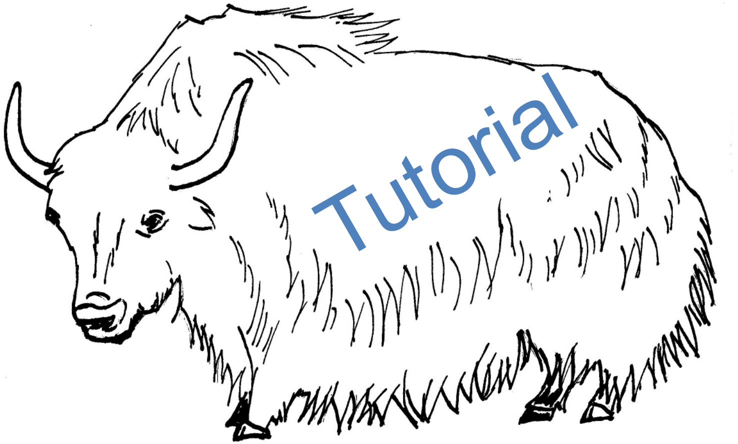
Yet Another Coupler – YAC

Version 2.0.0 – Jan 2021



Tutorial

Contact: Moritz Hanke (DKRZ)
René Redler (MPI-M)

Max-Planck-Institut
für Meteorologie

DKRZ
DEUTSCHES
KLIMARECHENZENTRUM

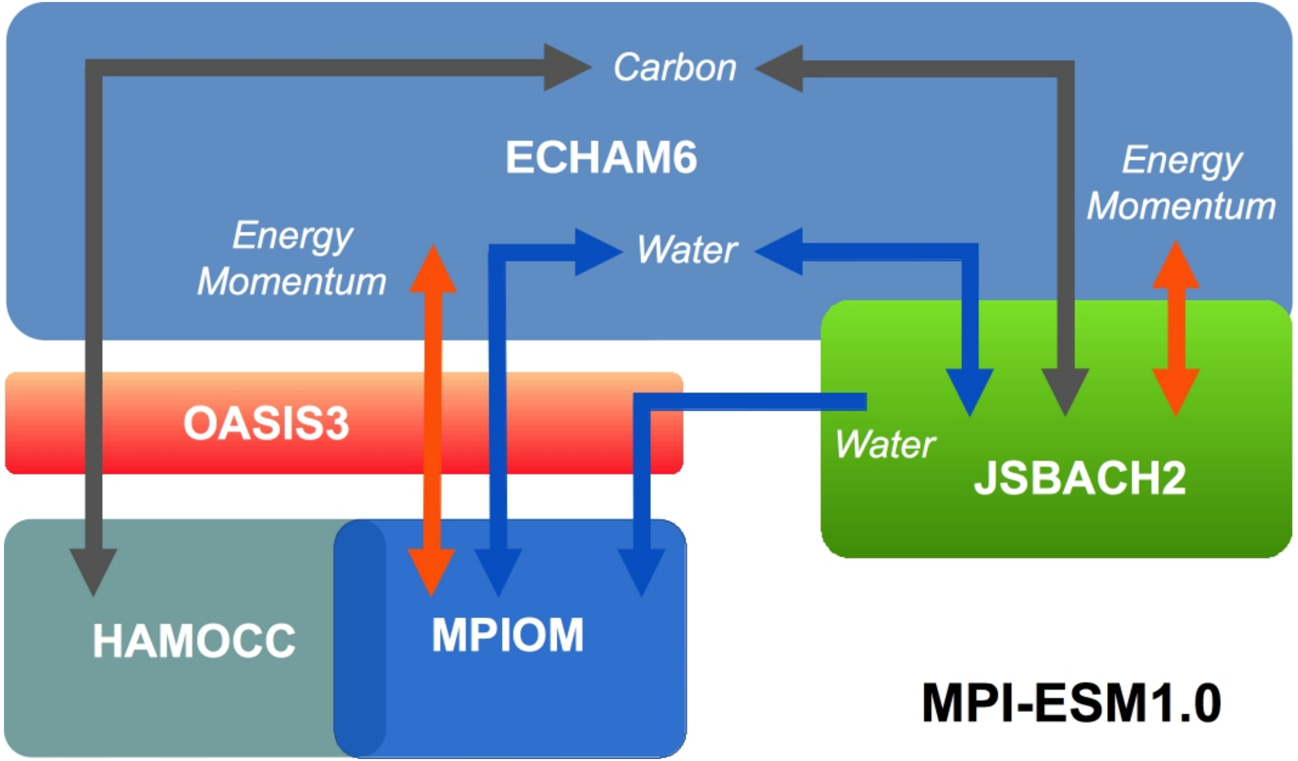
YAC – Team

Moritz Hanke (DKRZ)
René Redler (MPI-M)
Teresa Holfeld (MPI-M, student assistant)
Maxim Yastremsky (MPI-M, student assistant)

With contributions from

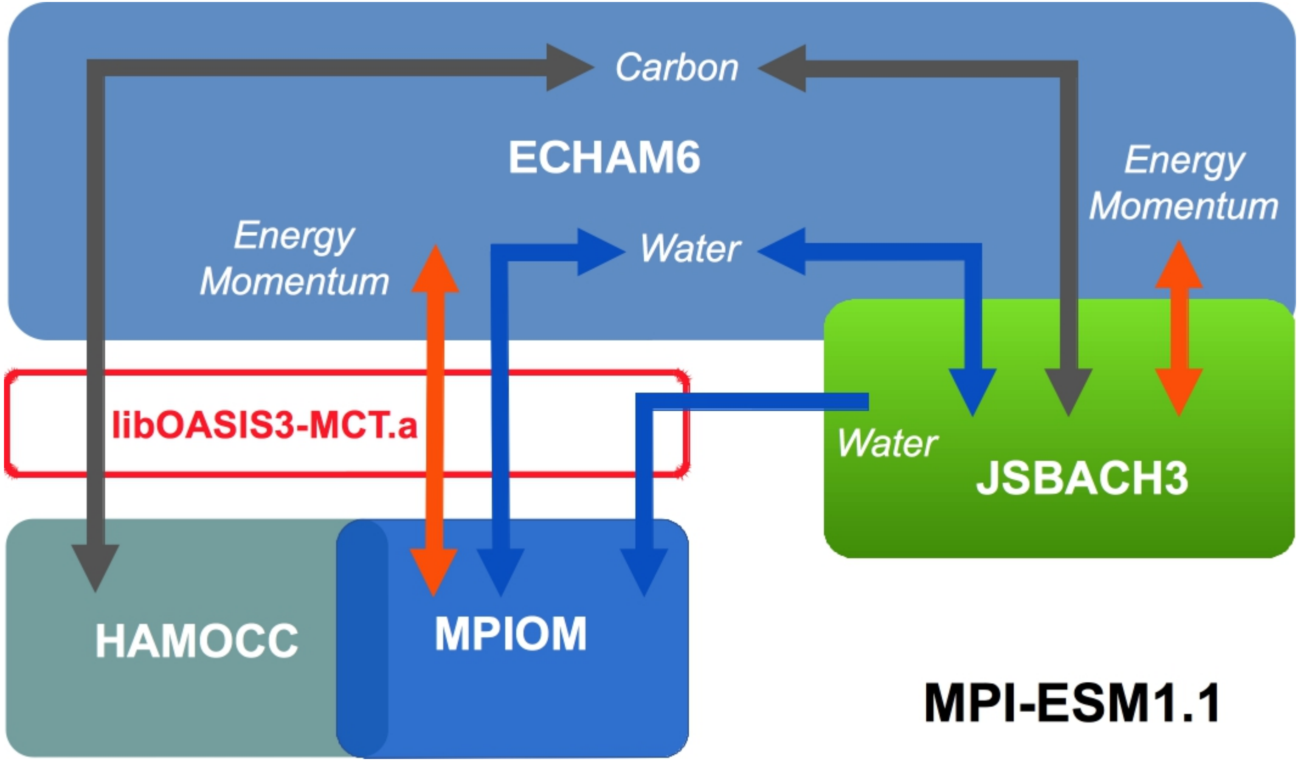
Thomas Jahns (DKRZ)
Uwe Schulzweida (MPI-M)
Hendrik Bockelmann (DKRZ)
Jörg Behrens (DKRZ)
Sergey Kosukhin (MPI-M)

YAC – General remarks



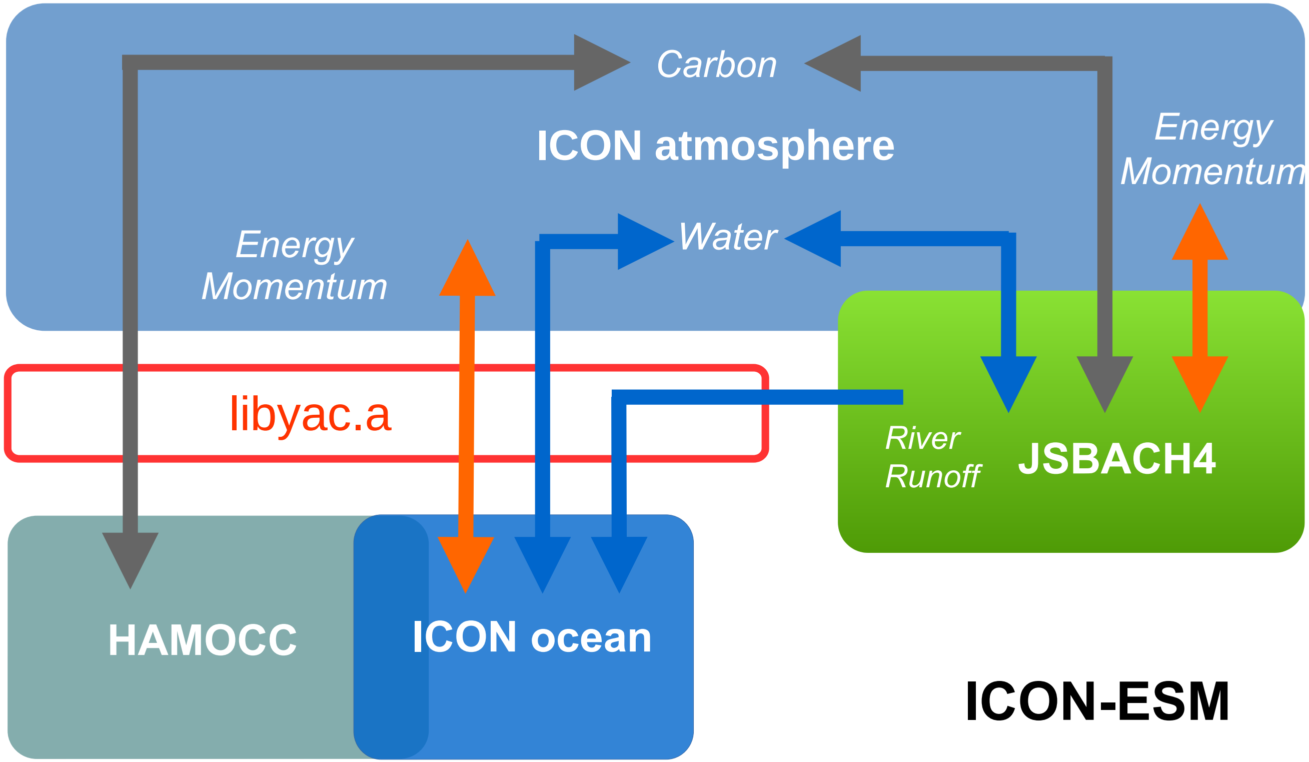
MPI-ESM1.0

YAC – General remarks



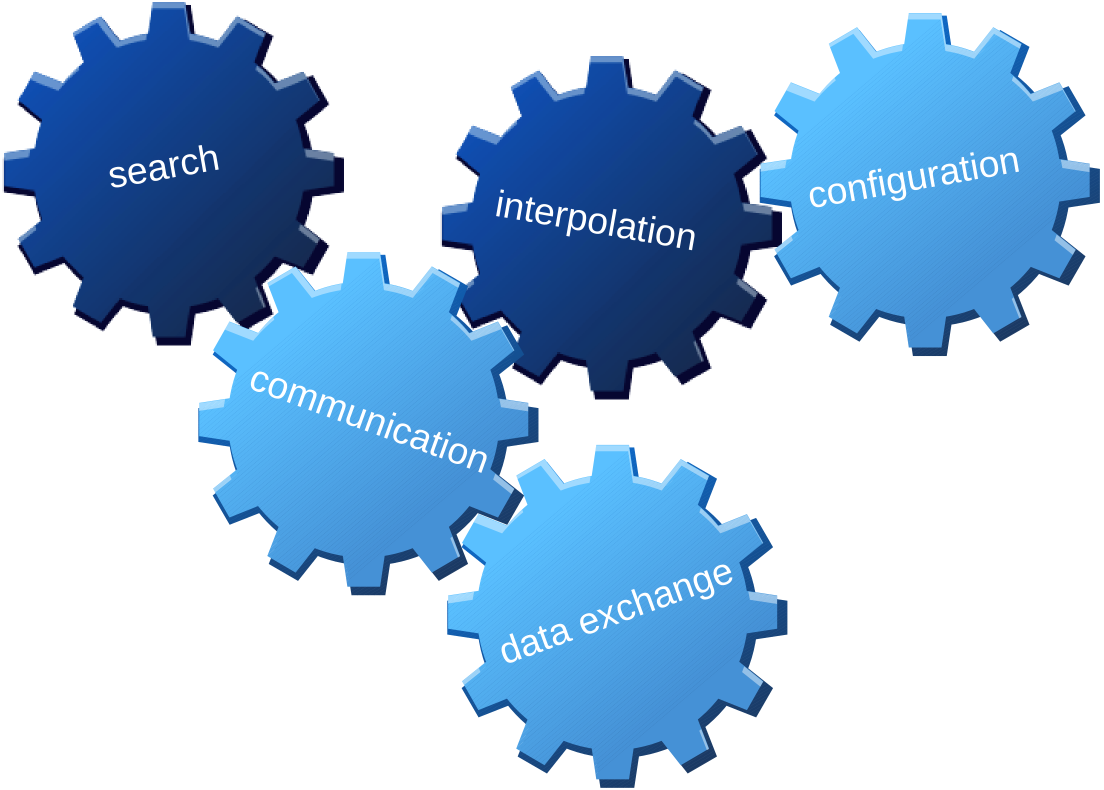
MPI-ESM1.1

YAC – General remarks



ICON-ESM

YAC – General remarks



YAC – General remarks

A coupling software not only for ICON

- Parallel search on (almost) arbitrary grids on the sphere
- Parallel interpolation
- Parallel data exchange

- Library
- BSD License
- Programming Language C
- Fortran and C user API
- Programming based on standards (C, MPI, XML, NetCDF)
- Git repository
- Autotools
- Valgrind testing
- Unit tests (~90% of lines covered)
- Fortran and C examples plus toy models
- XML coupling configuration file with GUI support

YAC – Search

required

- Geographical positions (λ, φ) of vertices and points

provided

- Initial scalable computation of global mapping
- Final scalable parallel interpolation specific search and calculation of interpolation weights

features

- Support for circles of latitude/longitude and great circles
- Search and interpolation in Cartesian coordinates
- Convex & moderately concave polygons
- Support for masked cells and points

YAC – Interpolation

Available 2-dimensional (horizontal) interpolation methods

- 1st – order conservative remapping (**conserv**)
- 2nd – order conservative remapping (**conserv**)
- Hybrid cubic spherical Bernstein-Bézier patch interpolation (**bernstein_bezier**)
- Distance-weighted N-nearest-neighbour (**n-nearest_neighbour**)
- N-nearest-neighbour average (**n-nearest_neighbour**)
- Gauss-weighted N-nearest-neighbour (**n-nearest_neighbour**)
- Radial Basis Functions (**radial_basis_function**)
- Source Point to Target Point Mapping (**source_to_target_map**)
- Fixed value (**fixed**)

YAC – Interpolation

Available 2-dimensional (horizontal) interpolation methods (continued)

- Patch recovery - polynomial fit (**patch_recovery**)
- Smoothed Patch recovery - polynomial fit (**smooth_patch_recovery**)
- Radial Basis Functions (**radial_basis_function**)
- Source Point to Target Point Mapping (**source_to_target_map**)
- Simple cell average (**average**)
- Distance-weighted cell average (**average**)
- File input (**user_file**)

YAC – Interpolation stack

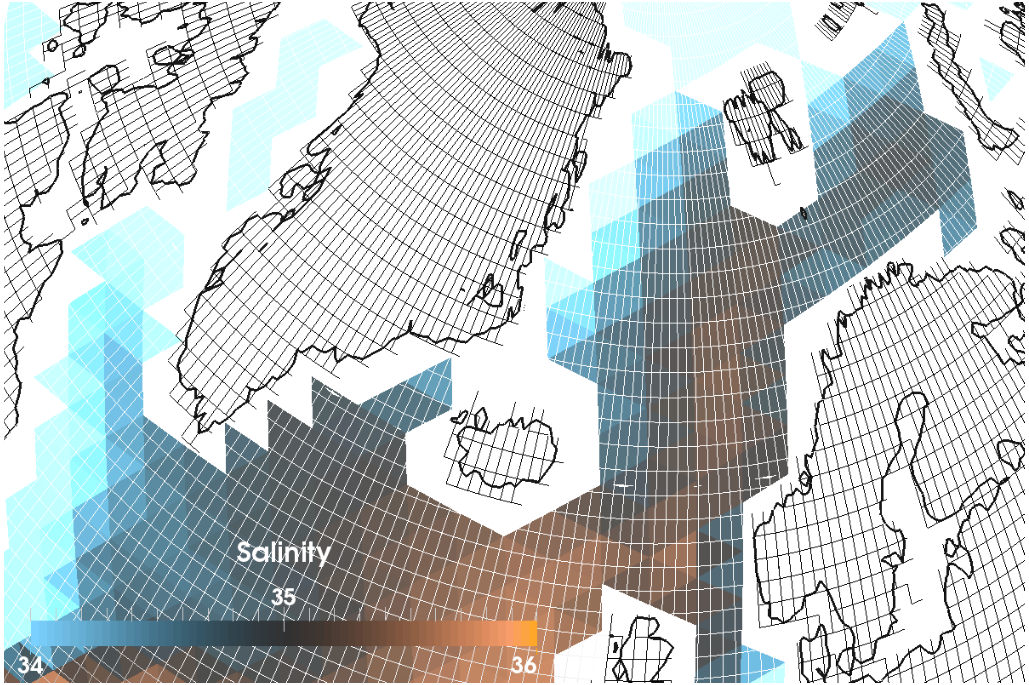
example

interpolation of World Ocean Atlas 2009 sea surface salinity onto an ICON R2B04 atmosphere grid.

1st-order conservative remapping
plus **patch recovery**
plus **fixed value**

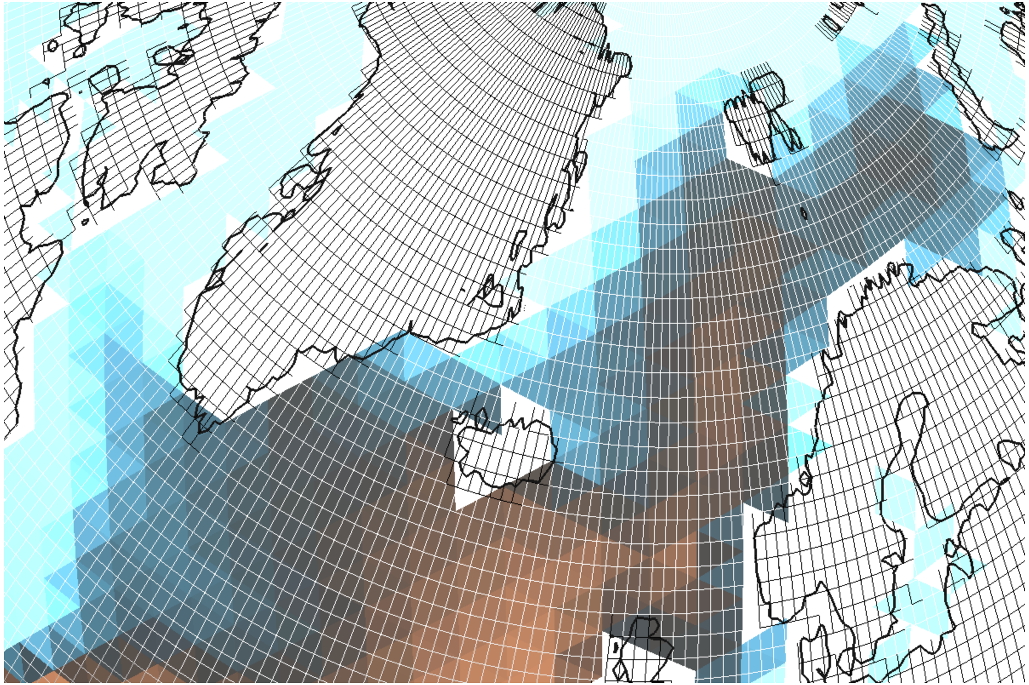
YAC – Interpolation stack

Step 1: 1st- order conservative remapping



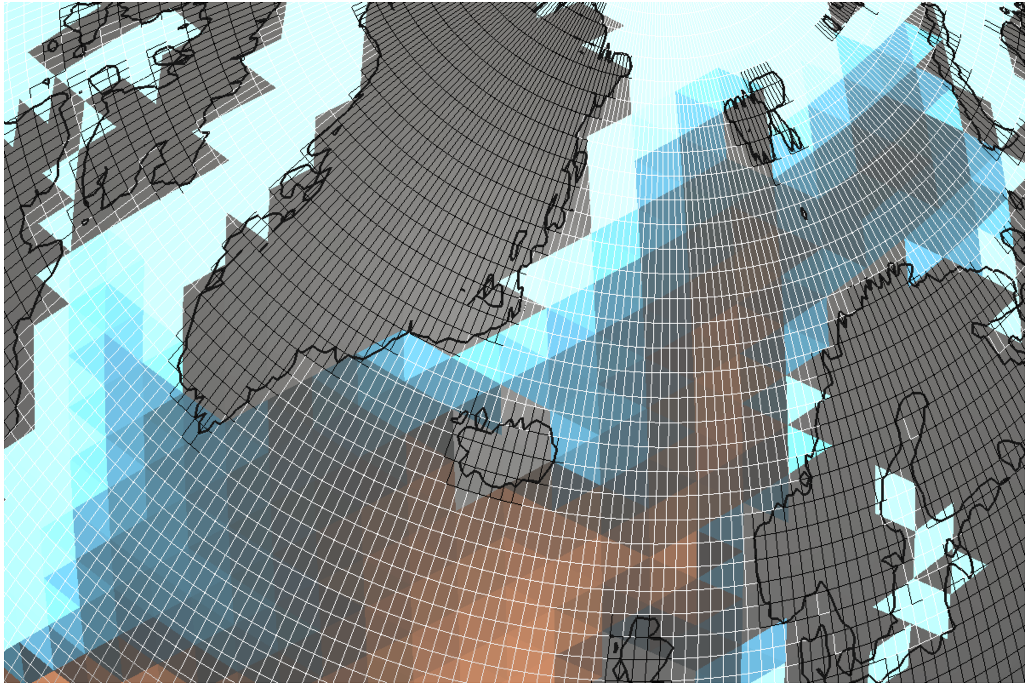
YAC – Interpolation stack

Step 1: … + patch recovery

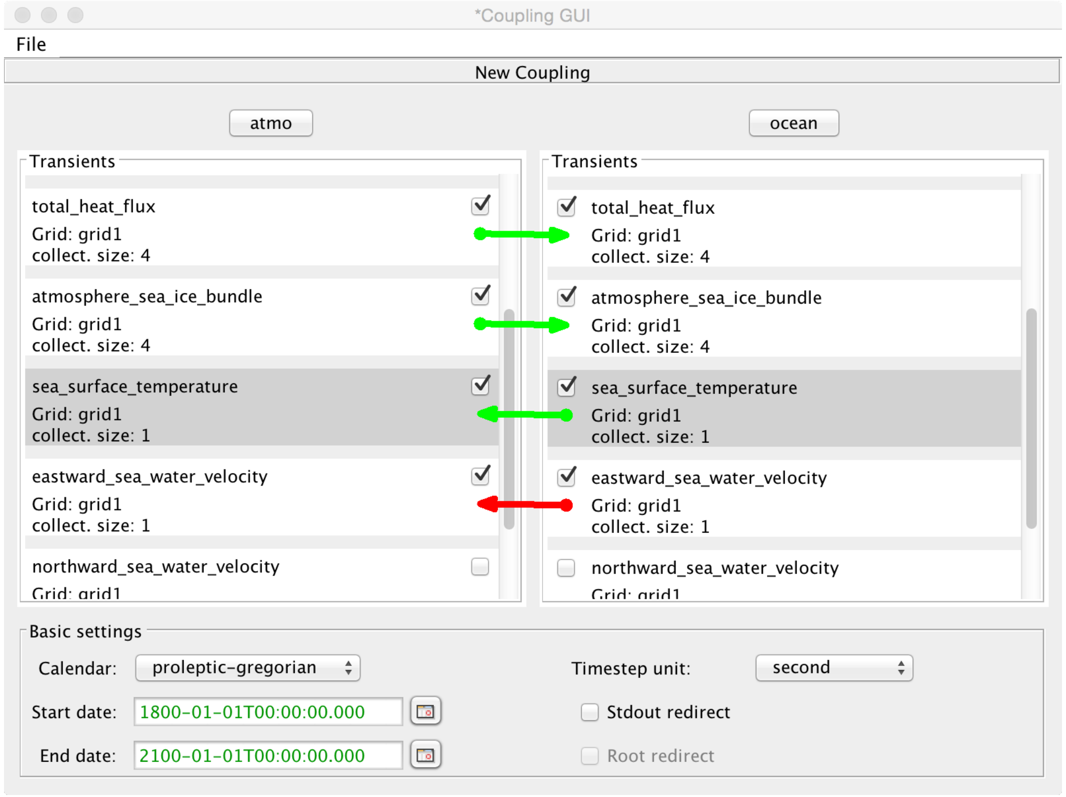


YAC – Interpolation stack

Step 1: … + fixed value



YAC – Graphical User Interface



Transients (atmo)

- total_heat_flux ☑
  Grid: grid1
  collect. size: 4
- atmosphere_sea_ice_bundle ☑
  Grid: grid1
  collect. size: 4
- sea_surface_temperature ☑
  Grid: grid1
  collect. size: 1
- eastward_sea_water_velocity ☑
  Grid: grid1
  collect. size: 1
- northward_sea_water_velocity ☐
  Grid: grid1

Transients (ocean)

- ☑ total_heat_flux
  Grid: grid1
  collect. size: 4
- ☑ atmosphere_sea_ice_bundle
  Grid: grid1
  collect. size: 4
- ☑ sea_surface_temperature
  Grid: grid1
  collect. size: 1
- ☑ eastward_sea_water_velocity
  Grid: grid1
  collect. size: 1
- ☐ northward_sea_water_velocity
  Grid: grid1

Basic settings

Calendar: proleptic-gregorian

Timestep unit: second

Start date: 1800-01-01T00:00:00.000

☐ Stdout redirect

End date: 2100-01-01T00:00:00.000

☐ Root redirect

Max-Planck-Institut
für Meteorologie

DKRZ
DEUTSCHES
KLIMARECHENZENTRUM

YAC – Fortran Interface

Initialisation Phase

- yac_finit
- yac_fdef_comp
- yac_fdef_datetime
- yac_fget_localcomm

Grid Definition

- yac_fdef_grid
- yac_fdef_points
- yac_fdef_index_location
- yac_fset_core_mask
- yac_fdef_mask
- yac_fdef_field

Search – End of Definition

- yac_fsearch

Data exchange

- yac_fget
- yac_fput

Termination

- yac_ffinalize

YAC – Initialisation phase

component initialisation

```
CALL yac_finit ( “coupling.xml”, “coupling.xsd” )
```

- will call MPI_INIT if not been called already

```
CALL yac_fdef_comp ( ”component_name” , component_id )
```

- local operations for initialising of YAC-internal data structures
- needs to be called by all processes

```
CALL yac_fdef_datetime   ( start_datetime = start_of_run_in iso_format,
                           end_datetime  = end_of_run_in_iso_format )
```

- overwrites start and end date set in coupling.xml
- if required it has to be called before calling yac_fdef_field
- time management inside yac using mtime

YAC – Definition Phase

grid definition *(example for an unstructured grid)*

```
CALL yac_fdef_grid   (   “grid_name“,
                         nbr_of_horizontal_vertices,
                         nbr_of_horizontal_cells,
                         nbr_vertices_per_cell,
                         array_of_vertex_longitudes,
                         array_of_vertex_latitudes,
                         connectivity,
                         grid_id  )
```

overloaded with respect to

- data type for coordinate arrays
- grid types

YAC – Definition Phase

grid definition

```
CALL yac_fset_global_index  (  array_of_global_indices,
                               YAC_LOCATION_CELL,
                               grid_id )
```

YAC – Definition Phase

grid definition

```
CALL yac_fset_core_mask  (  core_mask_array,
                            YAC_LOCATION_CELL,
                            grid_id )
```

YAC – Definition Phase

mask definition

```
CALL yac_fset_mask  (  mask_array,
                       point_id )
```

overloaded with respect to

data type (`Integer` or `Logical`) of mask array

mask_array

1 (`.TRUE.`) for valid data

0 (`.FALSE.`) for invalid data

YAC – Definition Phase

field definition

```
CALL yac_fdef_field   (   ”field_name”,
                          component_id,
                          grid_id,
                          array_of_cell_point_ids,
                          nbr_point_sets,
                          field_id    )
```

YAC – Communication Phase

search

```
CALL yac_fsearch  ( nbr_of_components,
                    array_of_component_ids,
                    nbr_of_fields,
                    array_of_field_ids,
                    error_status  )
```

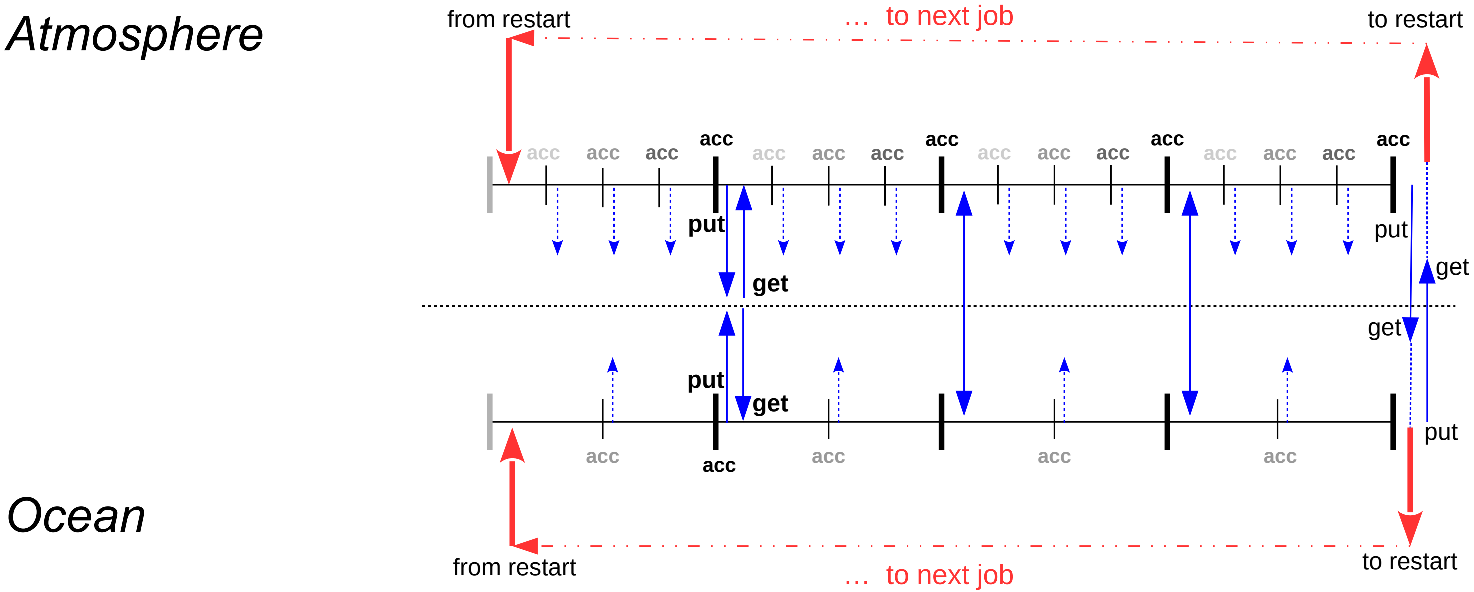
- includes collective MPI operations
- needs to be called by all processes
- accesses the coupling configuration
- invokes the neighbourhood search
- does the communicator splitting

```
CALL yac_fget_localcomm ( local_mpi_communicator, component_id )
```

YAC – Exchange Phase

data exchange

as it is implemented in ICON



YAC – Exchange Phase

data exchange

```
CALL yac_fput  ( field_id,
                 nbr_horizontal_points,
                 collection_size,
                 send_field,
                 info,
                 error_flag  )
```

- to be called at every time step
- at the “source timestep” interval specified in the xml file
- accumulation/averaging done inside yac_fput

YAC – Exchange Phase

data exchange

as it is implemented in ICON

```
! field_id(6) : Temperature

DO i_blk = 1, patch_horz%nblks_c
  nn = (i_blk-1)*nproma
  DO n = 1, nproma
    buffer(nn+n,1) = &
    ocean_state%p_prog(nold(1))%tracer(n,1,i_blk,1) + tmelt
  ENDDO
ENDDO

CALL yac_fput ( field_id(6), nbr_hor_points, 1,   &
      &         buffer(1:nbr_hor_points,1),       &
      &         info, ierror )
```

YAC – Exchange Phase

data exchange

```
CALL yac_fget (  field_id,
                 collection_size,
                 recv_field,
                 info,
                 error_flag  )
```

- to be called at every time step
- at the “source timestep” interval specified in the xml file
- accumulation/averaging done inside yac_fput

YAC – Exchange Phase

data exchange

as it is implemented in ICON

```
CALL yac_fget ( field_id(1), nbr_hor_points, 2,   &
      &         buffer(1:nbr_hor_points,1:2),     &
      &         info, ierror )

IF ( info > 0 .AND. info < 7 ) THEN
  DO i_blk = 1, patch_horz%nblks_c
    nn = (i_blk-1)*nproma
    DO n = 1, nproma
      atmos_fluxes%stress_xw(n,i_blk) = buffer(nn+n,1)
      atmos_fluxes%stress_x (n,i_blk) = buffer(nn+n,2)
    ENDDO
  ENDDO
  CALL sync_patch_array  …
ENDIF
```

YAC – Exchange Phase

data exchange

Return values for the info argument

```
enum, bind(c)
  enumerator :: NONE               = 0
  enumerator :: COUPLING           = 1
  enumerator :: RESTART            = 2
  enumerator :: GET_FOR_RESTART    = 3
  enumerator :: PUT_FOR_RESTART    = 4
  enumerator :: GET_FOR_CHECKPOINT = 5
  enumerator :: PUT_FOR_CHECKPOINT = 6
  enumerator :: OUT_OF_BOUND       = 7
end enum
```

YAC – Termination Phase

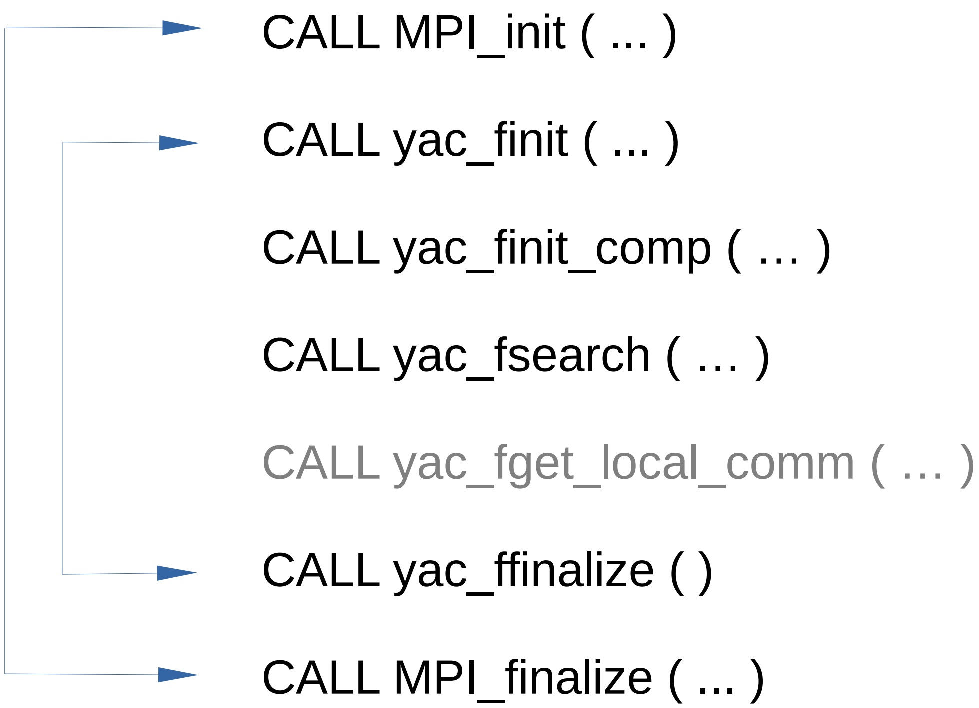
termination of coupling

```
CALL yac_ffinalize  ( )
```

- frees all internal data structures related to coupling
- MPI communicators may no longer be available
- will call MPI_FINALIZE
  if MPI_INIT has been called by yac_finit
  if MPI_FINALIZE has not already been called

YAC & MPI

Recommended calling sequence



```
CALL MPI_init ( ... )
CALL yac_finit ( ... )
CALL yac_finit_comp ( … )
CALL yac_fsearch ( … )
CALL yac_fget_local_comm ( … )
CALL yac_ffinalize ( )
CALL MPI_finalize ( ... )
```

YAC – Component XML Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<component
        xmlns="http://www.w3schools.com"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.w3schools.component.xsd">
        <id>1</id>
        <name>atmo</name>
        <model>ICON</model>
        <simulated>atmosphere</simulated>
        <transient_grid_refs>
            <transient_grid_ref collection_size="2" grid_ref="1" id="1" transient_ref="1"/>
            <transient_grid_ref collection_size="2" grid_ref="1" id="2" transient_ref="2"/>
            <transient_grid_ref collection_size="3" grid_ref="1" id="3" transient_ref="3"/>
            <transient_grid_ref collection_size="4" grid_ref="1" id="4" transient_ref="4"/>
            <transient_grid_ref collection_size="4" grid_ref="1" id="5" transient_ref="5"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="6" transient_ref="6"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="7" transient_ref="7"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="8" transient_ref="8"/>
            <transient_grid_ref collection_size="5" grid_ref="1" id="9" transient_ref="9"/>
        </transient_grid_refs>
        ...
```

YAC – Component XML Configuration

```
        ...

        <transients>
            <transient id="1" transient_standard_name="surface_downward_eastward_stress"/>
            <transient id="2" transient_standard_name="surface_downward_northward_stress"/>
            <transient id="3" transient_standard_name="surface_fresh_water_flux"/>
            <transient id="4" transient_standard_name="total_heat_flux"/>
            <transient id="5" transient_standard_name="atmosphere_sea_ice_bundle"/>
            <transient id="6" transient_standard_name="sea_surface_temperature"/>
            <transient id="7" transient_standard_name="eastward_sea_water_velocity"/>
            <transient id="8" transient_standard_name="northward_sea_water_velocity"/>
            <transient id="9" transient_standard_name="ocean_sea_ice_bundle"/>
        </transients>
        <grids>
            <grid id="1" alias_name="atmos_grid"/>
        </grids>
</component>
```

YAC – Component XML Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<component
        xmlns="http://www.w3schools.com"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.w3schools.component.xsd">
        <id>1</id>
        <name>atmo</name>
        <model>ICON</model>
        <simulated>atmosphere</simulated>
        <transient_grid_refs>
            <transient_grid_ref collection_size="2" grid_ref="1" id="1" transient_ref="1"/>
            <transient_grid_ref collection_size="2" grid_ref="1" id="2" transient_ref="2"/>
            <transient_grid_ref collection_size="3" grid_ref="1" id="3" transient_ref="3"/>
            <transient_grid_ref collection_size="4" grid_ref="1" id="4" transient_ref="4"/>
            <transient_grid_ref collection_size="4" grid_ref="1" id="5" transient_ref="5"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="6" transient_ref="6"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="7" transient_ref="7"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="8" transient_ref="8"/>
            <transient_grid_ref collection_size="5" grid_ref="1" id="9" transient_ref="9"/>
        </transient_grid_refs>
        ...
```

YAC – Component XML Configuration

```
<name>atmo</name>
<model>ICON</model>
<simulated>atmosphere</simulated>
```

```
CALL yac_fdef_comp ( “atmo”, comp_id )
```

YAC – Component XML Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<component
        xmlns="http://www.w3schools.com"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.w3schools.component.xsd">
        <id>1</id>
        <name>atmo</name>
        <model>ICON</model>
        <simulated>atmosphere</simulated>
        <transient_grid_refs>
            <transient_grid_ref collection_size="2" grid_ref="1" id="1" transient_ref="1"/>
            <transient_grid_ref collection_size="2" grid_ref="1" id="2" transient_ref="2"/>
            <transient_grid_ref collection_size="3" grid_ref="1" id="3" transient_ref="3"/>
            <transient_grid_ref collection_size="4" grid_ref="1" id="4" transient_ref="4"/>
            <transient_grid_ref collection_size="4" grid_ref="1" id="5" transient_ref="5"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="6" transient_ref="6"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="7" transient_ref="7"/>
            <transient_grid_ref collection_size="1" grid_ref="1" id="8" transient_ref="8"/>
            <transient_grid_ref collection_size="5" grid_ref="1" id="9" transient_ref="9"/>
        </transient_grid_refs>
        ...
```

YAC – Component XML Configuration

```
<transient_grid_refs>
   <transient_grid_ref collection_size="2" grid_ref="1" id="1" transient_ref="1"/>
   <transient_grid_ref collection_size="2" grid_ref="1" id="2" transient_ref="2"/>
   <transient_grid_ref collection_size="3" grid_ref="1" id="3" transient_ref="3"/>
   …
   <transient_grid_ref collection_size="5" grid_ref="1" id="9" transient_ref="9"/>
</transient_grid_refs>
```

```
CALL yac_fput ( field_id, nbr_hor_points, 5,   &
         &      buffer(1:nbr_hor_points,1:5), &
         &      info, ierror )

CALL yac_fget ( field_id, nbr_hor_points, 2,   &
         &      buffer(1:nbr_hor_points,1:2), &
         &      info, ierror )
```

YAC – Component XML Configuration

```
        ...

        <transients>
            <transient id="1" transient_standard_name="surface_downward_eastward_stress"/>
            <transient id="2" transient_standard_name="surface_downward_northward_stress"/>
            <transient id="3" transient_standard_name="surface_fresh_water_flux"/>
            <transient id="4" transient_standard_name="total_heat_flux"/>
            <transient id="5" transient_standard_name="atmosphere_sea_ice_bundle"/>
            <transient id="6" transient_standard_name="sea_surface_temperature"/>
            <transient id="7" transient_standard_name="eastward_sea_water_velocity"/>
            <transient id="8" transient_standard_name="northward_sea_water_velocity"/>
            <transient id="9" transient_standard_name="ocean_sea_ice_bundle"/>
        </transients>
        <grids>
            <grid id="1" alias_name="atmos_grid"/>
        </grids>
</component>
```

YAC – Component XML Configuration

```
<transients>
   <transient id="1" transient_standard_name="surface_downward_eastward_stress"/>
   <transient id="2" transient_standard_name="surface_downward_northward_stress"/>
   <transient id="3" transient_standard_name="surface_fresh_water_flux"/>
   ...
   <transient id="9" transient_standard_name="ocean_sea_ice_bundle"/>
</transients>
```

```
CALL yac_fdef_field &
      &        ( "surface_downward_eastward_stress",   &
      &          component_id, grid_id, point_id, &
      &          1, field_id(1) )
...
CALL yac_fdef_field &
      &        ( "ocean_sea_ice_bundle",               &
      &          component_id, grid_id, point_id, &
      &          1, field_id(9) )
```

YAC – Component XML Configuration

```
<grids>
   <grid id="1" alias_name="atmos_grid"/>
</grids>
```

```
CALL yac_fdef_grid ( “atmos_grid“,

                             [ … ],

                             grid_id )
```

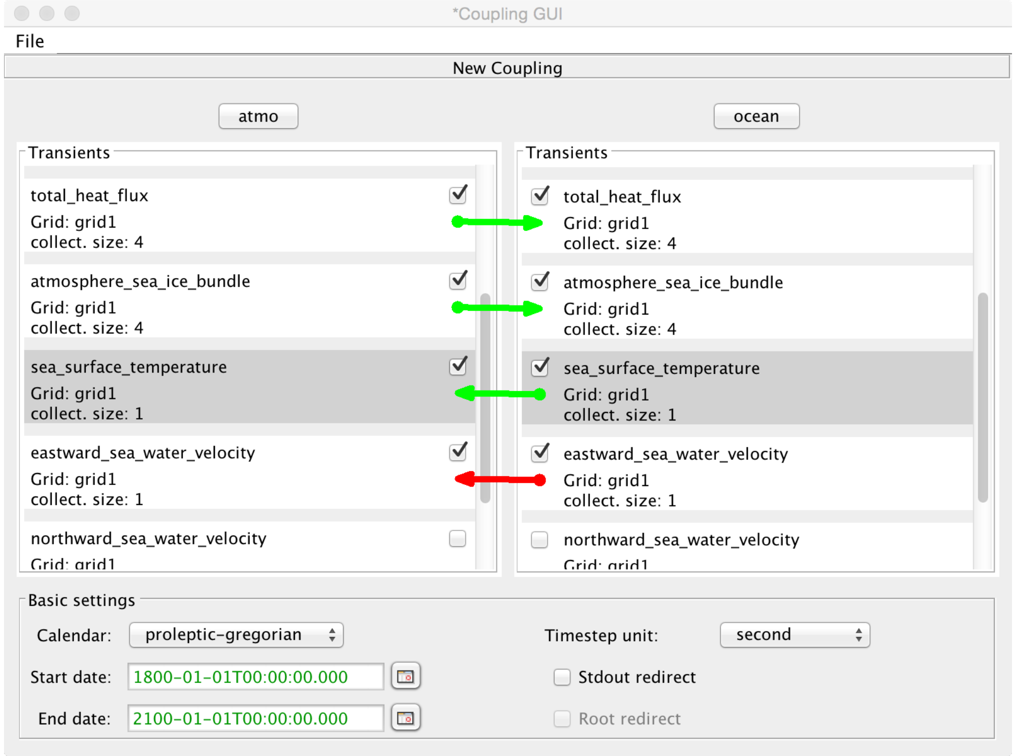
YAC – Component XML Configuration

```
<transient_grid_refs>
   <transient_grid_ref collection_size="2" grid_ref="1" id="1" transient_ref="1"/>
   <transient_grid_ref collection_size="2" grid_ref="1" id="2" transient_ref="2"/>
   <transient_grid_ref collection_size="3" grid_ref="1" id="3" transient_ref="3"/>
   <transient_grid_ref collection_size="4" grid_ref="1" id="4" transient_ref="4"/>
   …
   <transient_grid_ref collection_size="5" grid_ref="1" id="9" transient_ref="9"/>
</transient_grid_refs>

<transients>
   <transient id="1" transient_standard_name="surface_downward_eastward_stress"/>
   <transient id="2" transient_standard_name="surface_downward_northward_stress"/>
   <transient id="3" transient_standard_name="surface_fresh_water_flux"/>
   <transient id="4" transient_standard_name="total_heat_flux"/>
   ...
   <transient id="9"  transient_standard_name="ocean_sea_ice_bundle"/>
</transients>

<grids>
   <grid id="1" alias_name="grid1"/>
</grids>
```

YAC – XML Configuration



atmo – Transients	ocean – Transients
total_heat_flux ☑ Grid: grid1 collect. size: 4	☑ total_heat_flux Grid: grid1 collect. size: 4
atmosphere_sea_ice_bundle ☑ Grid: grid1 collect. size: 4	☑ atmosphere_sea_ice_bundle Grid: grid1 collect. size: 4
sea_surface_temperature ☑ Grid: grid1 collect. size: 1	☑ sea_surface_temperature Grid: grid1 collect. size: 1
eastward_sea_water_velocity ☑ Grid: grid1 collect. size: 1	☑ eastward_sea_water_velocity Grid: grid1 collect. size: 1
northward_sea_water_velocity □ Grid: grid1	□ northward_sea_water_velocity Grid: grid1

Basic settings

Calendar: proleptic-gregorian

Start date: 1800-01-01T00:00:00.000

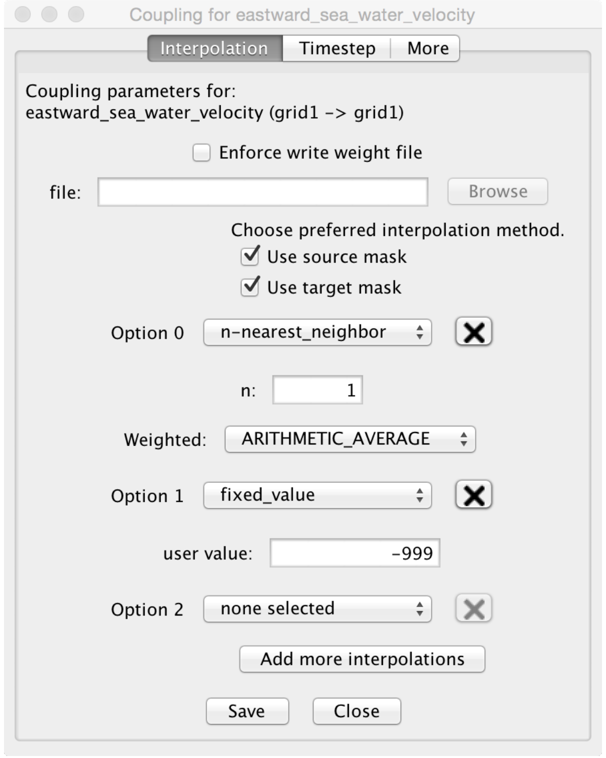
End date: 2100-01-01T00:00:00.000

Timestep unit: second

□ Stdout redirect

□ Root redirect

YAC – XML Configuration



Coupling parameters for:
eastward_sea_water_velocity (grid1 -> grid1)

☐ Enforce write weight file

file:

Choose preferred interpolation method.

☑ Use source mask

☑ Use target mask

Option 0: n-nearest_neighbor

n: 1

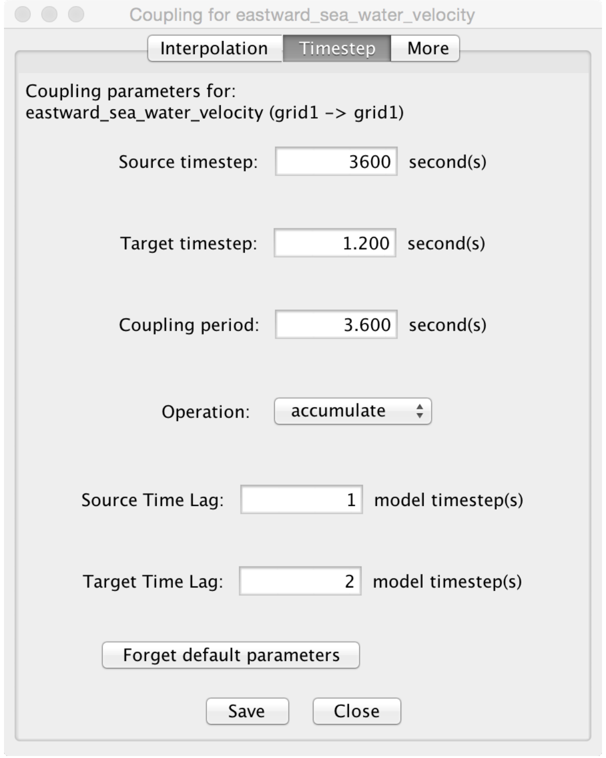
Weighted: ARITHMETIC_AVERAGE

Option 1: fixed_value

user value: -999

Option 2: none selected

YAC – XML Configuration



Coupling for eastward_sea_water_velocity
Interpolation
Timestep
More

Coupling parameters for:
eastward_sea_water_velocity (grid1 -> grid1)

Source timestep: 3600 second(s)

Target timestep: 1.200 second(s)

Coupling period: 3.600 second(s)

Operation: accumulate

Source Time Lag: 1 model timestep(s)

Target Time Lag: 2 model timestep(s)

Forget default parameters
Save
Close

YAC – XML Configuration

Source time step

- time interval between two consecutive calls to yac_fput

Target time step

- time interval between two consecutive calls to yac_fget

Requirement

Source or target time step must be equal to or an integer multiple of the other.

YAC – XML Configuration

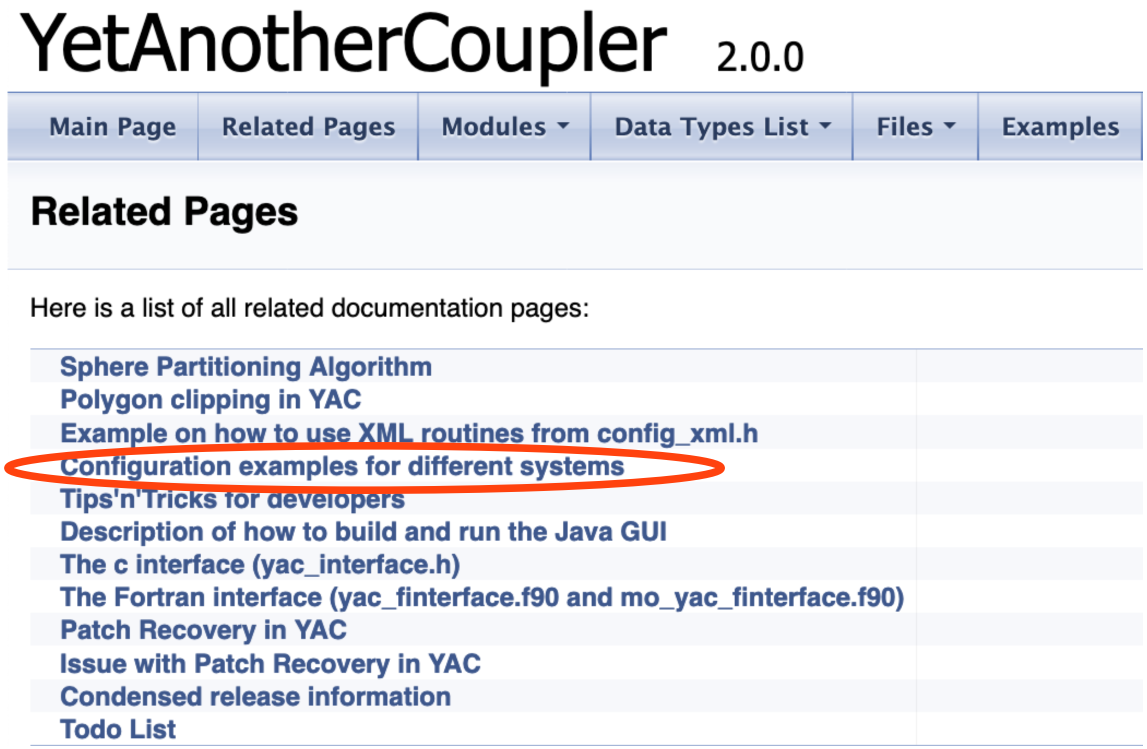
Coupling period

- Time interval at which data are exchanged (with internal calls to MPI_SEND and MPI_RECV)

Requirement

Coupling period must be an integer multiple of the source/target time step

YAC – Building



YetAnotherCoupler 2.0.0

Related Pages

Here is a list of all related documentation pages:

- Sphere Partitioning Algorithm
- Polygon clipping in YAC
- Example on how to use XML routines from config_xml.h
- Configuration examples for different systems
- Tips'n'Tricks for developers
- Description of how to build and run the Java GUI
- The c interface (yac_interface.h)
- The Fortran interface (yac_finterface.f90 and mo_yac_finterface.f90)
- Patch Recovery in YAC
- Issue with Patch Recovery in YAC
- Condensed release information
- Todo List

YAC – Documentation

Doxygen

http://dkrz-sw.gitlab-pages.dkrz.de/yac/

Source Code (version 2.0.0)

```
git clone -b 'release-2.0.0' --single-branch --depth 1 git@gitlab.dkrz.de:YAC/YAC.git
```

Latest version (untagged)

```
git clone git@gitlab.dkrz.de:YAC/YAC.git
```

Documentation with further Links

- https://www.geosci-model-dev.net/9/2755/2016/
- https://doi.org/10.5676/dwd_pub/nwv/icon_003
- https://code.zmaw.de/projects/mpiesm-2/wiki/ICON_Coupled_Model_Development
- https://www.mpimet.mpg.de/en/science/models/mpi-esm/