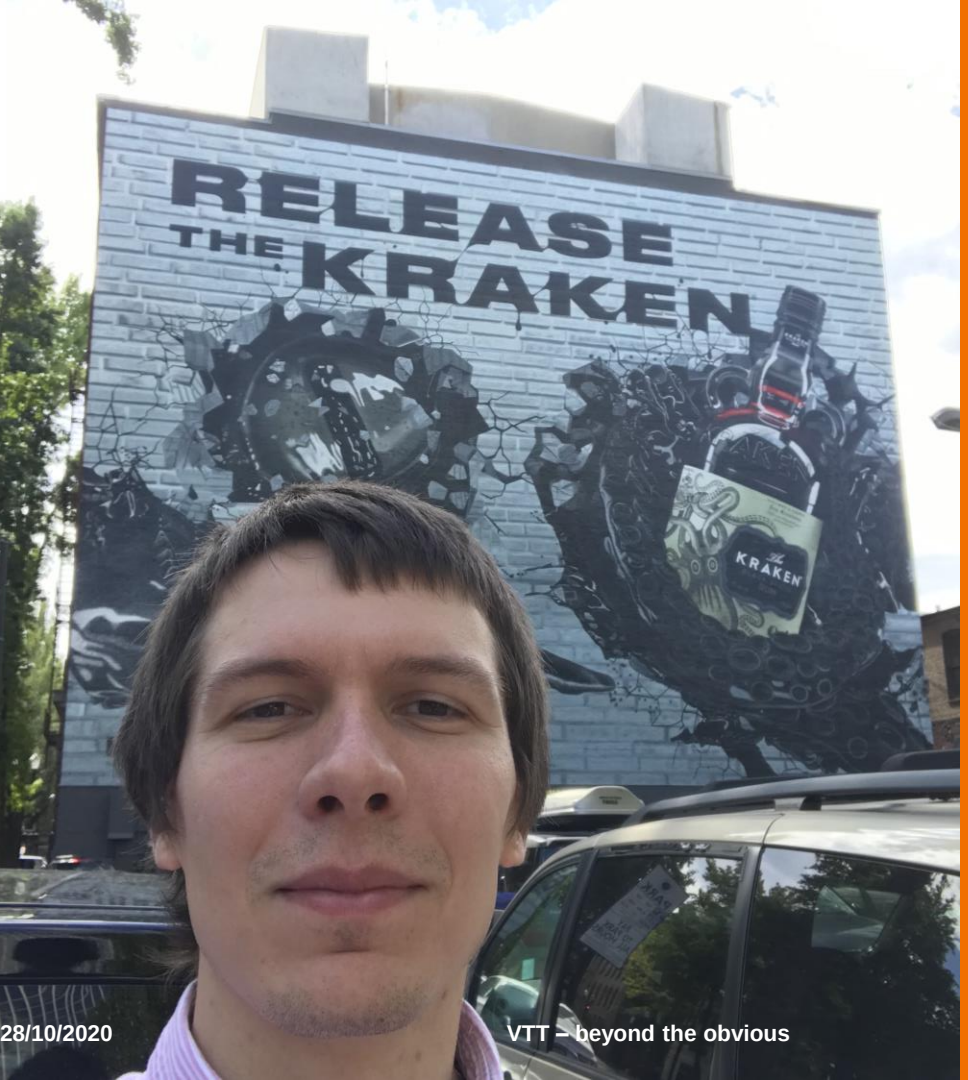
Future uses for the capabilities

- More output data can be exposed to be accessed from Cerberus.
- Additional input data can also be exposed with some limitations:
  - Updating some input data will need (re)processing.
  - The more complex the processing required, the more work it is to implement.
  - Straightforward (evaluated during transport):
    - Modifying surface parameters,
  - Complex (needs considerable processing):
    - Updating material compositions (excepting xenon, samarium etc.).
  - Problematic (needs memory allocation):
    - Adding new detectors, materials, surfaces etc. (can maybe be circumvented by adding some dummies in initial input)
- Some new “commands” will probably be included (e.g. write restart)

Summary and next steps

Summary and next steps

- The Kraken reactor analysis framework being developed at VTT uses Serpent as a particle transport (and burnup) module as a part of a bigger code system.
- This development also drives Serpent development into exposing larger parts of Serpent input/output variables to be modified/read during runtime.
- Serpent users and developers of Serpent based code systems can expect to benefit from this.
- After working through Serpent 2.2.0 licensing, we will look into licensing Cerberus and some other parts of the Kraken framework.



Thank you!

beyond
the obvious

Ville Valtavirta
ville.valtavirta@vtt.fi