Transactions of the Korean Nuclear Society Autumn Meeting
PyeongChang, Korea, October 30-31, 2008

Development of a Wrapper Object, TRelap, for RELAP5 Code for Use in Object Oriented Programs

Young Jin Lee
Korea Atomic Energy Research Institute, Dukjin 150, Yuseong, 305-600 Daejeon, Korea
yjlee1@kaeri.re.kr

1. Introduction

TRelap object class has been developed to enable object oriented programming techniques to be used where functionality of the RELAP5 thermal hydraulic system analysis code[1] is needed.

The TRelap is an object front for Dynamic Link Library (DLL) manifestation of the Relap5 code, Relap5.dll. In making the Relap5.dll, the top most structure of the RELAP5 was altered to enable the external calling procedures to control and the access the memory. The alteration was performed in such a way to allow the entire “fa” and the “ftb” memory spaces to be accessible to the calling procedure. Thus, any variable contained within the “fa” array such as the parameters for the components, volumes, junctions, and heat structures can be accessed by the external calling procedure through TRelap. Various methods and properties to control the RELAP5 calculation and to access and manipulate the variables are built into the TRelap to enable easy manipulation.

As a verification effort, a simple program was written to demonstrate the capability of the TRelap.

2. Methods and Results

When developing the Nuclear Plant Analyzer (NPA) and simulators, the integration of program routines in thermal hydraulics, computer graphics, man-machine interface and others are needed. However, these different routines are usually written in different program languages. When developing large scale programs that require integration of program routines from different disciplines, it is advantageous to use the modern programming languages with OOP (Object Oriented Programming)[2] feature. The OOP has advantages in code maintenance, re-useability, and ease-of-use and has been a popular programming technique for over a decade in various software development fields. The OOP begins to have substantial advantages when the program sizes and the required integration become large.

The nuclear thermal-hydraulic field has been very slow in accepting the OOP techniques. Part of the reason is that most thermal hydraulic system analysis codes were written in outdated Fortran language without OOP considerations. Although the RELAP5 code has a good solid modular structure amenable to transition to modern OOP language, it would be a considerable undertaking to convert RELAP5 to use the OOP features. Much easier and simpler transition to OOP is to develop a wrapper object class for the RELAP5. In this light, and the TRelap, which is a wrapper object of the DLL version of RELAP5, has been developed.

In this section some of the techniques used to develop the TRelap are described.

2.1 RELAP5.DLL

The first step in the development of TRelap was to develop the DLL version of the Relap5 code, Relap5.dll. The main requirements for the Relap5.dll was to facilitate the external “drive” routine to control the time-advancement, and have the read/write access to entire “fa” and the “ftb” memory spaces thereby effectively having near complete control of the RELAP5 program. The near complete control requirement was deemed essential if TRelap was to be used as thermal hydraulic engine for such programs as the NPA and simulator.

To realize these requirements, it was needed to modify the top-most structure of the Relap5. The relap5.f was re-coded to include the codings of trnctl.f and the tran.f. Also, some coding was necessary to calculate the “fa” and “ftb” memory locations. The overview of the coding structure change is shown in Figure 1.

The relap5.dll has just a single exported procedure having 7 parameters. 2 of the parameters are the memory addresses, and the rest are parameters used either for controlling the calculation flows or for showing the status of the calculation.

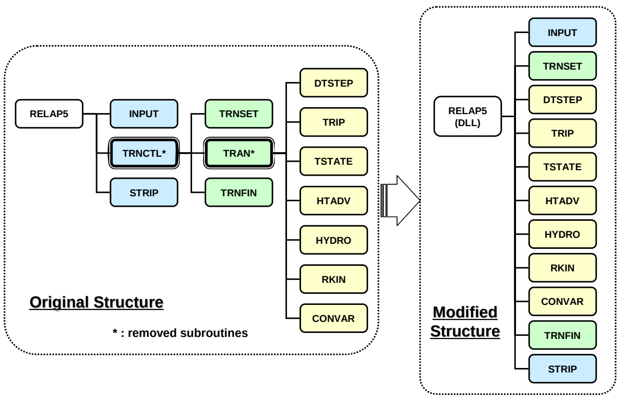

Figure 1. Re-coding of relap5.f to make relap5.dll

2.2 TRelap

TRelap is a wrapper for the Relap5.dll written in Delphi (Object Pascal)[3]. TRelap is a true object in the sense that it can be created, inherited, polymorphed and destroyed as needed, and multiple instances of TRelap can be created with careful input/output/rstplt file maintenance.

Major public properties of TRelap include:

- time step advancement count
- hydro time
- indices to access variables of components, volumes, trips, heat structures, etc.
- lists in text form of components, volumes, junctions, trips, etc.

Major public methods include:

- method to create an instance of TRelap
- method to destroy an instance of TRelap
- method to read input and carry out initialization
- method to run specified number of time steps
- method to run to the end of problem
- methods to obtain general information, volume information, junction information, heat structure information, component information, and others in text format

If more functionality is required, it is a simple matter of adding more methods or properties to the TRelap. In addition, because the TRelap is a true object of OOP, it is possible to make new object classes through inheritance and polymorphism where the variables and methods can be added, encapsulated, overloaded or overridden.

The detachment of the thermal-hydraulic calculation from other modules such as GUI, through the use of TRelap makes it simpler to modify the thermal hydraulic routines, as the modification would not, in most case, affect the other modules.

2.3 Demonstration program

In order to test and demonstrate the functionality of TRelap, a simple GUI (Graphic User Interface) based program was written. The program was written in Delphi and consists of TRelap and the GUI program modules. The program reads in the input deck, creates an instance of TRelap, controls the time step advancement (run/pause/stop), and examines the minor edit variables. Most of the functions can be carried out interactively using the GUI.

Figure 2 shows the screen captures obtained during the excution of the demonstration program. With the TRelap, the write up of the thermal hydraulic part of the program was a trivial effort. Most of the efforts in producing the demonstration program were in the coding of the GUI for the interactive program execution.

The TRelap performed in stable manner with only a slight degradation in execution time. The results of the calculations were identical to those calculated using the original relap5.

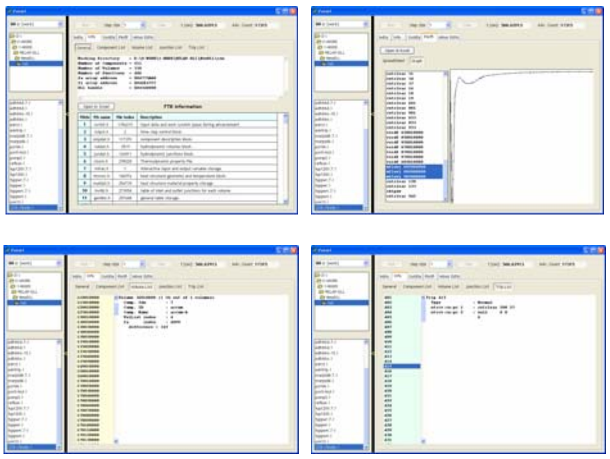

Figure 2. Screen Captures of the Demonstration Program

3. Conclusions

An object class, TRelap, for the best-estimate thermal hydraulic system analysis code, RELAP5, has been developed successfully.

The RELAP5 code has been an effective tool for safety and performance analyses, and with the vast improvement in computing power, the use of the RELAP5 code for personal computer based simulators and NPA has become practical.

Use of TRelap wrapper object has proven to be an efficient means of circumventing the difficulties of applying modern programming techniques such as OOP on Fortran based RELAP code. Although the TRelap has been created with least amount of re-coding of the original RELAP5, it is a true object in OOP.

The demonstration program has shown that TRelap can vastly simplify the integration of RELAP5 with the GUI program modules by effectively detaching the RELAP5 from the GUI modules. TRelap has shown that RELAP5 functionality can be added to programs with minimum efforts. TRelap has also shown that object oriented programming can improve code maintenance and re-useability.

REFERENCES

[1] "RELAP5/MOD3.3 Code Manual Volume VIII : Programmers Manual", Idaho National Laboratoty, NUREG/CR-5535, 2001.
[2] "Object Oriented Software Construction", Bertrand Meyer, Prentice Hall, 1997.
[3] Borland Software Corporation, Delphi 6 for Windows Developer's Guide, 1998.