RELOAD SAFETY ANALYSIS AUTOMATION TOOLS

F. Havlůj, J. Hejzlar, R. Vočka

ÚJV Řež, a.s.
Hlavní 130, Řež, 25068 Husinec, Czech Republic
http://orf.ujv.cz

Abstract: Performing core physics calculations for the sake of reload safety analysis is a very demanding and time consuming process. This process generally begins with the preparation of libraries for the core physics code using a lattice code. The next step involves creating a very large set of calculations with the core physics code. Lastly, the results of the calculations must be interpreted, correctly applying uncertainties and checking whether applicable limits are satisfied.

Such a procedure requires three specialized experts. One must understand the lattice code in order to correctly calculate and interpret its results. The next expert must have a good understanding of the physics code in order to create libraries from the lattice code results and to correctly define all the calculations involved. The third expert must have a deep knowledge of the power plant and the reload safety analysis procedure in order to verify, that all the necessary calculations were performed. Such a procedure involves many steps and is very time consuming.

At ÚJV Řež, a.s., we have developed a set of tools which can be used to automate and simplify the whole process of performing reload safety analysis. Our application QUADRIGA automates lattice code calculations for library preparation. It removes user interaction with the lattice code and reduces his task to defining fuel pin types, enrichments, assembly maps and operational parameters all through a very nice and user-friendly GUI. The second part in reload safety analysis calculations is done by CycleKit, a code which is linked with our core physics code ANDREA. Through CycleKit large sets of calculations with complicated interdependencies can be performed using simple and convenient notation. CycleKit automates the interaction with ANDREA, organizes all the calculations, collects the results, performs limit verification and displays the output in clickable html format.

Using this set of tools for reload safety analysis simplifies and organizes the whole process, reduces the chance of introducing error and saves an enormous amount of time as the calculations and results interpretation are performed almost with a click of a button.

Keywords: Reload Safety Analysis, Core physics calculations.

1. Introduction

Core reload safety analysis is crucial for NPPs from the operating point of view. It is vital in proving the safety of the core reload design. This is done by verifying that the limitations on power distribution, reactivity coefficients, the control protection system, etc. are met. At the Temelín NPP performing reload safety analysis is accompanied by the following challenges (many of which are pertinent to all plants).

a) The complicated nature of the limitations on core physics parameters leads to the fact, that the reload safety analysis is not a simple task. Performing this task requires running many core physics calculations, which are furthermore often inter-dependent. The process of evaluating the fulfillment of these limitations is just as challenging.

b) Unlike the safety analysis, which is performed once based on bounding core characteristics, the reload safety analysis is performed again for each new core reload design. It is a task done over and over again.

c) Performing core physics calculations involves using a diffusion core physics code. Since each core reload design at Temelin NPP comes with new fuel types this means, that doing a reload safety analysis involves preparing cross sections using a core lattice code.

Based on the challenges above it is clear that performing core reload safety analysis is a very demanding and time consuming process involving many steps. What more, it requires deep understanding of several areas including cross section preparation using the lattice code, core physics calculations using the diffusion code, plant operational regulations as well as programming skills, without which the whole task is nearly impossible.

To solve these many difficulties we have, at ÚJV Řež, a.s., developed a set of tools which are used to automate the whole process.

2. QUADRIGA

The first step in performing reload safety analysis is the preparation of homogenized cross sections for the core physics diffusion code through a transport-based lattice code. This process consists of the following steps.

1) Defining the input parameters (fuel geometry, pin enrichments and their placement, operational parameters)

2) Preparing the input deck for the lattice code

3) Running the lattice code and parsing the results

4) Preparing homogenized cross section libraries for the core physics code

QUADRIGA is a web application which accomplishes these tasks. Therefore the whole process is quite different when using QUADRIGA. First of all, a lattice code expert comes along and prepares a set of modular templates, which generate the input deck based on defined input parameters. The template language offered by QUADRIGA allows for the use of conditional statements, loops, temporary variable definitions and much more. This leads to the fact that templates themselves are quite short, which makes them easy to navigate and understand. Given the fact that their definition is quite general (they are not filled with particular values as the expert works with variables, which are later replaced by particular values), different fuels share identical templates making it much easier to make changes or extend existing fuel types. QUADRIGA is not tied to a particular lattice code and supports the use of multiple lattice codes.

Next off, another expert comes along and prepares physics models, namely fuel temperature and water density models. These models allow QUADRIGA to generate fuel temperature profiles and calculate water density based on the current operational parameters. In addition the expert can pre-define sets of operational parameters for different fuel types.

Now is the time where the regular user steps in. His big advantage is, that he does not need to interact with the underlying lattice code and can concentrate solely on the definition of a new fuel type. In most cases this just means:

1) Defining a new pin map, through a point and click interface (see Fig. 1).
2) Creating a new case for this pin map. This is where QUADRIGA dynamically generates the set of input operational parameters to be defined. In the production case these parameters are however already predefined by an expert and the task therefore reduces to picking the correct fuel type, pin map and naming the case.
3) Executing the case. QUADRIGA takes care of deploying the jobs on available computer clusters using its load balancer and monitoring the execution of the job.
4) Downloading the results.

All the cases are stored in QUADRIGA's case database (see Fig. 2). This is where users can organize cases into different folders, view case input parameters, view the lattice code input deck or output files, messages displayed during execution, the status of the case and much more.

3. CycleKit

The task of performing a reload safety analysis consists of verifying the reload safety analysis checklist (RSAC) which is a set of parameters and their limiting values which must be fulfilled. The steps needed to accomplish this process are as follows:

1) Preparation of a model for the cycle (reload pattern)
2) Execution of burnup calculations
3) Calculation of specified set of states
4) Exctraction of parameter values, application of uncertainties and comparison with limiting values

This process is not as straightforward as it appears. Rarely does the verification of one parameter involve just one calculation with the core physics code. Core characteristics are dependent on the current operational parameters including time in cycle, input temperature, flow, control rod positions, xenon distribution, etc. Therefore calculations may consist of steady-state power distributions for different rod positions and time in cycle or power transients such that the axial offset is within permissible bounds. As a consequence, the resulting set of calculations is quite large and the calculations themselves can be quite complex and inter-dependent.

The verification of compliance with limiting values itself is not always straightforward. Limited are for example local pin power distributions. The limiting values are dependent on local pin burnups. Furthermore pin power uncertainties are dependent on pin positions within the assembly. Therefore it does not suffice to find the maximum local pin power and compare it with some limiting value. The task of checking these criteria may also be demanding.

The general approach to solving these tasks is illustrated in Fig. 3. The user “on the fly” creates scripts for specific tasks. The pitfalls of this approach are apparent. First of all, the user is doing a lot of work, managing the different scripts, the burnup model, etc. This is not only time consuming but leaves a lot of potential for human error. Secondly, the scripts are often specifically tailored, which means they are not easily editable or extensible. Lastly, it is not easy to organize the process as a whole since there are a lot of things to look after.

CycleKit is a code which solves many of these downfalls. It can be used to perform the necessary calculations and to evaluate the RSAC. It is linked with the core physics code ANDREA, managing the input files, execution and the parsing of its output files. Users therefore do not need to interact with the core physics code directly, although some knowledge is beneficial. The process of performing the reload safety analysis with CycleKit is illustrated in Fig. 4. By having one code for everything code redundancy is removed and user interaction is reduced to a minimum.

The way CycleKit works is by allowing the user to define calculation sequences. These sequences may be performed at various times in the cycle and for various initial conditions. They may also be arbitrarily linked together.

A calculation sequence is defined by a general syntax, which is translated into calculations with the ANDREA code. The syntax allows for conditional decisions based on parameters of previous steps, loops, evaluation of results. It also allows for searching, i.e. changing calculation inputs until specified conditions are fulfilled. This may, for example, be used to find such an input temperature for which the moderator temperature coefficient is found to be negative.

The calculation sequences are then evaluated by CycleKit, checking the fulfillment of limits for specified parameters. The results of these evaluations are further processed into a clickable output file and a parameter checklist (see Fig. 5 and Fig. 6). These allow for viewing and interpreting the results.

Since the definition of the calculation sequences (thanks to the general nature of the syntax) is often a one-time process, performing the reload safety analysis reduces to the click of a button.

4. Conclusion

The tools developed by ÚJV Řež, a.s. simplify the process of reload safety analysis by

1) Reducing the expertise involved. This is done by:
    a. Removing user interaction with the lattice code for cross section preparation.
    b. Reducing interaction with the core physics code.
    c. Removing the requirement for advanced programming skills.
2) Automating repetitive tasks. This leads to error prevention by reducing human factor. It also converts valuable human time to machine time.
3) Organizing calculations and results. Web applications with GUI's aid in visualizing and understanding the underlying calculations without going into excruciating detail.

As a result, NPP personnel at Temelín can themselves verify their proposed reload pattern by performing the reload safety analysis a process, which has now become much more accessible, less error prone and not so time consuming.

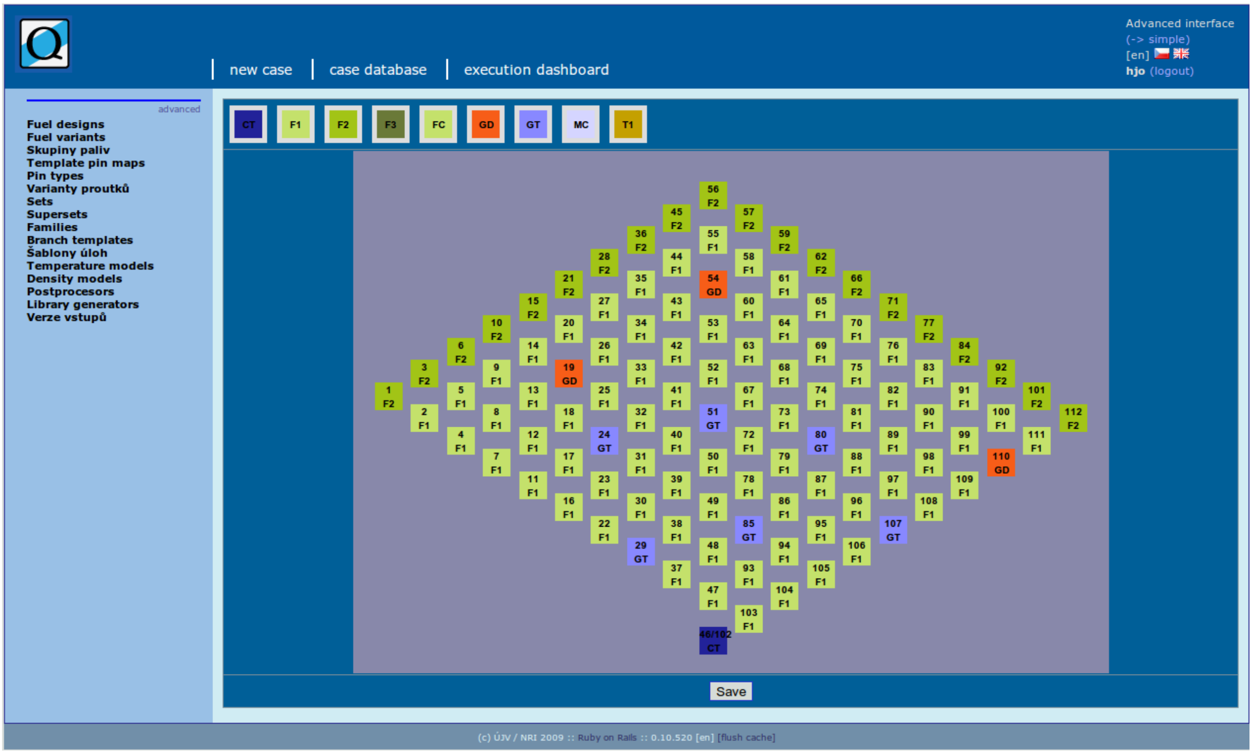

Fig. 1 QUADRIGA's point and click pin map definition interface

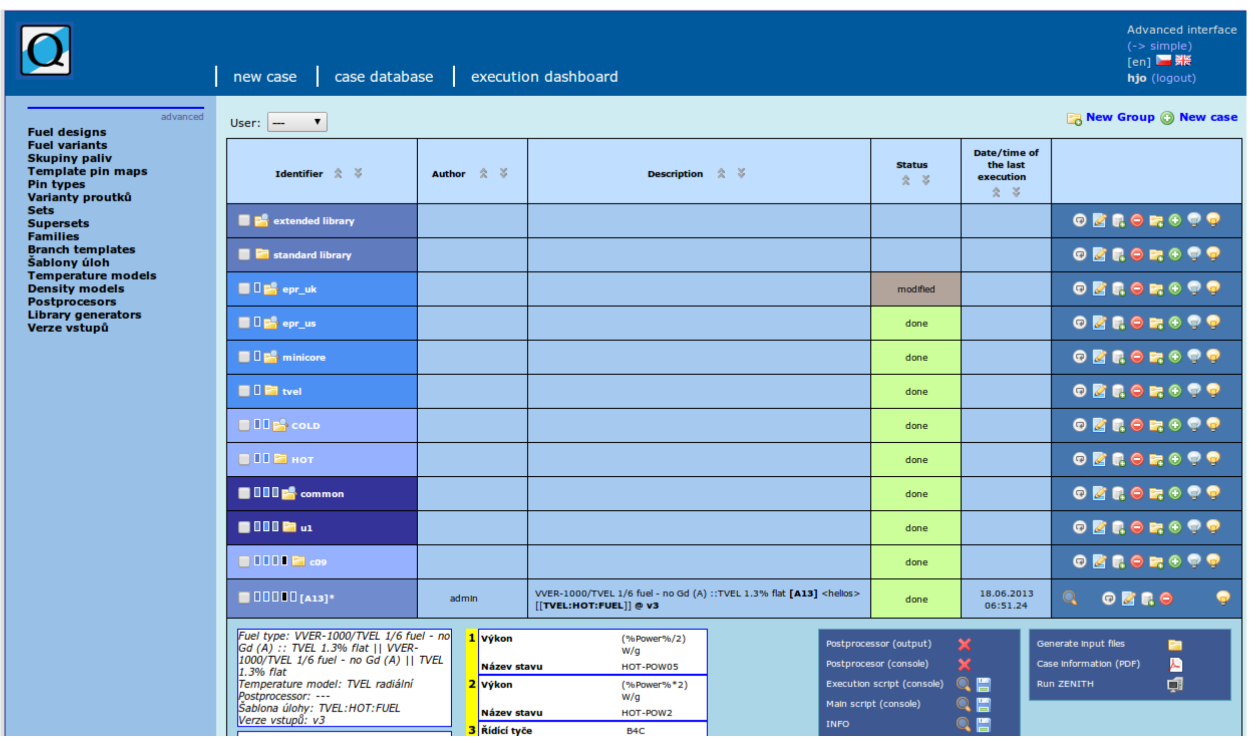

Fig. 2 QUADRIGA's case database

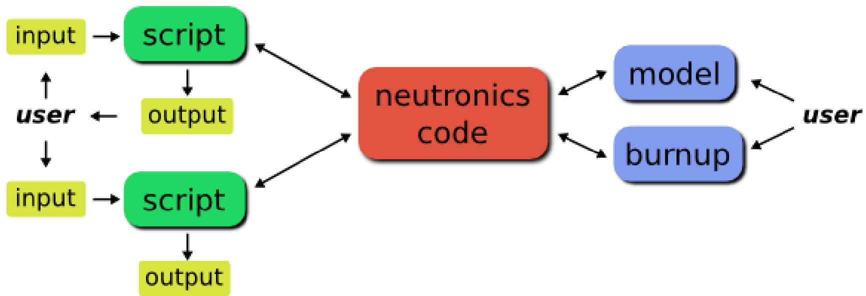

Fig. 3 General approach to reload safety analysis

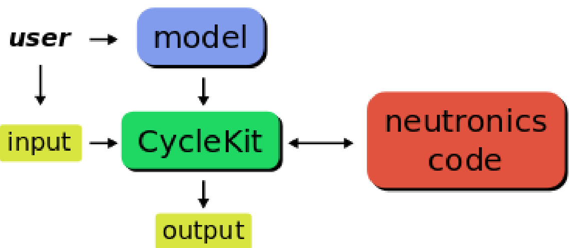

Fig. 4 Reload safety analysis schema when using CycleKit

Evaluation results

module	**Reactivity coefficients**	**start**	**Burnup calculations**	Calculation of reactivity coefficients		
section	**MTC**	**FTC**	**BCF**	moderator temperature and density reactivity coefficient		
start	**HFP ARO**	**HFP ARO Xe0**	**HFP ARO EOC**	**HZP RIL Xe1**	**HZP ARO**	HZP, All Rods Out, No Xenon
table	**MTC**	**MDC**	Moderator temperature reactivity coefficient			

[mtc] Moderator temperature reactivity coefficient [10^{-5}/°C] **cf / mtc / start.hzp_aro / mtc**

T_{eff}	T_{in}	BC	Power	Flow	CR10	Xe	MTC	uE	uF	min	max	lim-min	lim-max	margin
0.000	278.0	6.774	0.0	18268	373.0	2.53e-24	-1.27	3.00	1.10	-4.27	1.73	-55.00	0.00	-1.73
3.000	278.0	6.590	0.0	18268	373.0	2.53e-24	-1.82	3.00	1.10	-4.82	1.18	-55.00	0.00	-1.18
10.000	278.0	6.437	0.0	18268	373.0	2.54e-24	-1.17	3.00	1.10	-4.17	1.83	-55.00	0.00	-1.83
70.000	278.0	6.141	0.0	18268	373.0	2.59e-24	0.38	3.00	1.10	-2.62	3.38	-55.00	0.00	-3.38
150.000	278.0	5.169	0.0	18268	373.0	2.62e-24	-5.59	3.00	1.10	-8.59	-2.59	-55.00	0.00	2.59
300.331	278.0	2.763	0.0	18268	373.0	2.66e-24	-20.56	3.00	1.10	-23.56	-17.56	-55.00	0.00	17.56

Fig. 5 Calculation sequence evaluation results

Fig. 6 Parameter checklist generated by CycleKit