RAVEN as Control Logic and Probabilistic Risk Assessment Driver for RELAP-7

C. Rabiti, A. Alfonsi, D. Mandelli, J. Cogliati, R. Martineau

Idaho National Laboratory

INTRODUCTION

The Next Generation of System Analysis Code (NGSAC) [1] aims to model and simulate the Nuclear Power Plant (NPP) thermo-hydraulic behavior with a high level of accuracy. In this respect, Idaho National Laboratory (INL) is developing a NGSAC (known as RELAP-7) which will allow to model NPP responses for a set of accident scenarios (e.g., loss of off-site power).

The control of the RELAP-7 simulation (e.g., control logic and simulation of failure events) is directed by the RAVEN software tool. More precisely, the scope of RAVEN is to implement the control logic dictated by:

- Plant control logic
- Operator actions (procedure guided)
- Stochastic phenomena such as failure models of specific components (e.g., valves and pumps) and human interactions.

While the control logic of the first two bullets is deterministic (i.e., it is solved deterministically through partial or ordinary differential equations), the one indicated in the last bullet is purely stochastic. In order to model this stochastic behavior of the system, RAVEN will implement two methodologies: Monte Carlo (MC) [2] analysis and Dynamic Event Tree (DET) [3] analysis.

In this paper it will be shown how the structure of the RAVEN software models easily the control logic and how it is possible to overload most of the code to implement also MC analysis. It will be also illustrated a simple example of application of control logic.

DESCRIPTION OF THE ACTUAL WORK

The Equation Set for the Control Logic

The general equation describing a dynamic system is provided in (1). This equation models the trajectory of the system (i.e., time evolution of the system) in the phase space.

$$\frac{\partial \bar{\theta}}{\partial t} = \bar{H}\left(\bar{\theta}, t\right) \tag{1}$$

In (1) it is assumed the time differentiability of the trajectory that is not always true neither required for the methodologies that will be here considered. However, in this paper we will keep this notation for compactness and readability.

When control logic is included in the analysis, it is possible to split the vector $\bar{\theta}$ in two parts:

$$\bar{\theta} = \begin{pmatrix} \bar{x} \\ \bar{v} \end{pmatrix} \tag{2}$$

For our scope, the decomposition is carried in such a way that $\bar{x}$ represents the set of unknowns solved by RELAP-7 (which are fully differentiable) while $\bar{v}$ represents the set of variables (parameters) directly controlled by the control system. The governing equation (2) can now be rewritten as follows:

$$\begin{cases} \dfrac{\partial \bar{x}}{\partial t} = \bar{F}\left(\bar{x}, \bar{v}, t\right) \\ \dfrac{\partial \bar{v}}{\partial t} = \bar{V}\left(\bar{x}, \bar{v}, t\right) \end{cases} \tag{3}$$

As a consequence of this splitting, the components of the phase space in $\bar{x}$ are now all continuous while $\bar{v}$ contains both discrete and continuous variables. For example:

- Pressure and temperature in each point of the solution mesh belongs to $\bar{x}$
- On/off status of a pump (discrete), or the position of the control rods (continuous) belong to $\bar{v}$

A reasonable assumption is that the function $\bar{V}$, representing the control system is not depending on the whole space spanned by $\bar{x}$, but just on a subspace. In fact, we can imagine the control system acting only on a set of signals coming from the plant and not on the whole plant status.

Therefore, it is useful to introduce an appropriate subspace of $\bar{x}$, i.e., $\bar{C}$, from which the control logic can be fully derived. Thus, (3) is now re-cast as follows:

$$\begin{cases} \dfrac{\partial \bar{x}}{\partial t} = \bar{F}\left(\bar{x}, \bar{v}, t\right) \\ \bar{C} = \bar{G}\left(\bar{x}, t\right) \\ \dfrac{\partial \bar{v}}{\partial t} = \bar{V}\left(\bar{C}, \bar{v}, t\right) \end{cases} \tag{4}$$

Before moving forward it could be helpful to summarize functions and spaces introduced so far:

- $\bar{x}$: set of plant status variables (e.g., temperature, pressure, and velocity on each point of the mesh)
- $\bar{F}$: function which describe the temporal evolution of the plant status variables
- $\bar{C}$: monitored variables; usually they are the result of an integral operator (projection) applied to the

plant status variables (e.g., average temperature of a plant component, peak pressure in a pipe)

- $\bar{v}$: controlled variables. Variables affected by the control system (e.g. on/off pumps, control rod position, pump head, failure status of components, etc)
- $\bar{V}$: Control logic law

Time Dependent Integration

Classically the application of the control logic has been performed via an operator splitting approach with respect the time variable. The reason is that generally the actions decided by the application of the control logic always have an intrinsic delay due to the time required to apply them.

The operator split approach applied to system (4) is shown in (5)

$$\begin{cases} \dfrac{\partial \bar{x}}{\partial t} = \bar{F}\left(\bar{x}, \bar{v}_{t_{i-1}}, t\right) \\ \bar{C} = \bar{G}\left(\bar{x}, t\right) \\ \bar{v}_i = \bar{V}\left(\bar{C}_i, \bar{v}_{t_{i-1}}, t_i\right) \end{cases} \qquad t_{i-1} < t < t_i \quad (5)$$

As mentioned earlier, the space $\bar{C}$ represents an arbitrary construction that could be avoided and directly embedded in $\bar{V}$ but, as we will see later, its creation will be also useful in the software framework definition. In fact the constraint here imposed on the control law definition is that it should not contain any operator that requires the knowledge of the numerical schemes used to solve for $\bar{x}$. It is therefore necessary to embed in $\bar{C}$ all differential and integral operators in space and time applied to the plant status variable field needed for the definition of the control logic

Accounting for the Stochasticity of the System

As it has already been mentioned, failure laws are stochastic in nature. As an example, we can imagine the vessel failure being described by a probability distribution function (*pdf*) which depends on both time *t* and average pressure *p* inside the vessel at a certain point in time $pdf\left(p,t\right)$.

One of the most used methodologies to perform such analysis is the Monte Carlo approach where several simulation runs are performed choosing different failure probability thresholds for the failure of the vessel. The outcome histogram is used to assess the likelihood of the considered scenario.

As an example we can use the above-described case of the failure of the vessel:

1. Randomly pick a number p_{test} from the $pdf\left(p,t\right)$ of the vessel failure
2. For each time step verify:
    - If $t_i > t_{end}$ then end simulation without vessel failure and go to point 3, else
    - Compute average vessel pressure p_i and verify $p_{test} < pdf\left(p_i, t_i\right)$
        - If true: $i = i + 1$ (i.e., advance in time) and back to point 2
        - If false, end the simulation with vessel failure outcome and go to point 3
3. If the number of simulations is sufficient end sampling otherwise go to point 1

Note that the checks performed at point 2 are clearly equivalent to the application of specific control logic. As a consequence it is possible to overload the software emulating the control logic to perform the Monte Carlo simulation of the system.

Software Implementation

The development of both RAVEN and RELAP-7 are still in their beginning stages. RELAP-7 is built out of MOOSE [4] which is a middleware that provides the capabilities to solve system of partial differential equations (MOOSE embeds PETSc [5] and LIBMESH [6]). RELAP-7 orchestrates the assembling of the system of equations and provides the equation set describing each of the plant components.

Given the definition of the monitored variables provided in the above paragraph it is possible to build control logic software that is agnostic of the solution algorithm of RELAP-7 MOOSE but based only on the space of the monitored variables. Figure 1 shows the software architecture.

Test

The development of both RAVEN and RELAP-7 is still in its initial phase therefore at this moment it is not possible to provide a full example of the software implementation. Nonetheless we have performed a simple test to verify the correctness of the software architecture and its implementation.

The control system is used in this case to alter the thermal conductivity in the fuel-clad gap. The core is schematically represented by 2 channels (hot and cold), power is fixed with a cosine axial shape, and each channel represents the thermodynamic behavior of fuel, gap and clad. When the fuel temperature exceeds 500 K the thermal conductivity of the gap is set equal to the one of the fuel. While this example could be seen as simplified model for Pellet-Clad-Interaction, in our case was simply aimed to test the capability of the code.

The test used is a pseudo steady state problem where the system should reach the equilibrium situation provided power and inlet condition. Figure 2 shows the behavior of the clad temperature for the two core channels with or without the control logic. To be noticed the sudden spike in temperature due to the fuel contact while the clad while the asymptotic temperatures are, as expected, the same.

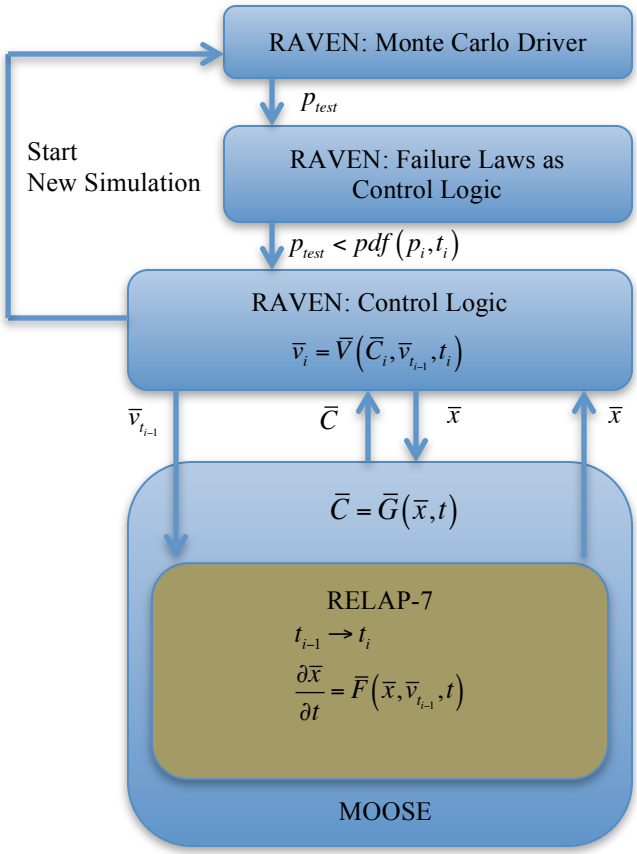

Figure 1: Software Layout

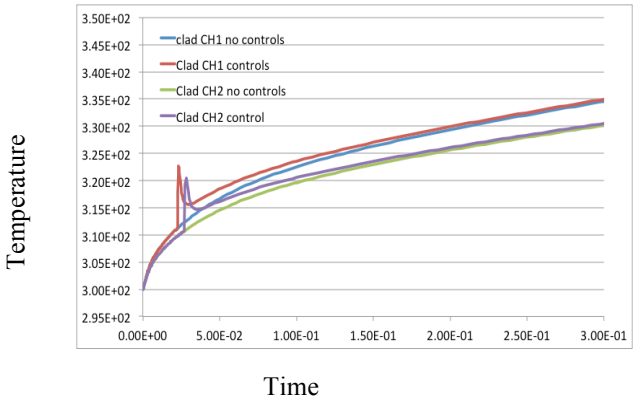

Figure 2: Clad Temperature comparison

CONCLUSIONS

The required mathematical model for the implementation of the control logic has been overviewed and its implementation in a modern software framework discussed. The chosen software structure has the advantage to allow overloading of most of the code so that the implementation of statistical failure models is straightforward. Currently implementation of more sophisticated control logic is ongoing for the simulation of the whole power plant.

ACKNOWLEDGMENTS

This work is supported by the U.S. Department of Energy, under DOE Idaho Operations Office Contract DE-AC07-05ID14517. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

REFERENCES

[1] U.S. Department of Energy, "Light Water Reactor Sustainability Research and Development Program Plan. Fiscal Year 2009–2013," (2009).

[2] M. Marseguerra, E. Zio, J. Devooght, P.E. Labeau, "A concept paper on dynamic reliability via Monte Carlo simulation", *Mathematics and Computers in Simulation*, Vol. 47, pp. 371-382 (1998).

[3] N. Siu and C. Acosta, "Dynamic event tree analysis: an application to STGR," *Probabilistic Safety Assessment and Management*, G. Apostolakis, Ed. New York: Elsevier Science Publishing Co., pp. 413–418, (1991).

[4] D. Gaston, C. Newman, G. Hansen, D. Lebrun-Grandié, "MOOSE: A parallel computational framework for coupled systems of nonlinear equations", *Nuclear Engineering and Design*, Vol. 239, Issue 10, pp. 1768-1778 (2009).

[5] Satish Balay and Jed Brown and and Kris Buschelman and Victor Eijkhout and William D. Gropp and Dinesh Kaushik and Matthew G. Knepley and Lois Curfman McInnes and Barry F. Smith and Hong Zhang, "PETSC Users Manual", ANL-95/11 - Revision 3.2, Argonne National Laboratory (2011).

[6] B. S. Kirk, J. W. Peterson, R. H. Stogner, and G. F. Carey. libMesh: A C++ Library for Parallel Adaptive Mesh Refinement/Coarsening Simulations. Engineering with Computers, 22(3-4), pp. 237-254, (2006)