NEMS-OCEAN coupling using the NUOPC Layer

Gerhard Theurich, NESII
NEMS code with the NUOPC Layer

- NUOPC Layer infrastructure is in the NEMS trunk version.
  
  https://svnemc.ncep.noaa.gov/projects/nems/trunk/

- More recent work (and this week's work) is under the NUOPC branch:
  
  https://svnemc.ncep.noaa.gov/projects/nems/branches/NUOPC

- Compiling and running NEMS with enabled NUOPC Layer:
  - Currently the NEMS/NUOPC configuration is targeted toward Zeus.
  - Need a recent ESMF version, ESMF v6.3.0r; available as a module via:
    module use /home/Gerhard.Theurich/Modulefiles
  - Configure the NEMS source directory to use the NUOPC Layer:
    ./esmf_version nuopc_zeus
  - Build NEMS with ATM and OCN options:
    gmake nmm_gsm OCN=mom5,dummy,hycom
  - Run regression test using: (can also handle all of the build details automatically)
    ./RT.sh nuopc test
Details: NEMS code with active NUOPC Layer

- External OCN component build dependencies:
  
  src/conf/configure.nems.Zeus.intel

  # OCN settings
  # NUOPC compliant OCN models supply a standard makefile fragment

  ocndummy_mk = /home/Gerhard.Theurich/OCN-INSTALLS/DummyOCN-version-03/ocn.mk
  hycom_mk = /home/Gerhard.Theurich/OCN-INSTALLS/HYCOM-rtofs0.25cfsr__branches__NUOPC.rev32555-ESMF630r/hycom_nuopc.mk
  mom5_mk = /home/Fei.Liu/OCN-INSTALLS/20131211/mom5.mk

- Regression testing: ./RT.sh nuopc test
  - NUOPC: NMM-B pure binary input
  - NUOPC: GSM with stepable Run() driven in 12h intervals
  - NUOPC: GSM <-2h-> Mediator <-6h-> DummyOCN
  - NUOPC: GSM + HYCOM side-by-side from data with 6h intervals
  - NUOPC: GSM + MOM5 side-by-side from data with 2h intervals
GSM <->2h-> Mediator
<-6h-> DummyOCN

```plaintext
####################################
###  NEMS Configuration File  ###
####################################

# ATM #
atm_model: gsm
atm_petlist_bounds: 0 31

# OCN #
ocn_model: dummy
ocn_petlist_bounds: 32 49

# MED #
med_petlist_bounds: 50 63
med_atm_coupling_interval_sec: 7200.0
med_ocn_coupling_interval_sec: 21600.0
```
NEMS Architecture based on NUOPC
NEMS Run Sequence with NUOPC
RunSequence in NEMS EARTH

! Implement the NEMS Earth Driver run sequence, replacing the default.
call NUOPC_RunSequenceDeallocate(superIS%wrap%runSeq, rc=rc)
! add two run sequence slots: runSeq(1) and runSeq(2)
call NUOPC_RunSequenceAdd(superIS%wrap%runSeq, 2, rc=rc)
! ocn2med into slot runSeq(1)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(1), i=ocn, j=med, rc=rc)
! med (phase=2) into slot runSeq(1)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(1), i=med, phase=2, rc=rc)
! med2ocn into slot runSeq(1)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(1), i=med, j=ocn, rc=rc)
! ocn into slot runSeq(1)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(1), i=ocn, rc=rc)
! LINK slot runSeq(2) into slot runSeq(1)
call NUOPC_RunElementAddLink(superIS%wrap%runSeq(1), slot=2, rc=rc)
! med2atm into slot runSeq(2)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(2), i=med, j=atm, rc=rc)
! atm into slot runSeq(2)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(2), i=med, j=med, rc=rc)
! atm2med into slot runSeq(2)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(2), i=atm, rc=rc)
! med (phase=1) into slot runSeq(2)
call NUOPC_RunElementAddComp(superIS%wrap%runSeq(2), i=med, phase=1, rc=rc)
GSM Export Fields

! exportable fields:
call NUOPC_StateAdvertiseFields(exportState, StandardNames=(/ &
   "mean_zonal_compt_moment_flx", &
   "mean_merid_compt_moment_flx", &
   "mean_sensi_heat_flx", &
   "mean_laten_heat_flx", &
   "mean_down_lw_rad_flx", &
   "mean_down_sol_rad_flx", &
   "mean_prec_rate", &
   "inst_zonal_compt_moment_flx", &
   "inst_merid_compt_moment_flx", &
   "inst_sensi_heat_flx", &
   "inst_laten_heat_flx", &
   "inst_down_lw_rad_flx", &
   "inst_down_sol_rad_flx", &
   "inst_temp_height2m", &
   "inst_spec_humid_height2m", &
   "inst_u_wind_height10m", &
   "inst_v_wind_height10m", &
   "inst_temp_height_surface", &
   "inst_pres_height_surface", &
   "inst_surface_height" &
   
   

/), rc=rc)
• **National Unified Operational Prediction Capability**
  - Consortium of U.S. operational weather and water prediction centers.

• Participants: NOAA, Navy, Air Force, NASA, and other associated modeling groups.

• Develop a Common Model Architecture (CMA) to:
  - Improve collaboration among agencies.
  - Accelerate the transition of new technology into the operational centers.

• NUOPC websites:
  - [http://www.weather.gov/nuopc/](http://www.weather.gov/nuopc/)
    → NUOPC Layer Reference Manual + NEMS Plan Roadmap
The NUOPC Layer adds to ESMF:

- A formalism that describes and splits the phases of *initialization*. This enables the initializations of different components to interact properly when they are combined in an application.
  - Automatic connections based on field matching.
  - Data-dependent initialization between components.
  - Grid and Mesh transfer between components.
- Field matching based on an extension of the Climate and Forecast (CF) conventions.
- A formalism for checking and reporting whether component input requirements are satisfied during a run sequence.
- Code reuse through generic library code for models, drivers, mediators, and simple connects. Templates for fundamental architectural cases.
- Compliance checking.
NUOPC Layer for Developers

- Writing a NUOPC compliant ESMF component the “easy way” by using the **NUOPC Layer**:
  - Software layer implemented on top of ESMF.
  - Generic components (Model, Mediator, Connector, and Driver) to derive from; Specialize by attachable methods.
  - Utility methods for common tasks.
  - Field dictionary (standard names and units) for matching.

- Compliance checker option, as a development & debugging tool.

- Part of ESMF; NUOPC Layer reference release is now in public release **ESMF v6.3.0r**.
**Basic Building Blocks**

**Model:**
- Typically implements a specific physical domain, e.g. atmosphere, ocean, ice.

**Connector:**
- Connects pairs of components in one direction, e.g. Model to/from Model, or Model to/from Mediator.
- Executes simple transforms (Regrid or Redist).

**Mediator:**
- Used for custom coupling code (flux calculations, averaging, etc.) between Models.

**Driver:**
- Provides a harness for Models, Mediators, and Connectors.
- Coordinates initialize and run sequences.
Architectural Options

**Driver: SIMPLE**
- Model: ATM
- Model: OCN

**Driver: COUPLED WAVE**
- Model: ATM
- Model: ICE
- Model: OCN
- Model: WAVE

**Driver: INTERACTIVE ENSEMBLE**
- Model: ATM
- Model: ICE
- Model: OCN
- Model: LAND

**Model: ATM ENSEMBLE**
- Model: ATM_1
- Model: ATM_2
- Model: ATM_3
- Model: ATM_n

**c.**
- ICE
- OCN
- LND
- MED
- ATM 1
- ATM 2
- ATM 3

**d.**
- ATM
- OCN
- WAV
- ICE
- MED

**Mediator**
- Look at

module_EARTH_GRID_COMP.F90

- Set petLists for ATM, OCN, MED according to run-time configuration (nems.configure).

- Select specific ATM and OCN models according to build- (configure.nems) and run-time (nems.configure) configuration.

- Plug a specific Mediator into the EARTH Driver.

- Define the connection matrix.
Specialization of Generic Components

- Going beyond “Initialize/Run/Finalize”
- Balance the need for additional complexity (for negotiation, compatibility checking, self-description) with the resulting burden on the developer.
- Generic components with clearly defined specialization points: a well established OOP pattern: “inheritance and abstract classes”
- Developer is guided to specific aspects that need to be implemented.
- Concept of application code called by the library is the basis of ESMF components; NUOPC Layer goes one step further.
ESMF Components

Application

...:
   call CompInit()
   ...
   ...
   end

subroutine app_special()
   ...
   end subroutine

ESMF

subroutine CompInit()
   ...
   call app_special()
   ...
   end subroutine
```
subroutine CompInit()
    ... call CompInit()
    ... end
end subroutine

subroutine generic_CompInit()
    ... ! generic code, e.g. checking
    call app_CompInit_point1()
    ... ! more generic code
    call app_CompInit_point2()
    ... ! generic code, e.g. time stamp
end subroutine

subroutine app_CompInit_point1()
    ... ! special code that goes beyond generic NUOPC code
end subroutine
```
NUOPC Layer Features

- Architectural choices
  - Parametrized in a Driver component.

- Field brokering
  - Field dictionary based on CF "standard name" metadata.

- Initialize sequence
  - Different levels are implemented.
  - Field brokering between producer and consumer components.
  - Resolving data dependencies between components during initialize.

- Run sequence
  - Parametrized run sequence.
  - Fully customizable (explicit, semi-implicit, implicit schemes).
  - Component based (run components, not code directly)
  - Support for multiple time scales.