Transactions of the Korean Nuclear Society Spring Meeting
Chuncheon, Korea, May 25-26 2006

A Restructuring of the CAV and FDI Package for the MIDAS Computer Code

S.H.Park a, D.H.Kim a, S.W.Cho b
a T/H Safety Research Division, KAERI, P.O.Box 105, Yusong, Daejon, 305-600 Korea, shpark2@kaeri.re.kr
b Korea Radiation Technology Institute Co. Kusongdong 19, Yusong, Daejon, 305-353 Korea

1. Introduction

As one of the processes for a localized severe accident analysis code, KAERI is developing a severe accident code MIDAS. The MIDAS code is being developed based on MELCOR. The existing data saving method of MELCOR uses pointer variables for a fix-sized storage management, and it deteriorates the readability, maintainability and portability of the code.

As a most important process for a localized severe accident analysis code, it is needed convenient method for data handling. So, it has been used the new features in FORTRAN90 such as a dynamic allocation for the restructuring[1,2]. The restructuring of the data saving and transferring method of the existing code makes it easy to understand the code. Before an entire restructuring of the code, a restructuring template for a simple package was developed and tested.

The target for the restructuring in this paper was the CAV and FDI packages. The CAV(cavity) package is responsible for modeling the attack on the basement concrete by hot core materials. The FDI(Fuel Dispersal Interactions) package is responsible for modeling both low and high pressure molten fuel ejection from the RPV into the reactor cavity, control volumes and surfaces. The verification was done through comparing the results before and after the restructuring.

2. Existing Structure

The MELCOR code is composed of 3 parts: MELGEN which checks the input data and makes the data file called restart needed for the calculation, MELCOR which calculates with time using the restart file and makes a log/plot file, and the PLOT- related programs.

Each part has information including the physical packages such as COR, HS, SPR, FL, CVH, DCH, RN1, RN2, and shares the data through each package, with dozens of subroutines per each package, and all the hundreds of subroutines which record the messages and control the execution flow.

At first the data saving and transfer method is analyzed[3]. The subroutines which read / write the restart file are MXXRS and MXXRSW, and they read / write according to the entry conditions. Through this process, arrays for 4 data types (real, integer, logical, and character) are read / written for each package.

MELCOR reserved and used 4 data types for the data storage and transfer effectively. Data is saved in the most effective way within a fixed array and it is transferred through 2 steps. In the first step, the information of each package for each data type (the start location and the size) is conveyed to the next step. At the second step, pointer variables used within each package are transferred as arguments, and they are used as local variables hereafter.

To analyze the pointer variables, comments and arguments passing within the subroutines which are related to data movements are analyzed. The pointer variables of the first step are applied similarly to all the packages and they are included as a common block named 'CAVDB' and 'FDIDB' for CAV and FDI package within several subroutines. The second step pointer variables are applied differently according to the packages, defined in the common blocks named CAVPNT, FDIPNT for CAV and FDI package, and it point out a specific location among the database.

The contents that the pointer variables convey can be searched in the subroutines within the packages. Based on these contents, the module is constructed for the CAV and FDI packages. Also the data was found to be referenced in the BH, CF, COR, CVH, EDF, RN1, RN2 packages besides the target package.

3. Development of the Restructured CAV and FDI

Before an entire restructuring of the code, a restructuring template for a simple package was developed and expanded into the entire MELCOR code[4,5]. the new features in FORTRAN90 such as a dynamic allocation was used to apply for code restructuring. The data transfer structure in the CAV and FDI packages were modularized from a single array into a derived data type for an easy understanding and usage. Also the variables used in these packages were transformed into a modularized type, and their efficiency was increased through a dynamic allocation.

200 subroutines in the CAV and FDI packages and 50 subroutines in the other packages using this data were analyzed, through a minimum manual job using an automatic conversion program(MeltoMid) related to the subroutines that were restructured[6].

3.1 Construction of the Module

In MELCOR, the subroutine CAVDBD and FDIDBD, which are the second level of the database manager, connects with subroutine CAVRUN and FDIRN1. The subroutine CAVRUN and FDIRN1 then apply the actual-meaning variables instead of the pointer variables. In the restructured CAV and FDI, these pointer variables were removed by using the module CAV_MDL and FDI_MDL which were constructed based on the analysis of the

pointer variable's usage. The ALLOCATE statement in the module allowed for a dynamic storage allocation.

3.2 Subroutine Reorganization

In advance of the restructuring of the subroutines, the whole existing MELCOR code written in FORTRAN77 was transferred into FORTRAN90 like the restructuring process of the other packages. It was confirmed that the results before and after a conversion were the same. Among the subroutines in the CAV and FDI package, only 33 and 8 subroutines were found to use the pointer variables, and the others used the local variables for data transfer. All the pointer variables in the 41 subroutines were transformed into meaningful variables defined in module CAV_MDL and FDI_MDL.

In the newly constructed module, most of the array variables were transformed into member variables which have a meaning for the implication. The comparison of the variables before and after a modification is shown in Table 1.

In subroutine CAVDBD and FDIDBD the pointer variables in an argument are removed and in subroutine CAVRUN and FDIRN1 the direct variables are used instead of the local variables.

The target package's data used in the BH, CF, COR, CVH, EDF, RN1, RN2 packages as well as the target packages, are also analyzed and restructured.

Table 1. Contents of the array change

Usage in existing method
NCAV(NUMCAV)
NCVCAV(NUMCAV)
ICAVBB(NUMCAV)
IDEP(NUMFDI)
LSEXIN(NUMFDI)
NFDCAV(NUMFDI)
Usage in new method
CAV_NM(NUMCAV)%NCAV
CAV_NM(NUMCAV)%NCVCAV
CAV_NM(NUMCAV)%ICAVBB
FDI_NF(NUMFDI)%IDEP
FDI_NF(NUMFDI)%LSEXIN
FDI_NF(NUMFDI)%NFDCAV

4. Result and Verification

In order to verify the new results, a two-step process was done. At first, a simple language conversion process from FORTRAN77 to FORTRAN90 was checked by comparing the major variables in the CAV and FDI packages before and after a conversion. To compare the results, the data for the most of packages are included in input files such as core fuel parameter, characteristics of core nodalization, flow path and containment volumes. Some sequences were calculated such as the steady state and SBO (Station Blackout) accident. The comparison was performed in a nuclear power plant of 4300MW_t until 15,000 sec and in APR1400 until 80,000 sec. The major variables were the same. Therefore the language conversion was confirmed to be successful.

The next step, which was the main part here, which was to verify the new CAV and FDI results against the current results. As only the CAV and FDI packages were restructured using the module and the derived-type variables, the interface program was developed for the data communication between the restructured target packages and the other original packages. The interface program was checked by comparing the values of the target packages variables with the original values, and it provided the same values for the target packages variables after the read/write processing.

5. Conclusions

To restructure the data storage and transfer scheme, the data management process was analyzed for the entire MELCOR code. Especially for the CAV and FDI packages, the data information and pointer variables were analyzed, and the modules were configured and applied to the target packages. In this procedure, the data structure was restructured to remove the pointer variables by using the direct variables through the FORTRAN90 features such as the MODULE and USE statements.

Using the reconstructed modules, the subroutines in the target packages were restructured. By comparing the important variables in the CAV and FDI packages, it was confirmed that the results were the same but differences appeared in the CPU time comparison. They are supposed to be caused by an array transformation in some subroutines within MELCOR. Based on the various results, the output of various executing results, and consulting other experiences, it has been confirmed that the CAV and FDI restructuring was done right.

Therefore, the propriety of the restructuring applied to the CAV and FDI packages were verified. Through the restructuring process, the base was constructed for an easy grasp of the variables everywhere in the code and it became easy for a code improvement and for an addition of new models. The restructuring process proposed in this paper will be extended to the entire code for the MIDAS development.

REFERENCES

[1] A Multi-Dimensional Thermal-Hydraulic System Analysis Code, MARS 1.3.1, Vol.31, Number 3, pp.344-363, June 1999.

[2] S.H.Park, H.D.Kim, D.H.Kim, Y.M.Song, B.D.Chung, Development of Restructuring Template for MELCOR, 5th PSAM Conference, Japan, 2000.11.26 – 2000.12.2

[3] D.H.Kim, S.H.Park, Analysis of MELCOR Code Structure, KAERI/TR-1543/00, June, 2000.

[4] D.H.Kim, S.H.Park, Y.M.Song, A Restructuring Proposal Based on MELCOR for Severe Accident Analysis Code Development, KAERI/TR-1536/2000, March, 2000

[5] D.H.Kim, et al., Experimental and Analytical Research on Severe Accident Phenomena, KAERI/RR-2216/2001, May, 2002.

[6] Y.M.Song, S.H.Park, D.H.Kim, Development of a Computer Program for Automatic Variable Conversion in MELCOR Code, Proceedings of the Korean Nuclear Autumn Meeting, 2000.