Overview of MELCOR Code Development

Larry L. Humphries, Randall Gauntt (SNL)
and
Hossein Esmaili (NRC)

Second European MELCOR User Group
Prague, Czech Republic
March 1-2, 2010
MELCOR Code Development

• MELCOR is developed by:
  – US Nuclear Regulatory Commission
  – Office of Nuclear Regulatory Research
  – Division of Safety Analysis (DSA)

• MELCOR development is also strongly influenced by the participation of many International Partners through the US NRC Cooperative Severe Accident Research Program (CSARP and MCAP)
  – Development Contributions – New models
  – Development Recommendations
  – Validation
Current MELCOR Development

**MELCOR 1.8.6 Branch**

- MELCOR 1.8.6
  - Molten pool models
  - Core Package upgrade
  - Released Fall 2005
  - Code Maintenance Only (no new modeling)
  - Current Workhorse
  - Maintenance release (Jan 2010)

**Current Version MELCOR 2.1 Branch**

- Source code in FORTRAN 95
- New input format
- All new development
- 2.0 beta version released Sept 2006
- 2.1 Release Sept 2008
- 2.1 Maintenance release (Jan 2010)
MELCOR 3.0 Code Development Thrust Areas

**Code reliability**
- Validation
- QA
- Numerical stability

**Code Enhancements**
- New/improved modeling
- Code performance

**User Utilities**
- Converter
- PTFREAD
- SNAP
- Uncertainty Engine

2nd European MELCOR User Group
Prague, Czech Republic  March 1-2, 2010
Code Development: Code Reliability

Code Reliability

MELCOR

User Utilities

Code Enhancements
Intel Visual FORTRAN Compiler

• Intel Visual FORTRAN has replaced Compaq Visual FORTRAN as our standard development platform
  – Did not want to make any changes until after 2.1 was released
  – Technical support for CVF no longer available
  – Problems with CVF rebuilding entire project
  – Problems with error checking

• Advantages to Intel Compiler
  – Able to build true 64-bit code for 64-bit operating systems
    • Performance improvements
  – Better support for F95 code
    • Improved error checking
  – More consistent support for Linux
  – Improvements to the programmer interface (Visual Studio.NET)
    • Automatic keyword completion
    • Integration with subversion
  – Capability to automatically convert CVF projects
Software Quality Assurance

• Sandia Corporate Process Requirement 001.3.6 (CPR 001.3.6)

• The software management framework adapted from two internationally recognized standards
  – the Capability Maturity Model Integration (CMMI) ®
  – and ISO 9001
  – These standards provide elements of traceability, repeatability, visibility, accountability, roles and responsibilities, and objective evaluation

• Areas showing improvement
  – Code Review (Code Collaborator)
  – Requirements Management
  – Configuration Management (Automated Build & Test)
  – Further Development of Internal Wiki
  – Development of SQA training materials

• Areas needing improvement
  – Improve debugging response time
  – Improve user access to documentation updates
  – Explore possibility of external Users’ Wiki

• Sandia’s commitment to SQA
Software Quality Assurance
Annual Re-evaluation

- Process areas
  - Project planning and oversight, PPO
  - Risk Management, RSK
  - Requirements Development and Management, RDM
  - Technical Solution, TS
  - Verification and Validation, VV
  - Development and Lifecycle Support, DLS
  - Configuration Management, CM
  - Measurement and Analysis, MSA
  - Integrated Product, IPD
  - Integrated Teaming, ITM

- Process Dimensions
  - Stakeholder Involvement, SI
  - Ongoing Process Monitoring and Control, PMC
  - Collected Improvement Data, CD
  - Objective Evaluations, OEV
  - Quantitative Objectives Defined for Processes, QPO
  - Stable Subprocess Performance, SSP
  - Training, TR
  - Problem Reporting & Corrective Action, RCA
<table>
<thead>
<tr>
<th>MELCOR 1.8.6 (since 7/07)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Bugs Reported</td>
<td>105</td>
</tr>
<tr>
<td>Number of Bugs Resolved:</td>
<td>72</td>
</tr>
<tr>
<td>Number of Bugs Unresolved:</td>
<td>33</td>
</tr>
<tr>
<td>Number of Bugs Reported (this year):</td>
<td>36</td>
</tr>
<tr>
<td>Number of Bugs Resolved (this year):</td>
<td>27</td>
</tr>
<tr>
<td>Number of Bugs Unresolved (this year):</td>
<td>9</td>
</tr>
<tr>
<td>Number of Unresolved User Bugs:</td>
<td>15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MELCOR 2.1 (since 9/07)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Bugs Reported</td>
<td>138</td>
</tr>
<tr>
<td>Number of Bugs Resolved:</td>
<td>98</td>
</tr>
<tr>
<td>Number of Bugs Unresolved:</td>
<td>40</td>
</tr>
<tr>
<td>Number of Bugs Reported (this year):</td>
<td>84</td>
</tr>
<tr>
<td>Number of Bugs Resolved (this year):</td>
<td>77</td>
</tr>
<tr>
<td>Number of Bugs Unresolved (this year):</td>
<td>7</td>
</tr>
<tr>
<td>Number of Unresolved User Bugs:</td>
<td>15</td>
</tr>
</tbody>
</table>
MELCOR Code Testing

- **Build Testing**
  - Automated to perform Nightly Builds
    - MELCOR 1.8.6 Windows Compaq Visual FORTRAN (CVF)
    - MELCOR 2.X Windows CVF
    - MELCOR 2.X Windows Intel Visual FORTRAN (IVF)
    - MELCOR 1.8.6 Linux IVF
    - MELCOR 2.X Linux IVF
  - Using CMAKE to generate Makefiles for use on Unix variants to extend building on other platforms

- **Code Testing**
  - Performed Daily (at least frequently)
    - Dedicated machine for performing auto testing to
  - Standard test cases chosen for physics coverage ~14 test cases
    - New cases will be added as validation calculations are run
    - Debug & optimized versions tested
    - Unix versions not tested as frequently (will test more frequently in future)
  - Comparison of results
    - Windows system
      - Latest 1.8.6 (CVF) and YT, 2.X(CVF)
      - Latest 2.X (CVF) and Latest 1.8.6 (CVF)
      - Latest 2.X (IVF) and latest 2.X (CVF)
    - Linux System (not yet realized)
      - Latest 2.X (IVF Windows) Latest 2.x (IVF Linux)

- **Optimization Testing (not yet realized)**
  - Run automated optimization studies to determine effects of optimization
    - Many routines can be optimized with no impact on answers
    - Order of arithmetic will mean some routines should not be optimized

<table>
<thead>
<tr>
<th>Case</th>
<th>BUR</th>
<th>CAV</th>
<th>CF</th>
<th>COR</th>
<th>CVH</th>
<th>DCH</th>
<th>FCL</th>
<th>FDI</th>
<th>FL</th>
<th>HS</th>
<th>NCG</th>
<th>PAR</th>
<th>RN</th>
<th>SPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-1-1 NoMix</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-1-1 SYM</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Loss</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Loss-1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vessel-M3</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Molten Salt</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHEBUS-B9</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FPT1</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOFT</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test loop</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SURRY</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(LBLOCA)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zee (SBO)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PunchBottom (SBO)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grand Gulf (SBO)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1-1: Physics Packages Coverage
# Automated Nightly Build and Test

## MELCOR Build and Test Report

### Current Status: 9/5/2009

<table>
<thead>
<tr>
<th>Revision</th>
<th>Submitter</th>
<th>Date</th>
<th>Time</th>
<th>Compiler</th>
<th>Build Status</th>
<th>Test Status</th>
<th>Compiler2</th>
<th>Build Status</th>
<th>Test Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>124</td>
<td>technical</td>
<td>8:25:44 AM</td>
<td>IVF 11.0/072</td>
<td>Failed</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Failed</td>
<td>Unk</td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>defect</td>
<td>1:15:16 PM</td>
<td>IVF 11.0/072</td>
<td>Failed</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Failed</td>
<td>Unk</td>
<td></td>
</tr>
<tr>
<td>1241</td>
<td>joonyu</td>
<td>9/2/2009</td>
<td>4:08:07 PM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1239</td>
<td>jroyno</td>
<td>9/2/2009</td>
<td>2:02:46 PM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1238</td>
<td>jroyno</td>
<td>9/2/2009</td>
<td>1:14:02 PM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1236</td>
<td>bhumph</td>
<td>9/1/2009</td>
<td>3:41:57 PM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1235</td>
<td>bhumph</td>
<td>9/1/2009</td>
<td>1:59:16 PM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1234</td>
<td>bhumph</td>
<td>9/1/2009</td>
<td>12:56:54 PM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1233</td>
<td>bhumph</td>
<td>9/1/2009</td>
<td>8:00:10 AM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1232</td>
<td>bhumph</td>
<td>9/1/2009</td>
<td>7:59:36 AM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1231</td>
<td>bhumph</td>
<td>9/1/2009</td>
<td>7:13:29 AM</td>
<td>IVF 11.0/072</td>
<td>Successful</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1229</td>
<td>nastya</td>
<td>9/1/2009</td>
<td>5:33:15 AM</td>
<td>IVF 11.0/072</td>
<td>Failed</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1228</td>
<td>nastya</td>
<td>9/1/2009</td>
<td>5:31:39 AM</td>
<td>IVF 11.0/072</td>
<td>Failed</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
<tr>
<td>1226</td>
<td>nastya</td>
<td>9/1/2009</td>
<td>1:54:27 AM</td>
<td>IVF 11.0/072</td>
<td>Failed</td>
<td>Unk</td>
<td>CVF 6.6C</td>
<td>Successful</td>
<td></td>
</tr>
</tbody>
</table>
MELCOR Regression Report Page

- Automated Regression Report Web page
  - Links to regression report with side-by-side comparisons of trend plots
  - Table of variables sorted by variance

MELCOR Regression Report Page

MELCOR 2.1 (IVF Compiler) Regression Results

Regression Reports

Following is a list of regression reports produced using the MELCOR 2.1 (IVF Compiler). The regression reports compare the MELCOR 2.1 (IVF Compiler) regression results with the regression results from the most recent regression test case.

- Automated for MELCOR 2.1 (IVF Compiler)
- Automated for MELCOR 2.1 (IVF Compiler)
- Automated for MELCOR 2.1 (IVF Compiler)
- Automated for MELCOR 2.1 (IVF Compiler)
- Automated for MELCOR 2.1 (IVF Compiler)

Current Regression Test Summary (IVF 1093, CVF 1093)

The following table summarizes regression results by regression type and regression test case. The table shows the number of variables that have been evaluated, the number of variables that have been evaluated with significance, and the number of significant results.

- Variables are sorted according to the magnitude of the regression test results.
- All values are rounded to three significant figures.
- Significant results are indicated by an asterisk (*).

<table>
<thead>
<tr>
<th>Test Case</th>
<th>&lt;5%</th>
<th>1% to 5%</th>
<th>5% to 10%</th>
<th>&gt;10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression Test Summary Table</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Case</td>
<td>&lt;5%</td>
<td>1% to 5%</td>
<td>5% to 10%</td>
<td>&gt;10%</td>
</tr>
<tr>
<td>Regression Test Summary Table</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This report compares results from the MELCOR regression tests for MELCOR 2.1 (IVF Compiler) against results obtained from tests for MELCOR 2.1 (IVF Compiler) using the same IVF Compiler version. The results are presented in a table format, with each test case listed in the left column and the regression results in the right column. The table shows the number of variables that have been evaluated with significance, and the number of significant results.
Changes to Documentation Published With Released Code Version

• Word documents stored locally in Subversion Repository
  - Documents available to all developers for modification
  - Word compare is readily used to ‘diff’ two revisions of a document

• PDF Generated from Working versions of Word Documents
  - Marked as draft version and revision numbers clearly marked in document.
  - SVN revision log is inserted after title page.
  - All modified content is highlighted in yellow
  - Style & formatting changes ignored.
  - All user font coloring is disabled.
Extended Code Pedigree Information

• MELCOR has always provided limited pedigree information
  – Available in output and on console
  
  MELCOR    BASE CODE VERSION
  2.1 DEC-16-2009
  DEVELOPMENT VERSION
  $Revision: 1660 $  

• MELCOR 2.1 now provides extended information
  – Includes compiler version, libraries used, compiler settings, and linker settings
  – Lists any files that were not optimized
  – Printed in output and also available from executable using command line
    • -CONFIG (command line argument)

MELCOR    BASE CODE VERSION
2.1 DEC-10-2009

DEVELOPMENT VERSION
BASE CODE YU $Revision: 1490 $  

Configuration information for MELCOR
Subversion revision: 1495
Compiler: IVF 11.1\051
Compiler settings
/nologo /O2 /Oy- /fpp /TC:\Program Files (x86)\Intel\Compiler\11.1\051\fortran
\include\ia32' /DLICENSING_INACTIVE /DWINDOWS
/recursion /real_size:64 /Qauto /align:rec8byte /Qtrapuv /iface:cfv
/module:/' /traceback /fp:source /Qprec-div /Qprec-sqrt
/assume:protect-parens /check:none /lib:static /threads /dbglibs /c

Linker settings:
Version:2.1 /INCREMENTAL:NO /MANIFEST /NOLOGO
/SUBSYSTEM:CONSOLE /TSAWARE:NO\ kernel32.lib\ Ws2_32.lib\ user32.lib\ mkl_solverquential.lib\ mkl_intel_c.lib\ mkl_sequential.lib\ mkl_core.lib

Files that were not optimized:
CORFZO, CORFZP, MXXDBD, POOL3_NSI, RES_NSI, CAVENR_NSI,

!!!LICENSING HAS BEEN DISABLED!!!
Continuous Assessments

- Continuous reassessment of many ISPs to assure consistency between code versions and effect of code improvements
  - LOFT
  - Cora-13
  - Phebus-B9
  - Lace
  - Vanam
  - SURC-2
  - DEHBI
  - NUPEC (M-8-1 & M-8-2)
  - Falcon (FAL-ISP 1)
  - LOFT (LP-FP-2)

- Additional assessments to be performed.
  - Intel allows us to test code coverage
  - Traceability matrix to enhance our code coverage
• All new code development will be performed in MELCOR 2.1
• MELCOR 1.8.6 – Code Maintenance only
Code Development
HTGR - COR Package Updates

- New reactor models (like present PWR, BWR)
  - PBR (pebble bed)
    - New components
      - Pebble fuel
    - New heat transfer coefficients
    - Effective thermal conductivity
      - radiation/convection/conduction
    - Coolant friction loss
  - PBR (prismatic)
    - New components
      - Reflectors
      - Hexagonal graphite blocks
    - cell-cell conductive/radiative heat transfer
- Graphite oxidation models
- Point Kinetics
- On-going work
  - Fission product release and transport from HTGR fuel
  - Plant demonstration calculations
PBR COR Model

- Pebble represented using fuel and clad components
  - Fuel radial temperature profile for sphere
    - Provides peak and surface pebble temperature
  - Fueled part of pebble (*Coated particles embedded in Graphite Matrix*) is fuel component
    - $\text{UO}_2$ with extra COR material as graphite
  - Unfueled shell (*5mm Graphite layer*) is clad component
    - Clad material becomes graphite
- Example Input:
  - ![graphite pebble](image)
  - ! RFUEL RCLAD DRGAP PITCH
  - COR_GP 0.025 0.03 0.0 0.0143
Radial cell-cell conductive/radiative heat transfer added
  - Uses effective bed conductivity (Zehner-Schlunder with thermal radiation)
  - New input record: COR_TKE (optional for PBR)
Axial cell-cell conduction modified to use effective bed conductivity
Coolant friction factor is for pebble bed with PBR model
  - Ergun correlation is default
  - KTA Rule (Achenbach) correlation can be used by changing SC1413
  - Pebble bed friction factor specified with options PBR-A or PBR-R on COR_BLK record
Coolant heat transfer uses pebble bed heat transfer coefficients
  - KTA Rule correlation can be entered with SC1231
PMR COR Models

- More “rod-like”, requires fewer changes to COR
- Fuel compacts represented as fuel component (blue)
- Part of hex block associated with a fuel channel is “thick” clad component (red)
  - Temperature gradient in clad as well as fuel
  - Important change is inclusion of “clad” resistance in HTC
- Can have significant effect on heat transfer to coolant
- Reflectors are new Reflector component
PMR Models

- Radial cell-cell conductive/radiative heat transfer used, specified on COR_TKE record
  - TKECF – optional temperature-dep TF, usually ‘NO’
  - PORCHAN – effective block porosity for Tanaka-Chisaka conductivity model (see COR RM)
    - “Porosity” is something like (coolant channel volume per block)/(block volume)
    - Conductivity in “porosity” is coolant conductivity
  - DBLK – block characteristic length
  - BLKGAP – gap width between blocks
    - Conductance across gap using coolant conductivity and radiation
Counter-Current Stratified Flow Modeling

- Counter-Current Stratified Flow Applications
  - Developed for HTGR air-ingress application
  - Modeling of natural circulation between the steam generator and the upper plenum of PWRs
  - Containment modeling
- Old concept used in MELCOR for many years
  - Split path in two to allow counter-current flow
  - Account for momentum exchange in flow equations
    - Implement as “pumps” with $\Delta P$ calculated from relative velocities
      \[ \Delta P = -\frac{2fL}{D_h} \rho (v_1 - v_2) |v_1 - v_2| = -C\rho (v_1 - v_2) |v_1 - v_2| \]
- New approach where $\Delta P$ is “internalized”
  - We have coded an appropriate $\Delta P$ calculation
    - Specific input was added to couple two flow paths
    - Form of $\Delta P$ has been generalized to better match published correlations
    - Terms added directly to flow equations
      - Increased stability because of implicit numerics
Ongoing or Recently Completed Code Model Development

- **Accumulator Model**
  - Replaces current control function approach to improve numerics and reduce errors and modeling effort of code user

- **Use a model to track activities.**
  - Existing MELCOR output doesn’t allow tracking of radionuclides and user has to perform calculations to estimate activities.

- **Heat and Mass Transfer Correlations**
  - MELCOR code should be able to model the CONTAIN correlations by default
  - Modify the MELCOR film tracking model and default model parameters based on the CONTAIN parity

- **Engineered Safety Features (ESF) Enhancements**
  - heat exchanger models
  - fan cooler models

- **Improvement of SPARC Models**
  - Review the SPARC98 model for possible improvements over the earlier SPARC90 model

- **Improvement of VANESA Models**
  - improvements for ex-vessel fission product release. Specifically, the modeling of Ru and Mo)releases

- **Others…**
Code Development: User Utilities
## Improvements to Converted Input

### September 2008

- **HS_ID** 'HS31014-UP' 31014  ! 186 name: UP
- ...
- '!* 103 - Next HS data

- **HS_ID** 'HS32001-LOWERPLE' 32001  ! 186 name: LOWERPLENUM
- **HS_GD** CYLINDRICAL YES  ! Type of geometry, Steady-state initialization
- **HS_EOD** 0.1445 1.0  ! HS Elevation and Orientation Data
- **HS_SRC** NO  ! No power source
- **HS_ND** 4 ! NXVALU NI XVALUE TEMPIN MATNAM
  1 1 2.2403                    - 'STAINLESS-STEEL'
  2 2 2.2874                    - 'STAINLESS-STEEL'
  3 3 2.3345                    - 'STAINLESS-STEEL'
  4 4 2.3816                    -
- **HS_LB** CalcCoefHS 'CV320-LOWER-PLEN' YES
- **HS_LBP** INT 5.0E-01 5.0E-01
- **HS_LBT** 100.0 100.0 1.0 1.0
- **HS_LBS** 1.0 0.701 0.5230
- **HS_RB** CoefTempTF 'TF10-PIPE INSULA' 'CV1-CAVITY' YES
- **HS_RBP** EXT 5.0E-01 5.0E-01
- **HS_RBT** 100.0 100.0 1.0 1.0
- **HS_RBS** 7.8262 1.0 0.5230
- **HS_FT** OFF
- ...
- **HS_ID** 'HS32002-LOWERPLE' 32002  ! 186 name: LOWERPLENUM

### Comments from original input

- Lower Vessel Wall - Core level 1
- Description from S/R5 Model HS 5062
- Inner Radius = 2.2403 m
- Outer Radius = 2.3816 m
- Inner Dhyd = 0.701 m
- DZ = 0.523

### More readable formatting

- **NODES CYL INPUT W4(I)**
- **HS_ID** 'HS32001-LOWERPLE' 32001  ! 186 name: LOWERPLENUM
- **HS_GD** CYLINDRICAL SS  ! Type of geometry, Steady-state initialization
- **HS_EOD** 0.1445 1.0  ! HS Elevation and Orientation Data
- **HS_SRC** NO  ! No power source
- **HS_ND** 4 ! NXVALU NI XVALUE TEMPIN MATNAM
  1 1 2.2403                    - 'STAINLESS-STEEL'
  2 2 2.2874                    - 'STAINLESS-STEEL'
  3 3 2.3345                    - 'STAINLESS-STEEL'
  4 4 2.3816                    -
- **HS_LB** CalcCoefHS 'CV320-LOWER-PLEN' YES
- **HS_LBP** INT 5.0E-01 5.0E-01
- **HS_LBT** 100.0 100.0 1.0 1.0
- **HS_LBS** 1.0 0.701 0.5230
- **HS_RB** CoefTempTF 'TF10-PIPE INSULA' 'CV1-CAVITY' YES
- **HS_RBP** EXT 5.0E-01 5.0E-01
- **HS_RBT** 100.0 100.0 1.0 1.0
- **HS_RBS** 7.8262 1.0 0.5230
- **HS_FT** OFF

### Steady State temperatures to be calculated

- Description from S/R5 Model HS 5062
- Inner Radius = 2.2403 m
- Outer Radius = 2.3816 m
- Inner Dhyd = 0.701 m
- DZ = 0.523

### Records positioned as in original input

- Lower Vessel Wall - Core level 1
- Description from S/R5 Model HS 5062
- Inner Radius = 2.2403 m
- Outer Radius = 2.3816 m
- Inner Dhyd = 0.701 m
- DZ = 0.523
• Input files can be selected from multiple methods
  – Directly typed or pasted
  – Navigation through file manager
  – ‘Drag and Drop’
  – Selected from recent history
• Executables can be selected in a similar manner
• Execution of MELGEN, MELCOR, or the Converter from one pane
• MELGEN/MELCOR input/output files can be accessed from hyperlinks
New MELCOR Output Format

HTML

- HTML Time Edits
  - Specified with global input
    - MEL_HTMLFILE ‘DEMON_Out.htm’
  - File for each time edit
  - Links to other time edits
  - Links to package edits/tables

- Other Links
  - SNL/Bugzilla
  - I/O files
  - Code Manuals
  - Graphical Diagrams
    - Node Diagram
    - Temperature contours
Named Comment Blocks (1)

• This feature was added to allow the user to include blocks of input records that can be ‘activated’ or skipped by a variety of input methods.

• User can encapsulate multiple versions of a test case within a single input deck.
  – Sensitivity calculations
  – Variations for a standard plant deck

• Analogous to the *EOR* capability for MELCOR 1.8.6
  – More flexible and powerful to use
MELCOR comment blocks were added as part of the code conversion project. In MELCOR 2.1, a block of input can be commented out by enclosing that block of input within a set of triple parenthesis.

Named comment blocks are either ignored or processed depending on user input.

- The example to the right demonstrates the input instructing MELGEN to read a block of records required for a SBO calculation.
  - Only the SBO comment block is read and the LBLOCA block is ignored.
  - Alternatively, if the second field on the CommentBlock record were LBLOCA, the LBLOCA records would be read.
- The CommentBlock record can also contain more than one field so that multiple comment blocks can be read, i.e.,
  - CommentBlock SBO LBLOCA.
Named Comment Blocks (3)

• There can be multiple comment blocks with the same name
  – Three comment blocks are given the name ‘SBO’ at right.
  – useful if you have many test case options with variations in multiple locations in the input deck

• Furthermore, the named comment block construct allows multiple names or pseudonyms for a comment block.
  – This would allow some commonalities between a subset of your multiple test options.

• Named Comment blocks can also be nested
  – Comment block ‘Nested’ can only be processed when both ‘LBLOCA’ and ‘Nested’ fields are present on the CommentBlock record.

CommentBlock SBO
...
PROGRAM MELGEN
...
(((SBO !Additional comments placed here
...These input records are not ignored
))))
...
(((LBLOCA !This is a large break LOCA scenario
...These input records are ignored
(((Nested !This is an example of a nested block
))))
))))
(((SBO
...These input records are not ignored
))))
(((SBO+LBLOCA
...These input records are not ignored for either SBO or LBLOCA cases
))))
Named Comment Blocks (4)

- Comment Blocks can be ‘activated’ through the following input:
  - CommentBlock Record
    
    ```
    CommentBlock case1 case2 case3
    ```
  - Command line arguments
    
    ```
    melgen i=inputfile c=case1+case2+case3
    ```
  - Interactive Method
    - Comments are echoed with comment block names
    - Multiple blocks can be specified by delimiting names with ‘+’
    - DefaultNamedCommentBlock record
      
      - Placed in the global variable section
      - When present, if the user does not select any comment blocks, those listed on the DefaultNamedCommentBlock record are used

- DefaultNamedCommentBlock 2HSNODE+BIGDT
Variable Input Fields (1)

• This feature allows the user to insert variable input fields into records which can then be updated through user input at the time of execution.

• This can be useful for testing ranges of input fields, managing variant assessment calculations, and could be used to aid interfacing MELCOR to other applications.

• Users can provide variable values through a variety of methods.
Variable Input Fields (2)

- Input fields are tagged so that MELCOR will preprocess the input file, replacing the tagged variables with a value specified by the user.
- These tagged variables can occur multiple places in the input deck and all instances are replaced by the same value.
- The tagged placeholder on an input record includes the variable name and the default value enclosed in a set of triple curly parenthesis, {{{}}}, i.e.,

<table>
<thead>
<tr>
<th>FL_SEG 1 !</th>
<th>SAREA</th>
<th>SLEN</th>
<th>SHYD</th>
<th>SRGH</th>
<th>LAMFLG</th>
<th>SLAM/CFNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{{{SAREA=1.0}}}</td>
<td>{{{SLEN=0.2}}}</td>
<td>0.24000E-01</td>
<td>0.5E-04</td>
<td>CONST</td>
<td>0.00</td>
</tr>
</tbody>
</table>

- In this example, two variables, SAREA and SLEN, are specified on the flow segment record. Default values are 1.0 and 0.2 respectively are provided.
Variable Input Fields (3)

- **Input Values for the Variable fields can be provided as follows:**
  - **Variable Input File**
    - A file can be specified in the global specifications, containing values for all or some of the variables found in the input deck.

    ```
    DefineVariablesFile Variables.dat !Values read from Variables.dat file
    ```
    - If some variable values are missing from this file, the user will be asked interactively to provide values.
    - If this file is missing, the user will be asked to interactively provide values.

    ```
    *{{{Var1=1.0}}:* !Additional comment
    *{{{Var2='Wetwell'}}:* !Additional comment
    *{{{Var3=3}}:* !Additional comment
    ```
  - **VariableValue record**
    - Values are provided on the VariableValue record located in the global variable section of the input.
    - Any data type can be provided but must be consistent with the requirements of the associated input record to avoid MELGEN errors.
    - The supplied value effectively replaces the {{{var=nnn}}} variable field.

  ```
  VariableValue
  {{{Var1=1.00}}} {{{Var2='Wetwell'}}} {{{Var3=3}}} !These 3 variables are evaluated
  ```
  - **Interactive Method**
    - A list of all variables is provided and the user is queried for the required value.
    - If the user does not specify a value, then the default value will be used unless it was specified by any of the previous methods.
Capability to Print Default SC Values

• **PrintDefaultSC** – Print Sensitivity Coefficient Default Values
  - Optional
  - Default values for all sensitivity coefficients in active packages are printed to the file named on this record. The default values printed are those defaults _after_ all global or package default records are processed, i.e., EXEC_GLOBAL_DFT, COR_DFT, etc. This report is printed during MELGEN and does not include user specification from XXX_SC input records. When this record is absent, no report is printed.

• **PrintCurrentSC** – Print Sensitivity Coefficient Default Values
  - Optional
  - Current values for all sensitivity coefficients in active packages are printed to the file named on this record. This report is printed just after the MELCOR input is read and includes all user specifications from XXX_SC input records after and any default set specifications (EXEC_GLOBAL_DFT, COR_DFT, etc.). When this record is absent, no report is printed.
Other Improved User Capabilities

♦ Smart Restart Capability
  ♦ Allow user to change CF and TF parameters from the restart without adding new control or tabular functions.

♦ Formula Control Functions
  ♦ Allow user to change CF and TF parameters from the restart without adding new control or tabular functions.

♦ ! ************** Formula version ***************
  ♦ CF_ID ‘sxjg’ 801 Formula
  ♦ CF_FORMULA 3 SQRT(MAX(ATM_FR*ATM_VEL,C0))
  ♦ ! ArgName Value
  ♦ 1 ATM_FR FL-VOID(‘flood’)
  ♦ 2 ATM_VEL FL-VEL(‘flood’,ATM)
  ♦ 3 C0 0.0 ! bound positive

♦ ! ************** Classic version ***************
  ♦ CF_ID ‘JG’ 501 MULTIPLY ! JG = atm_fr*atm_vel
  ♦ CF_SAI 1.0 0.0
  ♦ CF_ARG 2
  ♦ 1 FL-VOID(‘flood’) 1.0 0.0
  ♦ 2 FL-VEL(‘flood’,ATM) 1.0 0.0
  ♦ CF_ID ‘XJG’ 601 EQUALS ! XJG = MAX(JG,0)
  ♦ CF_SAI 1.0 0.0
  ♦ CF_ULB LW 0.0 ! Impose lower bound of 0.0
  ♦ CF_ARG 1
  ♦ 1 CF-VALU(‘JG’) 1.0 0.0
  ♦ CF_ID ‘XJG’ 701 SQRT ! SXJG = SQRT(MAX(JG,0))
  ♦ CF_SAI 1.0 0.0 !
  ♦ CF_ARG 1
  ♦ 1 CF-VALU(‘XJG’) 1.0 0.0
Supporting Applications

- **SNAP**
  - Symbolic Nuclear Analysis Package developed by API – MELCOR Plug-in

- **PTFREAD**
  - EXCEL add-in for generating plots, analyzing data, creating AVI’s, generation of regression reports

- **MELCOR 2.1 GUI & Converter**
  - Utility for generating MELCOR 2.1 input decks and converting existing MELCOR 1.8.6 decks to new format

- **Uncertainty Software**
  - Suite of tools for running MELCOR in batch, Monte Carlo sampling of variables and analyzing statistics

- **Best Estimate with Uncertainty Quantification**
- **Powerful tool for risk-informing regulations**

![Graph](image-url)
SNAP Development

- **MELCOR 2.1 Plugin**
  - Version 1.0.0 - Released 7/17/09
  - Will convert a 1.8.6 input deck to 2.x
  - Sandia is working with SNAP developers to recommend enhancements for MELCOR plug-in
- **A future workshop will focus on the use of SNAP**
- **Model Editor -Components**
  - Tree Structure organization
  - Arranged according to MELCOR package
  - ASCII view of object available
  - DIFF capability for components
- **Object Properties Window**
  - User Aids
    - User Guide Information
    - Field units
  - Tabular input
    - Can paste data from spreadsheet
  - Views
    - Trend plots
    - Custom animations
    - Others

2nd European MELCOR User Group
Prague, Czech Republic March 1-2, 2010
Code Performance Improvement Work(1)

- **Code Optimization**
  - Goal is to achieve maximum optimization of code, maintaining reproducibility of results, and consistency with debug code.
- **New capability to test code optimization on a subroutine level**
  - A number of compiler settings must be modified to achieve parity (/fp:source /Qprec-div /Qprec-sqrt /assume:protect-parens)
  - Only six routines remain un-optimized to give parity
    - "CORFZO", "CORFZP", "MXXDBD", "POOL3", "RES", "CAVENR"
  - Often code that does not optimize well has bugs.
    - Review existing routines and test future code for optimization issues.
- **Performance gains dependent on problem**
  - **BWR Plant decks**
    - LBLOCA
      - GrandGulf LBLOCA (1.43)
    - SBO
      - GrandGulfSBO (1.56), Peach Bottom (1.24)
  - **PWR Plant decks**
    - LBLOCA
      - Surry LBLOCA (1.55 – 1.78)
    - SBO
      - Zion SBO (1.82)
  - **Test Decks**
    - Quench 6 (2.2), CORA-13 (1.4), FALCON (1.85), LOFT (2.1), NUPEC (1.57), VANAM (1.7), LACE 7 (1.33)
Code Performance Improvement Work (2)

- **Solver Improvements**
  - The sparse matrix storage format changed to CSR format
    - A standardized format that will allow testing multiple solvers
    - Pardiso solver from Intel MKL has been tested

- **Parallelization (OpenMP)**
  - Currently focusing on CVH package
  - Restructuring of routines –
    - Eliminating of all the internal subroutines by removing them.
    - Treatment of shared data access
      - locally used variables removed from modules and passed through argument lists
      - Special treatment of accumulative variables
  - Implementation of OpenMP commands
  - Development of performance metrics
Code Performance Improvement Work (3)

• Developed as part of HTGR model development
  – Steady state initialization for HTGR may require lengthy computational time
    • Large heat capacity of core components
    • HTGR initialization can take 4 weeks CPU time without acceleration
    • Initialization reduced to less than a day
  – Works by reducing the heat capacity of solid HS and COR components
  – Care must be taken to ensure no melting of components or structures
  – User activates/de-activates model from input

<table>
<thead>
<tr>
<th>TstartSS</th>
<th>TendSS</th>
<th>Rfactor</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXEC_SS</td>
<td>-6000.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>0.0</td>
<td>0.01</td>
</tr>
</tbody>
</table>

TstartSS is the time to start model,
TendSS time to end model,
Rfactor is the CP reduction factor.

• Application to water reactors
Critical Flow Modeling

- Current Modeling
  - Hydro packages solve the flow equations without consideration of the critical flow limit.
  - Once the overall flow solution has converged, each flow is compared to the critical value for the flow path (from the correlations).
  - If one or more flows exceed the critical value, the entire equation set is re-solved, with those flows imposed.

- Improved Modeling
  - Include imposition of critical within the initial solution.
  - MELCOR hydrodynamics treats a flow resistance that is a non-linear in velocity by iterating until each calculated flow is in good agreement with the velocity that was used to calculate the flow resistance.
    - Choked flow can be captured by making the flow resistance very large for flows greater than critical.
Overview of MELCOR Code Development

Summary

Code reliability
- Validation
- QA
- Numerical stability

User Utilities
- Converter Improvements
- HTML Output
- Etc.

MELCOR Code Development

Code Enhancements
- New/improved modeling
- Code performance