Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatology MeteoSwiss

COSMO Software: fieldextra

Jean-Marie Bettems / MeteoSwiss

09.09.2015

COSMO GM

- **Core development** team

  Petra Baumann, Jean-Marie Bettems (SCA)

- With **contributions** from

  *Felix Ament, **Axel Barleben / DWD (new!)**, Philipp Glatt,*

  Christoph Hug, Pirmin Kaufmann (original code), Guy de Morsier,

  Donat Perler, Florian Prill / DWD, Anne Roches, Vanessa Stauch,

  Martin Schraner, Balazs Szintai, André Walser

Fieldextra
Identity card (1)

- **Generic tool** to process model data and gridded observations
  - Provide a set of primitive operations, which can be combined (**toolbox**)
  - Single **Fortran 2003 program** controlled by **namelists**

- **File based** input/output
  - Support **GRIB1** and **GRIB2** input (incl. files with mixed records)
  - Understand COSMO specificities (file name, GRIB **local** extensions)
  - Rich set of output format (GRIB1, GRIB2, NetCDF, CSV, XML ...)

- Primary focus is the **production environment**
  - **Robust** (no crash, recovery after exception, scale with problem size)
  - **Optimized** (time to solution, memory footprint, input/output)
  - **Safe** (sytematic check of operations and products consistency)
  - Comprehensive **diagnostic** and **profiling**

Fieldextra
Identity card (2)

• First release, May 1998		
• COSMO GM 2012 :	10.5.2	*~ 82k lines*
• COSMO GM 2013 :	11.1.0	*~ 102k lines*
• COSMO GM 2014 :	11.3.3	*~115k lines*
• **Next release, Oct 2015 :**	**12.1.0**	*~128k lines*

More than 13 FTE invested

Steady and sustained development effort

*About **1 FTE / year**, at MeteoSwiss and DWD*

Fieldextra

Identity card (3)

- **Availability**
  - Package on **COSMO web site** (main release only!)
    http://www.cosmo-model.org/content/support/software/default.htm
  - **Full installation at ECMWF on cca** (special UNIX group cfxtra)
    /perm/ms/ch/ch7/projects/fieldextra
  - **Full installation at CSCS** (for special users only)
    /users/tsm/project/fieldextra

- **Community support**
  - **Tutorial** with hands-on (no regular schedule, on-demand)
  - **Mailing list** (57 members, about 150 messages posted since last GM)
    cosmo-fieldextra@cosmo-model.org

Fieldextra

Developments since last COSMO GM (→ 12.1.0, Oct 2015)

About 80 new or updated features, driven by user needs …

- Fully compatible with new COSMO GRIB API environment
- Consolidate GRIB 2 support, in particular local use
- Optimizations of memory and time to solution (in particular for EPS)
- *Extend vertical interpolation and lateral re-gridding (CORSO-A)*
- Support of synthetic satellite fields, incl. operator (RTTOV 11.2.0)
- Many new operators…
- As in any new major release:
  - Bugs correction
  - Internal improvement of code
  - Consolidation and extension of existing features
  - Improvement of user interface and of diagnostic
  - Improvement of README.user

➢ **See the HISTORY file for the whole story !**

Fieldextra
Roadmap

Integrated information and development platform (→ Q1 2016)

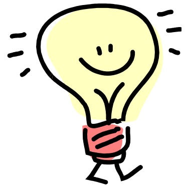
- Goal: support distributed development, get more developers on board
- Accessible from any COSMO center for selected users
- Code repository (Git), code review
- Regression test
- Roadmap, history, bug tracking, feature request

More (→ GM 2016)

- Improved documentation (in particular extended ‘cookbook’)
- Asynchronous input
- **Planning to be further defined … Feedback always welcome!**

Fieldextra

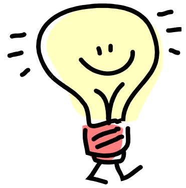
GRIB 2 and GRIB API



- Decoding and encoding GRIB 2 records is based on the system
  host software + GRIB API + definition files + sample
  - Full fledge implementation is **complex**
    (GRIB 2 : rich set of meta-information, incl. local usage;
    API : multiple software layers, poorly documented)
  - API is prone to **silent errors**
    (e.g. incorrect meta-information by changing the originating centre value)
  - System is hard to **debug**

Fieldextra

GRIB 2 and GRIB API



- Mitigating measures, as defined in COSMO GRIB2 policy
  - Build a **single system** compatible with all COSMO software and all COSMO centres
    → In progress: G API 1.13.1, fieldextra 12.1.0, COSMO 5.3(4), INT2LM …
  - Comprehensive and strict **testing** before approving any system change
    → COSMO Technical Test Suite
  - Extended **documentation**
    → New COSMO web page
    → Lobbying at ECMWF to improve document & debug mode
  - **Limit** the number of COSMO implementation
    → Focus on COSMO software, TAG coordinates software interfaces

Fieldextra
User case: DWD

- Contribution of Axel Barleben to core development (!)
  - Clear Air Turbulence (CAT) and other operators
- Installation on operational system (Cray Linux Cluster)
  - fieldextra release 11.3.3 (04.09.2014)
  - fieldextra pre-release 12.x
- Usage in operations
  - COSMO model data for German ATC (Deutsche Flugsicherung DFS) and FABEC (Functional Air Block Europe Central)
  - Postprocessing of ICON model

Thank you for your attention!

```
!+****************************************************************************
SUBROUTINE generate_output(multi_pass_mode, just_on_time, last_call,           &
                  datacache, data_origin, tot_nbr_input,                  &
                  out_paths, out_types, out_modes,                        &
                  out_grib_keys, out_spatial_filters,                     &
                  out_subset_size, out_subdomain, out_gplist, out_loclist, &
                  out_data_reduction, out_postproc_modules,               &
                  nbr_gfield_spec, gen_spec, ierr, errmsg                 )
!==============================================================================
!
! Root procedure to generate output files
!
!------------------------------------------------------------------------------
 ! Dummy arguments
 LOGICAL, INTENT(IN)                         :: multi_pass_mode     ! Multiple pass mode?
 LOGICAL, DIMENSION(:), INTENT(IN)           :: just_on_time        ! True if prod. now
 LOGICAL, INTENT(IN)                         :: last_call           ! True if last call
 CHARACTER(LEN=*), INTENT(IN)                :: datacache           ! Data cache file
 TYPE(ty_fld_orig), INTENT(IN)               :: data_origin         ! Data origin
 INTEGER, DIMENSION(:), INTENT(IN)           :: tot_nbr_input       ! Expected nbr. input
 CHARACTER(LEN=*), DIMENSION(:), INTENT(IN)  :: out_paths           ! Output files names
 TYPE(ty_out_spec), DIMENSION(:), INTENT(IN) :: out_types           !   types
 TYPE(ty_out_mode), DIMENSION(:), INTENT(IN) :: out_modes           !   modes
 INTEGER, DIMENSION(:,:), INTENT(IN)         :: out_grib_keys       !   grib specs
 INTEGER, DIMENSION(:), INTENT(IN)           :: out_subset_size     !   subset size
 INTEGER, DIMENSION(:,:), INTENT(IN)         :: out_subdomain       !   subdomain definition
 INTEGER, DIMENSION(:,:,:), INTENT(IN)       :: out_gplist          !   gp definition
 CHARACTER(LEN=*), DIMENSION(:,:), INTENT(IN) :: out_loclist        !   locations definition
 CHARACTER(LEN=*), DIMENSION(:,:), INTENT(IN) :: out_spatial_filters ! Condition defining filter
 TYPE(ty_out_dred), DIMENSION(:), INTENT(IN) :: out_data_reduction  ! Data reduction spec
 CHARACTER(LEN=*), DIMENSION(:), INTENT(IN)  :: out_postproc_modules ! Specific postprocessing
 INTEGER, DIMENSION(:,:), INTENT(IN)         :: nbr_gfield_spec     !+ Specifications of
 TYPE(ty_fld_spec_root), DIMENSION(:), INTENT(IN) :: gen_spec       !+ fields to generate
 INTEGER, INTENT(OUT)                        :: ierr                ! Error status
 CHARACTER(LEN=*), INTENT(OUT)               :: errmsg              !   error message

 ! Local parameters
 CHARACTER(LEN=*), PARAMETER           :: nm='generate_output: ' ! Tag

 ! Local variables
 LOGICAL                               :: exception_detected, exception, use_postfix
 LOGICAL                               :: unique_ftype, multiple_grid, exist
 LOGICAL, DIMENSION(3*mx_iteration+1)  :: tmp_fddata_alloc, tmp_gpdata_alloc
 LOGICAL, DIMENSION(3*mx_iteration+1)  :: tmp_value_alloc, tmp_flag_alloc
 INTEGER                               :: i1, i2, i3, i_fd, i_vd
 INTEGER                               :: nbr_input
 INTEGER                               :: out_idx, ios, idx_vd_defined
 CHARACTER(LEN=strlen)                 :: messg, temporal_res, out_path
 TYPE(ty_fld_type)                     :: out_ftype



 ! Initialize variables
 !---------------------
 ierr = 0  ;  errmsg = ''
 exception_detected = .FALSE.
 tmp_fddata_alloc(:) = .FALSE.  ;  tmp_gpdata_alloc(:) = .FALSE.
 tmp_value_alloc(:) = .FALSE. ; tmp_flag_alloc(:) = .FALSE.


 ! Create/update data cache file
 !------------------------------------------------------------------------
 ! The cache file must reflect the state of data(:) after the last call to
 ! collect_output (i.e. before any field manipulation done in prepare_pout)
```

```
! Loop over each output file
!---------------------------
output_file_loop: &
DO i1 = 1, nbr_ofile
  out_idx = data(i1)%ofile_idx
  nbr_input = COUNT( data(i1)%ifile_used )

  ! Skip bogus output
  IF ( data(i1)%ofile_bogus ) CYCLE output_file_loop
  ! Skip completed output
  IF ( data(i1)%ofile_complete ) CYCLE output_file_loop
  ! Skip empty data array
  IF ( ALL(.NOT. data(i1)%defined) ) CYCLE output_file_loop
  ! Only prepare output when all possible associated data have been collected
  ! or when 'just on time' production is active
  IF ( .NOT. last_call                  .AND.            &
       nbr_input < tot_nbr_input(out_idx) .AND.          &
       .NOT. just_on_time(out_idx)          ) CYCLE output_file_loop

  ! At this point the corresponding output file will be produced
  ! Keep track of completed output file
  IF ( nbr_input >= tot_nbr_input(out_idx) ) data(i1)%ofile_complete = .TRUE.

  ! Build name of output, considering a possible temporary postfix
  use_postfix = .FALSE.
  IF ( LEN_TRIM(out_postfix) /= 0 .AND. data(i1)%ofile_usepostfix .AND.    &
       .NOT. (data(i1)%ofile_firstwrite .AND. data(i1)%ofile_complete)  )  &
                                     use_postfix = .TRUE.
  out_path = out_paths(out_idx)
  IF ( use_postfix ) out_path = TRIM(out_path) // out_postfix

  ! Release memory allocated in previous call to prepare_pout (if any)
  DO i2 = 1, 3*mx_iteration+1
    IF ( tmp_value_alloc(i2) ) DEALLOCATE(data_tmp(i2)%values, data_tmp(i2)%defined)
    IF ( tmp_flag_alloc(i2) ) DEALLOCATE(data_tmp(i2)%flag)
    IF ( tmp_fddata_alloc(i2) ) THEN
      DEALLOCATE(data_tmp(i2)%field_type, data_tmp(i2)%field_origin,        &
data_tmp(i2)%field_name, data_tmp(i2)%field_grbkey,          &
                 data_tmp(i2)%field_trange,                       &
                 data_tmp(i2)%field_level, data_tmp(i2)%field_ltype,       &
                 data_tmp(i2)%field_prob, data_tmp(i2)%field_epsid,        &
                 data_tmp(i2)%field_vref, data_tmp(i2)%field_ngrid,        &
                 data_tmp(i2)%field_scale, data_tmp(i2)%field_offset,      &
                 data_tmp(i2)%field_vop, data_tmp(i2)%field_vop_usetag,    &
                 data_tmp(i2)%field_vop_nlev, data_tmp(i2)%field_vop_lev,  &
                 data_tmp(i2)%field_pop, data_tmp(i2)%field_hop,           &
                 data_tmp(i2)%field_top, data_tmp(i2)%nbr_level,           &
                 data_tmp(i2)%level_idx, data_tmp(i2)%nbr_eps_member,      &
                 data_tmp(i2)%eps_member_idx, data_tmp(i2)%field_idx       )
    ENDIF
    IF ( tmp_gpdata_alloc(i2) ) THEN
      DEALLOCATE(data_tmp(i2)%gp_coord, data_tmp(i2)%gp_idx,               &
                 data_tmp(i2)%gp_lat, data_tmp(i2)%gp_lon, data_tmp(i2)%gp_h)
    ENDIF
  END DO

  ! Prepare data for print out (calculate new fields, ... ; populate data_pout)
  ! * Info message
  IF ( just_on_time(out_idx) ) THEN
    messg = ' (just on time output)'
  ELSE IF ( nbr_input >= tot_nbr_input(out_idx) ) THEN
    messg = ' (all associated input collected)'
  ELSE
    messg = ''
  ENDIF
```