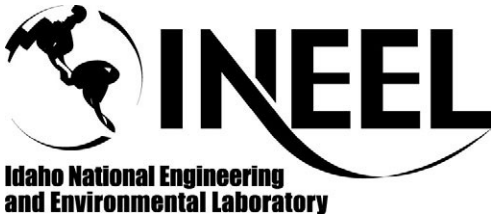

INEEL/CON-04-02223
PREPRINT

RELAP5-3D Architectural Development in 2004

Dr. George L. Mesina

August 24-27, 2004

RELAP5 International Users Seminar

This is a preprint of a paper intended for publication in a journal or proceedings. Since changes may be made before publication, this preprint should not be cited or reproduced without permission of the author.
This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, or any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for any third party's use, or the results of such use, of any information, apparatus, product or process disclosed in this report, or represents that its use by such third party would not infringe privately owned rights. The views expressed in this paper are not necessarily those of the U.S. Government or the sponsoring agency.

RELAP5-3D Architectural Development in 2004

Dr. George L. Mesina

Idaho National Engineering and Environmental Laboratory (INEEL)
Idaho Falls, ID 83145-3880, USA
Phone: 208-526-8612
Email: mesinagl@inel.gov

Proceedings of the RELAP5 International Users Seminar
Sun Valley, ID, USA, August 24-27, 2004

KEYWORDS: RELAP5-3D, Parallel Programming, Vectorization, Fortran 90

Abstract

Currently, RELAP5 is undergoing a transformation that will replace much of its coding with equivalent structured Fortran 90 coding. Four efforts are underway to modernize the code architecture of RELAP5-3D. These are parallelization, vectorization, code restructuring, and conversion to Fortran 90. The first two improve code run speed via on computer platforms of certain architectures. These code modifications have little effect on normal code performance on non-vector and non-parallel computers because they are mostly done with compiler directives.

The third and fourth efforts involve considerable rewriting of the source code. The third code improvement effort addresses code readability and maintainability. These are being greatly enhanced by application of a Fortran code-restructuring tool. The fourth effort is conversion to Fortran 90. The bulk of the coding is being rewritten in Fortran 90. This is a ground up reworking of the coding that begins with completely reorganizing the underlying database and continues with the source code. It will reach every part of RELAP5-3D.

Each of these efforts is discussed in detail in a different section. Section 1 relates background information. Section 2 covers the parallelization effort. Section 3 covers the efforts to vectorize the code. Section 4 covers the code restructuring. Section 5 covers the Fortran 90 effort.

Outline
Background: longevity, maintenance & development, reliability, speed
Parallelization: KAI to OpenMP, previous work & current, domain decomposition, done.
Vectorization: Speed - Fed init, vectors in PCs, INL Cray SV1, R5 Phant, EXV, results.
Code Restructuring: Reason to restructure, study of restruct, For Study: what it does,
Fortran 90: Modernization -

1.0 Background

There is a long history of code development for the RELAP5-3D program. Coding from every era of Fortran programming practices from the 1970s through the early 2000s is contained in RELAP5 and its derivative software products. These disparate forms of Fortran from numerous contributors in differing styles have resulted in code issues in the areas of readability, maintainability, and development. There are also issues of longevity, and code run speed that are of concern in general.

Longevity is an important long-term issue. Code life cycle has the design, development, maintenance and retirement stages. RELAP5-3D is still under development, but is simultaneously in the maintenance stage via the process of user trouble reports. Few codes have a pre-planned retirement stage. There are generally two causes for retirement. Either insufficient maintenance allows the state-of-the-art to bypass the program so that it does not run well or at all on modern platforms, or a competitor that is designed to make better use of modern computer technology supplants it.

To ensure that RELAP5-3D remains viable for at least the next decade, it must continue to be upgraded to stay abreast of recent developments in computing hardware, operating systems, and Fortran language changes.

As a succession of ANSI Fortran language standards have been created, RELAP5-3D has been modified to conform to them, and in some ways, to take advantage of them. This must be done for the code to continue to compile under new compilers. The latest ANSI standard, referred to as Fortran 2003, has been released. It has deleted some features that were part of previous Fortran standards. These deletions will have no effect on RELAP5-3D. The new standard also lists obsolescent language features that will be deleted in the next language release. These will pose a very serious issue to the longevity of RELAP5-3D.

The features of older Fortran standards that were deleted in Fortran 2003 are: real and double precision do loop index variables; branching to an ENDIF from outside its structured block; the PAUSE statement; ASSIGN and ASSIGNED GO TO statements; and the H format descriptor. It also lists obsolescent features that are due to be deleted in the next ANSI Fortran language release. These are:

- Alternate RETURN statement
- DATA statements among executable statements
- assumed length character functions
- the CHARACTER* form of CHARACTER declaration statement
- ARITHMETIC IF statement
- shared do termination on any statement other than CONTINUE or END DO
- computed GO TO
- statement functions
- fixed form source

These are listed in the order from least-impact to greatest for RELAP5-3D. In fact, the use of arithmetic if, computed GO TO, and statement functions is widespread throughout RELAP5-3D and the entire code is written in fixed form source.

As new computer architectures are developed and become available, RELAP5-3D has often been modified to take advantage of these. However, the programming that actually does this must often change as the standards for accessing these architectural features develop and become standardized. One example of this is parallel; it was pioneered by CRAY, developed by KAI and others, and has become standardized as OpenMP that is available as a compiler flag on all major vendor platforms. Another is distributed parallel computing; originally each vendor had its own communications library, then libraries that crossed platforms were developed, then ORNL developed PVM for all platforms, then the MPI standard was developed. RELAP5-3D makes use of the PVM communication protocol and must move on to MPI to stay current as PVM approaches end of life cycle.