HEMCO restructuring and coupling with CESM2

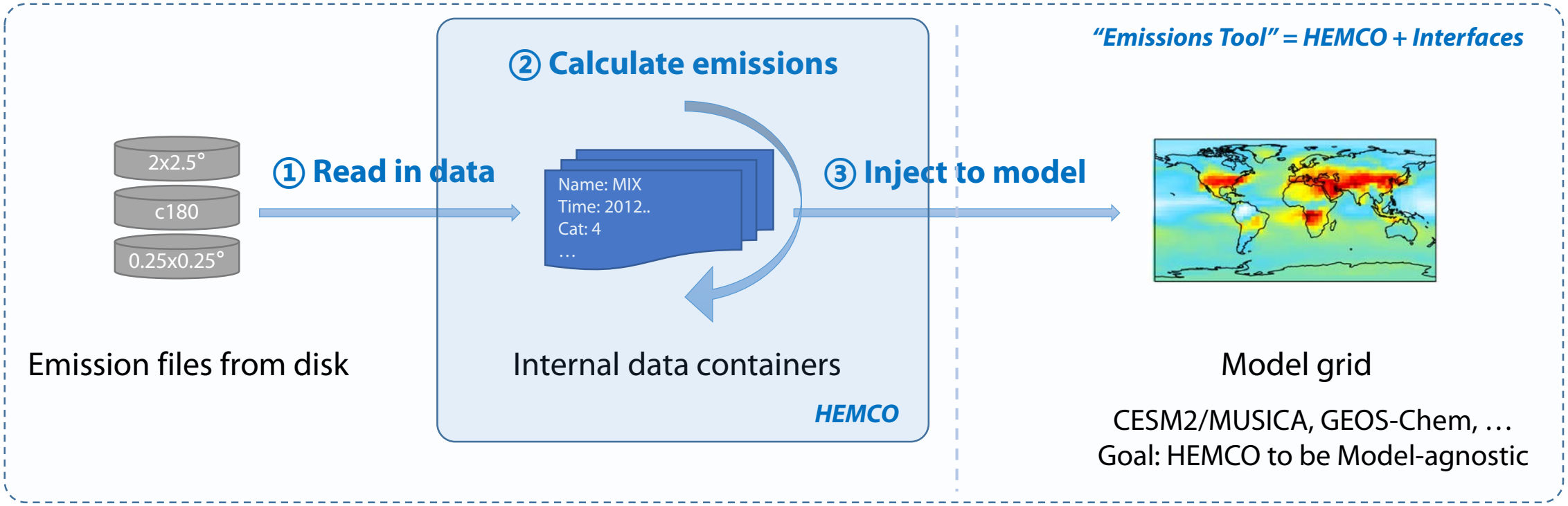
GEOS-Chem NCAR Visit

Haipeng Lin

Jan 8-9, 2020

Defining "HEMCO"

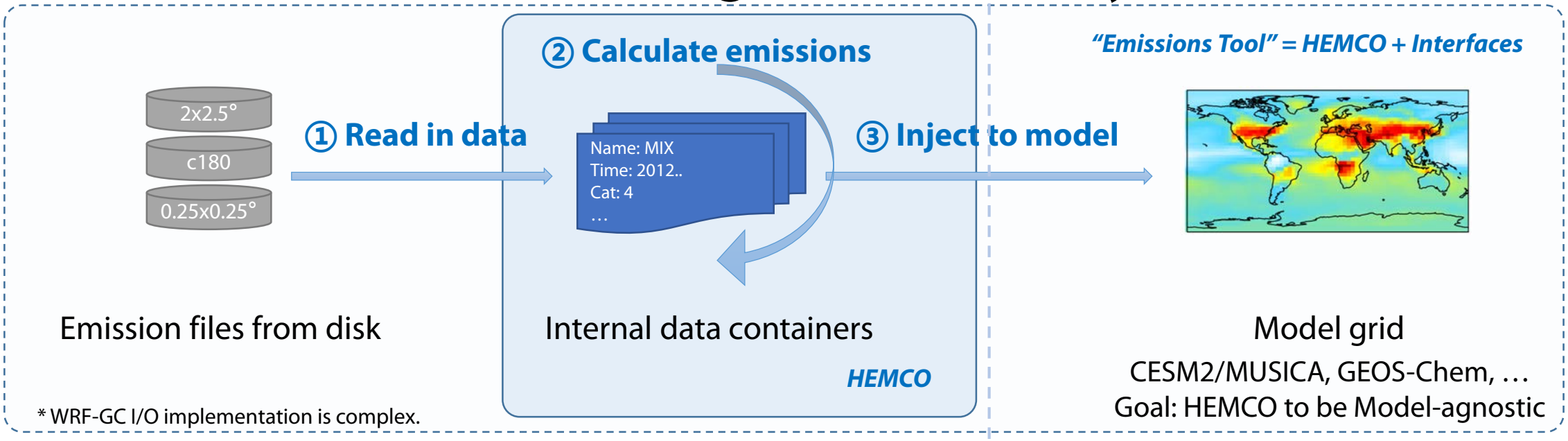
Any emissions tool needs to complete three tasks:



HEMCO will be the "emissions" component.
Interface with different IO/Regrid components (provided by each separate model)*

"Model" figure via GEOS-Chem benchmark

How do these three stages look today?

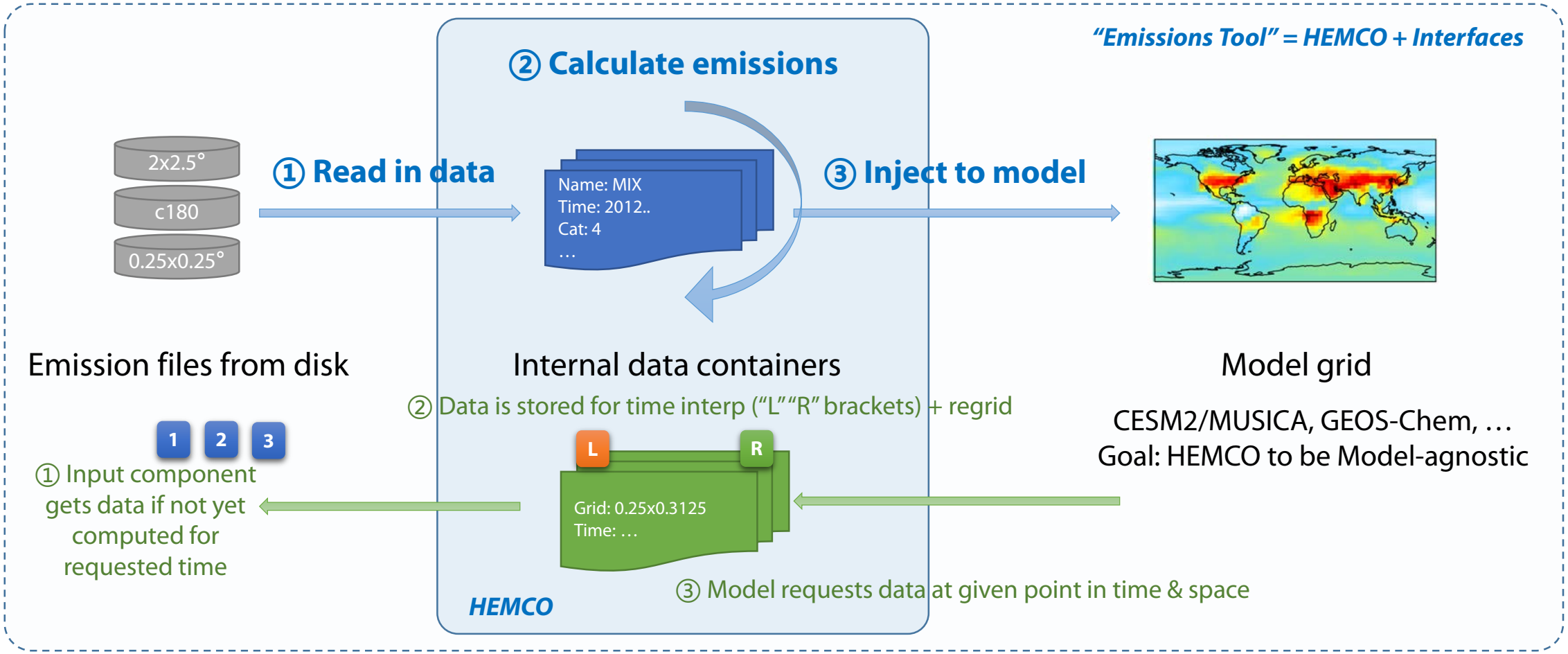


	Model	① I/O	② Emissions	③ Model interface / speciation
1	GC-Classic	NcdfUtil + A2A/MESSy Regrid	**HEMCO** Same code base for all models	HCO_GC_Main_Mod HEMCO + GEOS-Chem Interface
1	WRF-GC *			
2	GCHP	MAPL *ExtData*		HEMCO_GridComp in MAPL
2	GEOS-5			
	CESM2(-GC)	CESM2+ESMF		Two-stage implementation: (1) with GC, (2) generic

Time abstraction (brackets)

Blue: *Data flow*
Green: *Code flow*

Abstraction in time and space – model does not need to know indices!



Challenges and requirements

- Key goal: on-line emissions tool with support for many formats
  - Input: gridded, point, airplane, 2D, 3D, … *(* only gridded for now, see discussion for others)*
  - Output: (maybe adaptive) model grid.
- Isolate code that is specific to how HEMCO interfaces with the model
  - i.e. I/O, regrid (either as part of input which is in different grids & injection to model grid)
  - & data "injection" phase (interface with the atmospheric model)
  - Includes isolating frameworks (ESMF, MAPL…) to reduce external dependencies
- Foster collaboration through unified HEMCO code-base for all models

  Two-stage restructuring:
  - Build upon GEOS-Chem Classic first; CESM2-GC will be able to use GC-Classic HEMCO interfaces in a limited extent during transition period, for early science validation.
  - After standalone CESM2 I/O+Regrid interface built, can use those for model-agnostic emissions in CESM2

Intermediate grid issue

- Problem: (1) Multiple, adaptive grids; (2) process nonlinearity; (3) scaling & masking at higher resolutions
- Previous assumption: HEMCO operates on the **model grid**.
- Proposed solution: *Optional* intermediate grid becomes the new "model grid" - all HEMCO operations are performed on intermediate and re-gridded to model grid(s) at output

Key design characteristics

One unified interface for incorporating emissions in a ESM

→ Clear separation of duties: I/O, regrid, "emissions", data containers and flow

Two regrids with *intermediate grid* functionality

→ An *optionally hi-res* grid for intermediate emissions calculations

Three layers enabling independent development and fast dev cycle

→ Separating *model input, emissions and model output* layers, which can be swapped easily depending on model

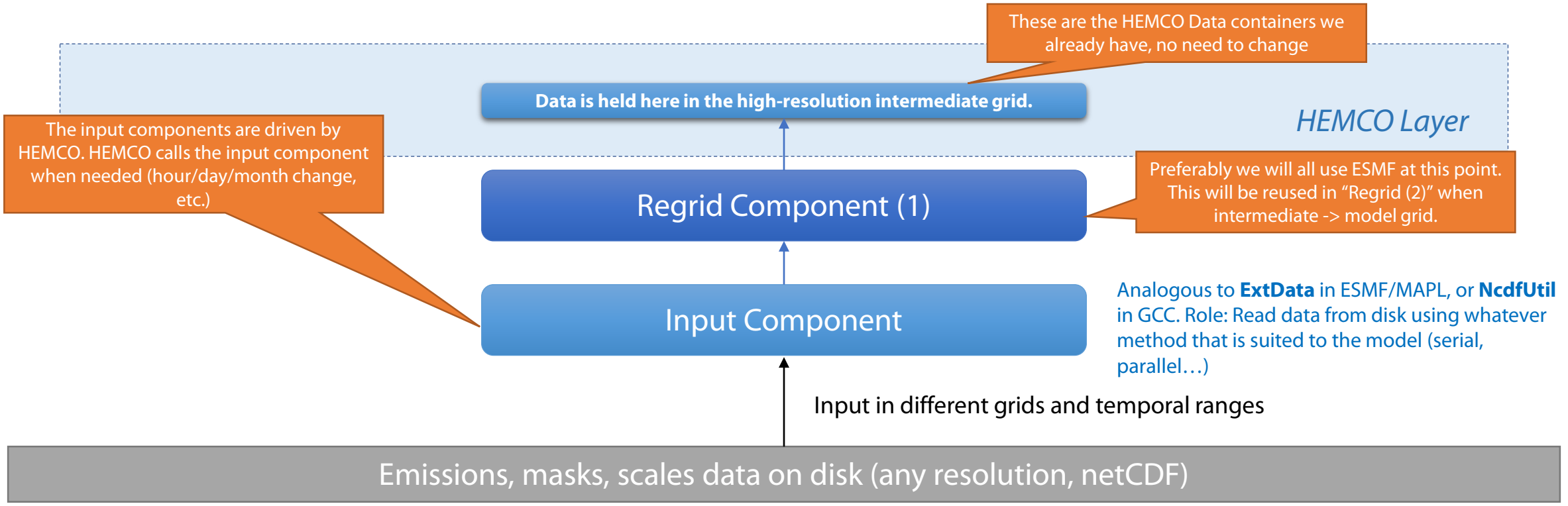
→ The "emissions" layer constitutes what we call "HEMCO"; everything else are coupling structures that are model-specific

Technical Overview: Interfaces

Layer 1: from IO to HEMCO

How it works:

- All timesteps are driven by the parent model. Parent model "heartbeat" -> call HEMCO -> calls the IO+Regrid layer below, if data update is needed.
- HEMCO calls input component to read data → regrid to intermediate grid → **data is stored in HEMCO memory, not IO component**.
- **In GC-Classic, the input component would be a wrapper of NcdfUtil instead of CESM2 stuff.**

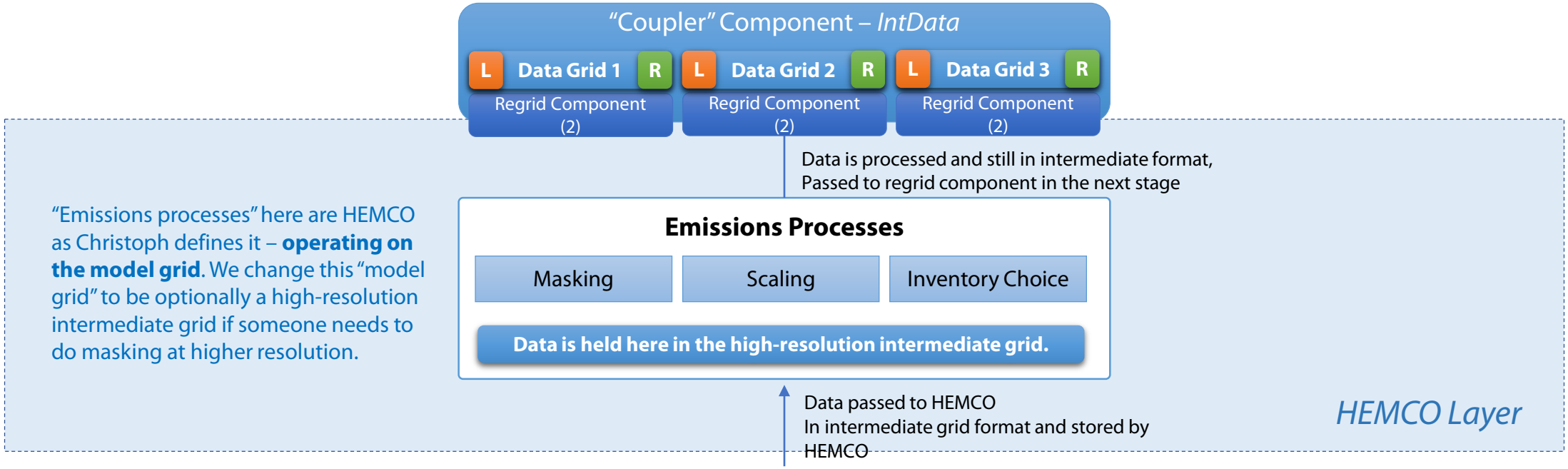


Technical Overview: Interfaces

Layer 2: Changes needed inside HEMCO

How it works:

- At this point everything is on a **intermediate grid,** which is the new "model grid" for HEMCO. Business as usual.
- **All scaled, masked, … emissions data is written through a regrid to a new "coupler component", which will hold the data in model grid.** (Talk about this in the next slide)
- Only need to do two main changes:
  - HCOIO_Read_Std_Mod, etc. will need to implement one single module format ("HCOIO_Read_Mod"), which will call different input components as shown in the previous slide. Cmake will decide which to compile.
  - **Add this new "coupler" component, which is essentially a new interface to HEMCO.**

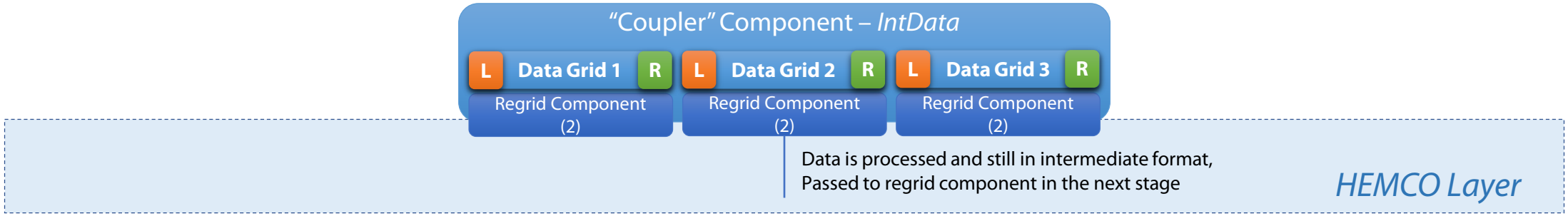


Technical Overview: Interfaces

Layer 3: the new "coupler" component, or IntData, or "The Thing" (credit: Sebastian Eastham)

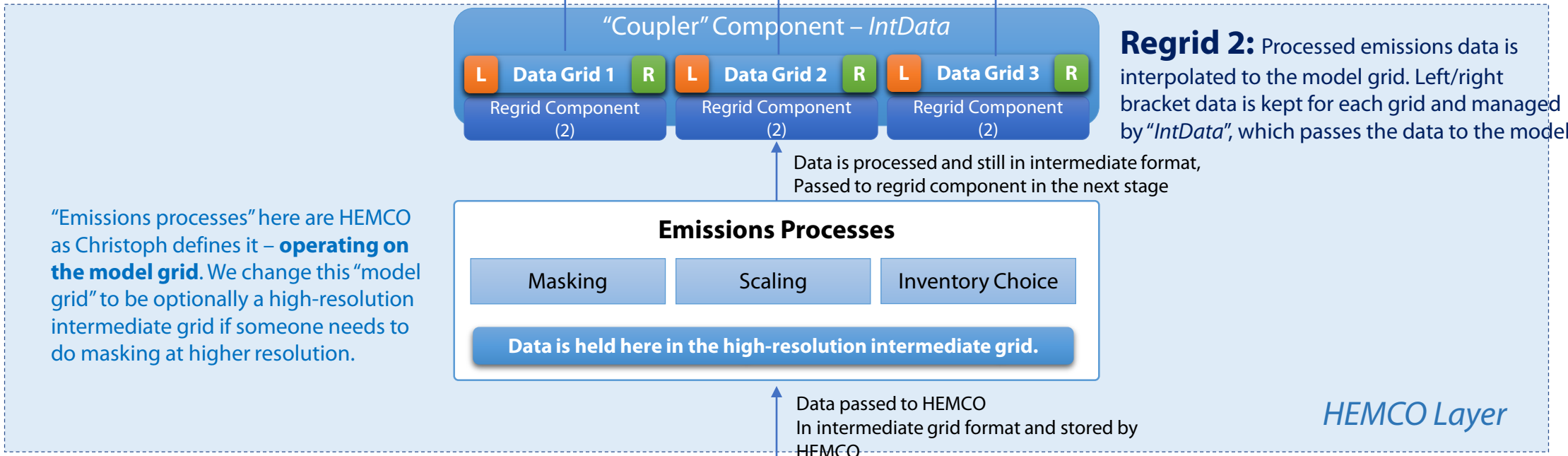
How it works:

- At this point everything has been processed by HEMCO on the "intermediate grid".
- The coupler component **will hold a copy of the data in model grid**, through a regrid.
  - This regrid component is ideally the CESM2 ESMF-based regridder.
- The model will pull data from this "coupler" component.
- In code workflow:
  - `Model decides its time to do emissions?`
    - `Yes – calls coupler component to fetch data for specified time.`
    - `Does specified time exist?`
      - `No – call HEMCO`
        - `HEMCO calls "IO" and "Regrid (1)" to pull data from disk to intermediate grid, via model (Layer 1)`
        - `HEMCO does calculations and returns data in intermediate grid (Layer 2)`
        - `"Coupler" regrids intermediate grid to model grid (Layer 3)`
      - `Yes – Just return the data from IntData (Layer 3)`



High-level overview
Data flow: Bottom to top

Data leaves "HEMCO" in the grid that the model operates on, which may be multiple grids. If not just ignore the below three horizontal boxes and consider them as one.



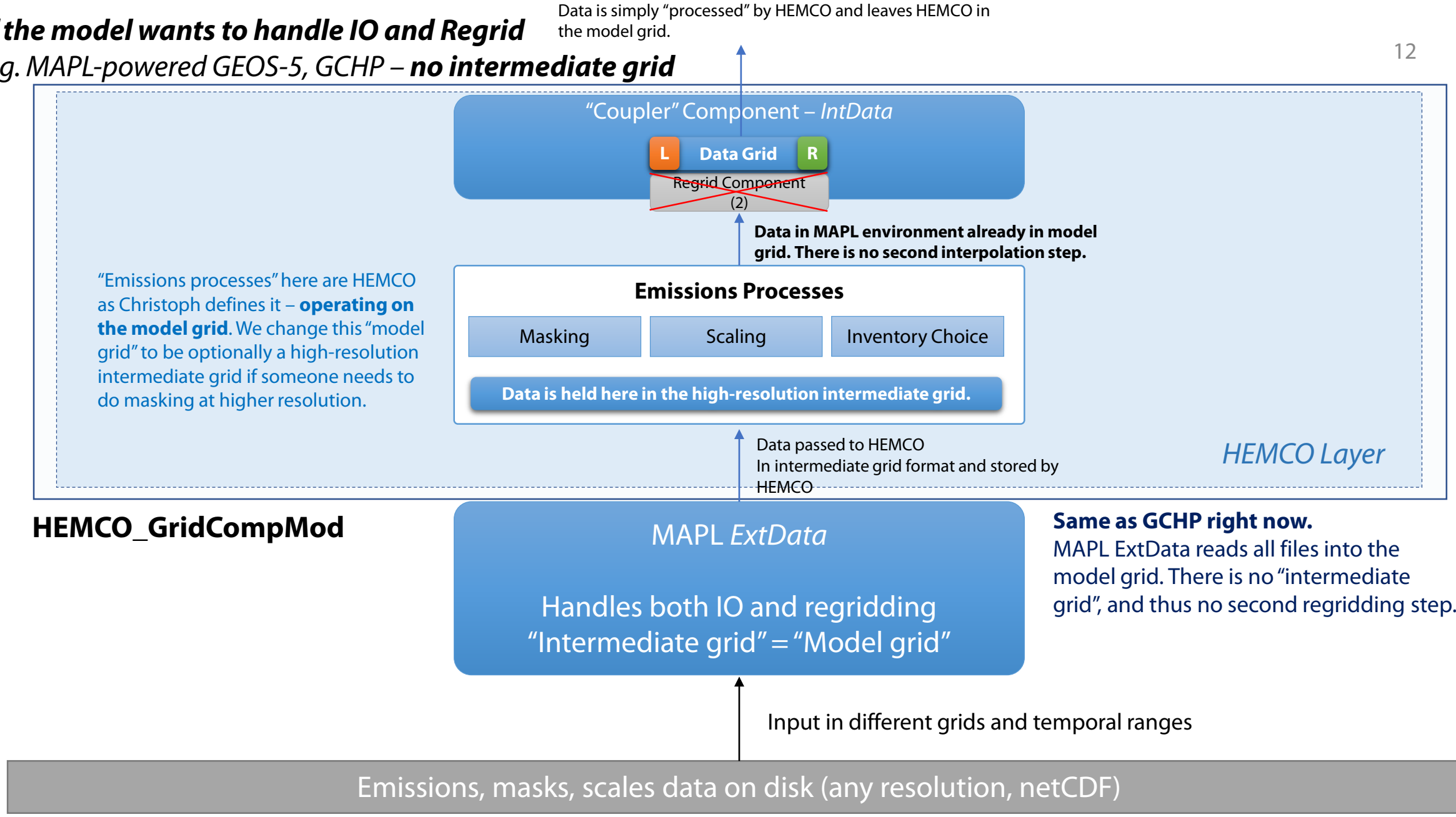
Regrid 2: Processed emissions data is interpolated to the model grid. Left/right bracket data is kept for each grid and managed by "*IntData*", which passes the data to the model.

"Emissions processes" here are HEMCO as Christoph defines it – **operating on the model grid**. We change this "model grid" to be optionally a high-resolution intermediate grid if someone needs to do masking at higher resolution.

Regrid 1: Regrid from input to unified grid, at user-defined "**intermediate resolution**". Can be model resolution if user has no special needs.

Analogous to **ExtData** in ESMF/MAPL, or **NcdfUtil** in GCC. Role: Read data from disk using whatever method that is suited to the model (serial, parallel…)

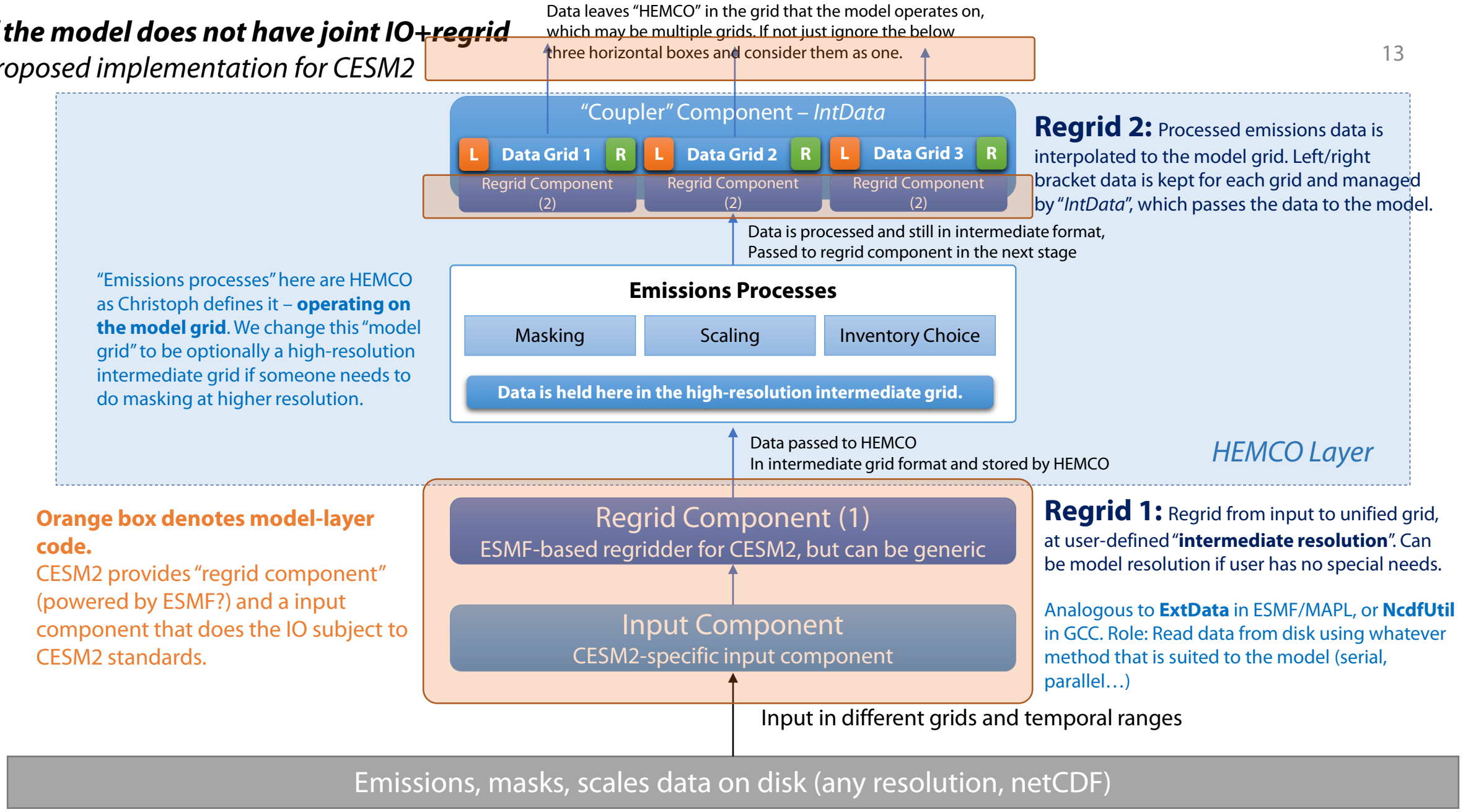
If the model wants to handle IO and Regrid
e.g. MAPL-powered GEOS-5, GCHP – ***no intermediate grid***



Same as GCHP right now.
MAPL ExtData reads all files into the model grid. There is no "intermediate grid", and thus no second regridding step.

If the model does not have joint IO+regrid

Proposed implementation for CESM2



Model specific items to coordinate with CESM2 team

1) **Grid specifications:** How to talk to HEMCO about time, grid (convert from CESM2 to HEMCO)
2) **ESMF/CESM2 Regrid Component**
3) **Design of *IntData:*** How is intermediate grid data regridded and stored in *IntData* (array of model grid-sized tendency containers)
   - Can probably use GEOS-Chem interface, if implementing only GC-CESM2-HEMCO; CESM2-HEMCO probably requires more fiddling with this interface
4) **CESM2 IO Component**: A single interface for IO, *that is not model-grid specific*, needs to be designed to fit within CESM2 architecture.
   - If we don't care about parallel I/O, can use transitional solution "hacked" based on NcdfUtil; it's not bad, even on HPC, from personal experience*

Expected timeline & working items

- **Now:** Confirming requirements, standalone HEMCO repository, restructuring groundwork
- **Phase 1:** Restructuring done within GC-Classic; isolate IO (NcdfUtil), Regrid (Map_A2A, MESSy), change directory structure & compile routines; **alter data flow to follow schematic, implementation of *IntData* container.**

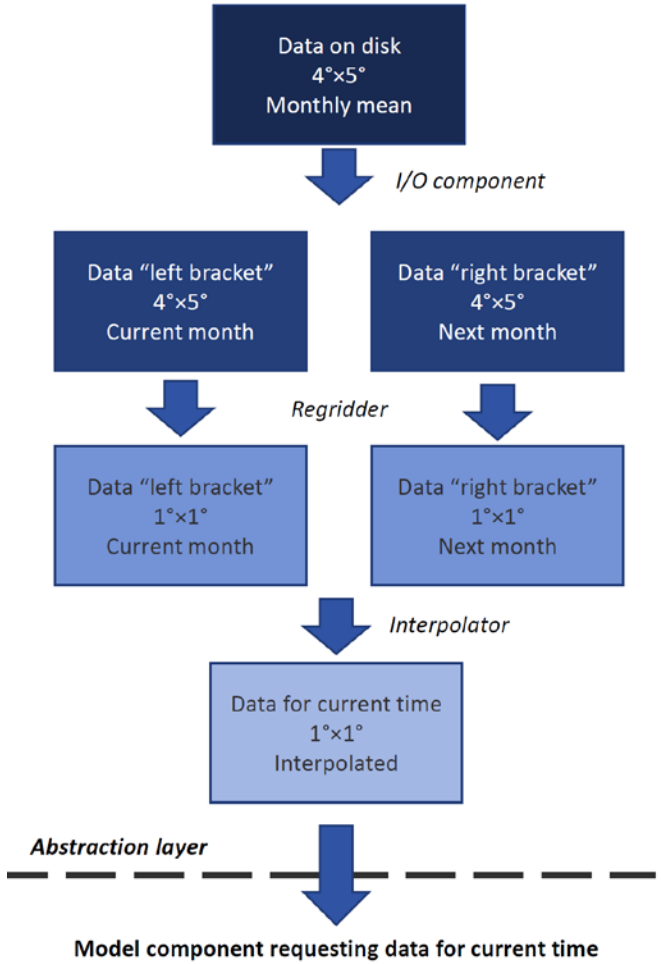
  (Work with GCHP should be fairly trivial, basically stubs)
- **Phase 2a:** Adapt *IntData* and model interface to CESM2 using prescribed emissions (no regrid) to test data flow
- **Phase 2b:** Once CESM2-ESMF Regrid tool complete, incorporate into "HEMCO"
- **Phase 3:** Validate…

Discussion items

- Intermediate grid issue (link)
- ESMF regridding tool in CESM2
  - Can it handle all NCAR use cases for 2-D?
  - 3-D?
  - Working schedule
- Where does HEMCO sit architecturally?
- Parallelization / MPI
  - Proposed: Concurrent I/O first; parallel I/O needs serious thought later

Backup slides & diagrams

“Brackets” (Seb Eastham, Feb 2019)



Proposal objectives