## SHELL-SCRIPTS

```
xecham (tcsh-script):
    ./configure
    gmake
    cd data
    ../messy/xmessy

messy/xmessy (sh-script):
    ...
    USE_XXX=.true.
    NML_XXX=xxx.nml
    NML_XXX_T=xxx_t.nml
    ...

    # ECHAM5.nml
    # MESSy.nml
    ...
    # xxx.nml
    # xxx_t.nml
    ...
    # run the model
```

## NAMELISTS

```
ECHAM5.nml:
...

MESSy.nml:
& CTRL
CHECK = .true.
...
USE_XXX = .true.
...
/

xxx.nml:
...
& CTRL
...
/
& CPL
...
/
&RGTEVENTS
...
/
&REGRID
...
/
&REGRID
...
/

xxx_t.nml:
&REGRID
...
/
&REGRID
...
/
```

## BASE MODEL LAYER (CALL TREE)

```
master (src/master.f90)
  p_start (modules/mo_mpi.f90)
  control (src/control.f90)
    initsd (modules/mo_start_dataset.f90)
    ...
    initialize (src/initialize.f90)
      ...
      messy_initialize
      messy_new_tracer (src/call_submodels.f90)
    init_geoloc (modules/mo_geoloc.f90)
    ...
    init_memory (modules/mo_memory_streams.f90)
      messy_init_memory
    ...
    ioinitial (src/ioinitial.f90) / iorestart (src/iorestart.f90)
      ...
      messy_init_tracer (src/call_submodels.f90)
    ...
    messy_init_coupling
    stepon (src/stepon.f90)
      ...
      scan1 (src/scan1.f90)
        m_bufscan ; m_bufsc1
        sym2 ... ewd ... fftj ... dyn ... tf2 ...
        ...
        messy_global_start
        <ADVECTION>
          tf1
          m_buftrow(jrow)
          messy_local_start
          gpc (src/gpc.f90)
            physc (src/physc.f90)
              ...
              cover
              radiation
              vdiff
                ...
                messy_vdiff
                ...
                surftemp
                ...
              radheat
              gwspectrum
              ssodrag
              ~~cucall~~ messy_convec
              cloud
              ...
              surf
              lake
              licetemp
              m1_ocean
              sicetemp
              collect
              hydrology_collect
              messy_physc
              ...
          messy_local_end
          si1
          ...
          fftd ... si2 ... sym1 ... ltd ...
          ...
          messy_global_end
        ...
        out_streams (mo_grib.f90)
        scan2 ...
        write_streams (mo_io.f90)
        ...
    ...
    free_memory (modules/mo_memory_streams.f90)
      messy_free_memory
    ...
  p_stop (modules/mo_mpi.f90)
```

color-legend:

| | |
|---|---|
| ECHAM5 specific | namelist |
| MESSy-convention | TIME LOOP |
| NCREGRID-interface | REGION LOOP |
| Files | GENERIC SUBMODEL |

## BASE MODEL INTERFACE LAYER

messy/e5/messy_main_control_e5.f90

**INITIALIZATION PHASE**

```
SUBROUTINE messy_initialize
  USE messy_main_switch
  USE messy_main_switch_e5, ONLY: main_initialize
  USE messy_xxx_e5, ONLY: xxx_initialize
  ...
  CALL main_initialize
  ...
  IF (USE_XXX) CALL xxx_init
  ...
END SUBROUTINE messy_initialize

SUBROUTINE messy_new_tracer
  USE messy_main_switch
  USE messy_main_tracer_e5, ONLY: main_tracer_new_tracer
  USE messy_xxx_e5, ONLY: messy_new_tracer
  ...
  CALL main_tracer_new_tracer(1)
  ...
  IF (USE_XXX) CALL xxx_new_tracer
  ...
  CALL main_tracer_new_tracer(2)
END SUBROUTINE messy_new_tracer

SUBROUTINE messy_init_memory
  USE messy_main_switch
  USE messy_main_data_e5, ONLY: main_data_init_memory
  USE messy_main_tracer_e5, &
      ONLY: main_tracer_init_memory
  USE messy_xxx_e5, ONLY: xxx_init_memory
  ...
  CALL main_tracer_init_memory
  CALL main_data_init_memory
  ...
  IF (USE_XXX) CALL xxx_init_memory
  ...
END SUBROUTINE messy_init_memory

SUBROUTINE messy_init_tracer
  USE messy_main_switch
  USE messy_main_tracer_e5, ONLY: main_tracer_init_tracer
  USE messy_xxx_e5, ONLY: xxx_init_tracer
  ...
  CALL main_tracer_init_tracer(1)
  IF (USE_XXX) CALL xxx_init_tracer
  ...
  CALL main_tracer_init_tracer(2)
END SUBROUTINE messy_init_tracer

SUBROUTINE messy_init_coupling
  USE messy_main_switch
  USE messy_xxx_e5, ONLY: xxx_init_coupling
  ...
  IF (USE_XXX) CALL xxx_init_coupling
  ...
END SUBROUTINE messy_init_coupling
```

**TIME LOOP**

```
SUBROUTINE messy_global_start
  USE messy_main_switch
  USE messy_xxx_e5, ONLY: xxx_global_start
  ...
  IF (USE_XXX) CALL xxx_global_start
  ...
END SUBROUTINE messy_global_start
```

**REGION LOOP**

```
SUBROUTINE messy_local_start
  USE messy_main_switch
  USE messy_xxx_e5, ONLY: xxx_local_start
  ...
  IF (USE_XXX) CALL xxx_local_start
  ...
END SUBROUTINE messy_local_start

SUBROUTINE messy_vdiff, convec, physc
  USE messy_main_switch
  USE messy_xxx_e5, ONLY: xxx_vdiff, convec, physc
  ...
  IF (USE_XXX) CALL xxx_vdiff, convec, physc
  ...
END SUBROUTINE messy_vdiff, convec, physc

SUBROUTINE messy_local_end
  USE messy_main_switch
  USE messy_xxx_e5, ONLY: xxx_local_end
  ...
  IF (USE_XXX) CALL xxx_local_end
  ...
END SUBROUTINE messy_local_end
```

**TIME LOOP**

```
SUBROUTINE messy_global_end
  USE messy_main_switch
  USE messy_xxx_e5, ONLY: xxx_global_end
  ...
  IF (USE_XXX) CALL xxx_global_end
  ...
END SUBROUTINE messy_global_end

SUBROUTINE messy_free_memory
  USE messy_main_switch
  USE messy_main_tracer_e5, ONLY: &
      main_tracer_free_memory
  USE messy_xxx_e5, ONLY: xxx_free_memory
  ...
  IF (USE_XXX) CALL xxx_free_memory
  ...
  CALL main_tracer_free_memory
END SUBROUTINE messy_free_memory
```

netCDF output

restart files

## GENERIC SUBMODELS INTERFACE / CORE LAYER

messy/e5/messy_main_data_e5.f90

```
MODULE MESSY_MAIN_DATA_E5
  ...
! DATA TRANSFER VIA stream-elements
! BETWEEN ECHAM5 and MESSy-SUBMODELS
! AND BETWEEN MESSy SUBMODELS
  ...
END MODULE MESSY_MAIN_DATA_E5
```

messy/e5/messy_main_transform_e5.f90

```
MODULE MESSY_MAIN_TRANSFORM_E5
  ...
! SUBROUTINES FOR
! TRANSPOSITION AND TRANSFORMATIONS
! BETWEEN DIFFERENT REPRESENTATIONS
! (GRIDPOINT, LAGRANGE, SPECTRAL,
! LEGENDRE, FOURIER, ...)
  ...
END MODULE MESSY_MAIN_TRANSFORM_E5
```

messy/e5/messy_main_switch_e5.f90

```
MODULE MESSY_MAIN_SWITCH_E5
  USE messy_main_switch
  USE messy_xxx, ONLY: modstr_xxx => modstr &
                     , modver_xxx => modver
  ...
CONTAINS
  SUBROUTINE main_initialize
    USE ...
    IF (p_parallel_io) THEN
      iou = find_next_free_unit(100,200)
      CALL messy_main_read_nml(status, iou)
      IF (status /= 0) CALL finish(substr)
    END IF
    ...
    CALL p_bcast(USE_XXX)
    ...
    CALL put_submodel_nc_att(USE_XXX, &
         modstr_xxx, modver_xxx)
    ...
    CONTAINS
      SUBROUTINE put_submodel_nc_att
  END SUBROUTINE main_initialize
END MODULE MESSY_MAIN_SWITCH_E5
```

messy/e5/messy_main_tools_e5.f90

```
MODULE MESSY_MAIN_TOOLS_E5

CONTAINS
  SUBROUTINE start_message_e5
  SUBROUTINE end_message_e5
  SUBROUTINE zonal_average
  ...
END MODULE MESSY_MAIN_TOOLS_E5
```

messy/e5/messy_main_tracer_mem_e5.f90

```
MODULE MESSY_MAIN_TRACER_MEM_E5
  USE messy_main_tracer

  CHARACTER(LEN=*), PARAMETER :: &
       GPTRSTR = 'gptrac' &
       LGTRSTR = 'lgtrac'
  TYPE(t_trinfo_tp), DIMENSION(:), POINTER ::&
       ti_gp, ti_lg
  INTEGER :: ntrac_gp, ntrac_lg
  REAL(DP), DIMENSION(:,:,:,:), POINTER :: &
       pxt, pxtte, pxtm1, pxtf &
       pxt_a, pxtte_a, pxtm1_a, pxtf_a
  REAL(DP), DIMENSION(:,:,:), POINTER :: &
       xt, xtte, xtm1, xtf &
       xt_a, xtte_a, xtm1_a, xtf_a
  ...
END MODULE
MESSY_MAIN_TRACER_MEM_E5
```

messy/e5/messy_main_tracer_e5.f90

```
MODULE MESSY_MAIN_TRACER_E5
  USE messy_main_tracer_mem_e5
  USE messy_main_tracer
  ...
CONTAINS
  SUBROUTINE  main_tracer_new_tracer
  SUBROUTINE  main_tracer_init_memory
  SUBROUTINE  main_tracer_init_tracer
  SUBROUTINE  main_tracer_free_memory
  SUBROUTINE  tracer_init
  SUBROUTINE  tracer_halt
  ...
END MODULE  MESSY_MAIN_TRACER_E5
```

messy/e5/messy_ncregrid_tools_e5.f90

```
MODULE MESSY_NCREGRID_TOOLS_E5

TYPE RGTEVENT
SUBROUTINE RGTEVENT_INIT_NML
SUBROUTINE RGTEVENT_READ

SUBROUTINE RGTEVENT_STATUS

SUBROUTINE RGTOOL_E5_READ_NCVAR
SUBROUTINE RGTOOL_E5_READ_NCFILE

END
MODULE MESSY_NCREGRID_TOOLS_E5
```

xxx.rst

netCDF input

netCDF output

**NCREGRID**

messy_ncregrid_control.f90
(messy_ncregrid_diag.f90)
messy_ncregrid_netcdf.f90
messy_ncregrid_geohyb.f90
messy_ncregrid_base.f90
messy_ncregrid_mpi.f90
messy_ncregrid_interface.f90

messy/src/messy_main_constants_mem.f90

```
MODULE MESSY_MAIN_CONSTANTS_MEM

  CHARACTER(LEN=*), PARAMETER :: &
       modstr = 'MESSy' , &
       modver = '0.1e'

  INTEGER :: PARAMETER :: &
       SP = selected_real_kind(6,37) , &
       DP = selected_real_kind(12, 307) , &
       I4 = selected_int_kind(9) , &
       I8 = selected_int_kind(14) , &
       WP = DP , &
       nout = 6 , &
       nerr = 6

  REAL(DP), PARAMETER :: ...
  ...
END MODULE
MESSY_MAIN_CONSTANTS_MEM
```

messy/src/messy_main_switch.f90

```
MODULE MESSY_MAIN_SWITCH
  USE messy_main_constants_mem, ONLY: &
      i4, modstr

  LOGICAL, PARAMETER,PUBLIC :: &
       CHECK = .true.,    &
       USE_XXX = .false.  &
       ....
  NAMELIST / CTRL / CHECK, &
       ..., USE_XXX, ...

CONTAINS
  SUBROUTINE messy_main_read_nml(status, iou)
    ...
    status = 1
    call read_nml_open( ..., iou, 'CTRL', modstr )
    ...
    READ(iou, NML= CTRL)
    call read_nml_check( ..., iou, 'CTRL', modstr)
    ...
    call read_nml_close( ..., iou, modstr)
    status = 0
  END SUBROUTINE messy_main_read_nml
END MODULE MESSY_MAIN_SWITCH
```

messy/src/messy_main_tools.f90

```
MODULE MESSY_MAIN_TOOLS

  TYPE PTR_ND_ARRAY (N=1 ... 5)
  ...

CONTAINS
  SUBROUTINE read_nml_open
  SUBROUTINE read_nml_close
  SUBROUTINE read_nml_check
  SUBROUTINE start_message, end_message
  SUBROUTINE iso2ind, ind2val
  SUBROUTINE int2str
  SUBROUTEIN strcrack
  ...
END MODULE MESSY_MAIN_TOOLS
```

messy/src/messy_main_tracer.f90

```
MODULE MESSY_MAIN_TRACER

  TYPE t_trinfo
  TYPE t_trinfo_list
  TYPE t_tracerset
  ...

CONTAINS
  SUBROUTINE new_tracer_set
  SUBROUTINE new_tracer
  SUBROUTINE get_tracer_set
  SUBROUTINE get_tracer_set_id
  SUBROUTINE setup_tracer_set
  SUBROUTINE clean_tracer_set
  SUBROUTINE print_tracer_set
  SUBROUTINE print_tracer_set_val
  SUBROUTINE get_tracer
  SUBROUTINE tracer_error_str
  ...
END MODULE MESSY_MAIN_TRACER
```

messy/src/messy_ncregrid_tools.f90

```
MODULE MESSY_NCREGRID_TOOLS

  TYPE NCRGCNT

CONTAINS
  SUBROUTINE RGTOOL_READ_NCVAR
  SUBROUTINE RGTOOL_READ_NCFILE
  SUBROUTINE RGTOOL_CONVERT
  SUBROUTINE RGTOOL_G2C
  SUBROUTINE RGTOOL_NCRGCNT_RST
  ...
END MODULE MESSY_NCREGRID_TOOLS
```

## SUBMODEL INTERFACE LAYER

messy/e5/messy_xxx_e5.f90

```
MODULE messy_xxx_e5
  USE messy_xxx  ! SUBMODEL CORE LAYER
  USE messy_ncregrid_tools_e5, ONLY: RGTEVENT
  IMPLICIT NONE
  PRIVATE
  PUBLIC :: xxx_initialize, xxx_new_tracer, xxx_init_memory, xxx_init_tracer, &
            xxx_init_coupling, xxx_global_start, xxx_local_start, xxx_vdiff, xxx_convec, &
            xxx_physc, xxx_local_end, xxx_global_end, xxx_free_memory
  !PRIVATE :: xxx_read_nml_e5
  ! GLOBAL PARAMETERS / VARIABLES / CPL-SWITCHES
  TYPE(RGTEVENT), DIMENSION(:), POINTER :: RGT
CONTAINS

SUBROUTINE xxx_initialize
  USE mo_filename,              ONLY: find_next_free_unit
  USE mo_mpi,                   ONLY: p_parallel_io, p_bcast, p_io
  USE mo_exception,             ONLY: finish
  USE messy_ncregrid_tools_e5,  ONLY: rgtevent_init_nml
  IMPLICIT NONE
  CHARACTER(LEN=*), PARAMETER :: substr='xxx_initialize'
  INTEGER :: status, iou
  ...
  IF (p_parallel_io) THEN
      iou = find_next_free_unit(100,200)
      CALL xxx_read_nml(status, iou)  ! READ / CTRL
      IF (status /= 0) CALL finish(substr, '...')
  END IF
  CALL p_bcast(..., p_io)  ! BROADCAST NAMELIST
  ...
  IF (p_parallel_io) THEN
      iou = find_next_free_unit(100,200)
      CALL xxx_read_nml_e5(status, iou)  ! READ / CPL/
      IF (status /= 0) CALL finish(substr, '...')
  END IF
  CALL p_bcast(..., p_io)  ! BROADCAST NAMELIST
  ...
  CALL rgtevent_init_nml(RGT, modstr)  ! read /RGTEVENTS/ in xxx.nml; -> xxx.rst
END SUBROUTINE xxx_initialize

SUBROUTINE xxx_new_tracer
  USE messy_main_tracer_mem_e5,  ONLY: GPTRSTR, LGTRSTR
  USE messy_main_tracer,         ONLY: get_tracer, new_tracer, TR_NEXIST
  USE messy_main_tracer_e5,      ONLY: tracer_halt
  IMPLICIT NONE
  CHARACTER(LEN=*), PARAMETER :: substr = 'xxx_new_tracer'
  INTEGER :: status
  ...
  CALL get_tracer(status, GPTRSTR, ...)
  IF (status == TR_NEXIST) THEN
      CALL new_tracer(status, GPTRSTR, ...)
  ELSE
      CALL tracer_halt(substr, status)
  END IF
END SUBROUTINE xxx_new_tracer

SUBROUTINE xxx_init_memory
  USE mo_memory_base, ONLY: t_stream, new_stream, add_stream_element
  IMPLICIT NONE
  CHARACTER(LEN=*), PARAMETER :: substr = 'xxx_init_memory'
  TYPE(t_stream), POINTER :: stream
  ...
  CALL new_stream(stream, modstr)
  ...
  CALL add_stream_element(stream, <GLOBAL POINTER>, ...)
  ...
  ALLOCATE(...) ! GLOBAL NON-STREAM-FIELDS
  ...
END SUBROUTINE xxx_init_memory

SUBROUTINE xxx_init_tracer
  USE messy_main_tracer_e5, ONLY: tracer_init
  IMPLICIT NONE
  ...
  CALL tracer_init(modstr)  ! read /REGRID/ in xxx_t.nml
  ...
END SUBROUTINE xxx_init_tracer

SUBROUTINE xxx_init_coupling
  USE messy_main_tracer,    ONLY: get_tracer
  USE mo_memory_base,       ONLY: t_stream, get_stream, get_stream_element
  ...
END SUBROUTINE xxx_init_coupling

SUBROUTINE xxx_global_start
  USE messy_ncregrid_tools_e5, ONLY: ...
  IMPLICIT NONE
  CALL ...    ! read /REGRID/ in xxx.nml; update xxx.rst
  ...
END SUBROUTINE xxx_global_start

SUBROUTINE xxx_local_start, xxx_vdiff, xxx_convec, xxx_physc, xxx_local_end
  USE mo_exception,             ONLY: finish
  USE [ECHAM5, MESSy]:          ONLY: ...
  IMPLICIT NONE
  CHARACTER(LEN=*), PARAMETER :: substr = 'xxx_...'
  INTEGER :: status
  ...
  CALL xxx_<SMCL-routine>(status, ...)
  IF (status /=0) CALL finish(substr, '...')
  ...
END SUBROUTINE xxx_local_start, xxx_vdiff, xxx_convec, xxx_physc, xxx_local_end

SUBROUTINE xxx_global_end
  IMPLICIT NONE
  CALL ...
END SUBROUTINE xxx_global_end

SUBROUTINE xxx_free_memory
  IMPLICIT NONE
  DEALLOCATE(...) ! GLOBAL-NON-STREAM-FIELDS
END SUBROUTINE xxx_free_memory
```

DATA TRANSFER ...

! ... to be continued on next page

Patrick Jöckel, 23.08.2004

## SUBMODEL INTERFACE LAYER (continued)

**messy/e5/messy_xxx_e5.f90**

```
! ... continued

SUBROUTINE xxx_read_nml_e5(status, iou) ! PRIVATE
    USE mo_mz_messy_tools,  ONLY: read_nml_open , read_nml_check, read_nml_close
    IMPLICIT NONE
    ! I/O
    INTEGER, INTENT(OUT) :: status
    INTEGER, INTENT(IN)  :: iou
    ! LOCAL
    CHARACTER(LEN=*), PARAMETER :: substr = 'xxx_read_nml_e5'
    LOGICAL     :: lex          ! file exists ?
    INTEGER     :: fstat        ! file status

    NAMELIST /CPL/ ... ! ECHAM5 specific GLOBAL PARAMETERS / SWITCHES

    status = 1  ! ERROR
    ...
    CALL read_nml_open(lex, substr, iou, 'CPL', modstr)
    IF (.NOT.lex) RETURN
    READ(iou, NML=CPL, IOSTAT=fstat)
    CALL read_nml_check(fstat, substr, iou, 'CPL', modstr )
    IF (fstat /=0) RETURN

    ! CHECK NAMELIST AND DIAGNOSTIC OUTPUT
    ...
    CALL read_nml_close(substr, iou, modstr)
    status = 0  ! NO ERROR
END SUBROUTINE xxx_read_nml_e5

END MODULE  messy_xxx_e5
```

## SUBMODEL CORE LAYER

**messy/src/messy_xxx.f90:**

```
MODULE MESSY_XXX
    IMPLICIT NONE
    PRIVATE
    CHARACTER(LEN=*), PARAMATER, PUBLIC :: &
              modstr='xxx',  modver ='1.0a'
    ...
    PUBLIC :: xxx_read_nml
    PUBLIC :: xxx_sub_01
    PUBLIC :: ...
    ...
    ! PRIVATE :: xxx_sub_sub_01
    ! PRIVATE :: ...
    ...
    ! GLOBAL PARAMETERS / SWITCHES
    ! GLOBAL VARIABLES

CONTAINS

SUBROUTINE xxx_read_nml(status, iou)

    USE mo_mz_messy_tools,  ONLY: read_nml_open, read_nml_check, read_nml_close
    IMPLICIT NONE
    ! I/O
    INTEGER, INTENT(OUT) :: status
    INTEGER, INTENT(IN)  :: iou
    ! LOCAL
    CHARACTER(LEN=*), PARAMETER :: substr = 'xxx_read_nml'
    LOGICAL     :: lex          ! file exists ?
    INTEGER     :: fstat        ! file status

    NAMELIST / CTRL / ...    ! GLOBAL ...
                             ! ... PARAMETERS / SWITCHES

    status = 1  ! ERROR
    ...
    CALL read_nml_open(lex, substr, iou, 'CTRL', modstr)
    IF (.NOT.lex) RETURN
    READ(iou, NML=CTRL, IOSTAT=fstat)
    CALL read_nml_check(fstat, substr, iou, 'CTRL', modstr )
    IF (fstat /=0) RETURN

    ! CHECK NAMELIST AND DIAGNOSTIC OUTPUT
    ...
    CALL read_nml_close(substr, iou, modstr)
    status = 0  ! NO ERROR
END SUBROUTINE xxx_ read_nml

... VIA PARAMETER LIST

SUBROUTINE xxx_sub_01(status, ...)
    IMPLICIT NONE
    ! I/O
    INTEGER, INTENT(OUT) :: status = 1
    ...
    CALL xxx_sub_sub_01(...)
    IF (ERROR) RETURN
    ...
    status = 0
END SUBROUTINE xxx_sub_01

...

SUBROUTINE xxx_sub_sub_01(...)
    IMPLICIT NONE
    ...
END SUBROUTINE xxx_sub_01

...
END MODULE MESSY_XXX
```