Raven Facing the Problem of assembling Multiple Models to Speed up the Uncertainty Quantification and Probabilistic Risk Assessment Analyses

Andrea Alfonsi[1], Cristian Rabiti[2], Diego Mandelli[3]

[1] *Idaho National Laboratory: 2525 Freemont Avenue, Idaho Falls, ID, 83400, andrea.alfonsi@inl.gov*
[2] *Idaho National Laboratory: 2525 Freemont Avenue, Idaho Falls, ID, 83400, cristian.rabiti@inl.gov*
[3] *Idaho National Laboratory: 2525 Freemont Avenue, Idaho Falls, ID, 83400, diego.mandelli@inl.gov*

The development of high-fidelity codes has undergone a significant acceleration in the last years. The newly deployed codes are characterized by remarkable improvements in the approximation of the underline physics (high approximation order and reduced use of empirical correlations). This improved fidelity is generally accompanied by a larger computational effort (computation time increased). This issue represents an obstacle in the application of traditional computational techniques of uncertainty quantification (UQ) and risk (RA) associated with the operation of any industrial plant (for example, a nuclear reactor).
In order to overcome these limitations, several techniques are under investigation and parallel development. In the RAVEN framework, these techniques heavily use the concept of Reduced Order Modeling in order to accelerate the Risk assessment and uncertainty quantification. The Reduced Order Models (ROMs) are mathematical representations of a system, used to predict a selected output space of a physical system. In other words, they are mathematical objects that can be "trained" in order to predict some selected Figure of Merits (FOMs) of a system, within its training domain.
The usage of the ROMs can represent a valid solution to speed up the UQ and RA process, overall if multiple ROMs are combined together in order to simulate all the physical aspects that can describe the system (e.g. neutronic FOMs, Thermal-hydraulic FOMs, etc). The aim of this paper is the description of a newly developed technique (RAVEN object) to assemble multiple ROMs, in particular, and any model (physical codes, post-processing algorithms, etc.), in general. This new RAVEN object allows the user to assemble multiple models in an entity that has been named "Ensemble-Model", identifying the input/output connections, and, consequentially the order of execution and which sub-models can be executed in parallel.
The " Ensemble-Model " concept represents a key development in the RAVEN code, since it allows to perform an integrated PRA and UQ analysis without the need to "interrogate" the high-fidelity codes if not strictly necessary, determining a remarkable speed-up of the whole process.

I. INTRODUCTION

RAVEN[1,2,3] is currently able to construct multi-targets Reduced Order Models (Ref.4), which are aimed to represent the response of a system (in a fixed configuration) for multiple Figures of Merits (FOMs) and time-dependent ROMs (see Ref.5). These capabilities represent the initial steps for a larger implementation about the interaction of multiple models. In fact, in several cases, multiple models need to interface with each other since the initial conditions of one are dependent on the outcomes of another.

To better understand the problem that here is solved, it is useful to consider two simple examples:

1) The following problem is considered: a weather forecast simulation code "a" is used to compute the external (i.e. ambient) temperature in a certain location. A second model "b" is inquired to compute the average temperature in a room having as boundary condition, among several others, the external ambient temperature. The response of the model "b" depends on the outcome of the model "a";
2) Two different simulation codes are considered: a) a code that is meant to compute the thermal conductivity of the ceramic Uranium Dioxide (UO_2) as function of the Temperature, and b) a Thermal-hydraulic (TH) code that is used to compute the Temperature field of a reactor, whose heat conduction depends on the thermal conductivity value. As easily inferable, the two models are mutual dependent, determining in a non-linear system of equations;

The two reported examples are only aimed to illustrate the reason why the creation of a framework to make interact different models is a key development for the advancement of RAVEN as a comprehensive calculation flow driver. Before reporting how the ensemble-models have been implemented, it is necessary to briefly describe the representative Model "entities" that are available in RAVEN.

II. MODELS IN RAVEN

The Model entity[1], in the RAVEN environment, represents a "connection pipeline" between the input and the output space. The RAVEN framework does not own any physical model (i.e. it does not posses the equations needed to simulate a generic system), but implements APIs by which any generic model can be integrated and interrogated. In the RAVEN framework four different model categories (entities) are defined:

- Codes;
- Externals;
- ROMs;
- Post-Processors.

The Code model represents the interface object that establishes the communication pipe between RAVEN and any driven code. Currently, RAVEN has APIs for several different codes:

- RELAP5-3D, the most widely used Safety Code (thermal-hydraulic);
- RELAP-7, safety code eventual future replacement of RELAP5-3D code;
- any MOOSE-based application;
- SAS4A/SASSYS-1, safety analysis code for fast reactors (Argonne)
- Modelica, object-oriented, declarative, multi-domain modeling language for component-oriented modeling of complex systems;
- MELCOR, engineering-level computer code that models the progression of severe accidents in light-water reactor nuclear power plants (coupling under development by the University of Rome "La Sapienza");
- MCNP, general-purpose Monte Carlo N-Particle code that can be used for neutron, photon, electron, or coupled neutron/photon/electron transport (under development);
- MAAP5, computer code that models the progression of severe accidents in light-water reactor nuclear power plants (coupling performed by the Ohio State University).

The data exchange between RAVEN and the driven code can be performed either by direct software interface or by files. If the system code is parallelized, the data exchanging by files is generally the way to follow since it can be much more optimized in large clusters.

The External model allows the user to create, in a Python file (imported, at run-time, in the RAVEN framework), its own model (e.g. set of equations representing a physical model, connection to another code, control logic, etc.). This model will be interpreted/used by the framework and, at run-time, will become part of RAVEN itself.

The ROM (Reduced Order Model) represents an API to several different algorithms. A ROM is a mathematical representation of a system, used to predict a selected output space of a physical system. The creation and sub-sequential usage of a ROM involves a procedure named "training". The "training" is a process that uses sampling of the physical model to improve the prediction capability (capability to predict the status of the system given a realization of the input space) of the ROM. More specifically, in RAVEN the ROM is trained to emulate a high fidelity numerical representation (system codes) of the physical system.

The Post-Processor model is aimed to manipulate the data generated, for example, employing a sampling strategy. In RAVEN several different post-processors are available: 1) Statistics Post-Processor, aimed to compute all the statistical figure of merits (e.g. expected values, variance, skewness, covariance matrix, sensitivity coefficients, etc.); 2) Limit Surface, which computes the Limit Surface, inquiring a goal function (i.e. a function that determines if a certain coordinate in the input space led to a failure or success), and so many others.

As already mentioned, in several cases multiple models need to interface with each other since the initial conditions of some are dependent on the outcomes of others. In order to face this problematic in the RAVEN framework, a new model category (e.g. class), named *EnsambleModel*, has been implemented. This class is able to assemble multiple models of other categories (i.e. Code, External Model, ROM), identifying the input/output connections, and, consequentially the order of execution and which sub-models can be executed in parallel.

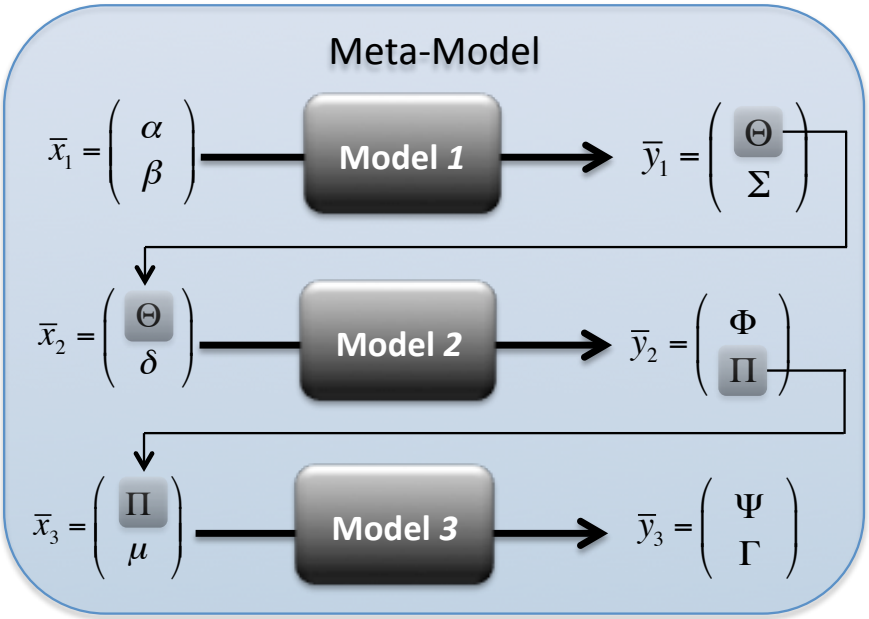

Figure 1 - Example of an Ensemble-Model constituted by 3 sequential sub-models.

Before analyzing in detail how the Ensemble-Model (also known as meta-model) capability has been developed in the RAVEN framework, it is worth to report a couple of schematic cases that show how the input/output interconnections determine the order of execution of the sub-models. **Figure 1** reports an example of a Meta-Model that is constituted by 3 sub-models (ROMs, Codes, or External Models). As it can be noticed:

- The *Model 2* is connected with the *Model 1* through the variable Θ (Model 1 output and Model 2 input);
- The *Model 3* is connected with the *Model 2* through the variable Π (Model 2 output and Model 3 input);

In this case, the Ensemble-Model is going to drive the execution of all the sub-models in a serial sequence, since each model (except the *Model 1*) is dependent on one of the outcomes of previously executed.

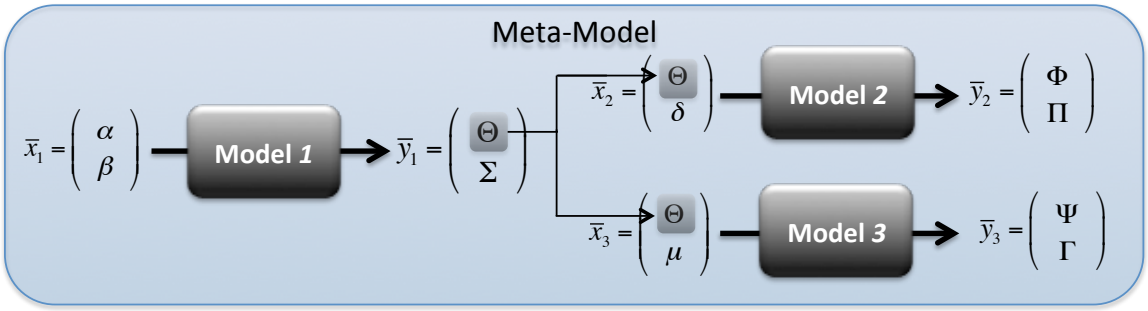

Figure 2 - Example of a Meta-Model constituted by 3 sub-models, 2 of which can be run independently.

In **Figure 2** another example is reported. In this case, the *Model 2* and *Model 3* depend on the Model 1 only through the output variable Θ and they are not connected between each other. For this reason, the Meta-Model executes the *Model 2* and *Model 3* in parallel, after inquiring the *Model 1*.

II.A. Implementation

Since the modularity of the RAVEN framework, the implementation of the Ensemble-Model entity has been straightforward. In RAVEN all the models' outputs (e.g. whatever code output, etc.) are collected in an internal containers (named *DataObjects*) that are aimed to store time-series and input/output data relations in a standardized fashion; in this way, the communication of the output information among different entities (i.e. Models) can be completely agnostic with respect

to the particular type of output generated by a model. The Ensemble-Model entity fully leverages this peculiarity in order to transfer the data from a Model to the other(s).

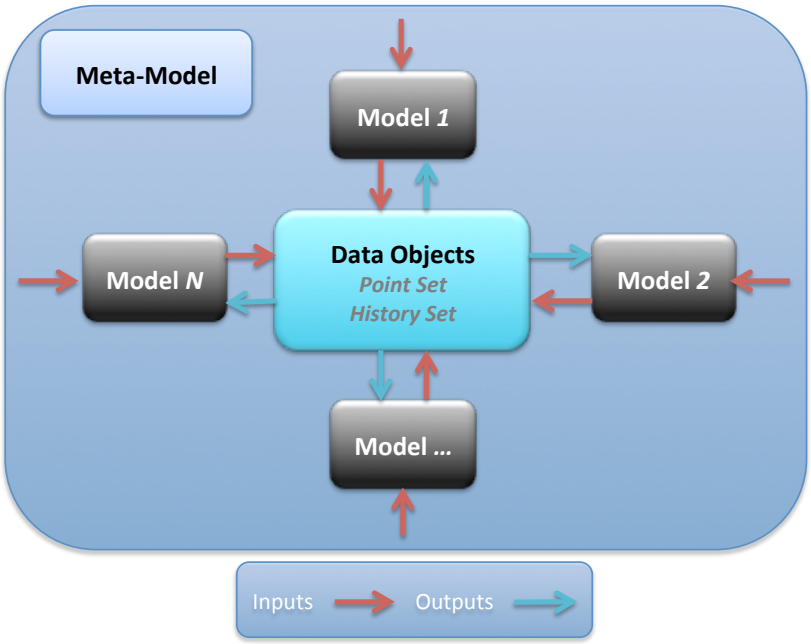

Figure 3 - Meta-model data exchange among sub-models.

Based on the Input/Output relations of each sub-models, the Ensemble-Model entity constructs the order of their execution and, consequentially, the links among the different entities.

Figure 3 schematically shows the communication piping established by the Ensemble-Model entity. It can be noticed how the sub-models share information (inputs/outputs data) using the *DataObjects* entity as communication network.

II.A. Ensemble-model resolving in a non-linear system

In several cases, the input of a model depends on the output of another model whose input is the output of the initial model. In this situation, the system of equation is non-linear and an iterative solution procedure needs to be employed. The Ensemble-Model entity in RAVEN is able to detect the non-linearity of the sub-models' assembling and activate the non-linear solver: Picard's iterative scheme. Since Picard's iterative scheme is well known, there is no mean to report its implementation here. **Figure 4** shows an example of when the Ensemble-Model entity activates the Picard's iteration scheme, which ends when the residue norm (between an iteration and the other) falls below a certain input-defined tolerance.

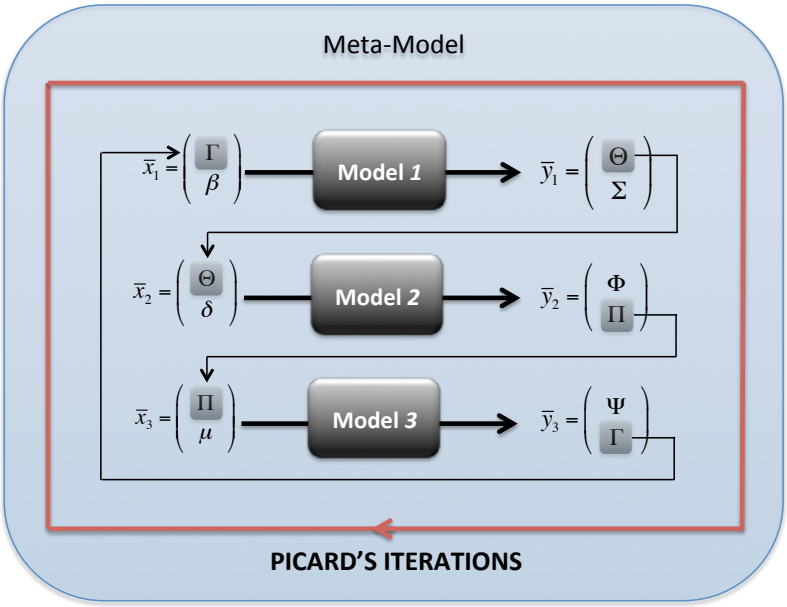

Figure 4 – Ensemble-Model resolving in a non-linear system of equations – Picard's iterations.

II. APPLICATION EXAMPLE

In order to test the newly developed capability, a simple 1-Dimensional heat conduction transient (in a slab of thickness L=1 m) has been employed. Two *External* Models (EMs) compose the Ensemble-Model:

1. *EM1*: is aimed to solve the heat conduction partial differential equation:

$$\begin{cases} \dfrac{dT(x,t)}{dt} = k\dfrac{d^2T(x,t)}{dx^2} \\ T(0,t) = T_{left} \\ T(L,t) = T_{right} \end{cases}$$

2. *EM2*: computes the thermal conductivity (input of *EM1*) as function of the average temperature in the slab, using the following correlation:

$$k = \frac{38.23}{129.2 + T} + 6.077E^{-13} * T$$

The transient is 10 seconds long and the initial temperature across the slab is set to 600 K. The so constituted Ensemble-Model has been sampled through a Grid strategy sampling the boundary conditions $T(0,t)$ and $T(L,t)$ with a uniform probability distribution between 500 and 1700 K.

Two cases have been run in order to test the functionality of the Ensemble-Model when it resolves in a chain of evaluations and in a non-linear system:

1. In the first test the model *EM2* uses the sampled boundary conditions to calculate the average temperature and consequentially the thermal conductivity that is fed in the heat conduction model *EM1*. This approach resolves in a chain of evolutions as shown in **Figure 5**.

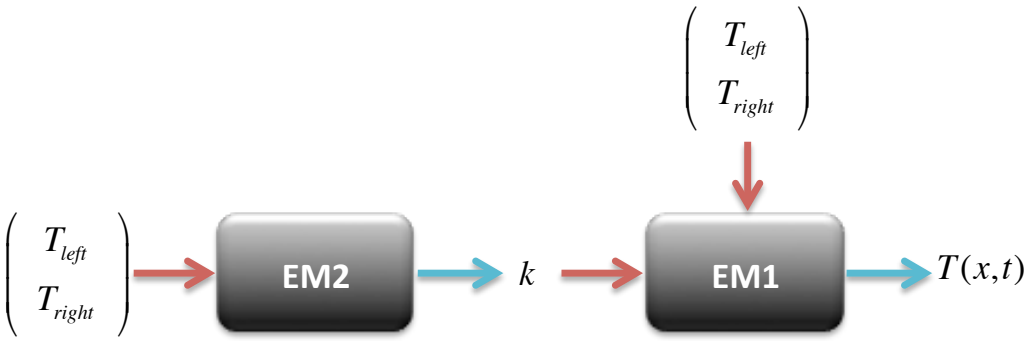

Figure 5 – Heat Conduction Ensemble-Model connections in chain of evaluations

2. In the second test the model *EM2* has as input the average temperature in the slab that is one of the outcomes of the model *EM1*. In addition, the model EM1 needs the output (thermal conductivity) of the model EM2 to compute the heat conduction problem. Hence, this scenario resolves in a non-linear system of equations. This is shown in **Figure 6**.

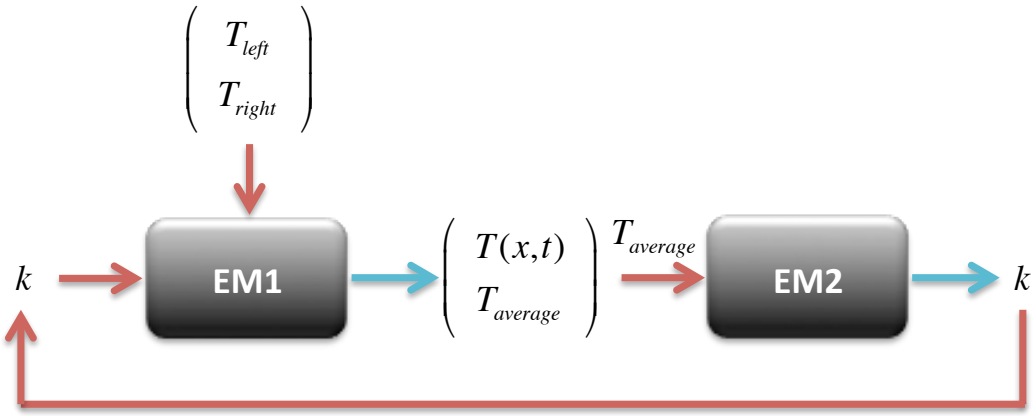

Figure 6 - Heat Conduction Ensemble-Model connections in non-linear system.

Figure 7 and **Figure 8** show the outcomes of the Ensemble-Model sampling in the two cases. As it can be seen, the obtained results match perfectly.

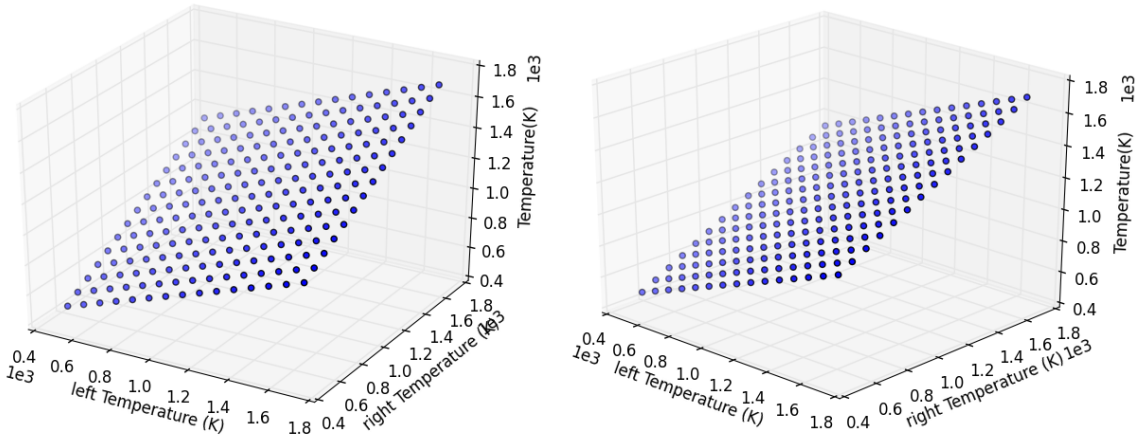

Figure 7 - Temperature at mid-plane: Sequential Model (Left) and Picard Iteration (Right).

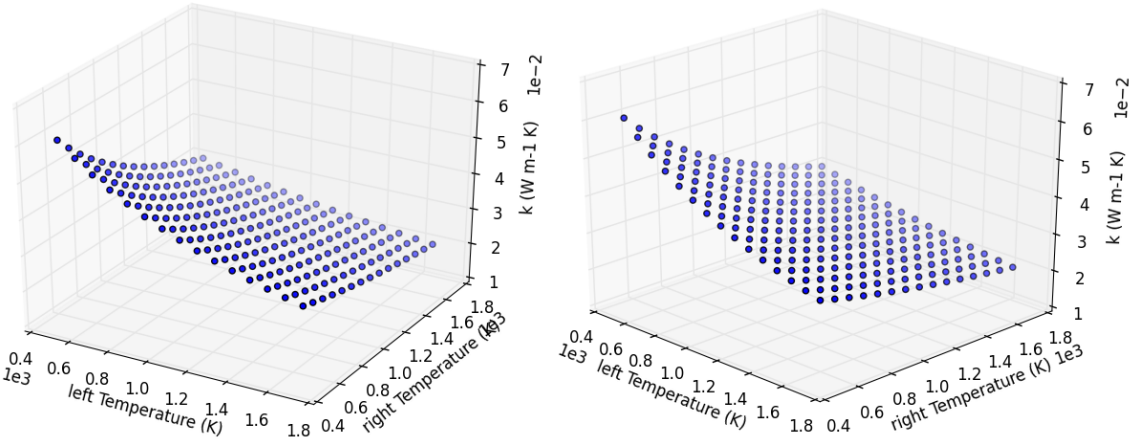

Figure 8 – Compute thermal conductivity: Sequential Model (Left) and Picard Iteration (Right).

II. CONCLUSIONS

In this paper, a newly developed capability of the RAVEN code has been shown. Through the Ensemble-Model entity, RAVEN is able to combine multiple models (i.e. Simulation Codes, Reduced Order Models, etc.), constructing a pipe network in order to transfer information among them. The addition of the Picard's iteration scheme lets the user solving combinations of models that resolve in non-linear systems.

The paper shows the early results for implementation of an ensemble approach for the coupling of surrogate models representing a multi-physics problem. While this is an initial implementation, the developed structure seems to support current needs and will be eventually extended in the future in order to couple codes whose input/output space is represented by high-density fields (e.g. temperature profiles in each nodal kinetic zones, etc.). This capability builds a complex system representation, even when the original models were not coupled, but just coupled their surrogate. Two applications are relevant for reliability analysis: (1) the possibility to build surrogate representation of a complex system, starting from libraries of surrogate models for each component, and (2) software implementation present during the first stage of surrogate model coupling when responses are high-density fields.

The Ensemble-Model capability in RAVEN is currently used to couple a fuel performance code (Bison) and the T-H system code RELAP5 in order to analyze LOCA scenarios, whose results are going to be presented during the conference presentation.

REFERENCES

1. A. Alfonsi, C. Rabiti, D. Mandelli, J. Cogliati, and R. Kinoshita, "RAVEN as a tool for dynamic probabilistic risk assessment: Software overview," in Proceeding of M&C2013 International Topical Meeting on Mathematics and Computation, CD-ROM, American Nuclear Society, LaGrange Park, IL, 2013.
2. Alfonsi, C. Rabiti, D. Mandelli, J. Cogliati, R. Kinoshita, A. Naviglio, "Dynamic Event Tree Analysis Through RAVEN", International Topical Meeting on Probabilistic Safety Assessment and Analysis (PSA 2013), September 22-26, Columbia, SC, USA, (2013).
3. A. A. Alfonsi, C. Rabiti, D. Mandelli, J. Cogliati, and R. Kinoshita, "RAVEN: Development of the adaptive dynamic event tree approach," Tech. Rep. INL/MIS-14-33246, Idaho National Laboratory (INL), (2014)
4. A. Alfonsi, C. Rabiti, D. Mandelli, J. Cogliati, S. Sen, and C. Smith, "Improving limit surface search algorithms in raven using acceleration schemes", INL/EXT-15-36100 (July 2015).
5. D. Mandelli, C. Smith, A. Alfonsi, C. Rabiti, J. Cogliati, H. Zhao, I. Rinaldi, D. Maljovec, P.Talbot, B. Wang, V. Pascucci "Reduced Order Model Implementation in the Risk-Informed Safety Margin Characterization Toolkit." INL/EXT-15-36649 (September 2015)
6. C. Rabiti, A. Alfonsi, D. Huang, F. Gleicher, B. Wang, H. S. Abdel-Khalik, V. Pascucci, and C. L. Smith, "System Reliability Analysis Capability and Surrogate Model Application in RAVEN", INL/EXT-16-37243 (November 2016).