1 of 1

Conf-930601--24

WSRC-MS-93-011

Migrating Criticality Safety Calculations From Mainframe Computing At Savannah River

by

J. F. Mincey*

* Westinghouse Savannah River Company
Savannah River Technology Center
P.O. Box 616, Bldg. 773-22A
Aiken, SC 29808

A summary of an invited paper proposed for presentation at the
ANS 1993 Annual Meeting and Embedded Topical "Nuclear Criticality Safety"
San Diego, California
June 20 - 24, 1993
and for publication in the proceedings.

Migrating Criticality Safety Calculations From Mainframe Computing At Savannah River

by

J. F. Mincey
Westinghouse Savannah River Company
Savannah River Technology Center
Aiken, SC 29808

Introduction

The Savannah River calculational environment is beginning migration away from IBM 3090/3081 and CRAY mainframe computing. Preliminary code migrations to workstations and personal computers have been made but such use has been limited to unofficial "scoping" calculations. Lessons learned suggest migration, while a boon to the end-user, presents a variety of quality assurance issues. These are more pronounced than in mainframe environments and suggest a need to rethink the code custodian's role.

Discussion: Current Status

Savannah River uses both adapted industry-standard codes and codes written by H. K. Clark within a modular code and data management system called JOSHUA[1]. Validated Savannah River criticality codes[2] for establishing subcritical limits are structured around JOSHUA database records and run on IBM 3090/3081 mainframes. A FORTRAN 77 (F77) version of JOSHUA has been certified and is necessary to migrate these codes into a workstation environment. For a few of these codes (and some SCALE[3] codes), personal computer versions[4,5] are run outside JOSHUA for purposes of familiarization with computing environment.

Work, some of which has been completed[6,7], is also in progress to validate MCNP 4.2[8] on a CRAY for dose assessment and subcritical limit calculations. Hardware and system support are currently being procured to migrate MCNP to RISC workstations for production calculations. Testing has been performed on RISC 550 and 530H machines.

The information contained in this article was developed during the course of work under Contract No. DE-AC09-89SR18035 with the U.S. Department of Energy.

Discussion: Historical Perspective

JOSHUA records were once a necessity due to limited mainframe computing hardware at Savannah River. Because manipulation and modification of JOSHUA records requires expertise largely unique to the site, a number of input processors were developed to improve the efficiency of calculational setup for the end-user not interested in record management niceties. These processors are quite powerful and can significantly automate input preparation for a long series of calculations. Unfortunately, the conversion of conventional input/output statements and preparation of record templates to convert industry-standard codes to run JOSHUA records is a non-trival exercise.

Inexpensive memory and mass storage now available in workstations makes conversion of industry-standard codes to JOSHUA records unnecessary. Resources would instead be better used improving the validations of all criticality safety codes used at Savannah River.

JOSHUA adapted/validated criticality codes will be maintained on a mainframe environment to permit reproducibility of past analyses. They will also be ported to a workstation/F77 JOSHUA environment. However, non-JOSHUA-adapted industry-standard codes (e.g., SCALE, MCNP, etc.) are expected to form the future nucleus of workstation criticality safety codes.

One taken-for-granted benefit of JOSHUA records was sensible and practical quality assurance arising as a natural byproduct. Because of the JOSHUA record structure, only a few people had the expertise to make changes to nuclear data libraries and source codes. These same people conducted training as requested for the end-user and independently verified the results of subcritical limit calculations. The result was and still is an enviable combination of practical configuration control and powerful code preprocessors to ease the end-user's task. It is desirable to promote a similar evolution on workstations and avoid force-fitting quality assurance in unnatural ways.

Discussion: Rethinking Quality Assurance Implementation

Ultimately the most important aspects of quality assurance on workstations are driven by the fact workstation end-users have no real dependence upon a central support cadre for the operating system. If this is recognized, it is possible to establish responsibilities of the code custodian, end-user, and technical reviewer in ways that make sense. A few issues that bear on assigning responsibilities follow.

Who is the Custodian, Custodian of What, and Why?

Control over a mainframe's operating system by someone without motivation to edit/modify controlled software has been the Savannah River approach to guard against unauthorized compilation of code revisions, editing of nuclear data, and other activities that would jeopardize validation. There is still debate as to whether this approach is viable on workstations; the view of this paper is that it is not generally viable.

Hence code custodianship must first be thought of as limited to distribution of binary code executables, data libraries, and documentation that meet site quality assurance requirements. Even with binary files, control cannot absolutely be guaranteed. Binary files will prevent inadvertent editing or stop casual modifications but will not deter a dedicated effort to alter controlled code/data libraries.

Under this assumption, custodianship is more a responsibility of ensuring the pedigree of the software and distribution of the software; not ensuring that the latest approved version is the only version available to the end-user in setting subcritical limits. That responsibility rests first with the end-user, controlling the software placed on his/her machine, and secondly with the technical reviewer.

An interesting combination is if the custodian is also the technical reviewer. Then there is less chance for using results obtained with the wrong data set or code version.

Who Determines When A Code is Validated, Validated for What, and Why?

The preceding scenario encourages the custodian to develop in-depth understanding of the code (provided adequate clerical help is available for making code distributions). The custodian's primary responsibilities would then naturally include code validation[9].

The technical reviewer also shares part of this responsibility. End-users will inevitably devise pre- and post-processor utility software for problem setup and automating display of the calculated results. Whenever this software potentially could misinterpret input or output (i.e., affect properly drawing conclusions), it has to be controlled. The technical reviewer must ensure that the custodian is alerted/involved with formalizing control of such utility software.

Again consider the custodian to also be the technical reviewer. If the custodian/technical reviewer also had access to supporting software writers, the end-user's functional needs could be met in a way that ensured quality assurance requirements were naturally incorporated rather than backfitted.

Finally, for codes such as MCNP, the custodian has to be technically diverse to validate the code in areas that extend beyond criticality safety. A group of custodians/technical reviewers may be required for general-purpose codes to cover all capabilities. It is technically meaningless to use the term "validated" unless the application for which a code is considered validated is specified.

Who Determines When An End-User Is Qualified, Qualified for What, and Why?

In a mainframe environment, it is possible to ensure only qualified users access controlled files. While this may prove undesirable on a mainframe for a number of reasons, on a work-station or personal computer it is not even an option. Control based upon the distribution of the software is also not adequate.

It is also clear that end-users should not be expected to know how to use all aspects of some codes. End-users of MCNP for subcritical limit calculations, for instance, have little need to perform electron transport calculations.

A practical form of control is for the custodian to build into the software a header that prints on each calculation the end-users qualified for specific code applications. The technical reviewer then verifies end-user qualification when the software is formally used. The technical reviewer, more than the custodian, is also in a position to judge when an end-user requires training.

Conclusion

The preceding discussion hopefully has shown that there is a natural role for the code custodian to play as validator, coordinator of utility program development, technical reviewer of calculations, and trainer in use of the codes.

However, the historical role of the custodian as responsible for ensuring the integrity of the software on an end-user's machine is not natural to the workstation/personnal computer environment. That responsibility must primarily become the end-user's.

References

1. H. C. Honeck, "The JOSHUA System", DPSTM-500, 9/73.

2. H. K. Clark, "User's Manual: JOSHUA Nuclear Criticality Safety Modules", DPSTM-86-700-3, 3/87.

3. CCC-545, "SCALE 4.1", RSIC 7/92.

4. CCC-548B, "KENO5A-PC", EG&G/Battelle Columbus/SAIC/ORNL, RSIC 5/30/91.

5. D. K. Parsons, "ANISN/PC Manual", EGG-2500, 4/87.

6. E. F. Trumble, "MCNP Certification Package", WSRC-TR-92-372, 8/92.

7. J. F. Zino, "Neutron and Photon Leakage Spectrum Benchmarking Calculations with MCNP", WSRC-TR-92-180, 4/92.

8. CCC-200, "MCNP 4.2", Los Alamos, RSIC 8/92.

9. ANSI/ANS-8.1-1983, *Nuclear Criticality Safety In Operations With Fissionable Materials Outside Reactors*, American Nuclear Society, reaffirmed 11/88.

DATE
FILMED
8/23/93

END