The HUNTER Dynamic Human Reliability Analysis Tool: Coupling an External Plant Code

Yunyeong Heo[a,c], Thomas A. Ulrich[c], Jeeyea Ahn[b,c], Jooyoung Park[c], and Ronald L. Boring[c]

[a] Ulsan National Institute of Science and Technology, Ulsan, South Korea, yyheo0207@unist.ac.kr
[b] Ulsan National Institute of Science and Technology
[c] Idaho National Laboratory, Idaho Falls, ID, US

Abstract: The Human Unimodel for Nuclear Technology to Enhance Reliability (HUNTER) is a framework that supports dynamic human reliability analysis (HRA) via the development of standalone software for performing dynamic HRA calculations. In the HRA, human actions in nuclear power plants (NPPs) are predicated by plant states, and these human actions in turn influence the plant. In other words, plant operations are necessarily recursive, and it becomes challenging to model complex human-plant interactions. Consequently, we have coupled two software simulations that complement those shortcomings. RELAP5-3D, which stems from the Reactor Excursion and Leak Analysis Program (RELAP) [1], is the foundational thermal-hydraulic software for modeling nuclear systems. Using RELAP5-3D, we simulated plant operations as they progressed in accordance with procedures developed to address emergent situations in NPPs. Plant operations include various actions such as operator checks of plant parameters, as well as actions that are continuously performed over time until a specific parameter reaches certain criteria. This means that HUNTER and RELAP5-3D exchange information with each other and should be carried out simultaneously over time. To simulate plant operations, which represent actual operator checks of plant parameters and the corresponding manual control actions, changes in plant status are identified via simulation and performed in accordance with the criteria and order of the procedure. Thus, the goal in coupling HUNTER with RELAP5-3D is to facilitate synchronous coupling, where human and plant models provide iterative feedback loops that drive the course of actions. The advantage of coupling these simulation frameworks so that RELAP5-3D can serve as the external environment module in HUNTER is that it allows for plant models to be customized and streamlined for specific applications. In this paper, we address the key features of the coupling, along with the coupling structure built to perform the feedback loops.

1. INTRODUCTION

The U.S. Department of Energy Light Water Reactor Sustainability program's Risk-Informed Systems Analysis pathway sponsors many HRA-related projects that aim to create better tools to support industry risk assessment needs. One such project involves the Human Unimodel for Nuclear Technology to Enhance Reliability (HUNTER) [1]. HUNTER is a framework that supports dynamic human reliability analysis (HRA) via the development of standalone software for performing dynamic HRA calculations. In the HRA, human actions in nuclear power plants (NPPs) are predicated by plant states, and these human actions in turn influence the plant. In other words, the plant operations are necessarily recursive, and it becomes challenging to model complex human-plant interactions.

Consequently, we have coupled two software simulations that complement those shortcomings. The Environment module in HUNTER represents the physical environment—in this case, a NPP—with which the virtual operator interacts. HUNTER's primary purpose is to simulate and analyze virtual operations and then derive HRA results. HUNTER must reflect the plant response to the plant operations operator. Plant status and plant operations are deeply interconnected, since procedures can change depending on the plant status, and plant parameters may change in accordance with operator actions. From a dynamic modeling point of view, coupling with a plant simulator is necessary for capturing human operations that vary depending on plant status or the detailed guidance provided at different steps in the procedures. We describe the coupling of HUNTER with a separate software code capable of plant simulation. This coupling reflects operator manipulations throughout the plant simulation, and includes the plant status needed for procedure progress. This paper focuses specifically on implementing the Environment module by using RELAP5-3D, which originates from the Reactor Excursion and Leak Analysis Program (RELAP).

2. DEMONSTRATION OF COUPLING BETWEEN HUNTER AND THE PLANT SIMULATOR

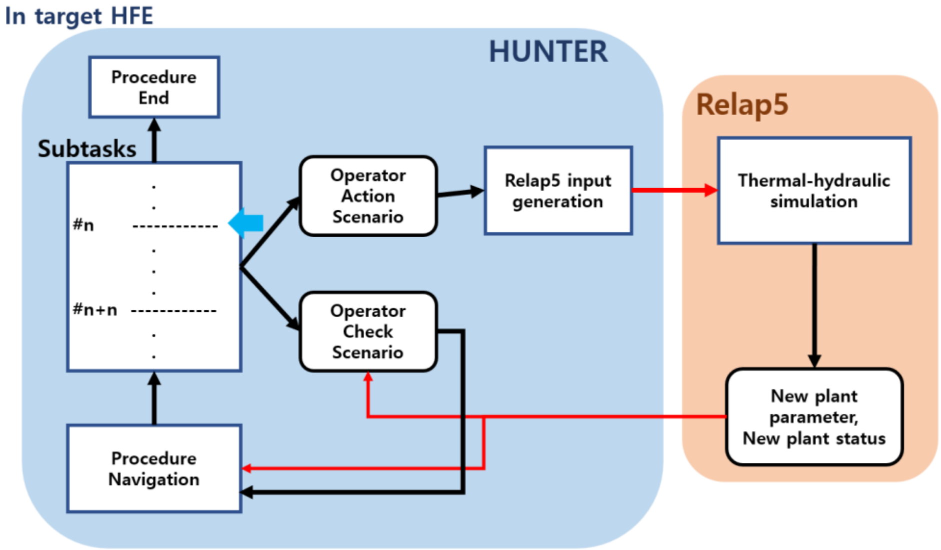
To reflect the interplay between plant state and operator response, HUNTER's virtual operator is linked to an external plant model. In HUNTER, operator manipulations are applied to the plant simulator or thermal-hydraulic code. It is necessary to perform actions (i.e., tasks) that directly affect plant operations, such as closing valves or operating pumps. In turn, the operator monitors the plant parameters. Some parameters are monitored regularly to determine routine or abnormal plant evolutions. The HUNTER virtual operator, guided by both continuous and one-off tasks specified in the procedures, checks for parameter changes or compares the plant parameters against the specific values given in the procedures. Additionally, when operators affect changes in the plant, they must confirm those changes are exactly what was desired.

Thus, the interface between the virtual operator in HUNTER and the virtual plant is a two-way feedback loop, as depicted in Figure 1. The operator must be able to receive parameters from the plant model, and must be able to manipulate or control aspects of the plant. To accomplish this interaction, we implemented a two-way synchronous coupling between HUNTER and a plant simulator:

- Monitor (Plant → Operator): HUNTER reads the parameter or component information reflected in the plant simulation.
- Control (Operator → Plant): In turn, HUNTER can alter the state of the plant simulation through virtual operator actions.

These two functions operate dynamically and in parallel. The plant model progresses with or without virtual operator input. The plant parameters are available to the virtual operator throughout this progression. When the operator intercedes (e.g., by closing a valve), this is reflected in the plant model as a change. The iterative feedback loop between the HUNTER virtual operator and the plant model progresses up to a particular stopping point for each given scenario. The human and plant models are synchronized through information exchange (i.e., parameter monitoring and operator actions) via the coupling mechanism. Hold points may be employed to allow time for particular information exchanges. For example, the operator model may be in wait-and-monitor mode awaiting a particular parameter level from the plant model. Similarly, the plant model may simulate plant functions for a particular interval up to a synchronization point and pause for potential input (e.g., operator actions) from HUNTER. Coupling may occur at regular intervals or may be irregular, depending on the type of plant activity being monitored or controlled.

Figure 1. Coupling scheme

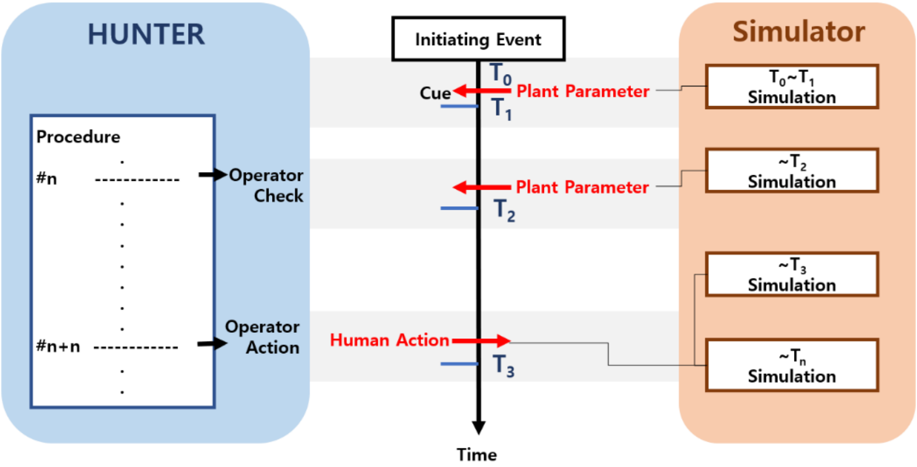


An irregular-interval coupling is depicted in **Figure 2**. The HUNTER virtual operator performs plant operations as specified in the procedure, with each procedure step taking different times according to the GOMS (goals, operators, methods, and selection rules) -HRA primitives. The plant also changes over time, with or without actions from the operator, reflecting the natural progression of plant conditions. Of course, plants are designed to require operator control actions, and the desired evolution of the plant is determined by human actions. For this reason, operator actions should be reflected in the simulator or thermal-hydraulic code, and the plant status or response transmitted to HUNTER. A typical course of action is:

- A procedure step directs the operator to take an action (e.g., closing a valve).
- HUNTER transmits the valve's changed state to the thermal-hydraulic code.
- The thermal-hydraulic code changes the plant state or model to reflect the closed valve.
- The operator is directed to confirm that the valve is closed.
- HUNTER retrieves the valve status from the thermal-hydraulic code.
- HUNTER performs the logic check on the valve status and proceeds to the next step (if the valve is closed) or to alternate RNO (response not obtained) steps (to deal with the stuck-open valve).

HUNTER and the thermal-hydraulic or plant simulator should exchange information in a manner similar to how the operator checks plant parameters and makes decisions. Both the operator and the plant function independently but synch up at various points. HUNTER uses task-level primitives in GOMS to estimate how long each procedure step takes to perform. The thermal-hydraulic code will pause at this point for synchronization, either a checking of the plant state or a manual action performed by the virtual operator in the plant model.

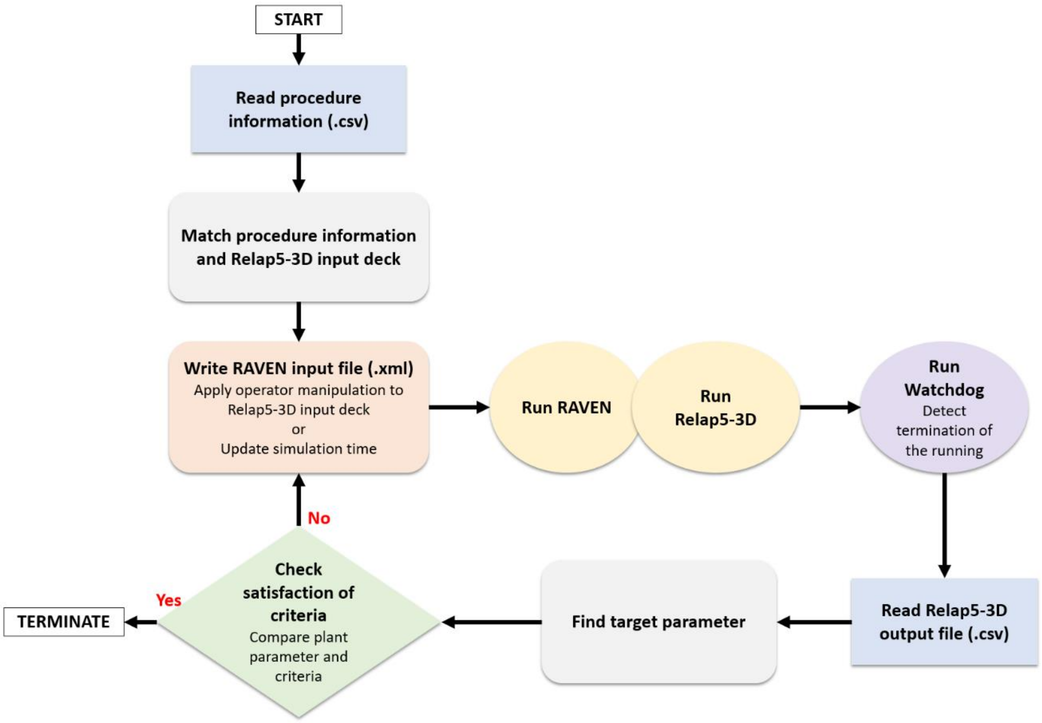
Figure 2. Conceptualization of the coupling between HUNTER and the plant simulator



3. COUPLING WITH RELAP5-3D

The coupling task described in this paper was performed using a computer simulation software dedicated to NPP operational thermal-hydraulics analysis. RELAP5-3D was developed at Idaho National Laboratory (INL) for analyzing transients and accidents in water–cooled NPPs and other related systems, as well as for analyzing advanced reactor designs [2]. RELAP5-3D was selected so we could analyze various types of plants (assuming the existence of corresponding plant models), whether developed relatively recently (e.g., small modular reactors) or based on established plant designs. Recent developments in RELAP5-3D include implementation in INL's supercomputer cluster, the High-Performance Computing (HPC) environment. The massively parallel HPC enables thermal-hydraulic code calculations to be conducted much faster than when using other plant simulators. We also developed codes dealing with RELAP5-3D by using the INL-developed Risk Analysis and Virtual Control Environment (RAVEN) [3]. RAVEN was coupled with RELAP5-3D so that thermal hydraulics could be used to streamline the risk analysis. The RAVEN interface simplifies the updating of RELAP5-3D input data and enables multiple simulations to be run in parallel. RAVEN receives data and regenerates input decks for RELAP5-3D. Once the regenerated simulation is complete, RELAP5-3D provides data back to RAVEN for post-processing [1][3][4][5]. Post-processing links the thermal-hydraulic output with time, saving the results in a CSV format for ready use by data analysis software. It also opens up the possibility of conducting various analyses in the future by enabling distribution samples to be used for uncertainty analysis.

Figure 3: Loop logic for coupling HUNTER and Relap5-3D


START
Read procedure information (.csv)
Match procedure information and Relap5-3D input deck
Write RAVEN input file (.xml)
Apply operator manipulation to Relap5-3D input deck
or
Update simulation time
Run RAVEN
Run Relap5-3D
Run Watchdog
Detect termination of the running
Read Relap5-3D output file (.csv)
Find target parameter
Check satisfaction of criteria
Compare plant parameter and criteria
No
Yes
TERMINATE

In regular RELAP5-3D simulations, the analyst sets every sequential propagation of an initiating event prior to running the simulation. We, however, implemented a simulation in this coupling that leaves all possibilities open to reflect operator interactions with the plant. In coupling HUNTER with RELAP5-3D, as briefly described above, the simulation begins with the plant parameters being monitored by the virtual operator, who follows procedure and performs plant actions in accordance with the plant response. The RELAP5-3D simulation runs for the time duration necessitated by the operator activities outlined in the procedures. It then pauses, updates the RELAP5-3D model, restarts, and continues on as before. A detailed description of this coupling is given below.

If the target task is an action, it can be terminated immediately by satisfying the procedure's criteria after the operator performs the action. If, on the other hand, the operator manipulation involves performing an action to a certain point (e.g., opening a valve to a certain level), this necessitates iteratively performing the action and then checking the resulting parameter value until the appropriate criterion is satisfied. It should not be decided a priori that the operator will always complete the task successfully. For example, another task could distract the operator from successfully completing the task, or the operator may fail to actuate the valve properly over time. Coupling between HUNTER and RELAP5-3D should be performed step by step, rather than assuming the best human outcome.

The coupling between HUNTER and the Environment module is shown as a flow chart in **Figure 3**. First, HUNTER receives information for each procedure step. The procedure specifies three possible activities relative to the Environment module:

- Read a particular parameter from the RELAP5-3D plant model.
- Change a characteristic (e.g., open a valve) in the RELAP5-3D plant model.
- Wait a particular interval, during which no interaction occurs between the plant model and HUNTER.

HUNTER checks the operator manipulation, as well as which parameter and component are the targets, through this file. Next, RAVEN (as controlled by HUNTER) performs a mapping to the RELAP5-3D input deck in order to apply target information for the plant simulations. RAVEN connects each component and parameter to the card number defined in the RELAP5-3D input deck. Third, RAVEN updates the connected RELAP5-3D input deck card and runs the simulation. Lastly, HUNTER compares the derived results against the criteria given in the procedure. If the criteria are not met, HUNTER increases the simulation clock and reruns the simulation. This synchronous back and forth is called loop logic.

3.1. Code Structure

We developed RAVEN and the RELAP code to accommodate the overall coupling. Scripts were developed to conduct the various tasks involved, such as to read the procedure CSV file provided by HUNTER, map procedure information and RELAP5-3D, and write and edit the RAVEN input files. Moreover, we composed a script to monitor the RELAP5-3D simulation, one to read the simulation results, and one that updates the simulation clock to re-run the simulation if the criteria are not met.

Mapping translates the component or plant parameters received from the procedure into RELAP5-3D information. This is completed by implementing operator manipulations in the RELAP5-3D model through procedure analysis, then linking the scripts with procedure information. Once the target plant is determined, analysts can easily add one-by-one mapping through the trip cards and component card in the RELAP5-3D input deck. The HUNTER Environment module finds and reads specific procedural information by using the parameters defined in the procedure file. When reading RELAP5-3D results, it finds and reads the appropriate parameters and times suitable for loop logic, as shown in **Figure 3**. In addition, we developed subroutines that update the XML file to modify and create the RAVEN input file. This changes the plant variables by adjusting the RELAP5-3D input deck. There is also a feature that accounts for operator time to complete tasks. Comparison of more than two plant parameters is accomplished in the library subroutines. Finally, since we cannot predict the required simulation time of RELAP5-3D, hold logic is implemented to synchronize HUNTER with RELAP5-3D. When the plant simulation result is saved as a CSV file, this file can be found and read. File generation using a server is performed in INL’s HPC environment; however, we developed the coupling between both HPC and a computer running a local copy of RELAP5-3D. In the script for monitoring a typical desktop computer, we used the Watchdog function, a Python API (application programming interface), and a shell utility to monitor file system events in order to detect every file and directory generated, modified, or deleted. The information is therefore exchanged through files such as input decks for RELAP5-3D and CSV parameter logs for HUNTER.

4.1. Proof-of-Concept Coupling Demonstration

4.1.1 Overview

To simulate the two-way coupling between the virtual operator in HUNTER and the plant response in RELAP5-3D, we must perform an operator manipulation and then check the plant status. Thus, as with the virtual operator, we must know how to imitate actions such as checking plant parameters, confirming changes, or waiting for plant parameter changes to reach specific set point. The code can perform plant simulation using loop logic to connect HUNTER and RELAP5-3D. To demonstrate this Environment module implementation, we present an arbitrary example defined via the following modeling criteria:

- Include an operator action.
- Include tasks that control simulations according to plant status, such as virtual operators.
- Check whether the criteria are met by monitoring two or more parameters simultaneously.

According to the above criteria, the example scenario for this demonstration is defined as follows:

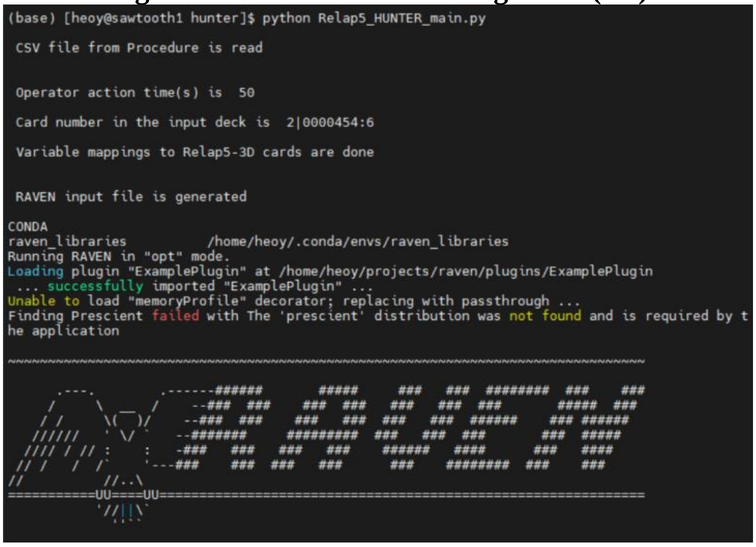
- Operators shut a main coolant pump in 50 seconds.
- Operators check the plant parameter changes and confirm their criteria.
    - Criteria for parameter 1 (center core channel temperature): over 629
    - Criteria for parameter 2 (lower plenum mass flow rate): under 818

We assumed that the operator stops the pump, checks the parameter change, checks the specific criteria to perform the next procedure task, then moves on to the next task. The RELAP5-3D model used is the INL Generic Pressurized Water Reactor, a generic 3-loop Westinghouse pressurized-water reactor model. Also, the model simulates SGTR (steam generator tube rupture, tube rupture occurs in 0 seconds), but when proceeding with the scenario, arbitrary operator actions independent of the procedure or event tree are simulated [6].

4.1.2 Results of Module Runs

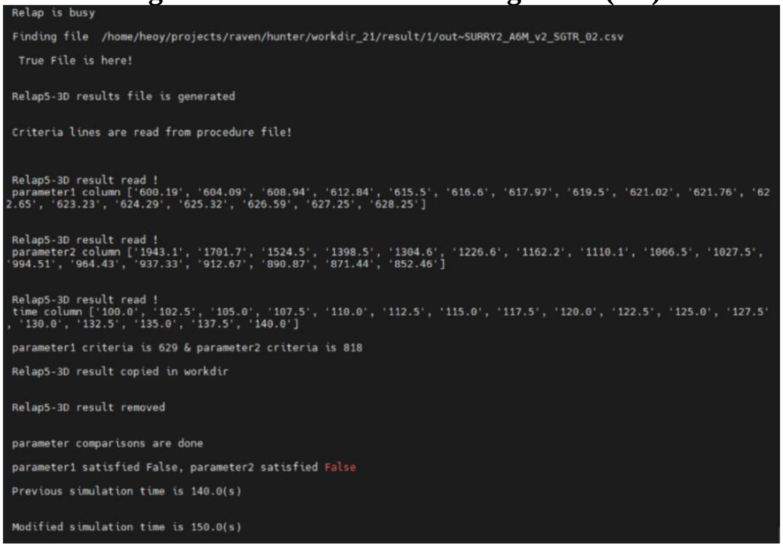
As shown in Figure 4, the information is first read from the procedure when we run the main script using Python. This is the step in which procedure information is received from HUNTER. Furthermore, the information is mapped directly to the Relap5-3D input deck and can be checked by analysts. Moreover, it generates the RAVEN input file by using the mapped information, then runs RAVEN. In INL HPC, RAVEN executes Relap5-3D by sending jobs to the server through Q-sub. When Relap5-3D completes the plant simulation calculation, it generates a CSV file. The input deck also determines which variable will be printed. Therefore, analysts must either insert the parameters of interest into the RAVEN input file for application to the Relap5-3D input deck, or otherwise pre-calibrate the Relap5-3D input deck.

Figure 4: Screenshot of running codes (1/4)



```
(base) [heoy@sawtooth1 hunter]$ python Relap5_HUNTER_main.py

 CSV file from Procedure is read


 Operator action time(s) is  50

 Card number in the input deck is  2|0000454:6

 Variable mappings to Relap5-3D cards are done


 RAVEN input file is generated

CONDA
raven_libraries          /home/heoy/.conda/envs/raven_libraries
Running RAVEN in "opt" mode.
Loading plugin "ExamplePlugin" at /home/heoy/projects/raven/plugins/ExamplePlugin
 ... successfully imported "ExamplePlugin" ...
Unable to load "memoryProfile" decorator; replacing with passthrough ...
Finding Prescient failed with The 'prescient' distribution was not found and is required by t
he application

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

      .---.        .------######       #####     ###    ###  ########  ###    ###
     /     \  __  /    --###  ###    ###  ###   ###    ###  ###       #####  ###
    / /     \(  )/    --###  ###    ###   ###  ###   ###  ######    ### ######
   //////    ' \/ `   --#######    #########  ###   ###  ###        ###  #####
  //// / // :    :   -###   ###   ###   ###    ######   ####       ###  ####
 // /   /  /`    '---###    ###  ###   ###      ###   ########  ###   ###
//          //..\
===========UU====UU=====================================================================
           '//||\`
             ''``
```

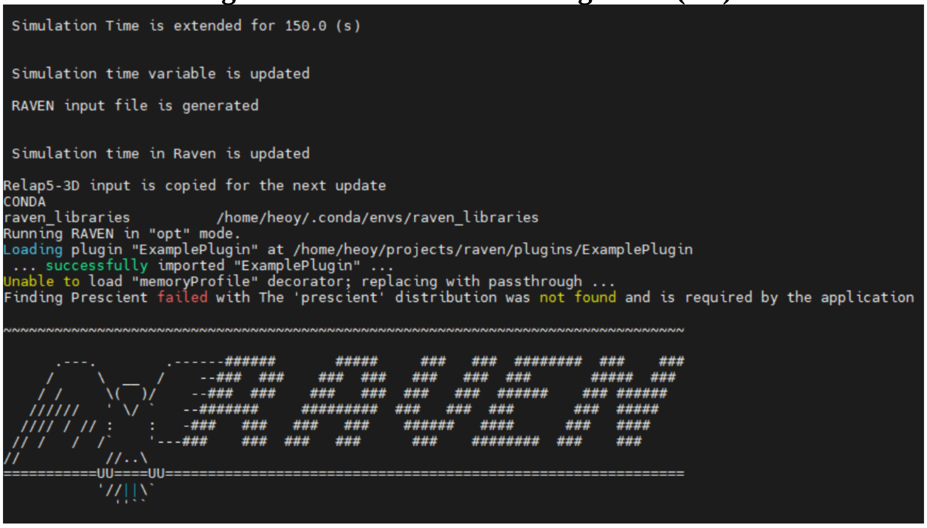
Figure 5: Screenshot of running codes (2/4)



```
 Relap is busy

 Finding file  /home/heoy/projects/raven/hunter/workdir_21/result/1/out~SURRY2_A6M_v2_SGTR_02.csv

  True File is here!


 Relap5-3D results file is generated


 Criteria lines are read from procedure file!



 Relap5-3D result read !
 parameter1 column ['600.19', '604.09', '608.94', '612.84', '615.5', '616.6', '617.97', '619.5', '621.02', '621.76', '62
2.65', '623.23', '624.29', '625.32', '626.59', '627.25', '628.25']


 Relap5-3D result read !
 parameter2 column ['1943.1', '1701.7', '1524.5', '1398.5', '1304.6', '1226.6', '1162.2', '1110.1', '1066.5', '1027.5',
'994.51', '964.43', '937.33', '912.67', '890.87', '871.44', '852.46']


 Relap5-3D result read !
 time column ['100.0', '102.5', '105.0', '107.5', '110.0', '112.5', '115.0', '117.5', '120.0', '122.5', '125.0', '127.5'
, '130.0', '132.5', '135.0', '137.5', '140.0']

 parameter1 criteria is 629 & parameter2 criteria is 818

 Relap5-3D result copied in workdir


 Relap5-3D result removed


 parameter comparisons are done

 parameter1 satisfied False, parameter2 satisfied False

 Previous simulation time is 140.0(s)


 Modified simulation time is 150.0(s)
```

Next, it reads the generated Relap5-3D result file and finds those criteria that are designated as flags in the procedure file. Figure 5 shows two target parameters and a time variable being read. As mentioned, the frequency of the parameter is printed at 2.5-second intervals, as defined in the Relap5-3D input. Next comes the process of checking whether the variable of interest satisfies the criteria. If not, it generates a False result and does not exit the loop. Furthermore, the simulation time must be increased to check the parameter change, so the simulation time is automatically updated and the plant simulation conducted once again. In addition, as shown in Figure 5, the Relap5-3D file is copied to the working directory for repeated calculation in the loop. It deletes the file in the existing folder as well.

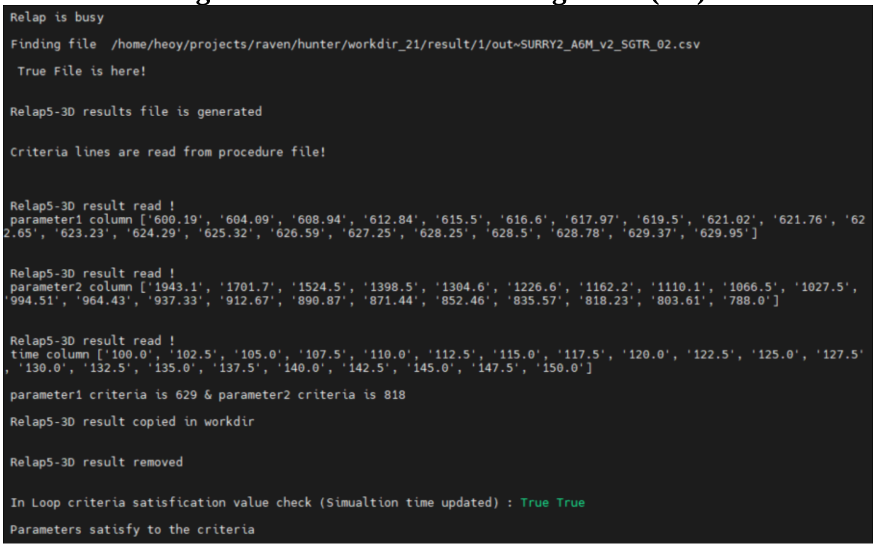
Figure 6. Screenshot of running codes (3/4)



```
 Simulation Time is extended for 150.0 (s)


 Simulation time variable is updated

 RAVEN input file is generated


 Simulation time in Raven is updated

Relap5-3D input is copied for the next update
CONDA
raven_libraries          /home/heoy/.conda/envs/raven_libraries
Running RAVEN in "opt" mode.
Loading plugin "ExamplePlugin" at /home/heoy/projects/raven/plugins/ExamplePlugin
 ... successfully imported "ExamplePlugin" ...
Unable to load "memoryProfile" decorator; replacing with passthrough ...
Finding Prescient failed with The 'prescient' distribution was not found and is required by the application

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

      .---.        .------######       #####     ###   ###  ########  ###    ###
     /     \  __  /    --###  ###    ###  ###   ###   ###  ###        #####  ###
    / /     \(  )/    --###  ###    ###   ###  ###   ###  ######    ### ######
   //////   ' \/ `   --#######    #########  ###   ###  ###        ###  #####
  //// / // :    :   -###   ###   ###   ###   ######   ####       ###   ####
 // /   /  /`    '---###    ###  ###   ###     ###    ########  ###    ###
//          //..\
===========UU====UU=============================================================
           '//||\`
             ''``
```

In this arbitrary example, we increase the simulation time by 10 seconds. Figure 6 shows the new simulation time as 150 seconds. Next, we rerun RAVEN in order to run Relap5-3D. Just as though restarting from the loop, it modifies the RAVEN input file, simultaneously changes the plant simulation, and monitors the completion of the simulation to read the result file. Afterward, the result is compared with the criteria once again, and if the target plant parameter in the loop satisfies the criteria, it exits the loop, ends the calculation, and saves the result. This process is shown in Figure 7 as well.

Figure 7. Screenshot of running codes (4/4)



```
 Relap is busy

 Finding file  /home/heoy/projects/raven/hunter/workdir_21/result/1/out~SURRY2_A6M_v2_SGTR_02.csv

  True File is here!


 Relap5-3D results file is generated


 Criteria lines are read from procedure file!



 Relap5-3D result read !
 parameter1 column ['600.19', '604.09', '608.94', '612.84', '615.5', '616.6', '617.97', '619.5', '621.02', '621.76', '62
2.65', '623.23', '624.29', '625.32', '626.59', '627.25', '628.25', '628.5', '628.78', '629.37', '629.95']


 Relap5-3D result read !
 parameter2 column ['1943.1', '1701.7', '1524.5', '1398.5', '1304.6', '1226.6', '1162.2', '1110.1', '1066.5', '1027.5',
'994.51', '964.43', '937.33', '912.67', '890.87', '871.44', '852.46', '835.57', '818.23', '803.61', '788.0']


 Relap5-3D result read !
 time column ['100.0', '102.5', '105.0', '107.5', '110.0', '112.5', '115.0', '117.5', '120.0', '122.5', '125.0', '127.5'
, '130.0', '132.5', '135.0', '137.5', '140.0', '142.5', '145.0', '147.5', '150.0']

 parameter1 criteria is 629 & parameter2 criteria is 818

 Relap5-3D result copied in workdir


 Relap5-3D result removed


 In Loop criteria satisfication value check (Simualtion time updated) : True True

 Parameters satisfy to the criteria
```

Coupling between HUNTER and RELAP5-3D was made successful by using RAVEN as an intermediary, as shown in the above figure. The example demonstrates how the virtual operator drives the plant via controls and monitoring. This process is simply scaled up when following a larger procedure, such as that for the SGTR scenario.

5. CONCLUSION

As described in the above proof of concept, we successfully coupled HUNTER with RELAP5-3D. To represent the virtual operators, we developed an iterative loop logic for application in the simulations. It involves reading procedures, writing Relap5-3D input deck with Raven, running simulations, modifying the input, and then re-running the simulations. However, there are two limitations to this approach. First, with a RELAP5-3D plant model we can simulate a variety of plants, but a RELAP5-3D code implementation is needed to represent operator manipulations accordingly. For operator manipulations, this requires an additional effort by the analyst to instantiate the initiating event in RELAP5-3D. Most RELAP5-3D models are simplified codes that do not represent the full spectrum of human actions. Second, for human error probability calculations or performance shaping factor determinations in HUNTER, we need an additional code for deriving the necessary plant information that exists outside the plant response data. In many cases, a variable such as available time should be derived. Available time is a function of how long an action takes relative to the amount of time available for performing that action before plant conditions begin to degrade. Available time may change according to the plant response after an operator manipulation is performed. Therefore, both simple plant simulation and multiple simulations through sampling must be performed simultaneously to determine the relevant variables. The duration of operator actions depends on the plant evolution, which is in turn a product of the operator actions. To derive available time, additional batch mode coupling should be developed for HUNTER, RAVEN, and RELAP5-3D.

One challenge in using RELAP5-3D is the need for synchronous coupling between the virtual operator in HUNTER and the thermal hydraulics in RELAP5-3D. This coupling has been demonstrated but represents the linking of several different pieces of code to achieve this functionality. For example, RELAP5-3D is fundamentally designed to run in batch mode, independent of starting and stopping. HUNTER makes use of RAVEN as a mediator, whereby RAVEN creates input decks for RELAP5-3D and outputs log files for use by HUNTER. RAVEN handles the starting and stopping, employing reruns with new configurations in response to operator manipulations. This is accomplished against the backdrop of RELAP5-3D being run by RAVEN in Monte Carlo repetitions according to a predefined distribution at the time of initiation. Such functionality is duplicative to the simulation run features built into HUNTER. RELAP5-3D exists as variants optimized for real-time dynamics with operator inputs. For example, RELAP5-HD from GSE Systems, a simulator vendor, provides real-time interactivity with RELAP5-3D [8]. RELAP5-HD provides the backend for synchronizing plant parameters with real human operator actions, thus affording a full-scope simulator for training purposes. A wide variety of human actions are included as initiating events in the plant models. Additionally, recent simulator implementations feature APIs to enable real-time monitoring and control interjection from third-party software applications (e.g., human-machine interfaces) or, by extension, software such as HUNTER. The fact that such functionality is not native to RELAP5-3D complicates efforts to model realistic human-centered scenarios in HUNTER. Future work should seek to introduce such functionality into the HUNTER-RELAP5-3D interface, or to couple HUNTER specifically with software simulations such as RELAP5-HD. Tools such as RELAP5-HD may be more readily available at utilities where HRA is performed, enabling easier adoption of HUNTER in industry. A trade-off in using a customized version of RELAP5-3D that is optimized for human integration is that the temporal synchronization may preclude faster-than-real-time analysis. Future work will strive to determine the best method for coupling HUNTER to the Environment module and provide guidance for real-time vs. faster-than-real-time applications of dynamic HRA.

Acknowledgements

This work was supported by the Nuclear Global Internship Program through the Korea Nuclear International Cooperation Foundation (KONICOF), funded by the Ministry of Science and ICT, the Korea Institute of Energy Technology Evaluation and Planning (KETEP), and the Ministry of Trade, Industry and Energy (MOTIE) of the Republic of Korea (No. 20203210100150).

References

[1] Aumiller, D.L., Tomlinson, E.T., & Bauer, R.C. *"A coupled RELAP5-3D/CFD methodology with a proof-of-principle calculation,"* Nuclear Engineering and Design, (2001).

[2] R. Boring, et al. *"Integration of Human Reliability Analysis Models into the Simulation-Based Framework for the Risk-Informed Safety Margin Characterization Toolkit,"* INL/EXT-16-39015, Idaho National Laboratory, (2016). https://doi.org/10.2172/1371517.

[3] Y.-J. Choi. *"Assessment of verification and validation status - RELAP5-3D and RAVEN,"* INL/EXT-19-56151, Idaho National Laboratory, (2019).

[4] Y.-J. Choi, et al. *"Demonstration of the Plant Fuel Reload Process Optimization for an Operating PWR,"* INL/EXT-21-64549, Idaho National Laboratory, (2021).

[5] Y.-J. Choi and C. Parisi. *"Risk-informed multi-physics best-estimate plus uncertainties (BEPU) application development of RELAP5-3D perturbation model,"* INL/EXT-20-59594, Idaho National Laboratory, (2020).

[6] Y.-J. Choi, *"Coupling of the Dynamic Human Reliability Assessment Capability with RELAP5-3D Thermal-Hydraulics Code,"* INL/EXT-21-64230, Idaho Falls, (2021).

[7] Z. Ma, et al. *"Risk-Informed Analysis for an Enhanced Resilient PWR with ATF, FLEX, and Passive Cooling,"* INL/EXT-19-53556, Idaho National Laboratory, (2019). https://doi.org/10.2172/1777257.

[8] GSE Systems. "*RELAP5-HD: A high-definition RELAP5-3D application*," Sykesville, MD: GSE, (2012).