



Schweizerische Eidgenossenschaft  
Confédération suisse  
Confederazione Svizzera  
Confederaziun svizra

Eidgenössisches Departement des Innern EDI  
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

# COSMO software

Jean-Marie Bettems / MeteoSwiss

St Petersburg, SMC, September 2018

# COSMO software



## *Official COSMO software*

- **COSMO** → WG6 / U. Schaettler
- **INT2LM** → WG6 / D. Rieger
- **EXTPAR** → WG3b / K. Osterried (*report on Thursday*)
- **Fieldextra** → WG4 / JM. Bettems
- **VERSUS** → WG5 / ???
- **SNOWE** → WG3b / I. Rozinkina

## *Additional utilities*

- **CALMO Meta-Model** → WG3b / I. Carmona
- **TERRA standalone** → WG3b / Y. Ziv

## *And more ...*

- **bufrx2netcdf, MEC, Rfdbk, DACE...**

# 20 years fieldextra...



## ... 1998

- **Swiss Model (HRM)** : *hydrostatic, 14km grid size, 125x125 points*
- **Cray J90 cluster at ETHZ** : *60' for 48h forecast*
- **UNICOS 9.0** : *restricted set of tools (C, fortran, sh, csh)*
- The Swiss Model popularity is increasing, tools are required to generate products out of the model output
- Decision to (1) use a *flat file system* as database, (2) generate the products *concurrently* to the computation of the model, (3) use a *single program* for all products

→ **May 1998, fieldextra is born!**

# ... 2018

- **Fieldextra** is the official **COSMO software** for model post-processing
- Many **users**, multiple **models**
  - *MeteoSwiss*: processing of KENDA, COSMO-1, COSMO-E, COSMO-7, IFS-HRES, IFS-ENS, IFS-SEAS, INCA
  - *DWD*: products based on COSMO-DE2, ICON-LAM & ICON-LAM-EPS
  - *COSMO-LEPS* production at ECMWF
  - And more : USAM, RHM, NMA, IMS...
- About **165k lines** of code
  - Still **actively developed code**  
(2 - 3 releases per year)

# Fieldextra

## Availability

- **Full code and documentation** on GitHub  
<https://github.com/MeteoSwiss-APN/fieldextra>
- **Standalone package** on COSMO web site (main releases 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>



# Fieldextra

## Releases

- Latest public release is **v12.7.1** (09.05.2018)
- **Next planned release is v12.8.0 (before year end)**

## Some highlights since COSMO GM 2017

- Add support for **ART products** (as requested by Jochen F)
- Computation of **vertical cross-section** along arbitrary path (NetCDF)
- Extended **lateral re-gridding** (conditional source points, generalized distance)
- Improve **OpenMP performances**
- **Cookbook** with more than 50 fully commented examples
- **NetCDF** import (*in progress*)

# Fieldextra

## Planned for v13.0.0 (→ Q1 2019)

- Migration from GRIB API to **ecCodes**
- Consolidation of **regression suite**
- Streamline code installation using **CMake**
- Better support of **complex namelist**



# Fieldextra

## Other actions

- Licences can now be granted to the **R&D community**
- Evaluation of **code life cycle**
  - Questionnaire has been distributed to the fieldextra community
  - Session at WG4 on Tuesday

## *Requested features, by priority*

- ☐ Full support of ICON grid
- ☐ Optimization for very large problems
- ☐ More interactive documentation (WiKi)



Schweizerische Eidgenossenschaft  
Confédération suisse  
Confederazione Svizzera  
Confederaziun svizra

Eidgenössisches Departement des Innern EDI  
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

# TERRA standalone



# TERRA standalone

- Available on **GitHub** and on **COSMO web**
- Based on **COSMO 5.03**
- Maintained by **IMS** (best effort)
- Recent **bug fixes** by Daniel Regenass / MCH
- Convenient tool for **soil spin-up**  
(CALMO, NWP test suite, hydrological cycle...)
- Some work required to port the software to the latest COSMO release  
**...but no resources currently assigned to this task**
- **Is this functionality integrated in ICON?**



Schweizerische Eidgenossenschaft  
Confédération suisse  
Confederazione Svizzera  
Confederaziun svizra

Eidgenössisches Departement des Innern EDI  
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

# CALMO meta-model



# CALMO Meta Model

- Available on **GitHub** and on **COSMO web**
- Being improved by IMS colleagues within the frame of **PP CALMO-MAX**
- Will be synchronized with **parallel developments at ETHZ**
- Plan to port the software to **Octave**  
(currently the software is written in MatLab)

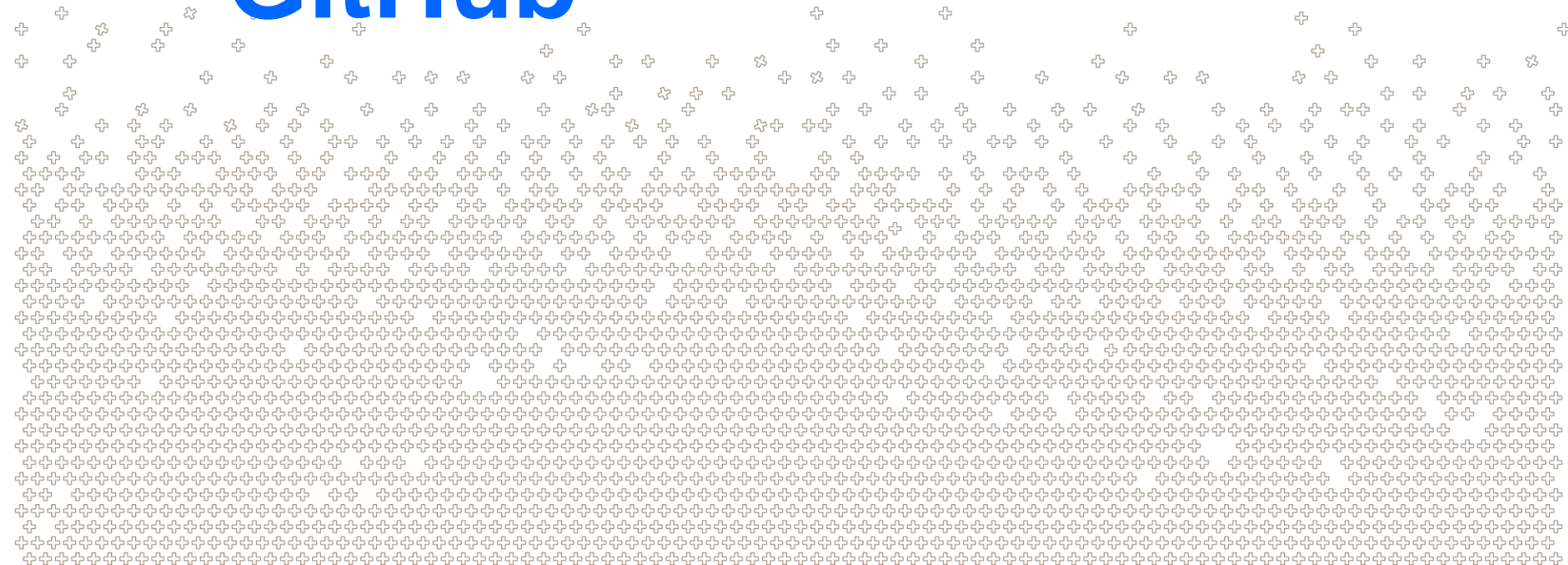


Schweizerische Eidgenossenschaft  
Confédération suisse  
Confederazione Svizzera  
Confederaziun svizra

Eidgenössisches Departement des Innern EDI  
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

# Sharing development

## GitHub



# GitHub

**GitHub** is a **web based service** for source code management, based on **git**  
**Git** is a distributed **version control system** (think SVN...)

## GitHub offers ...

- Tools for **sharing code development in a distributed environment**  
*code review , discussion board, tasks management, ...*
- **Easy to use information platform**  
*markup language, integrated WiKi and PDF reader, commits history...*
- Integration with **automatic testing procedure** (Jenkins)
- And more ...



# GitHub

Search or jump to...

Pull requests Issues Marketplace Explore

**COSMO-ORG**

Repositories 3 People 6 Teams 3 Projects 0

Search repositories... Type: All Language: All New

**terra-standalone** Private  
Contains standalone version of the terra soil model  
Fortran Updated on Jul 13

**int2lm** Private  
int2lm - the interpolation software for the COSMO-model  
Fortran Updated on Jul 9

**CALMO-MM**  
Matlab MIT Updated on Nov 20 2017

Top languages  
Fortran Matlab

People 6 >  
[Avatar 1] [Avatar 2] [Avatar 3] [Avatar 4] [Avatar 5] [Avatar 6]

© 2018 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About



```

[+*****
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_coord), DIMENSION(:,:), INTENT(IN) :: out_coord ! Data reduction
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)%file_firstwrite .AND. data(i1)%ofile_complete) ) &
     use_postfix = .TRUE.
```

```
out_path = out_paths(out_idx) &
```

```
              ( use_postfix ) || data(i1)%file_ext || out_idx ||
```

```
! 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
```