New Capabilities in MAPL

T. Clune

Atanas Trayanov, Arlindo da Silva,

Ben Auer, William Jamieson, Matt Thompson

Hamid Oloso, and Weiyuan Jiang

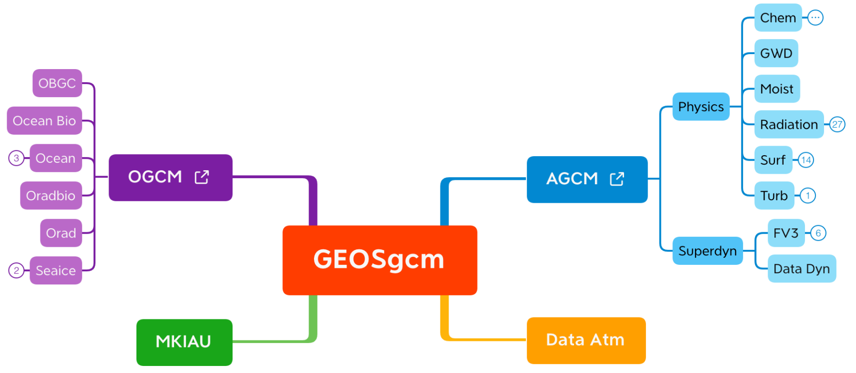
Outline

- What is MAPL
- What's new
  - mepo – holistic approach to multi-repo git
  - ESMF-enabled shortcut for hybrid MPI + OpenMP
  - MAPL-NUOPC interoperability layer
  - Provider-subscriber services
- What's next

What is MAPL

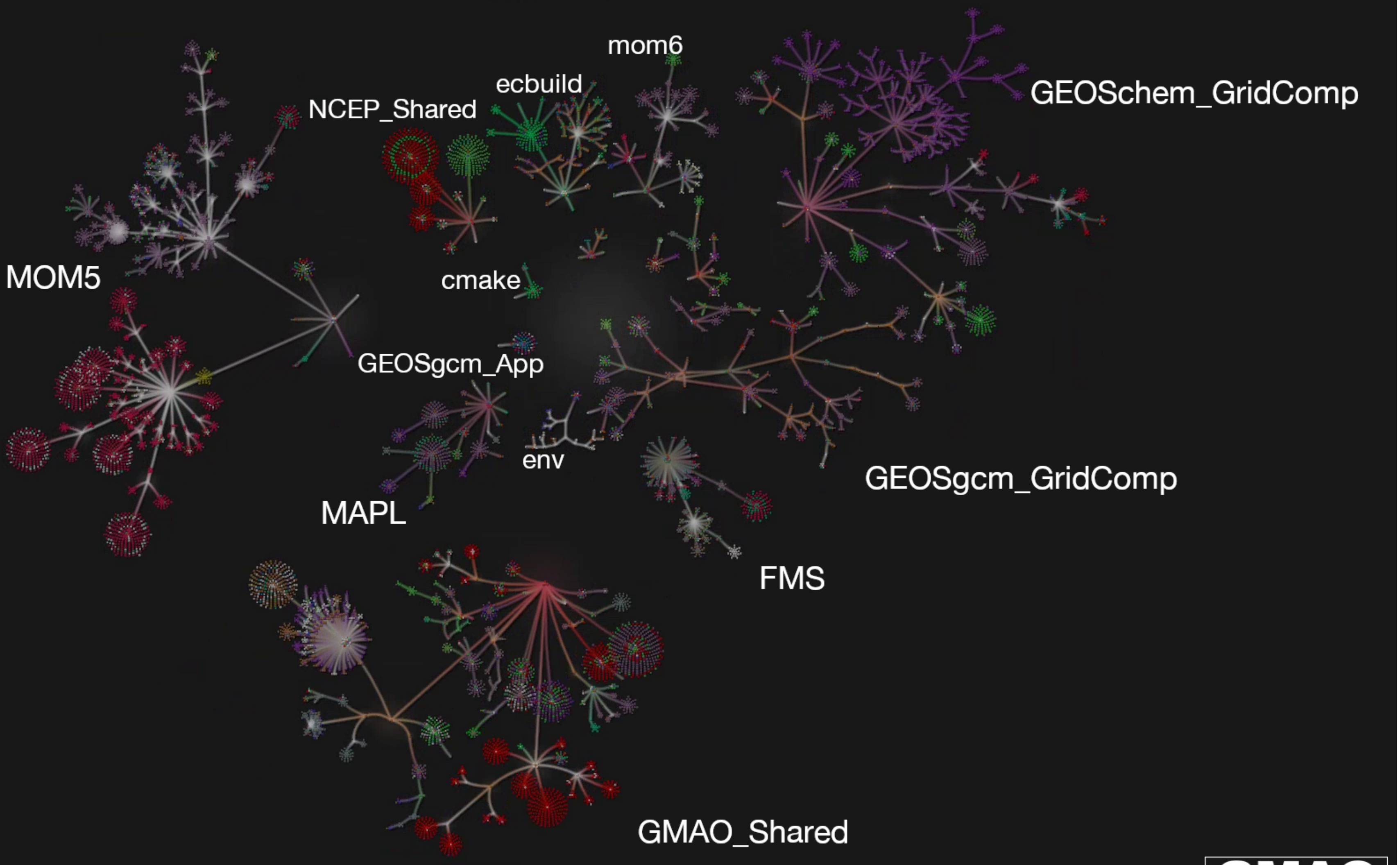
MAPL: a *usability layer* on top of ESMF to simplify the creation and use of ESMF gridded components in a hierarchical architecture

- Easy specification of import, export, and internal states
- Easy addition of child components
- Default implementation for checkpoint/restart.
- Framework standard gridded components:
  - History – run time configurable output
  - ExtData - configurable input – imports of last resort
  - Cap – drives hierarchy; manages History+ExtData
- Wrap user-specified ESMF entry points to
  - Manage instantiation of fields, esp. field connections
  - Profile (time and memory)
  - Insert couplers (when needed)
- Enforce conventions



<u>Similar motivations have led to NUOPC but unfortunately the details diverged.</u>

2020-09-15



GEOS Model Evolution

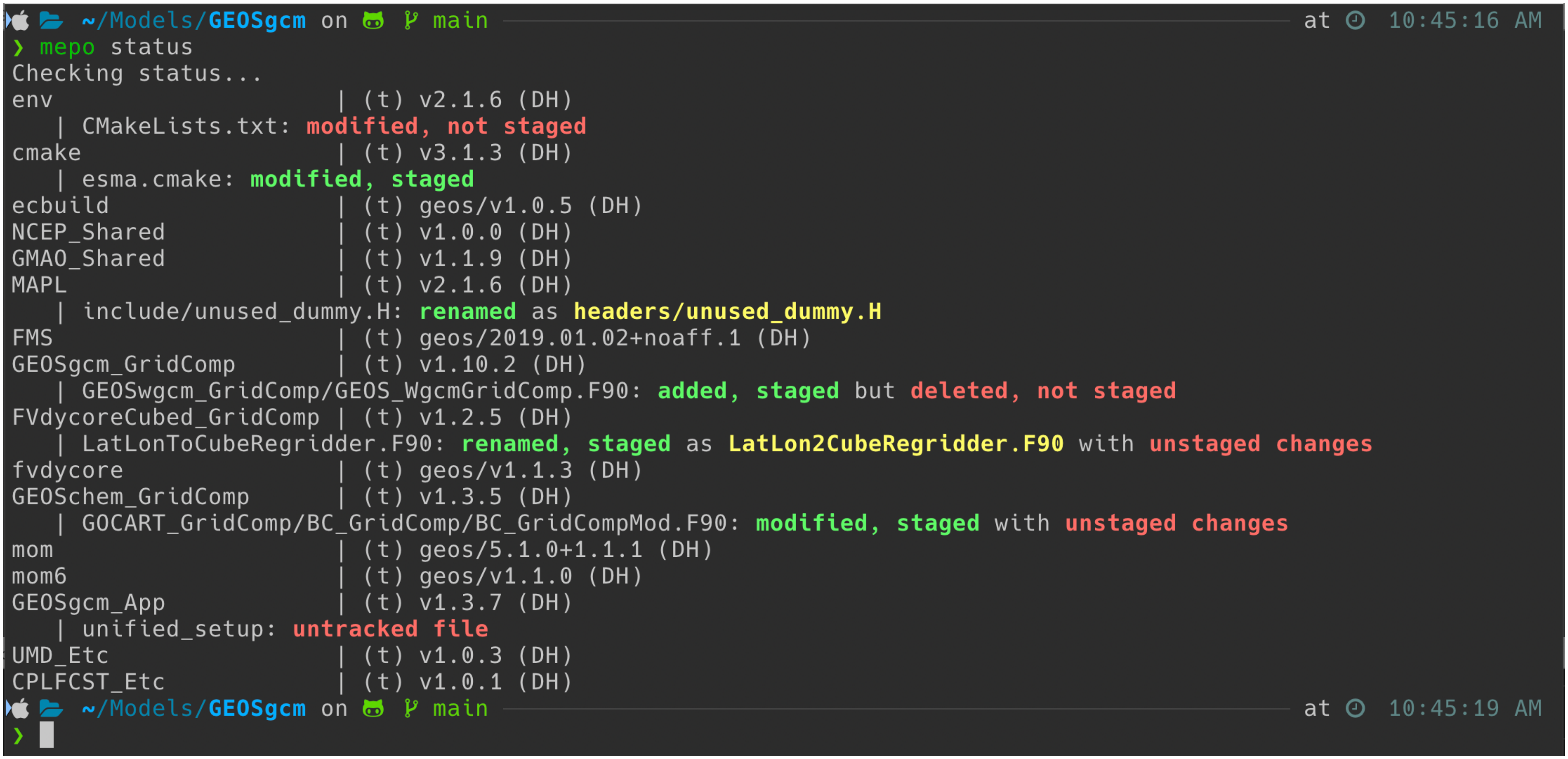
GMAO

mepo

- **Driver**: Simplify routine development with nested Git repositories
- **Available technologies**:
  - Git submodules
  - Git subtree
  - Manage Externals
- **Solution**: Extensible Python package **mepo** (https://github.com/GEOS-ESM/mepo)
  - Typical syntax: `$ mepo <git-command> <git-options> <repo-list>`
- Currently supported commands

- clone
- status
- diff
- fetch
- checkout
- branch
- commit
- push
- init
- stage
- tag
- stash
- *list*
- *checkout-if-exists*
- *develop*
- *compare*
- *whereis*
- *stage*
- *unstage*
- *save*
- *restore-state*
- *fetch-all*
- *pull-all*

M. Thompson, T. Clune,
P. Chakraborty, & A. da Silva



```
~/Models/GEOSgcm on  main ──────────────────────────── at  10:45:16 AM
❯ mepo status
Checking status...
env                    | (t) v2.1.6 (DH)
   | CMakeLists.txt: modified, not staged
cmake                  | (t) v3.1.3 (DH)
   | esma.cmake: modified, staged
ecbuild                | (t) geos/v1.0.5 (DH)
NCEP_Shared            | (t) v1.0.0 (DH)
GMAO_Shared            | (t) v1.1.9 (DH)
MAPL                   | (t) v2.1.6 (DH)
   | include/unused_dummy.H: renamed as headers/unused_dummy.H
FMS                    | (t) geos/2019.01.02+noaff.1 (DH)
GEOSgcm_GridComp       | (t) v1.10.2 (DH)
   | GEOSwgcm_GridComp/GEOS_WgcmGridComp.F90: added, staged but deleted, not staged
FVdycoreCubed_GridComp | (t) v1.2.5 (DH)
   | LatLonToCubeRegridder.F90: renamed, staged as LatLon2CubeRegridder.F90 with unstaged changes
fvdycore               | (t) geos/v1.1.3 (DH)
GEOSchem_GridComp      | (t) v1.3.5 (DH)
   | GOCART_GridComp/BC_GridComp/BC_GridCompMod.F90: modified, staged with unstaged changes
mom                    | (t) geos/5.1.0+1.1.1 (DH)
mom6                   | (t) geos/v1.1.0 (DH)
GEOSgcm_App            | (t) v1.3.7 (DH)
   | unified_setup: untracked file
UMD_Etc                | (t) v1.0.3 (DH)
CPLFCST_Etc            | (t) v1.0.1 (DH)
~/Models/GEOSgcm on  main ──────────────────────────── at  10:45:19 AM
❯
```

mepo (cont’d)

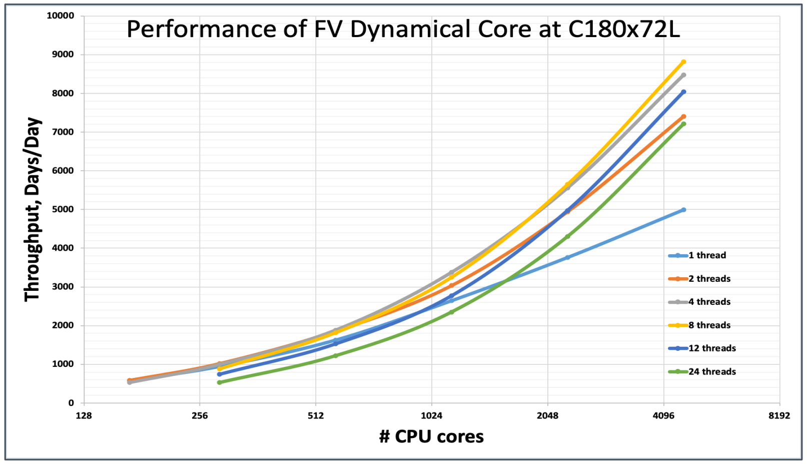
- Top level YAML config file
  - Mount point in src tree
  - URL of repo to clone
  - Tag/branch
  - Optional path to sparse checkout
  - Name of “develop” branch (for GitFlow)
- Caveats
  - Single config – not nested (design tradeoff)
  - Just Git; no SVN (or CVS)

```
GMAO_Shared:
  local: ./src/Shared/@GMAO_Shared
  remote: ../GMAO_Shared.git
  tag: v1.1.8
  sparse: ./config/GMAO_Shared.sparse
  develop: main

MAPL:
  local: ./src/Shared/@MAPL
  remote: ../MAPL.git
  tag: v2.1.6
  develop: develop

FMS:
  local: ./src/Shared/@FMS
  remote: ../FMS.git
  tag: geos/2019.01.02+noaff.1
  develop: geos/release/2019.01
```

ESMF-enabled MPI+OpenMP

Driver:

1. Hybrid MPI+OpenMPI improves perf of FV core by up to 50% at extreme scales.
2. Legacy parameterizations not (properly) instrumented with OpenMP.



Solution: Use ESMF to couple pure-MPI physics with hybrid FV.

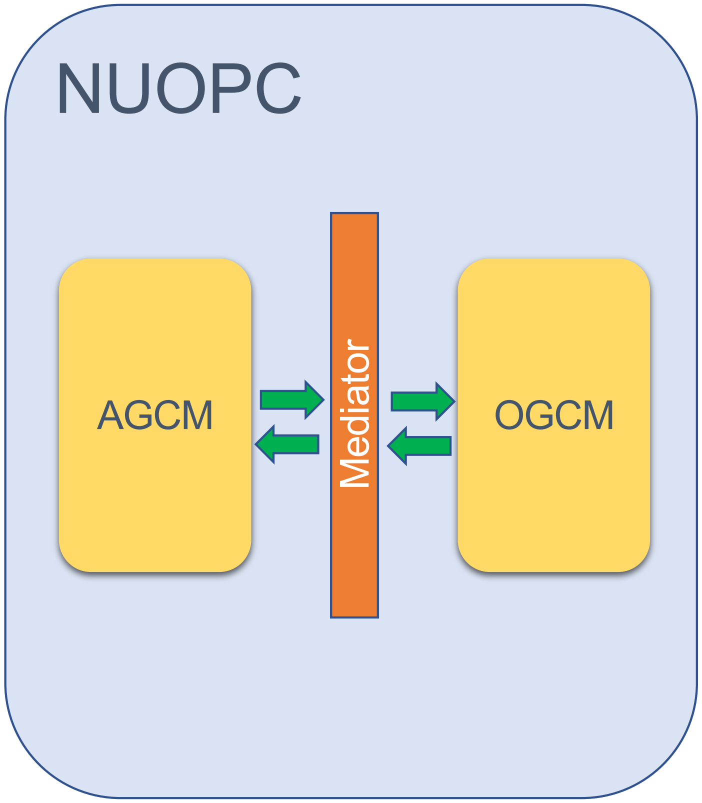
- ESMF idles subset of PETs; FV launches OMP threads during
- Data transferred using ESMF shared-image Field data storage
  - Low overhead – copy required due to halos anyway

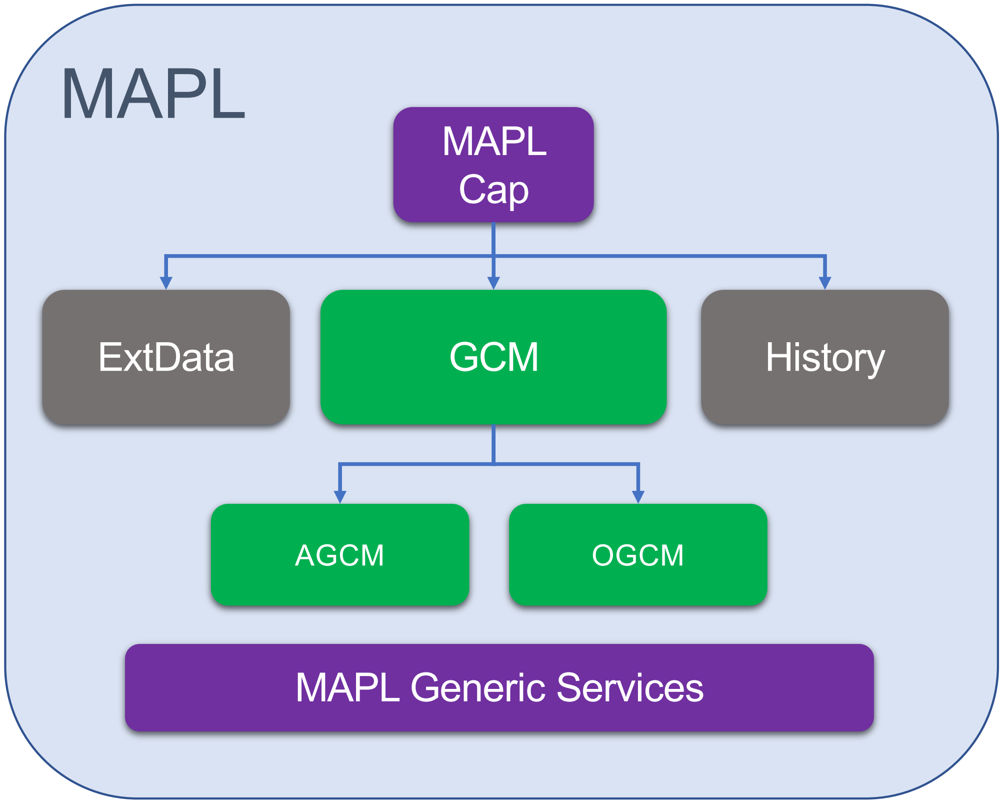
Results encouraging on synthetic example

Investigating why pinning seems to go wonky with full use case.

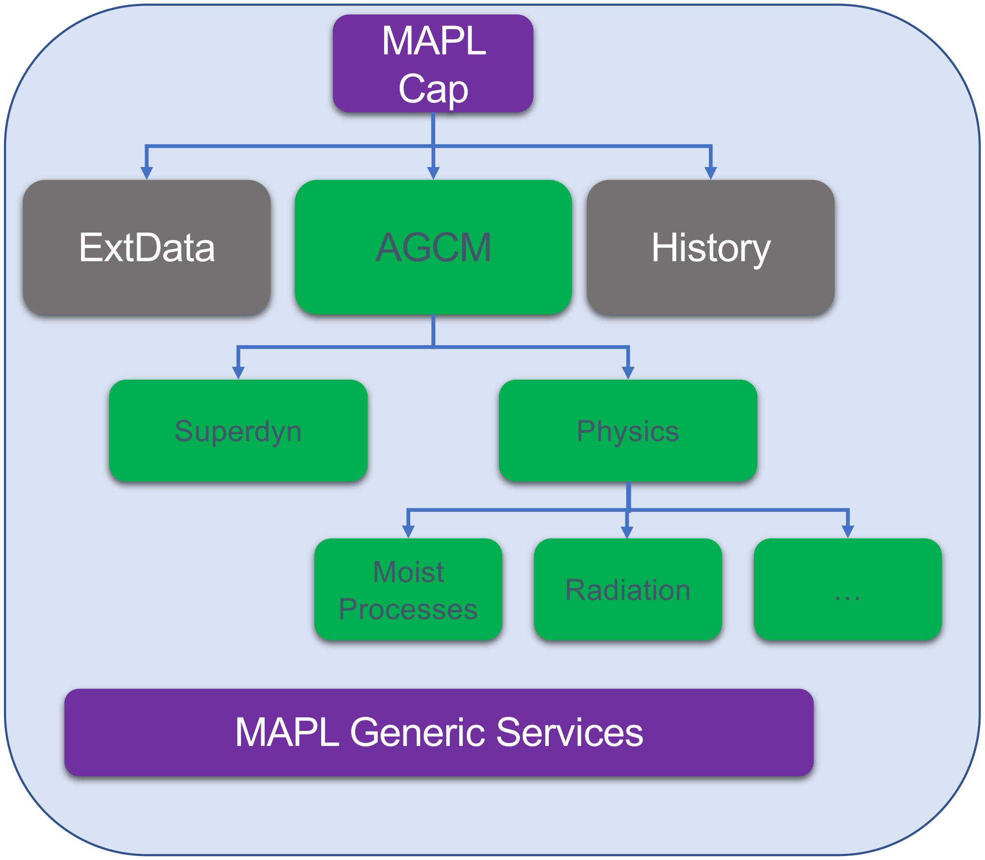
A. Oloso, T.Clune, & G. Theurich

NUOPC – MAPL Interoperability





NUOPC –MAPL Interoperability (cont'd)

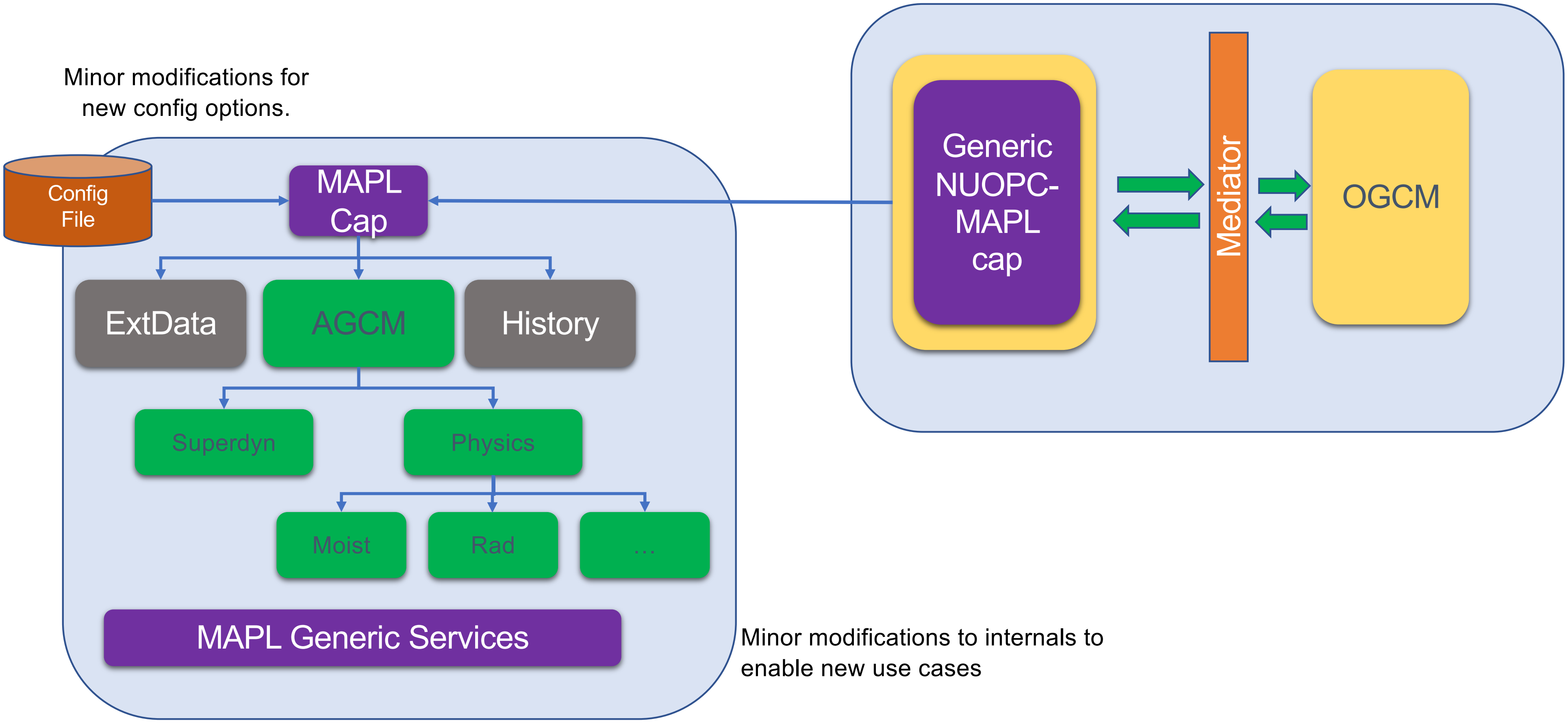


Drivers:

- MAPL component used in NUOPC architecture
  - E.g. GOCART aerosols exported to NOAA UFS
- Exploiting unique NUOPC capabilities
  - E.g. concurrent execution of MAPL components

NUOPC –MAPL Interoperability (cont’d)

Minor modifications for new config options.



Minor modifications to internals to enable new use cases

Provider – Subscriber Services

Driver: Consider a simple implementation of advection of chemistry tracers

1. Chemistry tracers passed as imports to dynamics
2. Advection tendencies passed as imports to chemistry

Can require > 150 3D arrays to store tendencies!

Concept: Allow components to “outsource” services as a mechanism to save memory

- Subscriber components (e.g. chemistry,) give explicit permission for state to be modified
- Provider components (e.g., DynCore) directly update bundle of tracers

Legacy mechanism:

- Little direct support from MAPL
- Some logic appears in otherwise uninvolved components (brokers)
- Somewhat inflexible/hardwired
- Fragile

A. Trayanov, T. Clune, & A. da Silva

Provider – Subscriber Services (cont'd)

New approach: Generalize existing MAPL "data service"

- ➢ MAPL provides high-level interface – simple to use/understand
- ➢ Elevates services to a component-level interface
- ➢ Flexible – e.g., send some tracers to DynCore and others to AdvecCore
- ➢ Applicable to many processes beyond advection
- ➢ Can detect unsatisfied service requests

Pseudo code:

1. "provider" component advertises service

   ```
   call advertise_service(self, 'advection', bundle_name)
   ```

2. "subscriber" component requests service with label and list of field names

   ```
   call subscribe_to_service(self, 'advection', my_fields)
   ```

3. common ancestor component specifies connections: {service, provider, subscriber}

   ```
   call add_connection('advection', provider=child_a, subscriber=child_b)
   ```

4. Under-the-hood: MAPL combines subscriber bundles into single import bundle for provider

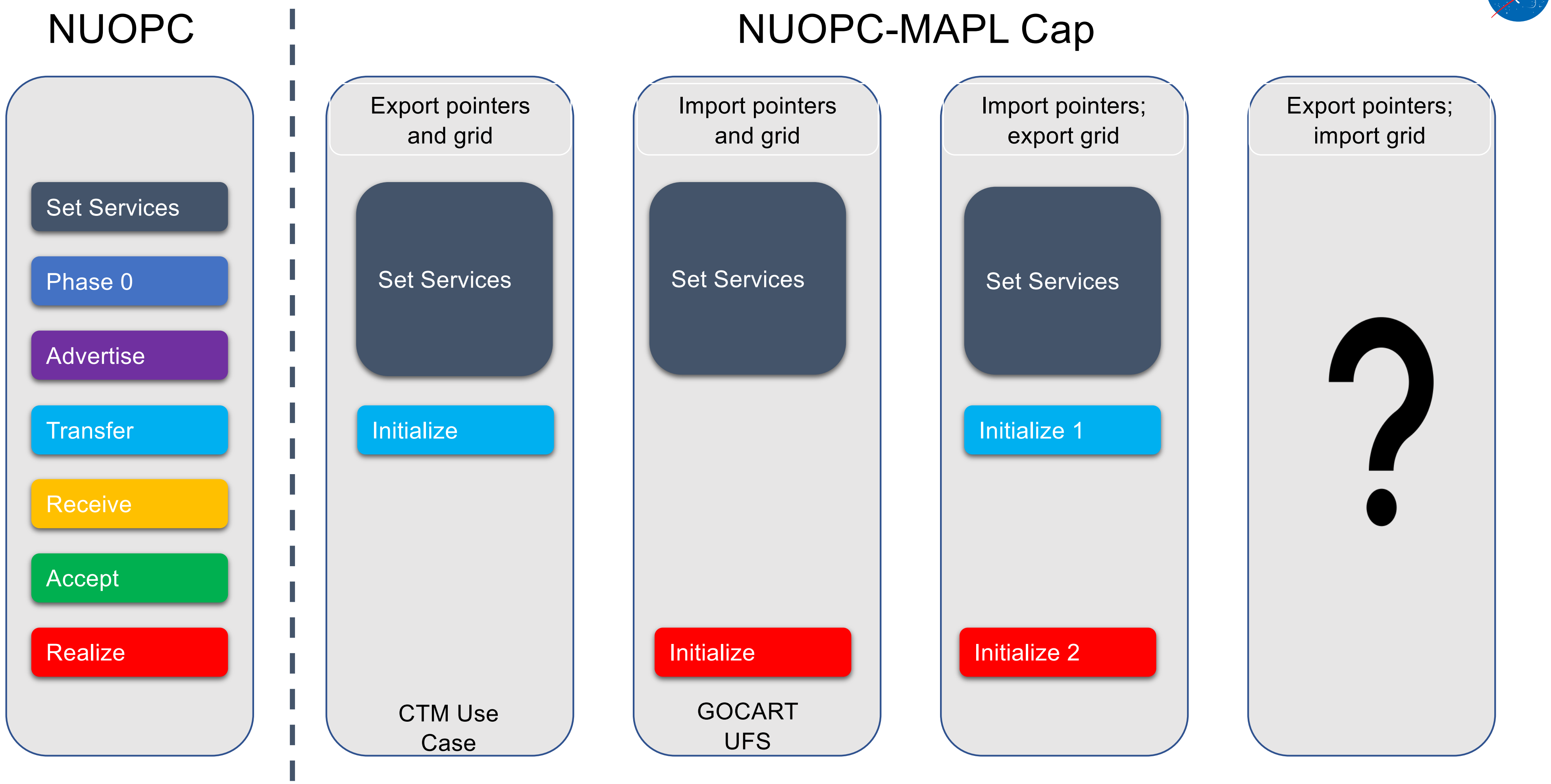
A. Trayanov, T. Clune, & A. da Silva

What's Next?

- Major refactoring of MAPL core layers underway
  - Clean OO design and comprehensive unit tests
  - Generalize a few increasingly inconvenient low-level assumptions
  - Switch from static to dynamic libraries
    - Enhanced run-time configurability
    - Eliminate dependencies between components (loose coupling)
- Improved interoperability with NUOPC
  - Either next-gen MAPL components are NUOPC caps, *or*
  - Next-gen MAPL components have factory method to generate a NUOPC cap
- Share ExtData and History gridded components with community
  - Establish comprehensive tests and refactor code base
  - Enable server-side ESMF regridding
  - Produce thorough tutorials/examples/documentation

Questions?

Extra materials



NUOPC

- Set Services
- Phase 0
- Advertise
- Transfer
- Receive
- Accept
- Realize

NUOPC-MAPL Cap

Export pointers and grid

- Set Services
- Initialize

CTM Use Case

Import pointers and grid

- Set Services
- Initialize

GOCART UFS

Import pointers; export grid

- Set Services
- Initialize 1
- Initialize 2

Export pointers; import grid

?

MAPL NUOPC Compatibility Layer

Drivers:

1. Concurrent execution of MAPL components – e.g. CTM driven by GCM
2. Run MAPL components within a NUOPC architecture – e.g. GOCART aerosols inside NOAA UFS

Major issues:

- Conflicts between initialization phasing
- Grid sharing
- Data pointer sharing (performance)

Near term solution – Develop a generic NUOPC-MAPL Cap

- New top-level config options:
  - Specify which fields are to imported/exported (using “standard” names).
  - Specifier for whether pointer is shared and/or which side allocates
- Extend MAPL VarSpec to include new config options
- Extend MAPL internal logic to
  - handle variant allocation cases
  - receive externally specified grid

W. Jamieson, T. Clune, & A. da Silva