Advancing HAFS Workflow with CIME

Common Infrastructure for Modeling the Earth

https://github.com/ESMCI/cime

Rocky Dunlap,
Mariana Vertenstein, Jim Edwards, Ufuk Turuncoglu

5 November, 2019

This material is based upon work supported by the National Center for Atmospheric Research, which is a major facility sponsored by the National Science Foundation under Cooperative Agreement No. 1852977.

Advancing HAFS Workflow with CIME

The *Common Infrastructure for Modeling the Earth* (CIME) is a community tool for configuring, building, and executing Earth system modeling workflows. It has been developed jointly by NCAR and DoE.

CIME is being brought into HAFS to extend and complement the existing workflow elements with new capabilities needed by the research community and to facilitate T2O benchmarking activities:

- **hierarchical model development** of HAFS
- **"data models"** (e.g., atmosphere, ocean, wave) that can stand in for active models to isolate coupling feedbacks
- **regression testing and verification** of standard configurations against baselines
- **usability and portability** to non-NOAA machines

Hierarchical Model Development

Hierarchical model development (HMD) enables **building up a complex, multi-component model incrementally**, testing individual components in isolation, systematically adding feedbacks between components, and evolving the system to a fully coupled model.

CIME facilitates HMD through ***compsets***: configurations of active and data components (DATM, DOCN, DWAV), including a coupler.

Building up to **FV3GFS+HYCOM+WW3**:

- FV3GFS + DOCN
- DATM + HYCOM
- FV3GFS + HYCOM + DWAV
- FV3GFS + HYCOM + WW3

Data and simplified model components isolate feedbacks to focus development on one active component at a time.

Major Components of CIME

Case Control System (CCS)

Create, configure, build and run **modeling workflows**;
Extensible unit and system **testing framework**

NUOPC-based Coupling Infrastructure

CMEPS:
Community Mediator for Earth Prediction Systems

Data Models

Hierarchical Model Development

Not shown are the statistical ensemble verification tool (for porting) and external libraries

CIME Case Control System (CCS)

The CIME CCS provides a user-friendly interface for configuring, building and executing Earth system modeling workflows.

- Provides **extensible set of out-of-the box tested configurations**
  - model (resolution, physics options, component coupling)
  - workflow with pre-processing, forecast, post-processing tasks
- Support for **user customization**
  - e.g., model source code changes and input namelist changes
- Integrated **data models** provide static forcings/BCs for testing
- Provides experiment **provenance for reproducibility** and documentation
- Facilitates **porting the workflow** to new machines (laptop to HPC)
- Generate **Cylc workflow suites**; extensible to other engines
- Comprehensive model **testing framework**
  - smoke test; baseline comparisons; exact restarts; reproducibility under different processor/thread counts; performance tests

Workflow Milestones (HSUP 1A-3-6a)

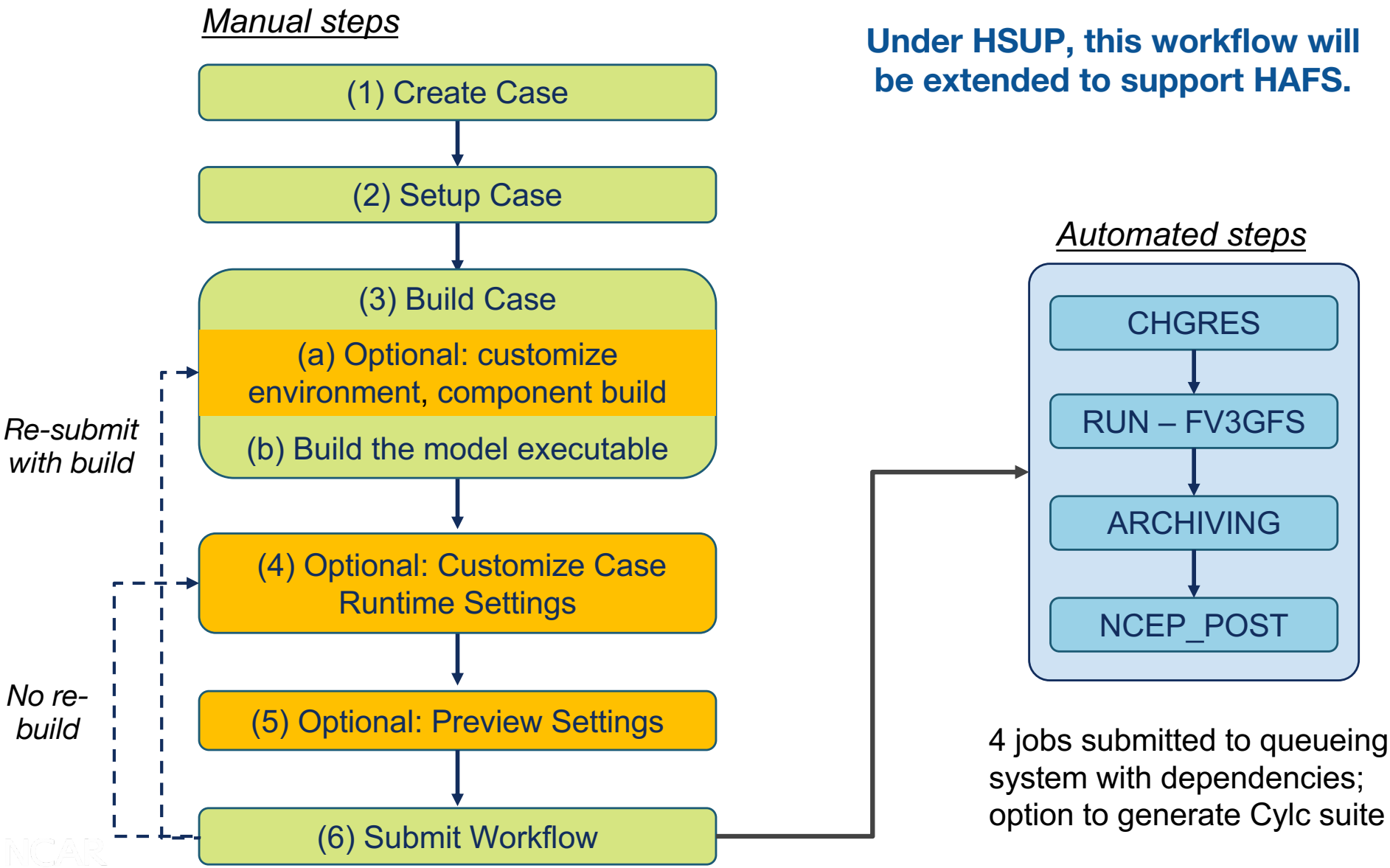
- *Q2FY20*: Make model components **CIME-compliant** (FV3GFS, HYCOM, WW3) with test runs on NCEP and non-NCEP machines
- *Q3FY20*: Demonstrate CIME can **run simple HAFS forecast** model configurations
- *Q4FY20*: CIME **data models** support domains, grids and forecast periods required for HAFS
- *Q2FY21*: Demonstrate HAFS **hierarchical testing workflow** with different configurations of active/data components
- *Q4FY21*: Demonstrate a HAFS workflow configuration that is suitable for **simplified benchmarking** of the forecast model, leveraging and extending CIME/CROW/HWRF/HAFS workflows

CIME Workflow in UFS Applications

Due to overlapping workflow requirements, CIME is being introduced as a workflow option in three UFS applications:

- **UFS Weather** (FV3GFS Global)
  - select by UFS release team as a community workflow
  - an initial prototype workflow is being extended and tested for the upcoming UFS release
- **UFS S2S** (FV3GFS-MOM6-CICE5)
  - university community access and testing of coupled system
- **UFS Hurricane** (FV3GFS-HYCOM-WW3)
  - will be introduced into HAFS as a community workflow to facilitate hierarchical model development and portability

CIME Workflow for UFS Global Weather



Under HSUP, this workflow will be extended to support HAFS.

4 jobs submitted to queueing system with dependencies; option to generate Cylc suite

(1) Create Case (create_newcase)

```
$ git clone https://github.com/ESCOMP/UFSCOMP.git
$ cd UFSCOMP

# To checkout the global weather workflow application:
$ git checkout app_fv3gfs

# Check out all model components and CIME
$ ./manage_externals/checkout_externals

# Go to CIME scripts directory
$ cd cime/scripts

# Create UFS Global Weather Workflow case
$ ./create_newcase --compset UFS_Weather --res C96 --case fv3gfs_global_app \
                   --workflow-id fv3gfs_global --driver nuopc --run-unsupported
```

- **compset**: describes the components that are used, here its FV3GFS without a mediator
- **res**: FV3GFS resolution
- **case**: user defined case name
- **workflow-id**: id of workflow to run - currently CIME supports out of box workflow that is configured for FV3GFS
- **driver**: use NUOPC driver

(2) Setup Case (case.setup)

```
Buildconf/
LockedFiles/
README.case
SourceMods/
Tools/
archive_metadata
case.build
case.cmpgen_namelists
case.qstatus
case.setup
case.submit
check_case
check_input_data
env_archive.xml
env_batch.xml
env_build.xml
env_case.xml
env_mach_pes.xml
env_mach_specific.xml
env_run.xml
env_workflow.xml
pelayout
preview_namelists
preview_run
xmlchange
xmlquery
```

```
.case.chgres*
.case.gfs_post*
.case.run*
.env_mach_specific.csh
.env_mach_specific.sh
Buildconf/
CaseDocs/
CaseStatus
Depends.intel
LockedFiles/
Macros.cmake
Macros.make
README.case
SourceMods/
Tools/
archive_metadata
case.build
case.cmpgen_namelists
case.qstatus
case.setup
case.st_archive*
case.submit
check_case
check_input_data
env_archive.xml
env_batch.xml
env_build.xml
env_case.xml
env_mach_pes.xml
env_mach_specific.xml
env_run.xml
env_workflow.xml
pelayout
preview_namelists
preview_run
software_environment.txt
user_nl_cpl
user_nl_fv3gfs
xmlchange
xmlquery
```

- Submission scripts for individual Workflow tasks
- Provenance of xmlchange
- Customize cmake
- Customize make
- Modified source code
- Submit case to batch queue
- Customize fv3gfs namelist

new directory fv3gfs_global_app/

(3a) Customize Component Builds

env_build.xml - case XML File that controls component specific build settings (e.g. CPP definitions)

SourceMods/ - directory generated by ./case.setup for placing source code modifications

```
src.drv/
src.fv3gfs/
src.share/
```

source files with modifications from fv3gfs placed in src.fv3gfs/ will be compiled instead of those that were checked out

(3b) Build the Model (case.build)

- The xml variable $EXEROOT controls where the model is built and run
- **case.build** will build the component libraries and the model executable in $EXEROOT/bld
- Build log output for every component will also appear here
- Timing for each build component is output to the screen

(4) Customize Runtime Settings

Customization for FV3GFS workflow – use **xmlchange** used to modify XML files

```
# Change configuration options such as start time, wall clock limit for schedular, simulation time
$ ./xmlchange DOUT_S=FALSE
$ ./xmlchange STOP_OPTION=nhours
$ ./xmlchange STOP_N=36
$ ./xmlchange RUN_REFDATE=2016-10-03
$ ./xmlchange RUN_STARTDATE=2016-10-03
$ ./xmlchange JOB_WALLCLOCK_TIME=00:30:00
$ ./xmlchange USER_REQUESTED_WALLTIME=00:30:00
```

<u>case XML files that control runtime environment</u>

env_run.xml – run time settings (e.g. length of run, start type, etc)

env_mach_pes.xml – PE layout

env_workflow.xml – settings for each workflow stage (4 here)

case.chgres, case.run, case.st_archive, case.gfs_post

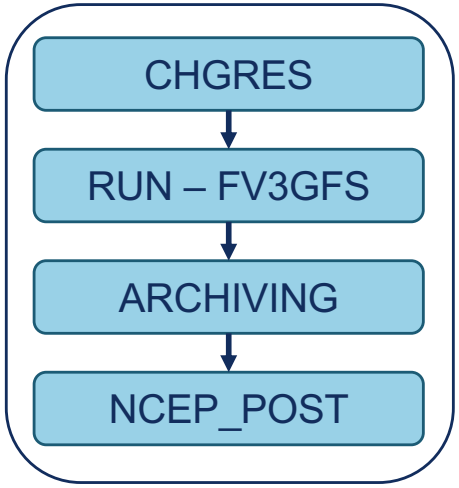
env_batch.xml – batch queue settings for target machine

Use **xmlquery** to get settings and definitions for xml variables

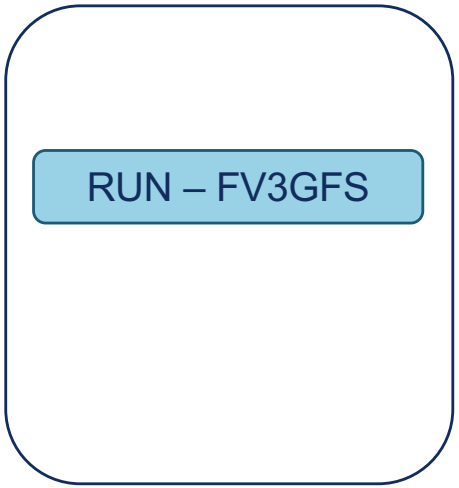
(6) Submit Workflow (case.submit)

- CCS has a simple dependency generator for workflow that uses the queuing system.
- By default this is what will be used in the UFS global weather release
- This is detailed in the case XML file **env_workflow.xml**

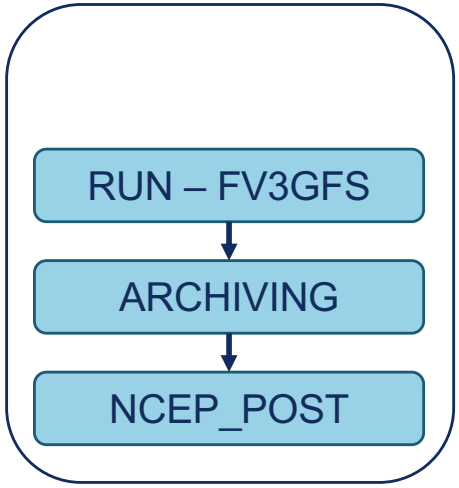
submit all workflow
case.submit



ONLY submit run phase
case.submit --only-job case.run



submit all workflow
STARTING FROM case.run
case.submit --job case.run



Graduate Student Testing of UFS S2S CIME-based Workflow

This test **evaluated the usability of a UFS Subseasonal-to-Seasonal coupled application** (GitHub link) comprised of:

- CIME workflow
- Community Mediator for Earth Prediction Systems (CMEPS)
- FV3GFS (atm), MOM6 (ocn), CICE5 (ice)
- Platforms: Cheyenne/NCAR, Stampede2/XSEDE, Theia/NOAA

Graduate Student Test criteria:

- Get code.
- Run code.
- Change code.
- Test code for correct operation.
- Evaluate code with standard diagnostic packages.
- Get documentation, user support, and training.
- Understand what is needed for their code changes to transition to operations.

NCAR
UCAR

Courtesy Cecelia DeLuca/Comms&Outreach WG

Graduate Student Testing of UFS S2S CIME-based Workflow

Specifically, the CIME/S2S test required that in 6 hours:

- get and run the FV3GFS-MOM6-CICE5 coupled application for 5 day forecast using CIME workflow
- modify the code to increase the SST provided by the ocean by 2 deg C
- re-run the application for 5 days with the modification
- visually compare results.

Student participants:

- 3 grad students/postdocs from U Albany, U Michigan, and GMU

Summary of results:

- **2 students were able to successfully complete the test within 6 hours**
- **1 student was unable to complete test, due to file permissions problems**

Courtesy Cecelia DeLuca/Comms&Outreach WG

CIME Open Development

CIME has been designed to facilitate and encourage **community collaboration**.

- Open collaboration on public GitHub repository: https://github.com/ESMCI/cime

- Multi-agency participation:
  - NCAR/CESM
  - DOE/E3SM
  - NORSC/NorESM
  - Upcoming UFS FV3GFS/global release

- Extensive, object-oriented Python design; XML-based configuration metadata

Summary

- CIME is being introduced as a workflow option in HAFS to facilitate research and development requirements including hierarchical model development, usability, regression testing, verification, and portability.
- Working with DTC/GSD collaborators, we will integrate CIME with existing HAFS workflow elements.
- CIME is being used for two other UFS applications, the Global Weather UFS release and S2S systems.
- Graduate student testing has demonstrated the ability of community users to get UFS model code, build, run and perform basic development tasks using CIME.

Thank You!

Extra Slides

CIME CCS in CESM

- CCS used in all CESM component model development process
- CCS used as the workflow in all CESM releases
- CCS used for CMIP6 Experiments (since August 2018)
  - Have run 979 different CESM cases.
  - Published 690 cases.
  - Generated ~1.3 PB of compressed (lossless) time series files.
  - Published ~435 TB of compressed CMIP6 files to ESGF.

Output after calling create_newcase

```
Compset longname is 2000_FV3GFS_SLND_SICE_SOCN_SROF_SGLC_SWAV
Compset specification file is /glade/work/mvertens/oct24_global/components/fv3//cime_config/config_compsets.xml
Automatically adding SESP to compset
Compset forcing is 1972-2004
================================================================================
ATM component is FV3GFS Atmosphere
LND component is Stub land component
ICE component is Stub ice component
OCN component is Stub ocn component
ROF component is Stub river component
GLC component is Stub glacier (land ice) component
WAV component is Stub wave component
ESP component is Stub external system processing (ESP) component
================================================================================
Pes     specification file is /glade/work/mvertens/oct24_global/components/fv3//cime_config/config_pes.xml
Machine is cheyenne
Pes setting: grid match    is a%C96
Pes setting: compset_match is FV3GFS
Pes setting: grid          is a%C96_l%null_oi%null_r%null_g%null_w%null_z%null_m%null
Pes setting: compset       is 2000_FV3GFS_SLND_SICE_SOCN_SROF_SGLC_SWAV_SESP
Pes setting: tasks       is {'NTASKS_ATM': 150, 'NTASKS_ICE': 1, 'NTASKS_CPL': 1,
                            'NTASKS_LND': 1, 'NTASKS_WAV': 1, 'NTASKS_ROF': 1,
                            'NTASKS_OCN': 1, 'NTASKS_GLC': 1}
Pes setting: threads     is {'NTHRDS_ICE': 1, 'NTHRDS_ATM': 1, 'NTHRDS_ROF': 1,
                            'NTHRDS_LND': 1, 'NTHRDS_WAV': 1, 'NTHRDS_OCN': 1,
                            'NTHRDS_CPL': 1, 'NTHRDS_GLC': 1}
Pes setting: rootpe      is {'ROOTPE_OCN': 0, 'ROOTPE_LND': 0, 'ROOTPE_ATM': 0,
                            'ROOTPE_ICE': 0, 'ROOTPE_WAV': 0, 'ROOTPE_CPL': 0,
                            'ROOTPE_ROF': 0, 'ROOTPE_GLC': 0}
Pes setting: pstrid      is {}
Pes other settings: {}
Pes comments: none
================================================================================
 Compset is: 2000_FV3GFS_SLND_SICE_SOCN_SROF_SGLC_SWAV_SESP
 Grid is: a%C96_l%null_oi%null_r%null_g%null_w%null_z%null_m%null
 Components in compset are: ['fv3gfs', 'slnd', 'sice', 'socn', 'srof', 'sglc', 'swav', 'sesp', 'drv', 'dart']
================================================================================
Using project from env PROJECT: P93300606
No charge_account info available, using value from PROJECT
cesm model version found: cmeps_v0.4.1-12-gc552246
Batch_system_type is pbs
job is case.chgres USER_REQUESTED_WALLTIME None USER_REQUESTED_QUEUE None
job is case.run USER_REQUESTED_WALLTIME None USER_REQUESTED_QUEUE None
job is case.gfs_post USER_REQUESTED_WALLTIME None USER_REQUESTED_QUEUE None
job is case.st_archive USER_REQUESTED_WALLTIME None USER_REQUESTED_QUEUE None
 Creating Case directory /glade/work/mvertens/oct24_global/cime/scripts/fv3gfs_global_app
```

(3a) Customizing the compiler build flags

Build flags can be customized in **Macros.cmake** or Macros.make

```
if("${COMPILER}" STREQUAL "gnu")
  set(MPIFC "mpif90")
  set(FFLAGS_NOOPT "-O0")
  set(MPICC "mpicc")
  set(SCC "gcc")
  set(MPICXX "mpicxx")
  set(HAS_F2008_CONTIGUOUS "FALSE")
  set(SUPPORTS_CXX "TRUE")
  set(FFLAGS "-fconvert=big-endian -ffree-line-length-none -ffixed-line-length-none")
  set(FIXEDFLAGS "-ffixed-form")
  set(CXX_LINKER "FORTRAN")
  set(FC_AUTO_R8 "-fdefault-real-8")
  set(CFLAGS "-std=gnu99")
  set(FREEFLAGS "-ffree-form")
  set(SFC "gfortran")
  set(SCXX "g++")
endif()
if("${COMPILER}" STREQUAL "intel")
  set(MPIFC "mpif90")
  set(FFLAGS_NOOPT "-O0")
  set(MPICC "mpicc")
  set(SCC "icc")
  set(MPICXX "mpicxx")
  set(CXX_LDFLAGS "-cxxlib")
  set(SUPPORTS_CXX "TRUE")
  set(FFLAGS "-qno-opt-dynamic-align  -convert big_endian -assume byterecl -ftz -traceback
  set(FIXEDFLAGS "-fixed")
  set(CXX_LINKER "FORTRAN")
  set(FC_AUTO_R8 "-r8")
  set(CFLAGS "-qno-opt-dynamic-align -fp-model precise -std=gnu99")
  set(FREEFLAGS "-free")
  set(SFC "ifort")
  set(SCXX "icpc")
endif()
```

(3b) Customizing the machine environment

env_mach_specific.xml - case XML File that controls machine environment

```
<header>
  These variables control the machine dependent environment including
  the paths to compilers and libraries external to cime such as netcdf,
  environment variables for use in the running job should also be set here.
</header>
<module_system type="module">
  <init_path lang="perl">/glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/init/perl</init_path>
  <init_path lang="python">/glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/init/env_modules_python.py</init_path>
  <init_path lang="csh">/glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/init/csh</init_path>
  <init_path lang="sh">/glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/init/sh</init_path>
  <cmd_path lang="perl">/glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/libexec/lmod perl</cmd_path>
  <cmd_path lang="python">/glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/libexec/lmod python</cmd_path>
  <cmd_path lang="sh">module</cmd_path>
  <cmd_path lang="csh">module</cmd_path>
  <modules>
    <command name="purge"/>
    <command name="load">ncarenv/1.2</command>
  </modules>
  <modules compiler="intel">
    <command name="load">intel/19.0.2</command>
    <command name="load">esmf_libs</command>
    <command name="load">mkl</command>
  </modules>
  ......
</module_system>
<environment_variables>
  <env name="OMP_NUM_THREADS">1</env>
  <env name="OMP_STACKSIZE">1024M</env>
  <env name="TMPDIR">/glade/scratch/$USER</env>
  <env name="MPI_TYPE_DEPTH">16</env>
  <env name="MPI_IB_CONGESTED">1</env>
  <env name="MPI_USE_ARRAY"/>
</environment_variables>
 ......
<resource_limits>
  <resource name="RLIMIT_STACK">-1</resource>
</resource_limits>
 ......
<mpirun mpilib="default">
  <executable>mpiexec_mpt</executable>
  <arguments>
    <arg name="labelstdout">-p "%g:"</arg>
    <arg name="num_tasks"> -np {{ total_tasks }}</arg>
    <arg name="zthreadplacement"> omplace -tm open64 </arg>
  </arguments>
</mpirun>
```

system modules

environment variables

resource limits

mpirun command

(5) Preview Settings (preview_run)

Preview information on each workflow state by calling **preview_run**
(the following is ONLY shown for case.run)

```
CASE INFO:
  nodes: 5
  total tasks: 150
  tasks per node: 36
  thread count: 1

BATCH INFO:
  FOR JOB: case.run
    ENV:
      module command is /glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/libexec/lmod python purge
      module command is /glade/u/apps/ch/opt/lmod/7.5.3/lmod/lmod/libexec/lmod python load \
                         ncarenv/1.2 intel/19.0.2 esmf_libs mkl mpt/2.19 \
                         netcdf-mpi/4.6.1 pnetcdf/1.11.0 ncarcompilers/0.5.0
      Setting Environment OMP_NUM_THREADS=1
      Setting Environment OMP_STACKSIZE=1024M
      Setting Environment TMPDIR=/glade/scratch/mvertens
      Setting Environment MPI_TYPE_DEPTH=16
      Setting Environment MPI_IB_CONGESTED=1
      Setting Environment ESMFMKFILE=/glade/work/turuncu/ESMF/8.0.0b48/lib/libO/Linux.intel.64.mpt.default/esmf.mk
      Setting Environment ESMF_RUNTIME_PROFILE=ON
      Setting Environment ESMF_RUNTIME_PROFILE_OUTPUT=SUMMARY
      Setting Environment UGCSINPUTPATH=/glade/scratch/turuncu/fv3gfs/fv3_gfdlmprad
      Setting Environment UGCSFIXEDFILEPATH=/glade/work/turuncu/FV3GFS/fix_am
      Setting Environment UGCSADDONPATH=/glade/work/turuncu/FV3GFS/addon
      Setting Environment CHGRES_NML=/glade/scratch/turuncu/fv3gfs/chgres.test/config.C96.nml
      Setting Environment CHGRES_BIN=/glade/work/turuncu/UFS/UFS_UTILS/exec/chgres_cube.exe
      Setting Environment NCEP_POST_BIN=/glade/work/turuncu/UFS/GLB_WRKF/sorc/gfs_post.fd/exec/ncep_post
      Setting Environment LD_LIBRARY_PATH=/glade/u/home/dunlap/YAML-INSTALL/lib:..........
      Setting Environment OMP_NUM_THREADS=1

    SUBMIT CMD:
      qsub -q regular -l walltime=12:00:00 -A P93300606 -v ARGS_FOR_SCRIPT='--resubmit' .case.run

    MPIRUN (job=case.run):
      mpiexec mpt -p "%g:"-np 150  omplace -tm open64  /glade/scratch/mvertens/fv3gfs_global_app/bld/cesm.exe >> cesm.log.$LID 2>&1
```

(5) Preview Settings (pelayout)

Preview processor usage and layout by calling **pelayout**

```
Comp   NTASKS   NTHRDS   ROOTPE
CPL :      1/      1;        0
ATM :    150/      1;        0
LND :      1/      1;        0
ICE :      1/      1;        0
OCN :      1/      1;        0
ROF :      1/      1;        0
GLC :      1/      1;        0
WAV :      1/      1;        0
ESP :      1/      1;        0
```

Obtaining timing information with every case

- Every job submission produces model timing output in a case timing/ directory

```
component       comp_pes    root_pe   tasks  x threads instances (stride)
---------        ------     -------   ------   ------  ---------  ------
cpl = cpl        144         0        144    x 1       1      (1     )
atm = datm       144         0        144    x 1       1      (1     )
lnd = clm        144         0        144    x 1       1      (1     )
ice = sice       144         0        144    x 1       1      (1     )
ocn = socn       144         0        144    x 1       1      (1     )
rof = mosart     144         0        144    x 1       1      (1     )
glc = sglc       144         0        144    x 1       1      (1     )
wav = swav       144         0        144    x 1       1      (1     )
esp = sesp       144         0        144    x 1       1      (1     )

total pes active           : 144
mpi tasks per node               : 36
pe count for cost estimate : 144

Overall Metrics:
  Model Cost:            3690.55   pe-hrs/simulated_year
  Model Throughput:         0.94   simulated_years/day

  Init Time   :      70.634 seconds
  Run Time    :       5.266 seconds      252.778 seconds/day
  Final Time  :       0.087 seconds


Runs Time in total seconds, seconds/model-day, and model-years/wall-day
CPL Run Time represents time in CPL pes alone, not including time associated with data exchange with other components

  TOT Run Time:       5.266 seconds      252.778 seconds/mday         0.94 myears/wday
  CPL Run Time:       1.132 seconds       54.326 seconds/mday         4.36 myears/wday
  ATM Run Time:       0.288 seconds       13.848 seconds/mday        17.09 myears/wday
  LND Run Time:       3.212 seconds      154.195 seconds/mday         1.54 myears/wday
  ICE Run Time:       0.000 seconds        0.000 seconds/mday         0.00 myears/wday
  OCN Run Time:       0.000 seconds        0.000 seconds/mday         0.00 myears/wday
  ROF Run Time:       0.759 seconds       36.446 seconds/mday         6.49 myears/wday
  GLC Run Time:       0.000 seconds        0.000 seconds/mday         0.00 myears/wday
  WAV Run Time:       0.000 seconds        0.000 seconds/mday         0.00 myears/wday
  ESP Run Time:       0.000 seconds        0.000 seconds/mday         0.00 myears/wday
  CPL COMM Time:      0.233 seconds       11.194 seconds/mday        21.15 myears/wday
```

Creating a workflow engine with Cycl

CCS also has a **utility generate_cylc_workflow.py** uses the dependency chain in **env_workflow.xml** to auto-generate a **Cylc** control file

New capability for other workflow engines, Rocoto, etc. can also be developed

```
usage: generate_cylc_workflow.py [-h] [-d] [-v] [-s] [--cycles CYCLES]
                                 [--ensemble ENSEMBLE]
                                 [caseroot]

Generates a cylc workflow file for the case.  See https://cylc.github.io for details about cylc

positional arguments:
  caseroot              Case directory for which namelists are generated.
                        Default is current directory.

optional arguments:
  -h, --help            show this help message and exit
  -d, --debug           Print debug information (very verbose) to file
  -v, --verbose         Add additional context (time and file) to log messages
  -s, --silent          Print only warnings and error messages
  --cycles CYCLES       The number of cycles to run, default is RESUBMIT
  --ensemble ENSEMBLE   generate suite.rc for an ensemble of cases,
                        the case name argument must end in an integer.
                        for example: ./generate_cylc_workflow.py --ensemble 4
                        will generate a workflow file in the current case, if that case is named
                        case.01,the workflow will include case.01, case.02, case.03 and case.04
```

CCS System Testing

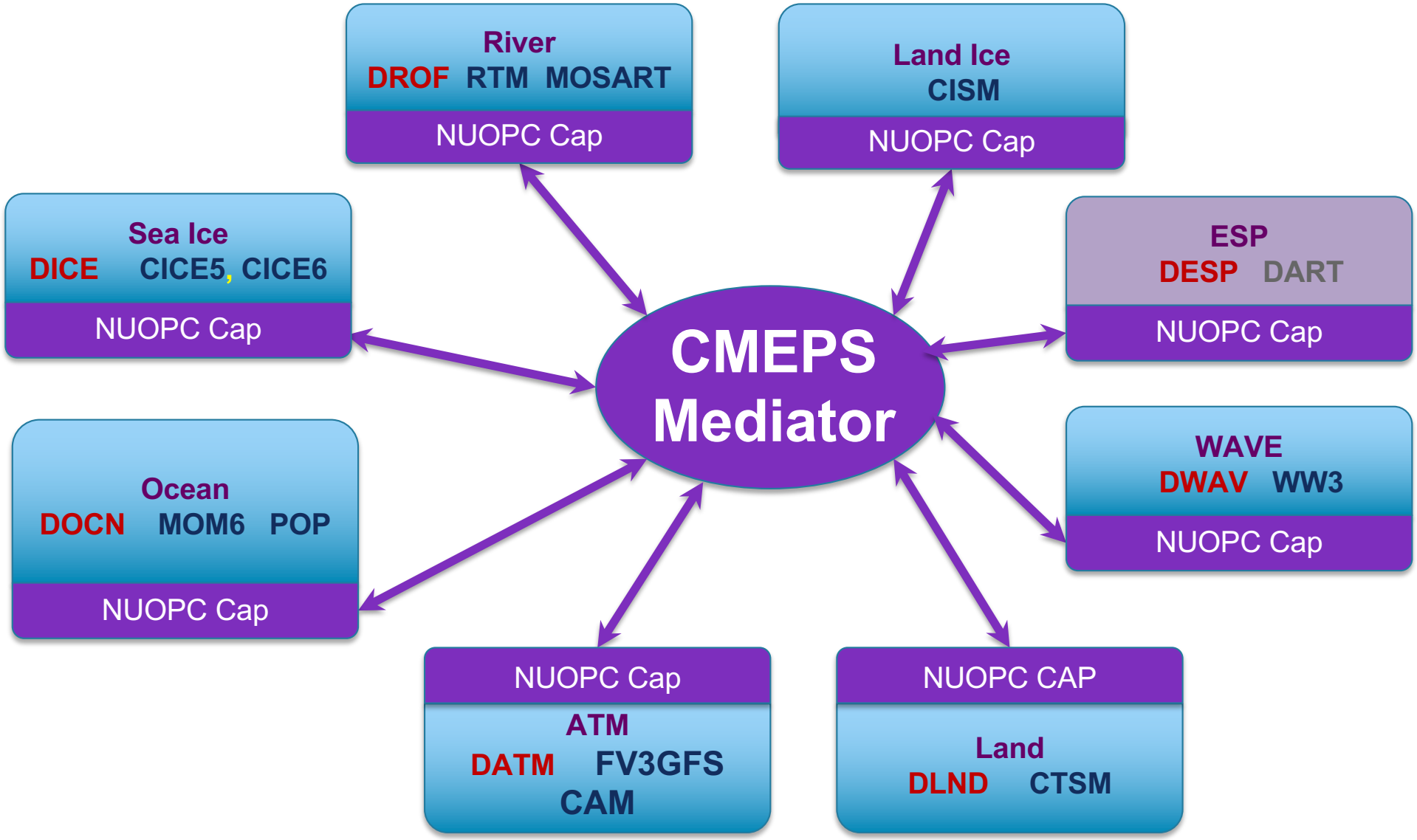
- Provides an extensible system test capability that is able to carry out tests for
  - Verifying that restarts are bit-for-bit
  - Determining model performance
  - Verifying that pe-layout changes are bit-for-bit
  - Verifying that threading implementation is correct
  - Comparing against previous saved baselines (tracks if unexpected answer changes appear)
  - Determining if memory leak is present
  - Determining if code changes result in a performance degradation
  - Tracking if model input parameter changes have occurred since last baseline

CIME documentation

The Common Infrastructure for Modeling the Earth (CIME - pronounced "SEAM") provides a Case Control System for configuring, compiling and executing Earth system models, data and stub model components, a driver and associated tools and libraries.

Table of contents

CESM architecture and CMEPS



Status of CMEPS

- CESM
  - All CESM components have NUOPC caps and current validation is underway to compare to older MCT/cpl7 infrastructure
  - Plan is to CESM migrate to CMEPS by May 2020
- NOAA/EMC
  - Validation of FV3GFS S2S season and Graduate Student Test carried out (using CIME Case Control System)

https://github.com/ESCOMP/UFSCOMP/wiki/Milestone:-CMEPS-0.5

https://github.com/ESCOMP/UFSCOMP/wiki/Milestone:-CMEPS-0.5-Appendix-Graduate-Student-Test-Evaluation-SST-Experiment

https://github.com/ESCOMP/UFSCOMP/wiki/Milestone:-CMEPS-0.5-Appendix-Graduate-Student-Test-Evaluation-Wind-Stress-Experiment

Advantages of CMEPS vs. NEMS

https://vlab.ncep.noaa.gov/redmine/projects/emc_fv3-mom6-cice5/wiki/Running_With_CMEPS

Aspect	CMEPS	NEMS Mediator	Answer Changing?	Comments
Internal geometric structure	ESMF_Mesh	ESMF_Grid	No	ESMF_Mesh is used in CMEPS because it is the most flexible and can handle both structured and unstructured grids. Grids are automatically converted to meshes.
Conservation and fractional surface cells	Option for running fully conservative by allowing fractional surface cells. Also option for using nearest neighbor fills to match existing NEMS Mediator behavior.	Nearest neighbor fills were used for BM1 and BM2 runs. Fractional surface cells being tested on separate branch, but has not been validated.	Yes	Fully conservative approach working in CESM but has not been tested in NEMS. Short term goal is to match existing NEMS Mediator behavior with nearest neighbor fills in place.
Nearest neighbor (NN) fills to extrapolate missing values	Currently only available for conservative remapping.	Available for conservative, bilinear, and patch remapping	Yes	Plan is to extend CMEPS for missing NN fills. However, long term these are only needed when there are land/sea mask differences, e.g., when using a data atmosphere with fixed land mask. There NN implementations are different between the two and need to be reconciled.
Flux computation	Multiple options for where to compute fluxes: (1) compute fluxes and merge by surface fraction in the Mediator or (2) send ocn state and ice flux from ice model and atmosphere (FV3) computes and merges fluxes or (3) compute atm/ocn flux in Mediator and merge with atm/ice from ice model, open water fluxes computed in the atmosphere.	NEMS currently implements option (3) from CMEPS but desire is to use option (2).	Yes	Goal is for FV3 to compute and merge fluxes except for those coming in from CICE.
Accumulation/averaging of ice fields	Provides mean(ifrac *F) to ocean.	Provides mean(ifrac)*mean(F) to the ocean.	Yes	CMEPS approach is more accurate and will be adopted into S2S system. Accumulation/averaging of ice required since ice runs at faster coupling interval than ocean.
Coupling field exchanges	All field exchanges including how to remap and merge defined in one file.	Field exchanges, remapping, merging defined throughout Mediator code.	No	
Mediator phases	Each mediator phase has its own file	All phases are included in a single large file	No	
Remapping, merging and normalization	Generic functions are provided that automatically handle remapping, merging, and normalization. These are called based on field exchange definition metadata.	Remapping, merging, and normalization are handled on a per-field bundle basis directly in specific mediator phases. These functions are not generic.	No	
History and diagnostic output	All fields output into one NetCDF file	One NetCDF file output per field	No	
NUOPC field dictionary	All standard names defined in an external YAML file	All standard names defined in NEMS Driver code	No	Community is moving towards the YAML approach, which should be easier to maintain and share