A new concept / interface for the halo-exchange

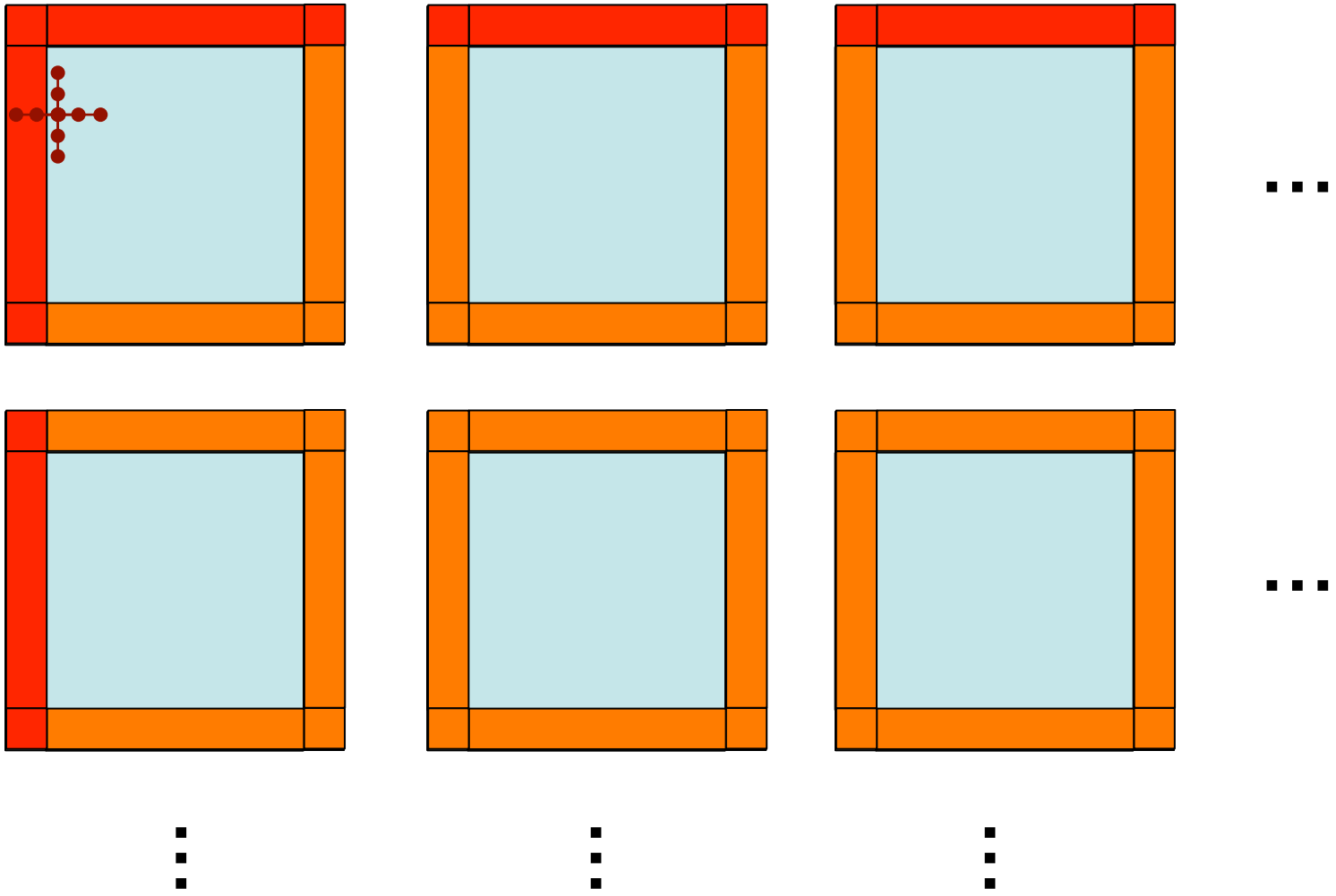
Oliver Fuhrer (MeteoSwiss)

Inspired by work done on the C++ dycore together with Tobias and Carlos

COSMO GM 2012, Lugano, WG6 Parallel Session

Halo-Exchange

0. Computation (Stencil)
1. Halo-Update
2. Boundary Condition
3. Computation (Stencil)



Halo-Exchange Properties

- **Order**
  - … ➔ Computation 1
  - Halo-exchange 2 ➔ Boundary Condition 2 ➔ Computation 2
  - Halo-exchange 3 ➔ …
- **Number of lines (nlines)**
  - nlines depends on next stencil
  - There are use-cases i-only or j-only update
  - Maximum nlines determined by largest stencil
  - Multiple exchanges can be avoided by more computation

Current Implementation

```
SUBROUTINE exchg_boundaries(
  icase, sendbuf, isendbuflen, imp_type, icomm, num_compute,  &
  idim, jdim, kdim, jstartpar, jendpar, nlines, nboundlines,  &
  neighbors, lperi_x, lperi_y, l2dim, ntag, lmpi_types, ntype,&
  ierror, yerrmsg,                                            &
  var01, var02, var03, var04, var05, var06, var07, var08,     &
  var09, var10, var11, var12, var13, var14, var15, var16,     &
  var17, var18, var19, var20, var21, var22, var23, var24      &
)
```

- Subroutine in environment.f90
- Based on MPI library calls
- ~1250 lines of code
- Separated from parallel_utilities.f90
  (and thus does not have access to domain decomposition parameters setup by init_par_utilities() method)
- 3 communication strategies
  (ISend/Recv, IRecv/Send, SendRecv)
- 2 packing strategies (MPI datatypes, explicit packing)

Current Implementation

```
SUBROUTINE exchg_boundaries(
  icase, sendbuf, isendbuflen, imp_type, icomm, num_compute,  &
  idim, jdim, kdim, jstartpar, jendpar, nlines, nboundlines,  &
  neighbors, lperi_x, lperi_y, l2dim, ntag, lmpi_types, ntype,&
  ierror, yerrmsg,                                            &
  var01, var02, var03, var04, var05, var06, var07, var08,     &
  var09, var10, var11, var12, var13, var14, var15, var16,     &
  var17, var18, var19, var20, var21, var22, var23, var24      &
)
```

- **Communication parameters**
  - **icase** **tag for exchange scenario**
  - sendbuf buffer used for packing messages
  - isendbuflen length of buffer
  - imp_type MPI type
  - icomm MPI communicator
  - ntag tag of message
  - lmpi_types implicit/explicit packing
  - ntype communication strategy

internal

global

Current Implementation

```
SUBROUTINE exchg_boundaries(
  icase,                                                num_compute,   &
  idim, jdim, kdim, jstartpar, jendpar, nlines, nboundlines,   &
  neighbors, lperi_x, lperi_y, l2dim,                                  &
  ierror, yerrmsg,                                                     &
  var01, var02, var03, var04, var05, var06, var07, var08,      &
  var09, var10, var11, var12, var13, var14, var15, var16,      &
  var17, var18, var19, var20, var21, var22, var23, var24       &
)
```

- **Domain-decomposition parameters**
  - num_compute — buffer used for packing messages
  - neighbors(4) — MPI task IDs of neighbors
  - lperi_x/y — periodicity in x-/y-direction
  - l2dim — 2-dimensional setup

} **global**

Current Implementation

```
SUBROUTINE exchg_boundaries(
  icase,                                                          &
  idim, jdim, kdim, jstartpar, jendpar, nlines, nboundlines,      &
                                                                  &
  ierror, yerrmsg,                                                &
  var01, var02, var03, var04, var05, var06, var07, var08,         &
  var09, var10, var11, var12, var13, var14, var15, var16,         &
  var17, var18, var19, var20, var21, var22, var23, var24          &
)
```

- **Exchange parameters**
  - **nlines** **number of lines to exchange**
  - jstartpar start/end of border width
  - jendpar …in j-direction
  - nboundlines total halo width

} **global**

Current Implementation

```
SUBROUTINE exchg_boundaries(
  icase,                                                    &
  idim, jdim, kdim,                          nlines,        &
                                                            &
  ierror, yerrmsg,                                          &
  var01, var02, var03, var04, var05, var06, var07, var08,   &
  var09, var10, var11, var12, var13, var14, var15, var16,   &
  var17, var18, var19, var20, var21, var22, var23, var24    &
)
```

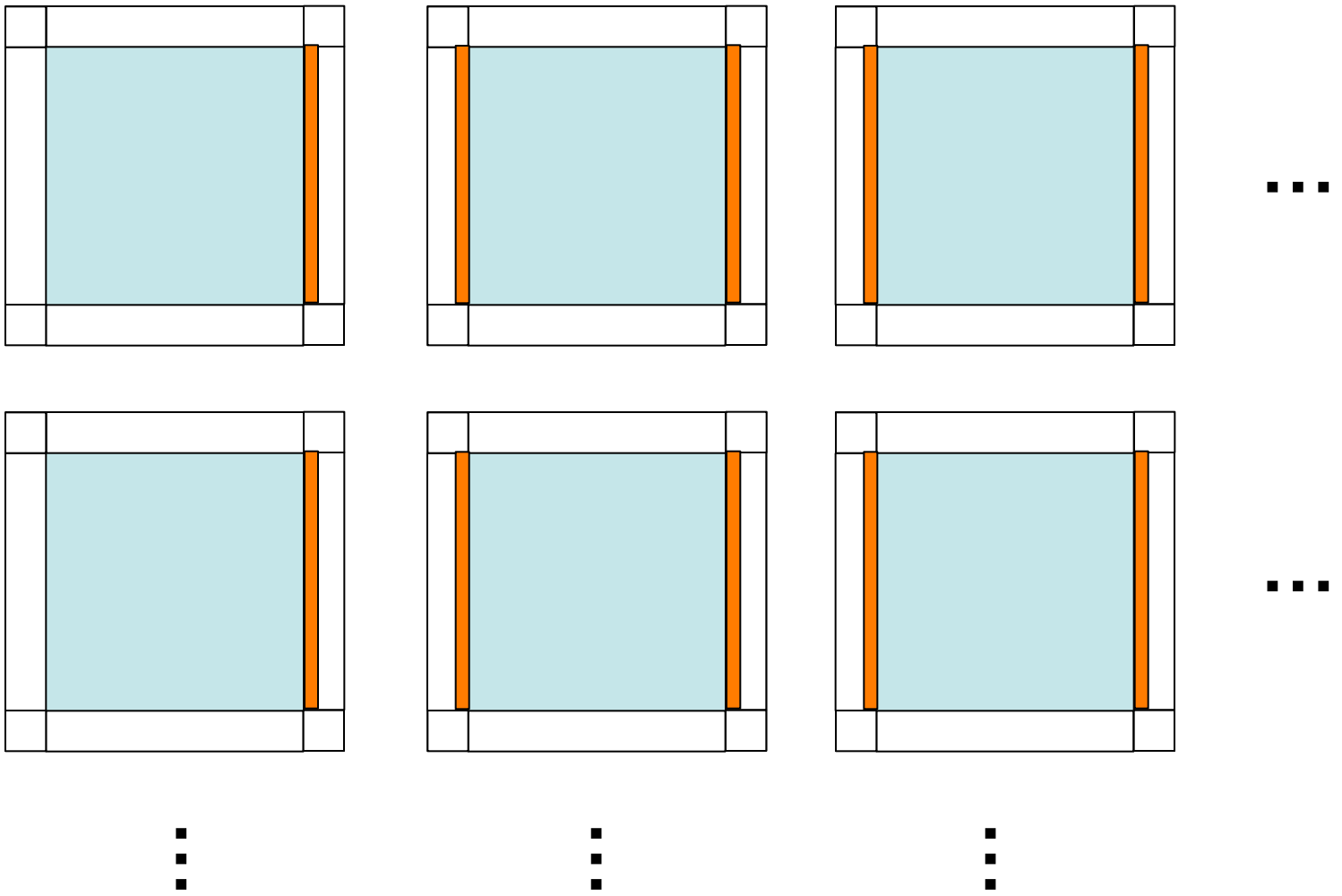
- **Field parameters**
  - idim, jdim — size in i-/j-direction
  - kdim(:) — number of levels of each field
  - **var01 - 24** — **fields to be exchanged**

} **redundant**

Corner Exchanges

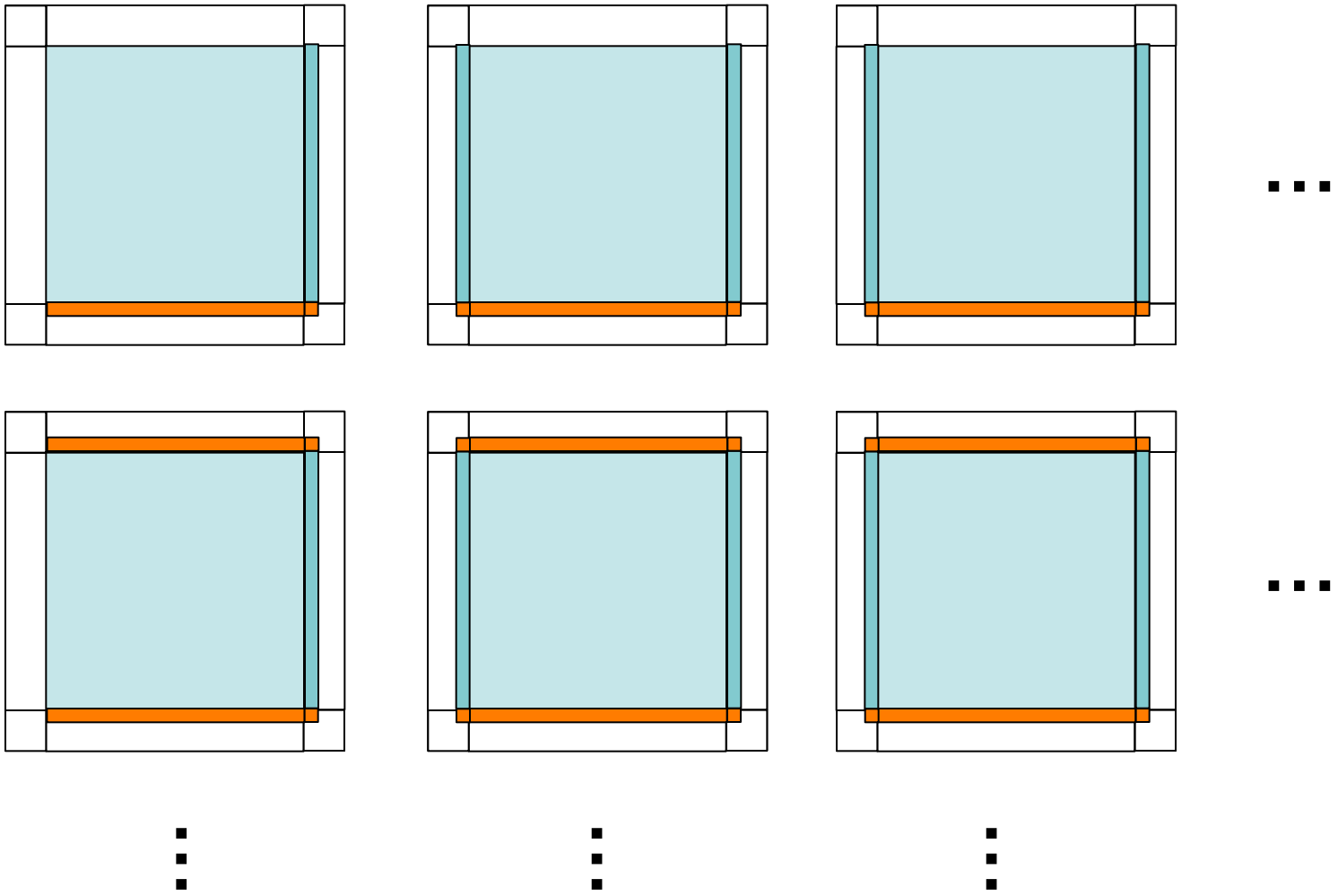
(Imagination)

1. EW-Exchange



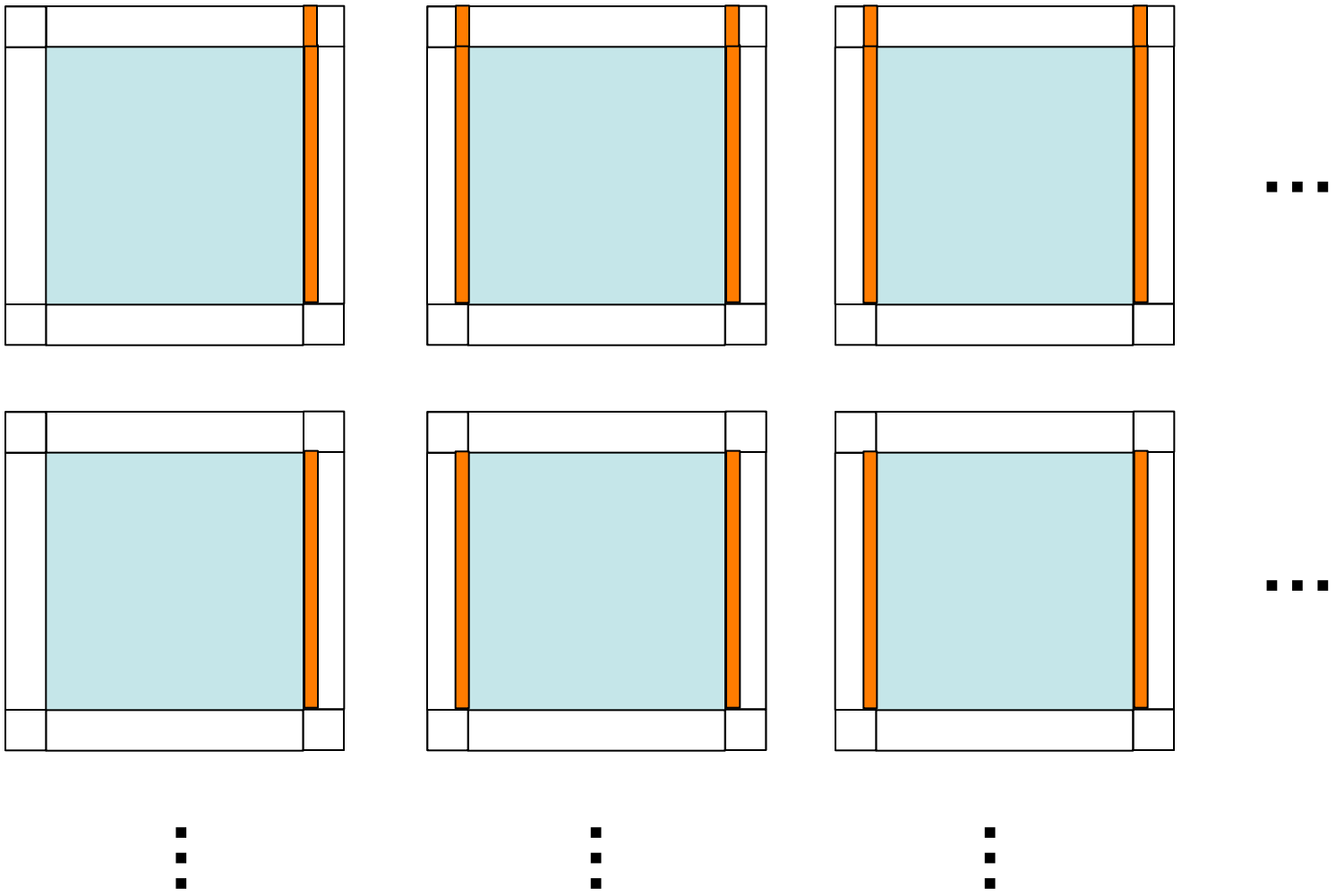
Corner Exchanges
(Imagination)

1. EW-Exchange
2. NS-Exchange



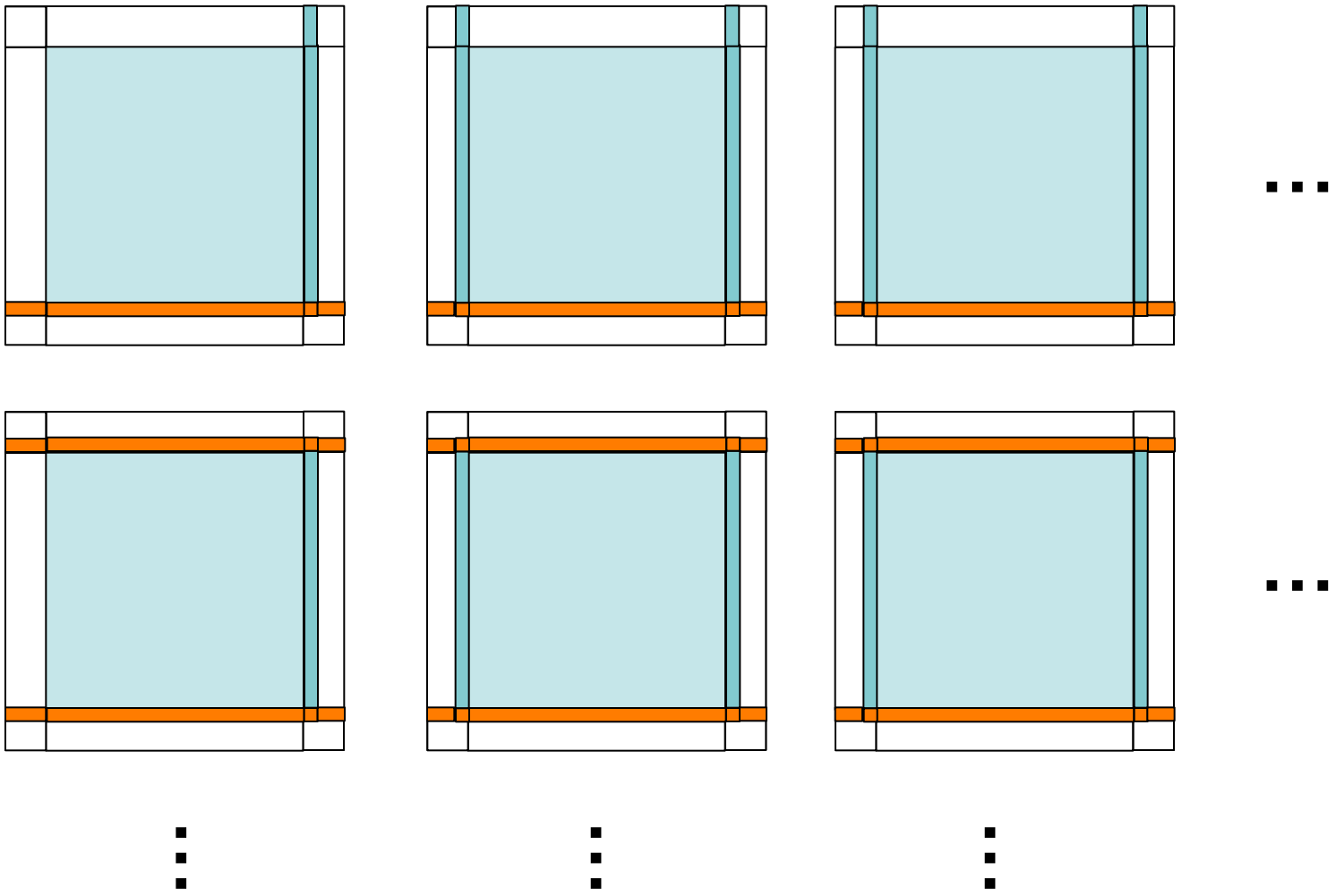
Exchanges “Reality”

1. EW-Exchange



Exchanges “Reality”

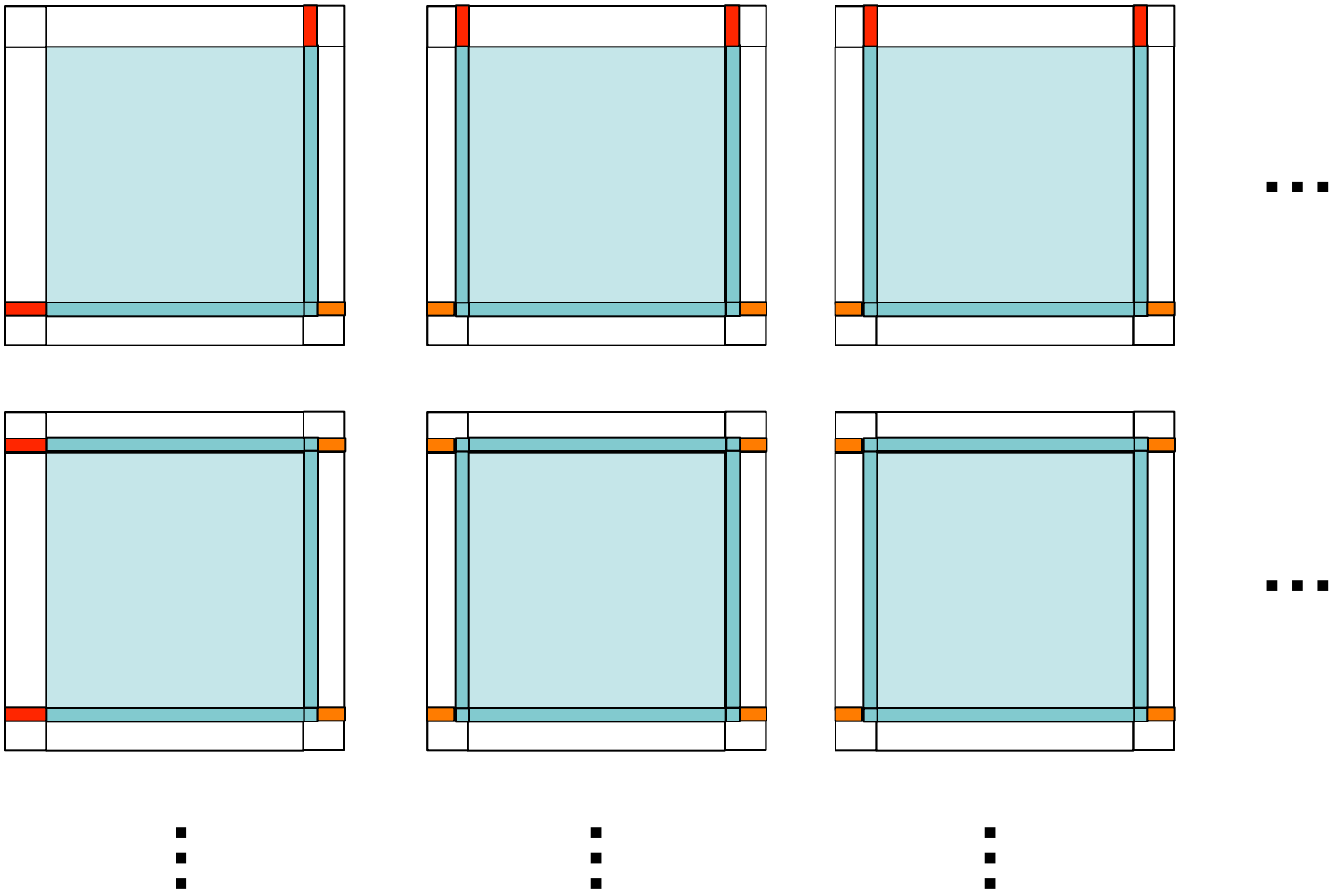
1. EW-Exchange
2. NS-Exchange



Exchanges “Reality”

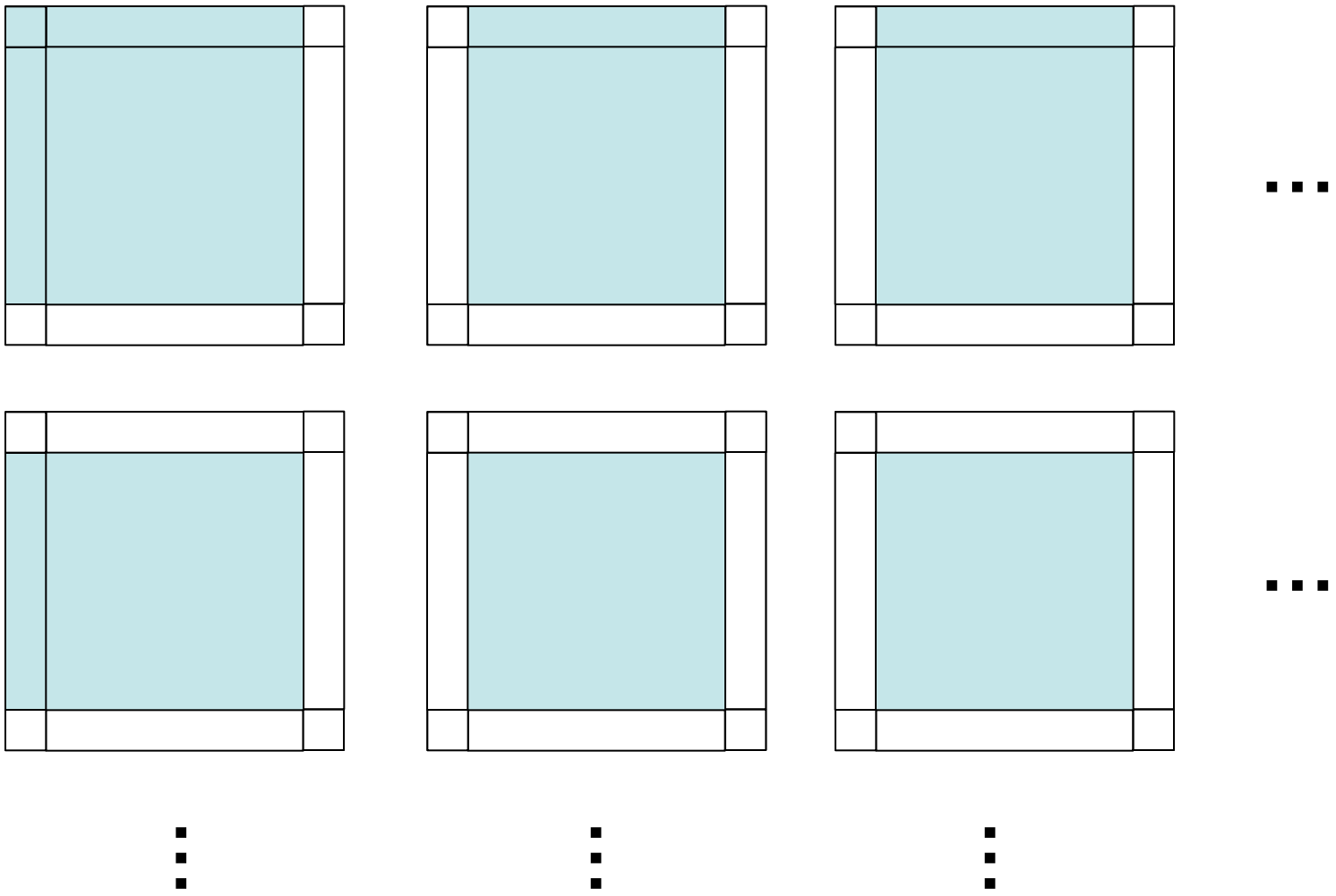
“Modified” boundary points

“Invalid” halo-points



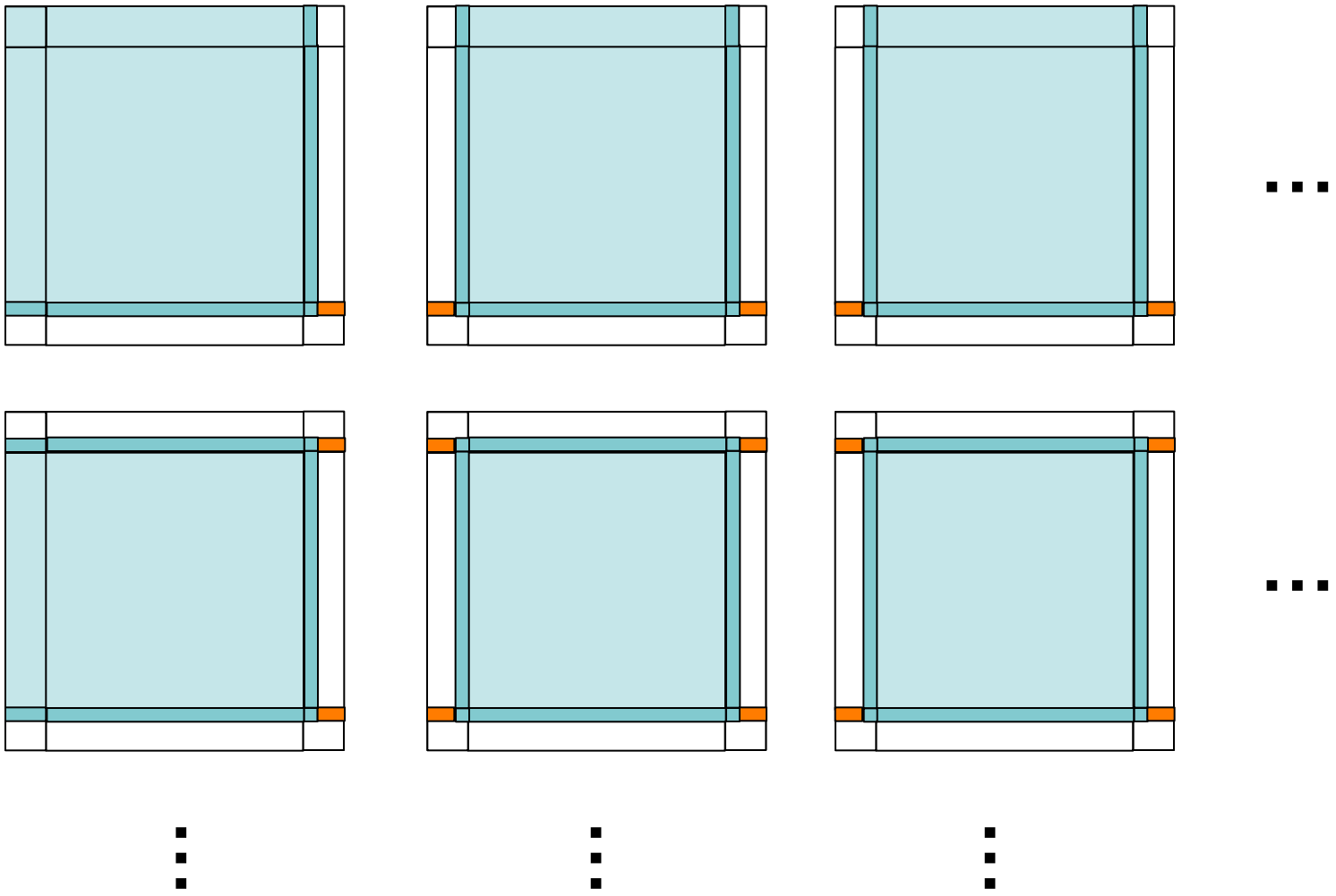
Application: Extended computations

(e.g. istartpar/endpar instead of istart/iend)



Application: Extended computations

(“artifacts” disappear only if nlines=nboundlines)



Code redundancy (error prone)

```
IF (lprog_qi .AND. lzconv) THEN
  kzdims(1:24) =                                                    &
     (/ke,ke,ke,ke,ke1,ke1,ke,ke,ke,ke,ke,ke,                       &
       ke,ke,ke,ke,ke,ke,ke,ke,ke,ke,1,0/)
  CALL exchg_boundaries                                             &
     (nnew+39, sendbuf, isendbuflen, imp_reals, icomm_cart, num_compute,  &
      ie, je, kzdims, jstartpar, jendpar,                           &
      nbl_exchg, nboundlines, my_cart_neigh,                        &
      lperi_x, lperi_y, l2dim,                                      &
      20000+ntstep, ldatatypes, ncomm_type, izerror, yzerrmsg,      &
      u (:,:,:,nnow), u (:,:,:,nnew), v (:,:,:,nnow), v (:,:,:,nnew),   &
      w (:,:,:,nnow), w (:,:,:,nnew), t (:,:,:,nnow), t (:,:,:,nnew),   &
      qv(:,:,:,nnow), qv(:,:,:,nnew), qc(:,:,:,nnow), qc(:,:,:,nnew),   &
      qi(:,:,:,nnow), qi(:,:,:,nnew), qr(:,:,:,nnow), qr(:,:,:,nnew),   &
      qs(:,:,:,nnow), qs(:,:,:,nnew), pp(:,:,:,nnow), pp(:,:,:,nnew),   &
      qrs(:,:,:)    , dqvdt(:,:,:)  , qvsflx(:,:) )
  IF (itype_gscp==4) THEN
    kzdims(1:24) =(/ke,ke,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0/)
    CALL exchg_boundaries                                           &
     ( 0, sendbuf, isendbuflen, imp_reals, icomm_cart, num_compute,     &
      ie, je, kzdims, jstartpar, jendpar,                           &
      nbl_exchg, nboundlines, my_cart_neigh,                        &
      lperi_x, lperi_y, l2dim,                                      &
      20000+ntstep, .FALSE., ncomm_type, izerror, yzerrmsg,         &
      qg (:,:,:,nnow), qg (:,:,:,nnew) )
  ENDIF
ELSEIF (lprog_qi .AND. .NOT. lzconv) THEN
  kzdims(1:24) =                                                    &
     (/ke,ke,ke,ke,ke1,ke1,ke,ke,ke,ke,ke,ke,                       &
       ke,ke,ke,ke,ke,ke,ke,ke,ke,0,0,0/)
  CALL exchg_boundaries                                             &
     (nnew+36, sendbuf, isendbuflen, imp_reals, icomm_cart, num_compute,  &
      ie, je, kzdims, jstartpar, jendpar,                           &
      nbl_exchg, nboundlines, my_cart_neigh,                        &
      lperi_x, lperi_y, l2dim,                                      &
      20000+ntstep, ldatatypes, ncomm_type, izerror, yzerrmsg,      &
      u (:,:,:,nnow), u (:,:,:,nnew), v (:,:,:,nnow), v (:,:,:,nnew),   &
      w (:,:,:,nnow), w (:,:,:,nnew), t (:,:,:,nnow), t (:,:,:,nnew),   &
      qv(:,:,:,nnow), qv(:,:,:,nnew), qc(:,:,:,nnow), qc(:,:,:,nnew),   &
      qi(:,:,:,nnow), qi(:,:,:,nnew), qr(:,:,:,nnow), qr(:,:,:,nnew),   &
      qs(:,:,:,nnow), qs(:,:,:,nnew), pp(:,:,:,nnow), pp(:,:,:,nnew),   &
      qrs(:,:,:)    )
  IF (itype_gscp==4) THEN
```

- subroutines
  exchange_runge_kutta()
  exchange_leapfrog()
- ~320 lines of code
- kzdims ↔ fields
- IF (lprog_qi) ELSE ENDIF
- lumping
- copy & paste of arguments

Lumping not possible

```
IF (l_cosmo_art) THEN
  IF (lgas) THEN
    DO isp = 1,isp_gas
      kzdims(1:24) =                                                    &
         (/ke,ke,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0/)
      CALL exchg_boundaries                                             &
         ( 0, sendbuf, isendbuflen, imp_reals, icomm_cart, num_compute,  &
          ie, je, kzdims, jstartpar, jendpar,                           &
          nbl_exchg, nboundlines, my_cart_neigh,                        &
          lperi_x, lperi_y, l2dim,                                      &
          20000+ntstep, .FALSE., ncomm_type, izerror, yzerrmsg,         &
          cgas(:,:,:,isp,nnow), cgas(:,:,:,isp,nnew))
    ENDDO
  ENDIF
```

- no lumping for 200 fields!

New design proposition

- Split exchg_boundaries() into two new methods
- Add a field to an halo-exchange job

```
CALL exchg_addjob( field, nlinesx, nlinesy, &
                   lcorner, lborder, ierror, yerrmsg )
```

- Executed halo-exchange for all pending jobs

```
CALL exchg_do( icase, ierror, yerrmsg )
```

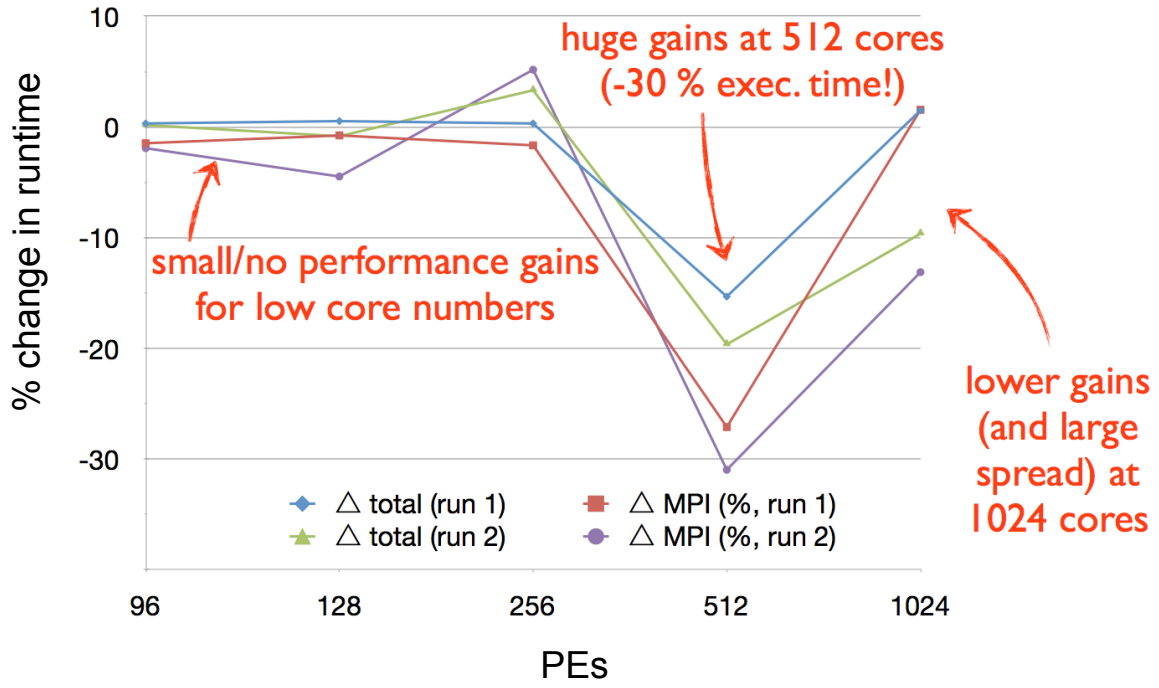
- Move to parallel_utilities.f90 instead of environment.f90

Usage (e.g. exchange_runge_kutta)

```
SUBROUTINE exchange_halos

  ! add time-dependent fields at nnew
  CALL exchg_addjob(  u(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob(  v(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob(  w(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob(  t(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob( pp(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob( qv(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob( qc(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  CALL exchg_addjob( qr(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  IF (lprog_qi) THEN
    CALL exchg_addjob( qi(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  ENDIF
  IF (itype_gscp > 1) THEN
    CALL exchg_addjob( qs(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  ENDIF
  IF (itype_gscp == 4) THEN
    CALL exchg_addjob( qg(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  ENDIF
  IF (lprog_tke) THEN
    CALL exchg_addjob( tke(:,:,:,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
  ENDIF
  ! add time-dependent fields at nnow for Leapfrog core
  IF (.NOT. l2tls) THEN
    ! ...same as above but with nnow
  ENDIF
  ! add time-independent fields
  CALL exchg_addjob( qrs, nbi, nbj, lzb, izerr, yzerrmsg )
  IF (lzconv) THEN
    CALL exchg_addjob( dqvdt, nbi, nbj, lzb, izerr, yzerrmsg )
    CALL exchg_addjob( qvsflx, nbi, nbj, lzb, izerr, yzerrmsg )
  ENDIF

  CALL exchg_do( 50+nnew, izerr, yzerrmsg )
  IF ( izerror /= 0 ) THEN
    WRITE(*,*) ' *** ERROR in executing the exchange jobs *** '
    RETURN
  ENDIF

END SUBROUTINE exchange_halos
```

Usage (e.g. tracers)

```
#ifdef COSMOART
  IF (l_cosmo_art) THEN
    IF (lgas) THEN
      DO isp = 1,isp_gas
        CALL exchg_addjob( cgas(:,:,:,isp,nnew), nbi, nbj, lzb, izerr, yzerrmsg )
      ENDDO
    ENDIF
  END IF
  CALL exchg_do( 50+nnew, izerror, yzerrmsg )
#endif
```



Courtesy: Christoph Knote (EMPA)

Advantages

- Much **less error prone** (less code, avoid copy&paste errors)
- Extensively **use lumping** (often more efficient)
- **Improved flexibility** for core tasks
  - specification of fields
    (number, rank, size)
  - specification of halo-exchange
    (number of lines, i/j-direction, corners, borders)
- **Integrated into parallel utilities**

Status

- First prototype implemented (v4.23 + tracer)
  - https://cosmo.cscs.ch/cosmo/branch/mch/olifu/feature/cosmo_newhaloexchg
  - Works!
  - Is not yet feature complete
  - Much better performance for tracer module and COSMO-ART demonstrated on Cray XT4

- **Next steps**
  - Feedback from you!
  - Code cleanup / extension
  - Full integration in COSMO
  - Testing

Discussion

- Any reasons why this will not work?
- Error handling?
  - Many calls to exchg_addjob()!
  - Is it not better to fail instead of returning ierror/yerrmsg which nobody checks anyway?
- Other?