CIME Case Control System in UFS MRW App v1.1.0

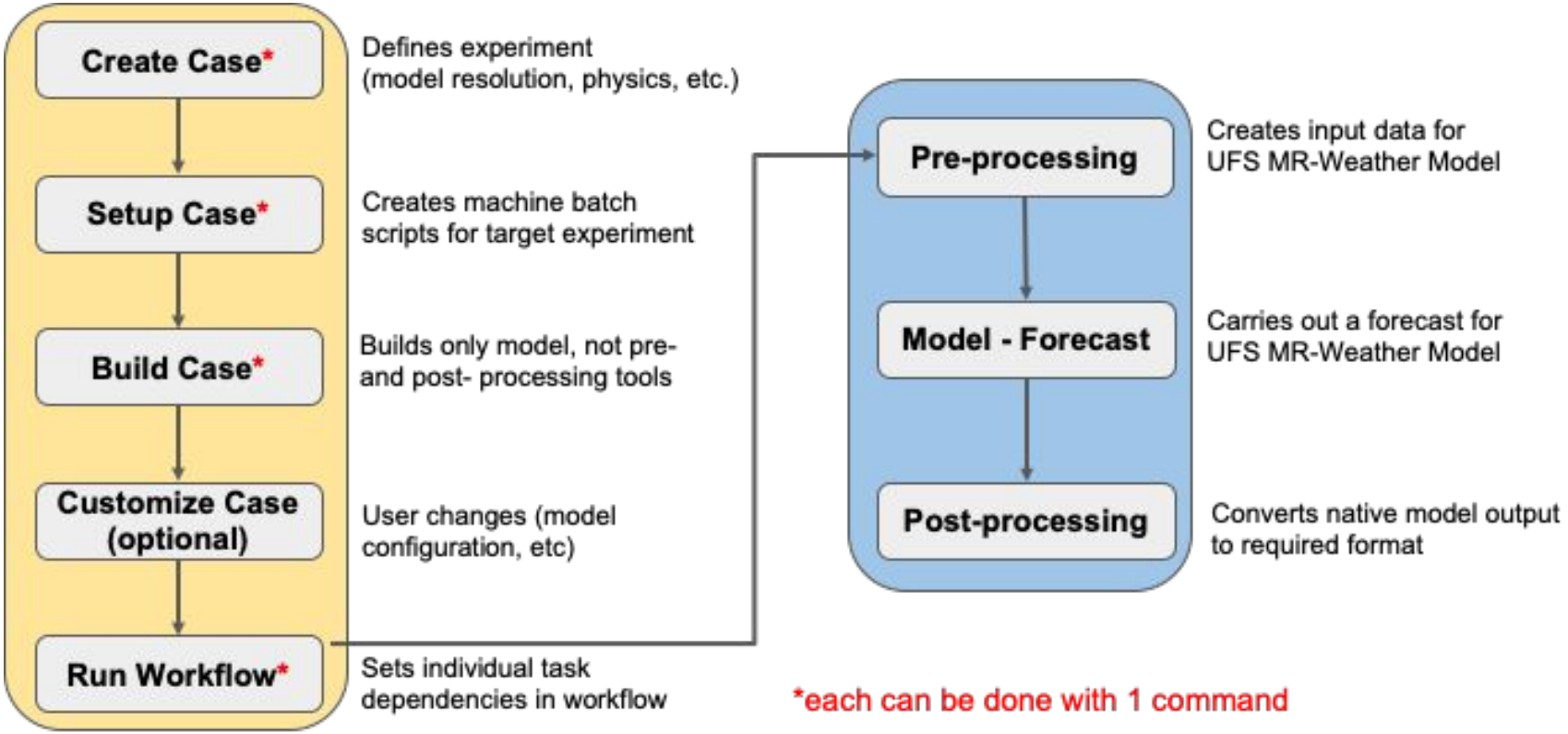
Ufuk Turuncoglu, Jim Edwards, Rocky Dunlap and Mariana Vertenstein

NCAR, CGD

Workflow and CIME

- UFS MR Weather Application workflow is based on the **Common Infrastructure for Modeling the Earth** (CIME).
- **CIME** was initiated in 2015 as a joint collaboration between DOE/E3SM and NCAR/CESM to create an object oriented python scripting infrastructure to deal with the complexities of a coupled earth system model.
  - Python based scripting infrastructure that provides a user-friendly, reproducible workflow
  - Experiments can be created easily (see Laurie Carson slides)
  - Open community collaboration in a public GitHub repository
  - Multi-agency (NSF, DOE, NOAA) and multi-model (CESM, E3SM, NORESM and UFS)
- Use of CIME in UFS MR Weather App release demonstrates the success of the NCAR/NOAA MOA

Overview of UFS workflow using CIME



Pre-processing

- **Pre-processor function:**
  - Conversion of raw input file to the required model input format, including change of resolution
  - It uses **chgres_cube** provided by UFS UTILS to create input for model
  - Three raw input data format are supported:
    - GFS GRIB2 (default)
    - NEMSIO
    - NetCDF
- **CIME workflow:**
  - Automatically generates the input namelist (fort.41) needed by the pre-processor (dependent on resolution and number of vertical layers)
  - Executes the pre-processor (chgres_cube)

Forecast

- **Forecast function:**
  - Running model in forward to make forecast
- **CIME workflow:**
  - Automatically creates experiment relevant model configuration settings (input.nml, model_configure etc.)
  - Provides ability for user to customize the initialization date, forecast length, model parameters, processor layout, and other aspects of the run
    - Recognizes if this is a run that is continued from a previous startup (restart)
    - Submits the forecast job to the batch system for execution
  - As part of submission, automatically stages all required input datasets (output from the pre-processor)

Post-processing

- **Post-processor function:**
  - The tool that is used by the workflow is EMC_post
  - The EMC_post repository contains source code and supporting files for the Unified Post Processor (UPP)
  - It converts native NetCDF output from the model to the GRIB2 format on standard isobaric coordinates in the vertical
  - Also calculates additional diagnostic fields that are not part of model output
    - such as precipitation, PBL and cloud related fields
- **CIME workflow:**
  - The CIME automatically creates the required scripts and namelist files for the post-processing as part of the workflow and run them after successful forecast run

Build and Platform Support

More information about installation of required tools

Level 1 - Pre-configured Platforms

- Prerequisites and libraries installed
- Workflow and model build/run out of the box
- Comprehensive testing before release

NCAR Cheyenne
NOAA Hera / Jet / Gaea

Level 2 - Configurable Platforms

- Prerequisites and libraries expected to install successfully
- Workflow and model expected to build/run
- Comprehensive testing

MSU Orion
TACC Stampede

Level 3 - Limited-test Platforms

- Prerequisites and libraries expected to install successfully
- Workflow and model should build and run
- Limited testing

MacOS
Generic Linux (Ubuntu)

UFS MR Weather Application

The CIME workflow builds the UFS MR Weather Model using a single command.

Prerequisite Libraries

Upgraded build process simplifies building all software libraries required by the model, pre-processing and post-processing utilities - already installed on level 1 platforms.

CIME

Basic CIME Commands

CIME documentation

- Commands under ***$CIMEROOT/scripts***
  - ***./query_config*** - displays information about available compsets, component settings, grids and/or machines such as *./query_config --machines*
  - ***./create_newcase*** - creates CIME experiments and requires minimum input such as compset, model grid and case directory such as *./create_newcase --case CASENAME --compset COMPSET --res GRID*
  - ***./query_testlists -*** list available tests of the application for each individual platform such as UFS

```
prealpha    : SMS_Lh3.C96.GFSv15p2.cheyenne_intel
prealpha    : SMS_Lh3.C96.GFSv15p2.cheyenne_gnu
prealpha    : SMS_Lh3.C96.GFSv15p2.orion_intel
prealpha    : SMS_Lh3.C96.GFSv16beta.cheyenne_intel
prealpha    : SMS_Lh3.C96.GFSv16beta.cheyenne_gnu
...
prealpha_p1: SMS_Lh3.C96.GFSv15p2.stampede2-skx_intel
prealpha_p1: SMS_Lh3.C96.GFSv16beta.stampede2-skx_intel
prealpha_p1: SMS_Lh3_D.C96.GFSv15p2.stampede2-skx_intel
...
```

  - ***./create_test -*** runs available tests such as *qcmd -l walltime=3:00:00 -- "export UFS_DRIVER=nems; CIME_MODEL=ufs ./create_test --xml-testlist ../../src/model/FV3/cime/cime_config/testlist.xml --xml-machine cheyenne --workflow ufs-mrweather_wo_post -j 4 --walltime 03:00:00"*

Basic CIME Commands (2)

- Commands under ***$CASEROOT***
  - ***./case.setup -*** script used to set up the case (create the case.run script, Macros file and user_nl_xxx files)
  - ***./case.build*** - script to build component and utility libraries and model executable
  - ***./xmlquery*** - script to query values in the xml files such as *./xmlquery -p RUN_STARTDATE*
  - ***./xmlchange -*** script to modify values in the xml files such as *./xmlchange STOP_OPTION=nhours, STOP_N=36*
  - ***./preview_namelists*** - script for users to see their component namelists in *$CASEROOT/CaseDocs* before running the model. It also creates the *$RUNDIR*
  - ***./preview_run*** - script for users to see batch submit and mpirun command. It also includes individual steps of the workflow

Basic CIME Commands (3)

- Commands under ***$CASEROOT***
  - ***./check_input_data -*** script for checking for various input data sets and moving them into place such as *./check_input_data --download*
  - ***./pelayout -*** Script to query and modify the *NTASKS*, *ROOTPE*, and *NTHRDS* for each component model. This a convenience script that can be used in place of *xmlchange* and *xmlquery*

Single component configuration

```
Comp   NTASKS   NTHRDS   ROOTPE
ATM :    108/      1;        0
```

Default layout for C96

```
Comp   NTASKS   NTHRDS   ROOTPE
ATM :    684/      4;        0
```

Default layout for C768

Total MPI tasks

Threading (OpenMP) is activated and uses 4 threads for each MPI task

Detailed look for *$CASEROOT*

- XML files
  - ***env_mach_specific.xml*** - sets a number of machine-specific environment variables as well as modules that will be used for building and/or running. You CANNOT use *xmlchange* to modify variables in this file.

```
<modules compiler="intel" mpilib="mpt">
  <command name="use">/glade/p/ral/jntp/GMTB/tools/modulefiles/intel-18.0.5/mpt-2.19</command>
  <command name="load">NCEPlibs/1.1.0</command>
</modules>
<modules DEBUG="FALSE" comp_interface="nuopc" compiler="intel" mpilib="impi">
  <command name="use">/glade/p/cesmdata/cseg/PROGS/modulefiles/esmfpkgs/intel/19.0.5</command>
  <command name="load">esmf-8.1.0b24-ncdfio-intelmpi-O</command>
</modules>
```

    The changes that are made in ***env_mach_specific.xml*** will be case specific.
    - can be used for experimenting modifications such as testing newer version modules, compilers
    - *./case.setup --keep env_mach_specific.xml* command can be used to keep the modifications

Detailed look for *$CASEROOT*

- XML files
  - ***env_build.xml -*** sets model build settings. This includes component resolutions and component compile-time configuration options.
    - You must run the case.build command after changing this file.
  - ***env_run.xml -*** sets runtime settings such as length of run, frequency of restarts, output of coupler diagnostics, and short-term and long-term archiving.
    - This file can be edited at any time before a job starts.
  - ***env_batch.xml -*** sets batch system settings such as wallclock time and queue name
- Directories
  - ***SourceMods -*** Top-level directory containing subdirectories for each compset component (such as src.ufsatm) where you can place modified source code for that component. You may also place modified buildnml and buildlib scripts here.

Detailed look for *$CIMEROOT/config/ufs*

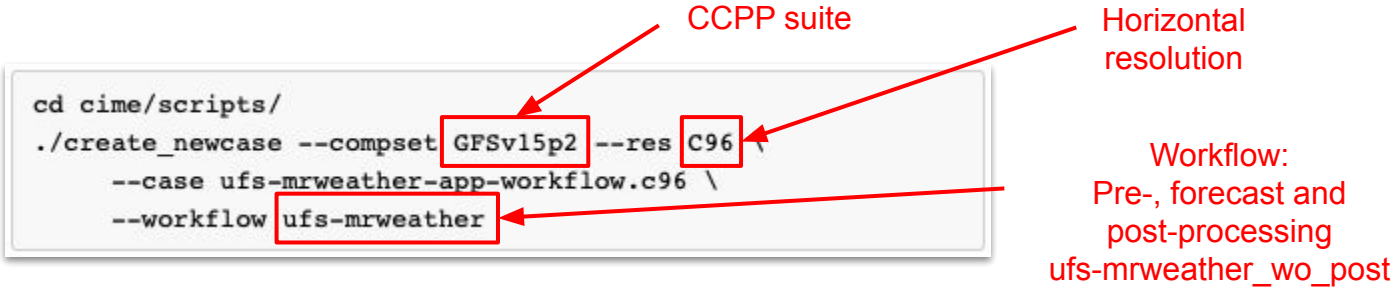
- XML files:
  - ***config_files.xml*** - contains all model-specific information that CIME uses to determine compsets, compset component settings, model grids, machines, batch queue settings, and compiler settings.
  - ***config_grids.xml*** - contains UFS specific grid definitions such as C96, C192, ...
  - ***config_inputdata.xml*** - defines remote sites that could be used to retrieve input files. The current release (version 1.1) only supports retrieving static input files.
- Directories:
  - ***machines/***
    - machine specific file - ***config_machines.xml***
    - Template workflow scripts for chgres and EMC post
    - Batch system and compiler specific files - ***config_batch.xml*** and ***config_compilers.xml***

Practical Session

Common Tasks

Changing CCPP Suite and Resolution

- Supported model configurations:
  - The application supports two different physics configurations: GFSv15p2 and GFSv16beta CCPP suites
  - in four different horizontal resolutions: C96 (~100 km), C192 (~50 km), C384 (~25 km) and C768 (~13 km)
- To create a case, the user need to provide the name of the case, CCPP suite, model resolution and desired workflow (with or without post-processing)



Customize the case

- By interacting with CIME interface, user could customize common configuration options easily
- For this purpose, the *xmlchange* command can be used to customize the case

```
# To change default simulation date
./xmlchange RUN_STARTDATE=2020-02-02,START_TOD=0

# To change default simulation lenght (5-days) to 36-hours
./xmlchange STOP_OPTION=nhours,STOP_N=36

# To change job wall clock time for forecast
./xmlchange JOB_WALLCLOCK_TIME=00:30:00

# To change job wall clock time for chgres
./xmlchange JOB_WALLCLOCK_TIME=00:30:00 --subgroup case.chgres
```

- The model configuration options can be modified via *user_nl_ufsatm*

Restarting the case

- The model can be restarted by issuing following command in *$CASEROOT* directory

```
./xmlchange CONTINUE_RUN=TRUE
```

- In this case, model will restart from previous forecast and run additional 5 days (the default simulation length)
  - both input.nml and model_configure are modified for restart simulation
  - The files from *RESTART/* directory are copied over *INPUT/*

```
./xmlchange STOP_N=1
./xmlchange STOP_N=6,STOP_OPTION=nhours
```

restart and run 1 day

restart and run 6 hours

Usage of *SourceMods*

- Let's assume that we would like to modify model code
  - The used does not need to modify main source code under *src/model*
  - The modified code can be placed under *SourceMods/src.ufsatm/*
  - Once the new source codes placed to *SourceMods/src.ufsatm/* the model needs to be build again using *./case.build*
  - and run model by issuing *./case.submit* command
- This allow to isolate the modifications from the model code and same source directory can be used for different experiments
- It also enables easy to exchange the information in terms of used code base for the experiment and allow to reproduce the experiment easily.

Supported Input Files and ICs

More information about fixed and raw initial condition files

- The **CIME automatically downloads the fixed file set**
  - The download process may be triggered manually by issuing *./check_input_data --download* command in the *$CASEROOT* directory.
  - However this step is automatically done by *case.submit*
- The raw initial conditions need to be downloaded manually (different than release 1.0) and placed under *icfiles* directory using following convention

```
$UFS_INPUT/ufs_inputdata/icfiles/YYYYMM/YYYYMMDD
```

- and it must follow following naming convention

Horizontal resolution (~13 km)

```
NEMSIO: sfc.input.ic.nemsio and atm.input.ic.nemsio
NetCDF: sfc.input.ic.nc and atm.input.ic.nc
GRIB2: atm.input.ic.grb2
```

0.5° (gfs_4 prefix) and 1.0° (gfs_3 prefix) files. *chgres_cube* automatically recognize the resolution

Supported Input Files and ICs (2)

More information about fixed and raw initial condition files

- The same initial condition directory may have raw data for different file formats (NEMSIO, GRIB2 and NetCDF), CIME interface uses GRIB2 by default.
- To force CIME to use specific data format, user needs to edit *user_nl_ufsatm* file under *$CASEROOT* and add one of the following namelist options
  - *input_type = "gaussian_nemsio"* for NEMSIO
  - *input_type = "gaussian_netcdf"* for NetCDF
  - *input_type = "grib2"* for GRIB2 format (it is default)
- In this case, the CIME interface automatically generates required namelist file for *chgres_cube* under run directory

Supported Input Files and ICs (3)

More information about fixed and raw initial condition files

- How can I use different data format in the existing case that has already processed input files via *chgres_cube*?
  - In this case, the user needs to delete the pre-processed input files from $RUNDIR/INPUT directory.
  - Otherwise, the workflow uses existing files and chgres_cube will not be triggered again
  - The raw data with new data format need to be retrieved first

```
# Go to run directory
cd `./xmlquery --value -p RUNDIR`

# Remove previously processed files by chgres_cube
rm -f INPUT/*_00.gfs_ctrl.nc
rm -f INPUT/*_00.*_data.tile*.nc

# Back to case directory
cd `cat CASEROOT`

# Edit user_nl_ufsatm and set new input type such as
input_type = "gaussian_netcdf"

# Mark run as cold run if it is restarted previously
./xmlchange CONTINUE_RUN=FALSE

# Submit chgres_cube job only
./case.submit --only-job case.chgres
```

Default PE Layout Configurations

- Default configurations for Linux:

Resolution	(A) Total Num. Proc.	Number of Threads	(B) Layout	(C) Write Task Group	(D) Write Tasks
C96	108	1	4x4	1	12
C192	180	1	4x6	1	36
C384	252	1	6x6	1	36
C768	684	4	12x8	3	36

- The logic:

A = Number of tiles * B + C * D

- **Note:** The model resolution also needs to divide evenly with the layout pair. For the given configuration of C192 resolution, 192/4=48 and 192/6=32

- Example for C192 (number of tiles is always 6 for global configuration):

180 = 6 * 4 * 6 + 1 * 36

Modifying Default PE Layout

- Let's assume that we would like to modify default PE Layout for C192

Resolution	(A) Total Num. Proc.	Number of Threads	(B) Layout	(C) Write Task Group	(D) Write Tasks
C192	180	1	4x6	1	36

- Change layout to 6x6

252 = 6 * 6 * 6 + 1 * 36

- Required modifications in the application:
  - Edit *user_nl_ufsatm* and change layout by adding *layout = 6,6*
  - Use *xmlchange* command to change total number of processor as *./xmlchange NTASKS_ATM=252*
  - You need to issue also *./case.setup --reset; ./case.build --clean-all and ./case.build*

Modifying Default PE Layout (2)

- CIME basically checks the combination of
  - Total Number of Processor
  - Layout
  - Number of Write Task Group
  - Write Tasks for each group
- If there is an inconsistency, it throws a message such as

```
ERROR: Total number of PE need to be consistent with the model namelist options:
        Total number of PE (ntask_atm) = 252
        Decomposition in x and y direction (layout) = 6x6
        Number of tile (ntiles) = 6
        Number of I/O group (write_groups) = 1
        Number of tasks in each I/O group (write_tasks_per_group) = 12
```

write_tasks_per_group need to be set as 36

Alternative solutions: (1) set write_groups as 3, **(2)** decrease NTASK_ATM to 228

Advanced Topics

CIME Workflow Scripts

- CIME workflow defines pre-processing, forecast and post-processing
  - The main scripts (bash) for pre- and post-processing steps can be found in
    - *$CIMEROOT/config/ufs/machines/template.chgres.run*
    - *$CIMEROOT/config/ufs/machines/template.gfs_post.run*
  - If user needs to modify those scripts, which is not recommended!), it must be done under *$CIMEROOT/config/ufs/machines/*
  - The CIME copies them to $CASEROOT as ***.case.chgres*** and ***.case.gfs_post*** again when user issues ./case.submit or ./preview_namelist

```
# Run it only if it is not restart and INPUT/ directory has no input for the given date
if [ "$isrestart" != "TRUE" -a ! -f "$rundir/INPUT/${prefix}.gfs_ctrl.nc" ]; then
  # Link namelist file
  ln -sf config.nml fort.41

  # Get current date
  LID=`${date_cmd} +%y%m%d-%H%M%S`

  # Run chgres
  runcmd='{{ mpirun }}'
  mpirun=`echo $runcmd | awk '{print $1}'`
  eval "$mpirun -n $np $blddir/chgres_cube.exe 1> chgres_cube.$LID.log 2>&1"

  # Move output files to input directory
  mv -f gfs_ctrl.nc INPUT/${prefix}.gfs_ctrl.nc
  mv -f out.atm.tile1.nc INPUT/${prefix}.gfs_data.tile1.nc
  mv -f out.atm.tile2.nc INPUT/${prefix}.gfs_data.tile2.nc
  mv -f out.atm.tile3.nc INPUT/${prefix}.gfs_data.tile3.nc
  mv -f out.atm.tile4.nc INPUT/${prefix}.gfs_data.tile4.nc
  mv -f out.atm.tile5.nc INPUT/${prefix}.gfs_data.tile5.nc
  mv -f out.atm.tile6.nc INPUT/${prefix}.gfs_data.tile6.nc
  mv -f out.sfc.tile1.nc INPUT/${prefix}.sfc_data.tile1.nc
  mv -f out.sfc.tile2.nc INPUT/${prefix}.sfc_data.tile2.nc
  mv -f out.sfc.tile3.nc INPUT/${prefix}.sfc_data.tile3.nc
  mv -f out.sfc.tile4.nc INPUT/${prefix}.sfc_data.tile4.nc
  mv -f out.sfc.tile5.nc INPUT/${prefix}.sfc_data.tile5.nc
  mv -f out.sfc.tile6.nc INPUT/${prefix}.sfc_data.tile6.nc
else
  echo "Skip running CHGRES!"
  echo "Restarted? - $isrestart"
  echo "Input already exists or processed? - $rundir/INPUT/${prefix}.gfs_ctrl.nc"
fi
```

Part of template.chgres.run

Porting CIME

- Steps needed to port the CIME workflow to a new platform:
  - Add the new machine description to ***config_machines.xml***
  - Add the batch system to ***config_batch.xml***
  - (Optional) Build and install the “cprnc” tool. It is used by the tests to compare with the baseline runs.
  - Verify that the port is working by running a simple test
- The existing ports such as Cheyenne, Orion, Stampede can be used as reference to include new platforms
- The user requires detailed information specific for the desired platform such as
  - Type of batch system used in the platform and details (queues, allocations etc.)
  - How software modules are handling? Which software modules are available?

Porting CIME (2)

- Add the new machine description to *config_machines.xml*
  - Edit the file *$CIMEROOT/config/ufs/machines/config_machines.xml* and add a new <machine/> entry under the root XML element.
  - A good approach to this is to copy an existing <machine/>description and modify it to match the new machine to which you are porting CIME.
  - The file can be verified as
    cd $CIMEROOT
    xmllint --noout --schema config/xml_schemas/config_machines.xsd config/ufs/machines/config_machines.xml

```
<machine MACH="orion">
  <DESC>NOAA orion system</DESC>
  <NODENAME_REGEX>Orion-login-\d+.HPC.MsState.Edu</NODENAME_REGEX>
  <OS>LINUX</OS>
  <COMPILERS>intel</COMPILERS>
  <MPILIBS>impi</MPILIBS>
  <PROJECT>nems</PROJECT>
  <SAVE_TIMING_DIR/>
  <CIME_OUTPUT_ROOT>$ENV{UFS_SCRATCH}</CIME_OUTPUT_ROOT>
  <DIN_LOC_ROOT>$ENV{UFS_INPUT}/ufs_inputdata</DIN_LOC_ROOT>
  <DIN_LOC_ROOT_CLMFORC>${DIN_LOC_ROOT}/atm/datm7</DIN_LOC_ROOT_CLMFORC>
  <DOUT_S_ROOT>$CIME_OUTPUT_ROOT/archive/$CASE</DOUT_S_ROOT>
  <BASELINE_ROOT>$ENV{UFS_INPUT}/UFS_BASELINES</BASELINE_ROOT>
  <GMAKE>make</GMAKE>
  <GMAKE_J>8</GMAKE_J>
  <BATCH_SYSTEM>slurm</BATCH_SYSTEM>
  <SUPPORTED_BY>NCEP</SUPPORTED_BY>
  <MAX_TASKS_PER_NODE>80</MAX_TASKS_PER_NODE>
  <MAX_MPITASKS_PER_NODE>40</MAX_MPITASKS_PER_NODE>
  <PROJECT_REQUIRED>TRUE</PROJECT_REQUIRED>
  <mpirun mpilib="default">
    <executable>srun</executable>
    <arguments>
      <arg name="num_tasks">-n $TOTALPES</arg>
    </arguments>
  </mpirun>
  <mpirun mpilib="mpi-serial">
    <executable></executable>
  </mpirun>
```

```
  <module_system type="module">
    <init_path lang="sh">/apps/lmod/lmod/init/sh</init_path>
    <init_path lang="csh">/apps/lmod/lmod/init/csh</init_path>
    <init_path lang="python">/apps/lmod/lmod/init/env_modules_python.py</init_path>
    <cmd_path lang="sh">module</cmd_path>
    <cmd_path lang="csh">module</cmd_path>
    <cmd_path lang="python">/apps/lmod/lmod/libexec/lmod python</cmd_path>
    <modules compiler="intel">
      <command name="purge"/>
      <command name="load">intel/2018.4</command>
      <command name="load">netcdf/4.7.2</command>
      <command name="load">cmake/3.15.4</command>
    </modules>
    <modules mpilib="impi">
      <command name="load">impi/2018.4</command>
    </modules>
  </module_system>
```

Porting CIME (3)

- Add the batch system to *config_batch.xml*
  - Edit file *$CIMEROOT/config/ufs/machines/config_batch.xml* and add a <batch_system/> element describing the batch system on the new machine.
  - To verify correctness of the *config_batch.xml* file, use the command:
    cd $CIMEROOT
    xmllint --noout --schema config/xml_schemas/Config_batch.xsd config/ufs/machines/config_batch.xml

```
<batch_system MACH="orion" type="slurm" >
  <batch_submit>sbatch</batch_submit>
  <submit_args>
    <arg flag="--time" name="$JOB_WALLCLOCK_TIME"/>
    <arg flag="-q" name="$JOB_QUEUE"/>
    <arg flag="--account" name="$PROJECT"/>
  </submit_args>
  <directives>
    <directive>--partition=orion</directive>
  </directives>
  <queues>
    <queue default="true" walltimemax="08:00:00" nodemin="1" nodemax="210">batch</queue>
    <queue walltimemax="00:30:00" nodemin="1" nodemax="210">debug</queue>
  </queues>
</batch_system>
```

Questions?

Documentation for more detailed information:

https://ufs-mrweather-app.readthedocs.io/en/ufs-v1.1.0/