Quality Control (QC) of Climate Model Meta-data

Dr. Heinz-Dieter Hollweg

Abteilung Datenmanagement

Deutsches Klimarechenzentrum GmbH (DKRZ)

Bundesstraße 45a • D-20146 Hamburg • Germany

Email: hollweg@dkrz.de

Benefits

1. **Modellers Site**

   reduced costs.
   avoid transfer of flawed data sets.

2. **Archives and Data Nodes (ESGF)**

   ensurance that stored data is reliable.
   touch data only once.

3. **Customers**

   no programming to adapt downloaded files,
   get what is expected,
   facilitated handling for intercomparisons.

4. **Harvesters**

History

<2007: (from a Radiative Past)

- C++ frame-program with netCDF operations.

2007 – 2009: QC-0.2 (German Project CLM)

- Automatic checks of large data sets (netCDF).
- Fine grained selection of data sub-sets and files (RegExpr).
- Parallel processing in a Bash environment.
- Focus on correct time values and data bodies.
- Consistency of meta-data of sequential (sub-temporal) files.

History

2010 – 2012: QC-0.3 (CMIP5)

- Evolutionary coding of algorithms and checks.
- Meta-data of files are compared to the CMIP5 standard-output table.
- Consistency checks across sequential experiments.
- Time period table for the various CMIP5 experiments.
- SVN repository and distribution of the QC package.
- Optional permanent surveillance of a drop-zone where users put tasks, which are automatically executed (cron-job triggered).

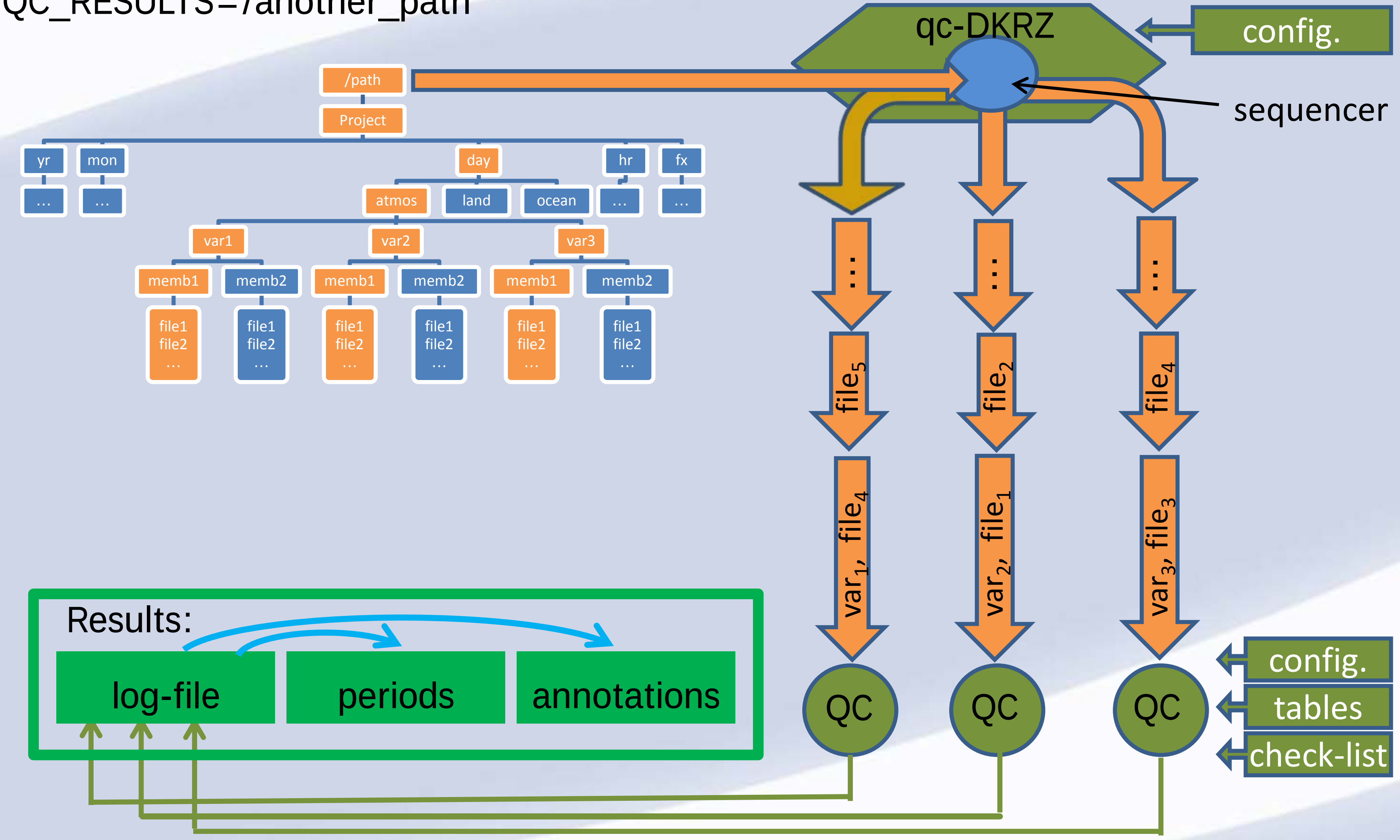
History

2012 - 2014: QC-0.4 (CORDEX)

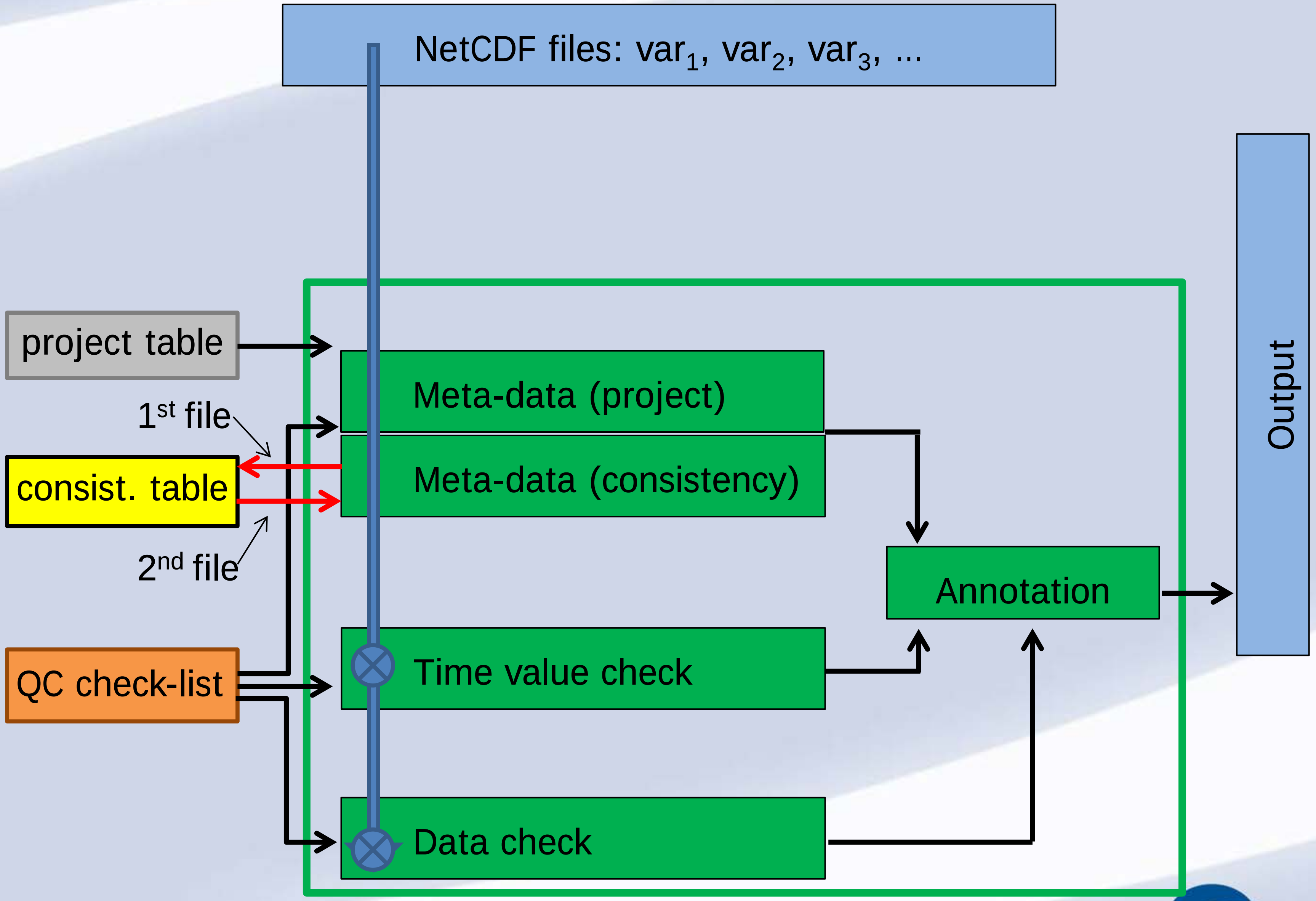
- Easy installation in User Space.
- Modularisation of checks:
  annotation, meta-data of a project, time values, data.
- Table for the CORDEX Archive Design:
  DRS, CV, required features of attributes, domains, time coordinate, etc.
- CORDEX variable requirements table.
- Cascaded config-files: general, project, experiment task.
- Adjustable check-list (with a default for the project).
- Automatic updates of QC and updating of modified CORDEX tables.

Work-flow

```
PROJECT_DATA=/path/Project
SELECT day/atmos/.*/memb1
QC_RESULTS=/another_path
```

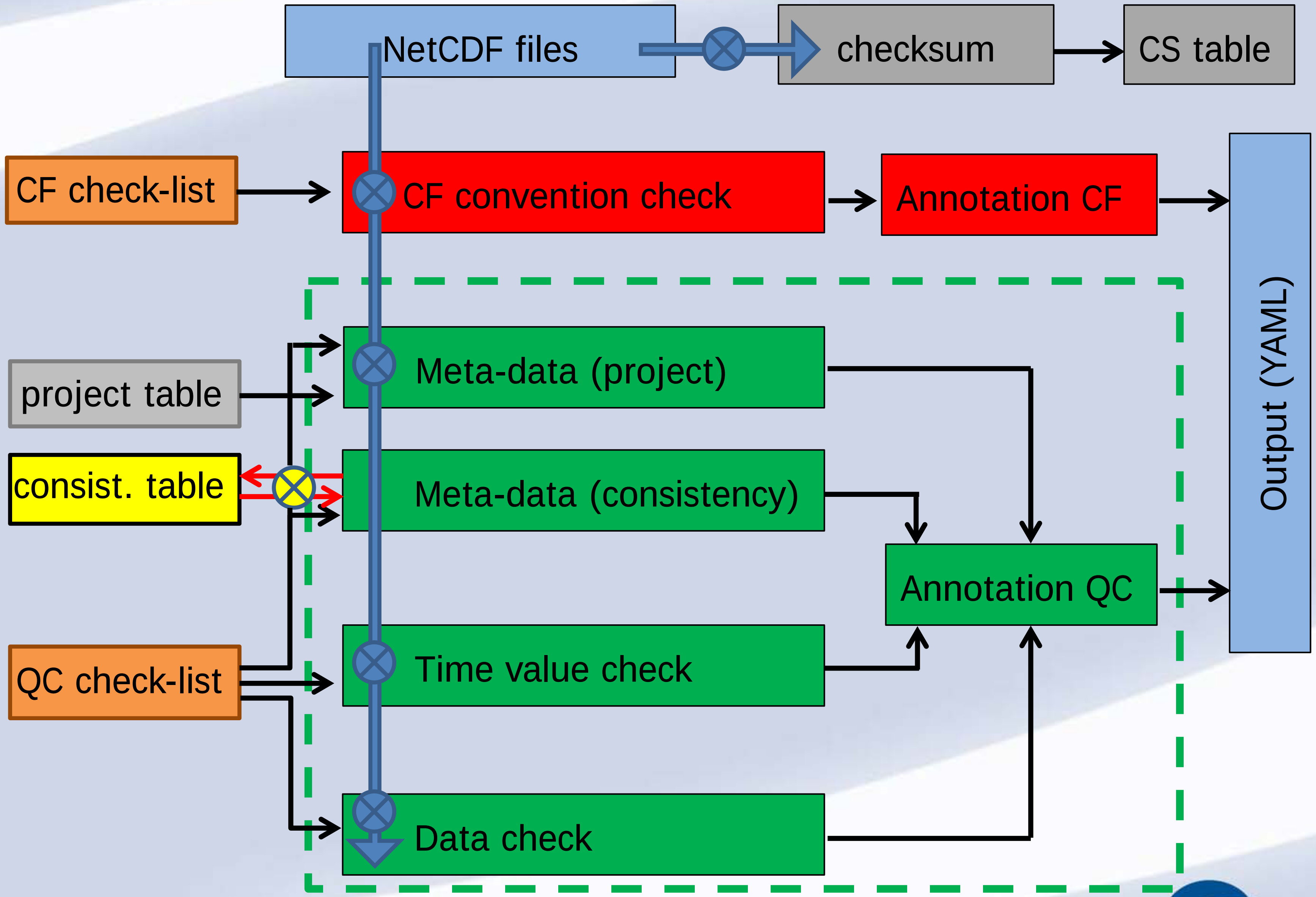


QC-0.4


NetCDF files: var$_1$, var$_2$, var$_3$, ...
project table
1st file
consist. table
2nd file
QC check-list
Meta-data (project)
Meta-data (consistency)
Time value check
Data check
Annotation
Output

DKRZ

QC-0.5


NetCDF files
checksum
CS table
CF check-list
CF convention check
Annotation CF
Output (YAML)
project table
Meta-data (project)
consist. table
Meta-data (consistency)
Annotation QC
QC check-list
Time value check
Data check

Requirements

Libraries

• zlib	CORDEX	www.zlib.net
• hdf5	CORDEX	www.hdfgroup.org/HDF5
• netcdf-	3: CMIP5	www.unidata.ucar.edu/netcdf
	4: CORDEX	
• udunits2	CF Conv.	www.unidata.ucar.edu/software/udunits

Requirements

CMIP5 http://cmip-pcmdi.llnl.gov/cmip5/docs

• CMIP5 DRS & CV	⟹ hard-coded
• Standard_output.xls (includes time table info)	⟹ csv-table

CORDEX https://github.com/IS-ENES-Data/IS-ENES-Data.github.io

• cordex_archive_specifications	⟹ csv-table
• CORDEX_variables_requirement_table	⟹ csv-table
• CORDEX_ToU_RCMModel.txt	⟹ as is

CF Conv. http://cfconventions.org

• cf-conventions.pdf (v1.4 – v1.6, draft v1.7)	⟹ hard-coded
• cf-standard-name-table.xml	⟹ as is
• standardized-region-names.html	⟹ as is
• area-type-table.html	⟹ hard-coded

DKRZ

Package Structure

```
QC-DKRZ
├── bin
├── CF-TestSuite
│   ├── Pass
│   └── Fail
├── doc
├── example
├── include
├── scripts
├── src
└── tables
    ├──        <-- place for user-defined modifications
    └── projects
        ├── CF
        ├── CMIP5
        ├── CORDEX
        └── NONE
```

Package Structure

```
└── tables
    ├── projects
    │   ├── CF
    │   │   ├── CF_check-list.conf
    │   │   ├── cf-standardized-region-names.txt
    │   │   └── cf-standard-name-table.xml
    │   ├── CMIP5
    │   │   └── ...
    │   ├── CORDEX
    │   │   ├── CORDEX_archive_design.csv
    │   │   ├── CORDEX_check-list.conf
    │   │   ├── CORDEX_GCMModelName.txt
    │   │   ├── CORDEX_qc.conf
    │   │   ├── CORDEX_RCMModelName.txt
    │   │   └── CORDEX_variables_requirement_table.csv
    │   │
```

Package Structure

```
└── tables
    ├── CORDEX_check-list.conf
    ├── CMIP5
    │   └── CMIP5_qc.conf
    ├── projects
    │   │   ├── CF
    │   │   │   ├── CF_check-list.conf
    │   │   │   ├── cf-standardized-region-names.txt
    │   │   │   └── cf-standard-name-table.xml
    │   │   ├── CMIP5
    │   │   │   └── CMIP5_qc.conf
    │   │   ├── CORDEX
    │   │   │   ├── CORDEX_archive_design.csv
    │   │   │   ├── CORDEX_check-list.conf
    │   │   │   ├── CORDEX_GCMModelName.txt
    │   │   │   ├── CORDEX_qc.conf
    │   │   │   ├── CORDEX_RCMModelName.txt
    │   │   │   ├── CORDEX_time_table.csv
    │   │   │   └── CORDEX_variables_requirement_table.csv
```

user-defined modifications

could be only lines Which have been modified.

Check-list

Check-list File

- A file for each project.
- Interface between the user and the QC program
- One entry for each check.
- A general descriptive text is provided; the real annotation is specific.
- A default setting is provided for each project.

Check-list

```
#  Syntax:  text & tag, level, [task], [variable], [constraint]
#
#  Brace grouping {}:
#  ...
#  Example: given: a,b=1{x=v{D(x),y,b=2}},{u,v},w
#           result: 'a,b=1,w', 'x=v,a,b=1,w', 'y,b=2,a,w', 'u,v,a,b=1,w,'

#   Key words: L1, L2, L3, D, EM, ST, PT, flag, var, V=value, R=record
#
#   Level:  L1 – L4
#           ...
#   Tag:      Has to match a flag for each check in the QC sources.
#   Task:     Email notification (EM), discard the check/test (D)
#   Variable: A list of comma-separated acronyms of variables;
#             directive is only applied to the variable(s).
#   Value:    Constraining value, e.g. {flag,D,V=0,var} discards a test
#             for variable var only if value=0
```

Check-list

Examples from `CORDEX_check-list.conf`:

```
Height requires units=m.
        & 55_1,L1

Near-surface height must be 0 - 10m.
        & 55_2,L1,{D,rlut,rsdt,rsut}

Suspecting a replicated record
        & R3200,L1{D, sund},{D,V=0,clivi,mrfso,prsn,sftgif}
```

CF Convention

CF Convention (1.4 – 1.6):

- QC-0.4: only a few checks within the QC,
  hope that modellers checked convention compliance.
- QC-0.5: complete CF check,
  with annotations in the QC style,
  with an adjustable CF check-list.

CF Convention

http://cfconventions.org/compliance-checker.html

Compliance Checker

This utility checks that a netCDF file which you supply complies with the CF comformance requirements and recommendations.

CF Compliance Checker (BADC)

CF Compliance Checker (READING)
http://puma.nerc.ac.uk/cgi-bin/cf-checker.pl

Example from the CF_Test Suite at DKRZ:

CF Convention

CF Compliance Checker (READING). Truncated output

```
File name:      cf_5.2b.nc   Output of CF-Checker follows...
CHECKING NetCDF FILE: /tmp/24612.nc

Checking variable: lat
INFO: attribute 'comment' is being used in a non-standard way
INFO: attribute 'coordinates' is being used in a non-standard way

Checking variable: rh
INFO (3.1): No units attribute set.  Please consider adding a units attribute for
completeness.
```

CF Compliance Check (DKRZ). Full output

```
path: QC-DKRZ/CF-TestSuite/Nc/Fail/chap5
file: cf_5.2b.nc:   FAIL

L1-CF_13b: auxiliary coordinate variable=lat should not have a coordinates
attribute.
```

DKRZ

Example from the CF_Test Suite at DKRZ:

CF Compliance Checker (READING). Truncated output.

```
File name:      cf_4.1b.nc      Output of CF-Checker follows...

ncopen: Can't open HDF5 attribute

COULD NOT OPEN FILE, PLEASE CHECK THAT NETCDF IS FORMATTED
CORRECTLY.
ERRORS detected: 1
```

CF Compliance Check (DKRZ). Full output

```
path: QC-DKRZ/CF-TestSuite/Nc/Fail/chap4
file: cf_4.1b.nc: FAIL

L1-CF_12a: all values of coordinate variable=time have to be set,
found _FillValue at index=3.
```

CFC-DKRZ

CF Compliance Checker (CFC) within the QC:

- The core CFC is a C++ object embedded in the QC.
- CFC gets the state of meta-data, which the QC relies on.
- Annotations raised by the CFC feed neatly into QC results.

Stand-alone CF Compliance Checker of the DKRZ :

- Part of the QC package.
- Installation: `QC-DKRZ/install CF`
- Execution: `QC-DKRZ/scripts/cf-checker [options]`

DKRZ

Efficiency:

- QC-0.4:

  Each checked file is opened twice.

  A new QC process is launched for each sub-temporal file.

  I/O of ASCII tables.

- QC-0.5:

  A new QC process for the suite of files for a given variable.

  GAIN: less back-ground processes.

  I/O of binary tables.

QC-DKRZ

Installation Documentation

```
https://redmine.dkrz.de/projects/cordex/wiki/DKRZ_QC_Tool
```

QC-0.4 (CORDEX)

```
svn co http://svn-mad.dkrz.de
      /svn/mad/Model/QualCheck/QC/branches/QC-0.4
```

QC-0.5-beta

```
git clone https://github.com/h-dh/QC-DKRZ
```