Fortran 95 Project Summary

George L Mesina
RELAP5-3D Development Team

RELAP5 International Users Seminar
Aug 10-13, 2009
Park City, UT

Outline

- ***Summary of Purpose***
- ***Summary of User Benefits***
- ***Comparisons Pre- and Post-F95***
- ***Timing Measurements***
- ***Recognition***

Purpose / Usefulness of Conversion

- ***Machine independence via F90/95 intrinsics***
- ***Machine independent plot, strip, fluids files***
  - ***Separation of restart from plot files***
  - ***Multiple formats for plot files***
- ***Elimination of many memory restrictions***
- ***Modernize code for longevity considerations***
  - ***Modern language for current-day developers***
  - ***Code easier to read and understand***

Conversion to F95

- ***Database modification***
  - ***Data analyzed & reorganized. Comdeck -> module***
  - ***Module is like Object Oriented Programming Class***
    - ***Data & procedures specific to the data***
    - ***Allocate, eliminate, restart, others***
- ***Source code conversion***
  - ***Convert code to use module data***
  - ***Apply Fortran 95 constructs***
  - ***Streamline – Restructure, rewrite, recombine***

Conversion to F95 – User View

- ***Eliminate memory restrictions***
  - ***Fixed-size** FA container-array -> **scalable** database*
    - ***F95-pointers replace FA-indexing trick***
    - ***Elimination of old memory management library & memory tricks***
    - ***Simplify (rework) loop indexing***
  - ***New database sizes itself to exact amount user needs. (some few exceptions)***

Conversion to F95 – User View

- **Machine Independence**
  - **F95 intrinsic library replaces machine-specific libraries for timers, bit-manipulation, . . .**
  - **Machine independent binary files in the eXternal Data Representation (XDR) format**
  - **XDR *Fluid-property* files on one platform can be ported to and used on another**
  - ***Plot* file can be ported to another platform**
    - **XDR Binary for RELAP-specific plot tools**
    - **ASCII for use with generic plot tools**

Supported Platform Information

- ***Windows XP platforms***
  - ***32-bit installation (32-bit integers, 64-bit floats)***
    - ***Still support 4-byte word chips***
  - ***Intel compiler with 64-bit floating point capability***
- ***Linux Platforms***
  - ***32- and 64-bit platforms***
  - ***Intel compiler***
    - ***Other compilers not fully debugged***
- ***Unix capability still exists***

Comparing Pre-F90 to F95

- ***F95 version 2.9.2***
  - ***Further developments are ongoing***
  - ***The ensuing numbers are preliminary***
- ***Pre-F90 version: 2.4.3 and 2.4.4***
  - ***Version 2.4.3 is the most recent release to IRUG***
  - ***Version 2.4.4 is not in the developmental main line***
    - ***Its use allows apples-to-apples comparison with 2.9.2 on same O/S and compiler***

Comparing Pre- & Post-F95 Code

Category	Pre-F90 **(2.4.3)**	F95 **(2.9.2)**
Platforms	Windows, Unix	Linux, Windows, Unix
Portability	Machine dep. binary	Machine indep. Binary plot, fluid
	O/S specific bit & timer utilities	F95 intrinsic library
Compiler	Many	Intel Fortran (9.1)
Modularity	Unstructured	Strongly Modular – Structured & Modules
Dead Code	Many unused source files	Removed 162 unused files

Testing and Coverage

- *Test Case Coverage*
  - *Coverage provides information on how much application code is exercised during execution.*
  - *Coverage obtained by compiling code with coverage analysis options enabled*
    - *Can analyze single test case or set of input*
  - *Coverage can analyze by number/percentage of source files entered, functions, or code blocks*
    - *Function = subprogram or internal subprogram*
    - *Block = code w/in if branch, or executable stmt*
- *Comparison against 2.4.4 for same O/S and compiler as 2.9.2.*

Testing and Coverage

- ***Conversion #1 goal: Do not change calculations***
  - ***Conversion did not change a single character in the output file <u>between versions</u>***
    - ***Exceptions: bug fixes, developments***
- ***Test set expanded from Pre-F90 to F95 versions***
  - ***Test newly developed capability***
  - ***Expand coverage***
- ***Significant improvements in F95 over Pre-F90***

Testing and Coverage Comparison

Category	*Pre-F90*	*F95*
Test Files	*217*	*240 + 53 (DA*)*
*Developmentally Assessed **	*No*	*Yes*
Source File Coverage		
Relap Directory	*63.87 %*	*80.37 %*
Envrl Directory	*35.46 %*	*54.24 %*
Pvmexec Directory	*57.89 %*	*61.11 %*

- *Not part of F95 Conversion Project*

Testing and Coverage Comparison

Directory	Pre-F90	Post-F95
Function Coverage		
Relap	**62.19 %**	**77.15 %**
Envrl	**40.12 %**	**59.26 %**
Pvmexec	**67.86 %**	**71.56 %**
Block Coverage		
Relap	**44.72 %**	**61.51 %**
Envrl	**38.91 %**	**51.91 %**
Pvmexec	**81.30 %**	**82.91 %**

Testing and Coverage

- ***Testing coverage greater***
  - ***Increase in what test cases cover***
  - ***Removal of dead code (subroutines, functions, blocks of statements)***
- ***Coverage is generally greater than numbers show***
  - ***NONE of the diagnostic coding was accessed***
  - ***User options (card 1) largely untested***
- ***Note that coverage percentages depend on customer class installation option***
- ***You can help coverage. Submit your input files!***

Run Speed

- ***Performance change is mixed. Some problems run slower, others faster. Some are virtually unchanged.***

	Pre F90			*F95*		
Model	*Req Att*	*CPU (sec)*	*CPU / Req Attempt*	*Requested Attempts*	*CPU (sec)*	*CPU / Req Attempt*
ANS79	*828*	*.03*	*3.5e-5*	*828*	*.04*	*.4.8e-5*
ENCLSS	*800*	*.27*	*.3.4e-4*	*800*	*.46*	*.5.8e-5*
FLDRN2	*2000*	*.31*	*1.55e-4*	*2000*	*.36*	*1.8e-4*
HXCO2	*1000*	*1.29*	*.00129*	*1000*	*1.72*	*.00172*
CMT11N	*20001*	*3.97*	*1.98e-4*	*20001*	*5.38*	*2.69e-4*
NEPTUNUS21	*5242*	*1.63*	*3.1e-4*	*5242*	*2.32*	*4.43e-4*

Run Speed

- *These are faster in F95*

	Pre F90			F95		
Model	Req Att	CPU (sec)	CPU / Req Attempt	Requested Attempts	CPU (sec)	CPU / Req Attempt
CSTEST2	4506	.05	1.1e-5	4419	.02	4.5E-6
HEX2DK	20	1.39	.069	20	1.32	.066
RTSAMPN	12	.80	.067	12	.79	.0658
TYPPWRR2	12	.35	.029	12	.34	.028
T0301	2500	5.85	.0023	2500	4.95	.00198
TANK	2301	44.76	.0195	2301	36.17	.0157

Run Speed

- *Representative Large Problems*
  - *AP600: over 600 volumes and 1000 junctions*
- *F95 version is comparable or faster*

Model	Att 244	CPU (sec)	CPU / Attempt	Attempts 292	CPU (sec)	CPU / Attempt
AP600PMPS	139	12.26	.0882	139	12.46	.0896
AP600PWRS	455	34.28	.0753	519	33.36	.0643
AP600SBS	419	31.74	0.758	419	28.08	.0670

Speed Discussion

- ***The nearly implicit are generally slower.***
- ***Larger problems are generally faster.***
- ***A short study of one problem (TYP1200) showed***
  - ***Some subroutines were much slower (up to a factor of 5 times)***
  - ***Some subroutines were faster (up to a factor of 2)***
- ***The use of pointers may contribute to slow-down***
- ***The rewriting of some subroutines may have sped up***
- ***No compiler flag optimization done***

Thanks to Contributors

The following have spent several months on the project each:

- *Dr. Richard Riemke: Heat and reflood, robustness*
- *Dr. Walter Weaver: Kinetics & PVM*
- *Nolan Anderson: Scratch, fluids, robustness*
- *Richard Wagner: R-, I-level and Heat*
- *Peter Cebull: Fluid properties*
- *Hope Forsmann: Machine Independent Plot*
- *Summer students: Restructuring, Environmental*

Future Work

- ***Increase coverage***
  - ***You too can contribute to this worthy cause***
- ***Study speed issues***
  - ***Find the actual cause of slow downs and correct***