The CMIP5/AR5 Data Quality Control of Level 2

M. Stockhause, H.-D. Hollweg

18. Apr. 2011

1. Introduction

In the CMIP5 process quality assurance is performed to a larger extend than in former intercomparison projects. An overall block diagram of the CMIP5 data ingest and publication process in the ESG Federation (ESGF) together with tasks for data acceptance, documentation and quality control is provided in Figure 1 (Lautenschlager et al., 2010).

Within this document the functionality of the automated quality control checks of level 2 (QC L2) are described and its embedding into the overall data publication process. More information on quality control for CMIP5: http://cmip5qc.wdc-climate.de.

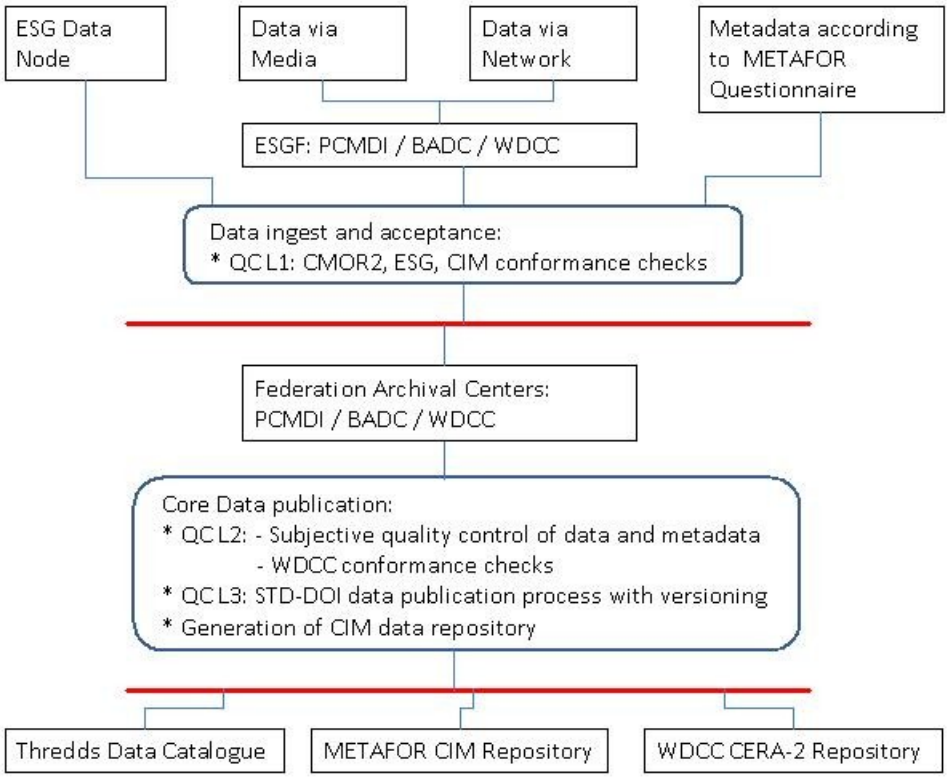

Figure 1: CMIP5/AR5 data ingest and publication process.

2. Quality Control Level 2

The CMIP5/AR5 quality control level 2 is performed for the core data after the initial checks for CMOR2 and ESG conformance (QC L1) are passed (http://cmip5qc.wdc-climate.de). It consists of automated checks by a tool and subjective metadata and data controls. These quality checks fulfill most of the testing properties for the DOI data publication review process.

The statistical checks of the QC Tool checks the following criteria:
a) File consistency
1) in the end files will have the right number of records. The number is given in the metadata.
2) strictly regular time steps (ESG checker allows for time gaps)

b) Metadata consistency (check of consistency between metadata of the standard_output table and the metadata of the file headers)

c) Physical properties of variables
3) minimum and maximum are checked against specified ranges (default for an invalid current exterm value of a global field:

mean:
mextr(t) = 1 / N * sum_{i=0}^{N=t / (delta t) - 1} extr_{i}
N:= index over all time steps 'delta t' up to the actual time t

Default (case for all variables in the standard_output table of K. Taylor):
mextr_{N-1} - extr_{N} > order_of_magnitude(mextr_{N-1} * 10^{5}

4) time series are calculated for:

- min
- max
- globally weighted mean (in case of no _FillValue)
- area weighted mean (in case of existing _FillValue; reasonable, e.g., for temperature of snow)
- standard deviation of the globally weighted mean.

If the statistical QC Tool fails to read and check the data, a record check is performed, which checks for every record individually, if the value is inside a certain plausible range. The ranges are defined in QCWrapper/table/range.txt.

The WDCC Conformance checks are completed by subjective quality controls of data and metadata. A logfile of the quality checks for level 2 is stored in the metadata repository for further use in QC L3 and in the data publication process.

2.1 Statistical QC Tool for L2 checks

Availability: http://svn-mad.zmaw.de/svn/mad/Model/QualCheck/QC/branches/QC-0.x

Software: C++, C and bash

Requirements: linux, C/C++ Compiler
netcdf-3.6.3 library
latest standard table in csv format based on
http://cmip-pcmdi.llnl.gov/cmip5/docs/standard_output.xls
(see Readme.txt in svn for detailed information)
Installation details in C1 Installation of QC Tool.

Functionality: QC tool is designed to run during the model simulation. It crawls through the all subdirectories under the specified data location for netCDF data and checks the new data. So, if new data is received and identical QC run will only check the additional data. The QC investigates the Atomic Datasets.

Input: Configure file

Start: qcManager -f <configure file>

Output: experiment log – list checked data files (chunks)
QC result netCDF file for every Atomic Dataset with global fields of
the monthly average, minimum, maximum, standard deviation
and
an error flag (see A. QC netCDF Result Example for details)
QC error or warning messages if occurred for Atomic Datasets
session log – program execution messages
version log – 'svn info' information file
atomic dataset logs – bash command echos
possibly protocols and an altered standard_table

2.2 Wrapper, QC Record Checks and Result DB

NEW: ***Production QCDB now in Oracle!***
Switch between postgres and Oracle by setting:
QCDB_TYPE="ORACLE"
QCDB_TYPE="POSTGRES"
default (at the moment postgres)

Availability: http://svn-mad.zmaw.de/svn/mad/Model/QualCheck/QCWrapper

Software: python

Requirements:
python >= 2.4 + modules sqlalchemy
ncdump (check of validity of QC netCDF results)
openssl (DB user password decryption)

for postgres: **send the IP of your machine** for access of the central QC DB
python module psycopg2

for oracle: oracle instant client packages "basic" and "sdk"
set softlinks for version independent libs, e.g.
libclntsh.so -> libclntsh.so.11.1
libocci.so -> libocci.so.11.1
set or add to environment variables:
ORACLE_HOME=<instant client directory>
LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$ORACLE_HOME"
PATH="$PATH:$ORACLE_HOME

for qcDbselect.py –outcera*:** xmgrace (netcdf-3.6.3***, ps, pdf support)
cdo (***netcdf-3.6.3*** support)

Installation details in C2 Installation of QC Wrapper, QC Record Checks.

2.2.1 QC Tool Wrapper – qcWrapper.py

Functionality: set-up QC result database QCDB: DB schema in Figure 2
run QC and fill QC results in QCDB together with log files and result files, and if any error and warning messages
assure unique result directory paths of QC
list errors of QC for an DRS experiment
runs only on latest published version directory

Input: enlarged QC tool configure file
Example configure file in D. QCDB Configure File.

Start:
QC Statistic: **qcWrapper.py --configure=<enlarged configure file>**
QC Record: **qcWrapper.py --configure=<enlarged configure file> --rcheck**

Output: log file as specified in configure file
results stored in QCDB

2.2.2 Check QC Results – qcDbselect.py

Functionality: check QC results for an experiment
check specific atomic dataset
optional list only atomic datasets with errors and warnings
optional download of QC netCDF file(s) for investigation
optional download of QC log files
optional plot data with xmgrace and export metadata

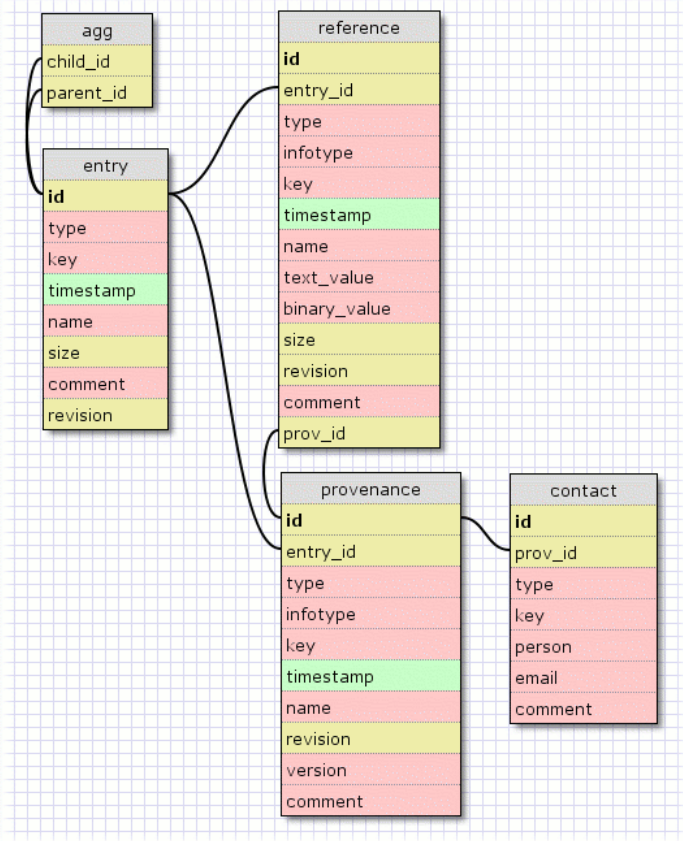

Figure 2: QC Result DB Schema.

Input: command line options
mandatory: --postgresdb
with values 'test1' for testing and 'qc1' for production

Start: **qcDbselect.py <options>**

a. Check experiment, list only errors and download QC log files:
qcDbselect.py --postgresdb=test1 --error --experiment=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45 –log=/tmp/mydowndir

b. Check atomic dataset result and download QC netCDF result:
qcDbselect.py --postgresdb=test1 --result=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45/mon/atmos/tas/r1 –-download=/tmp/mydowndir

c. Listing of
1. all experiments QC checked in the QCDB by:
qcDbselect.py --postgresdb=test1 –-list

2. history of a specific experiment:

qcDbselect.py --postgresdb=test1 –-list --experiment= CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45

```
# EXPERIMENT     QC_STATUS COMMENT       ERR_STATUS      QC_DATE         QC_CONDUCTOR
QC_TOOL_VERSION

cmip5/output/MPI-M/ECHAM6-MPIOM-TR/amip          QCL2_assigned QC Level2 assigned at Fri Jan  7 08:44:10
CET 2011      EXCEPTION   2011-01-03 07:57:52          Martina Stockhause:martina.stockhause@zmaw.de
QC Tool in svn revision: 2571 by author: heinz-dieter on date: 2010-11-25 19:42:43 +0100 (Thu, 25 Nov 2010)

cmip5/output/MPI-M/ECHAM6-MPIOM-TR/historical   ---  ---      EXCEPTION   2010-12-07 13:29:24     Martina
Stockhause:martina.stockhause@zmaw.de   QC Tool in svn revision: 2583 by author: heinz-dieter on date: 2010-12-06
18:25:26 +0100 (Mon, 06 Dec 2010)
```

d. Plotting of netCDF results with xmgrace using the given parameter

template qc_xmgrace.par in table subdirectory and export of metadata:

qcDbselect.py --postgresdb=test1 --experiment=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45 --outcera=/qcWrapper/table/qc_xmgrace.par

Note: At the moment, the metadata export is a preliminary mapping to CERA xml input files

e. Check for new published datasets after QC performance (date=date of QC performance):

qcDbselect.py --date=2010-08-31 --postgresdb=test1 – experiment=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45

Output: standard output and logfile: list of atomic datasets passed QC checks or error log
optional: save experiment log or result files in specified directories
optional: pdf plotfile and tar-ed qc-Results

2.2.3 Delete QC results from QCDB – qcDbdelete.py

Functionality: delete all QC result entries of an atomic dataset or an experiment including all atomic datasets belonging to it

Input: command line options

Start: **qcDbdelete.py <options>**

qcDbdelete.py --postgresdb=test1 --entry=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45 – log=myqclogfile --drsfile=drsnew.txt

or:

qcDbdelete.py --postgresdb=test1 --entry=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45/mon/atmos/tas/r1 --log=myqclogfile --drsfile=drsnew.txt

Be careful with the usage of sql wildcard '%'. A single '%' as entry deletes all QC results from the QCDB.

Output: information in log file added

2.2.4 Assign QC Level 2

Functionality: send CIM quality report and experiment log (extracted out of QCDB) and given quality result plot to to CIM repository

update QCDB with QC flag and add quality result plot

Input: command line options
CIM quality report XML (from qcDbselect --outcera)
optional: quality result plot (from qcDbselect --outcera)

Start: **qcAssignL2.py <options>**

qcAssignL2.py –postgresdb=<qc database> --experiment=<DRS name of experiment> --cimxml=<CIM quality report> --plot=<quality result plot>

Output: -

2.3 Quick Start

1. Install QC tool:
 install netcdf library netcdf-3.6.3 with configuration
 export CC="/usr/bin/gcc"
 export CFLAGS="-static -O2"
 export CPPFLAGS="-DNDEBUG"
 export CXX="/usr/bin/g++"
 export CXXFLAGS="-static -O2"
 export FC=""
 export F90=""
 ./configure --prefix=path/netcdf-3.6.3 --disable-examples --disable-docs-install --disable-fortran-compiler-check \
 cd QC/src
 edit compiler instruction file zg++
 run ./zg++

2. Ensure QC tool is installed an running by:
 qcManager -E_CHECK_TOOLS

3. Run test for QC tool:
 cd QC/example
 edit qc_test.conf
 run ../QC/scripts/qcManager -f qc_test.conf
3a. postgres: Check accessibility of qc.dkrz.de on port 5432 through your firewall by:
telnet qc.dkrz.de 5432
3b. oracle: Check accessibility of oracle db by
src/access_oracle.py

4. Ensure python modules sqlalchemy, psycopg2 and openssl are installed and running and you sent the IP of your machine to martina.stockhause@zmaw.de.
Check QC DB access by:
qcDbselect.py --postgresdb=test1 –list
5. Try an QC DB ingest by editing qc.conf example and running:
qcWrapper.py –conf=qc.conf
6. Ensure for plotting (qcDbselect --outcera) that ncdump, xmgrace,psmerge and cdo are installed and working

2.4 Run QC Tool

1. Edit the sample configure file given in QCWrapper main directory. Use db 'test1' for testing and 'qc1' for production runs.

2. Start statistical QC Tool by: **qcWrapper.py --conf=<configure file>**
   Start record checks if not all data was checked by QC tool or in case of severe QC tool problems by: **qcWrapper.py --conf=<configure file> --rcheck**

2.5 Subjective data and metadata controls

1. Check QC results for errors and warnings, most warnings indicate errors or inconsistencies in the data

   a. Check whole experiment by:

   qcDbselect.py --postgresdb=test1 --experiment=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45 --log=/tmp/mydowndir --drsfile=drsnew.txt

   Examine the logfiles in the specified download directory for errors.
   Flags: 'ERROR': netcdf file not readable, 'WARNING': qc ok, QC Tool reported errors or warnings during check, 'UNCHECKED': no qc result written
   'INFO': nothing reported

   b. If any errors or warnings occurred (stdout list of a.), check the QC result for these specific atomic datasets by:
   qcDbselect.py --postgresdb=test1 –-result=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45/mon/atmos/tas/r1 --download=/tmp/mydowndir

c. Check the global monthly mean statistical time series of netcdf results ('ave':average,'min':minimum,'max':maximum) and if the error flag ('flag') is <>0 at a time step (error flag definition in B1. Error flag value in netCDF result file). Cross-check on data. You can use the xmgrace routine implemented:
qcDbselect.py --postgresdb=test1 --experiment=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45 --outcera=qcWrapper/table/qc_xmgrace.par

2. Analyse and check results – to be defined
3. Assign QC level 2:

   qcAssignL2.py --postgresdb=test1 --experiment=CMIP5/output/MPI-M/ECHAM6-MPIOM/rcp45 --cimxml='cimqr_CMIP5_output_MPI-M_ECHAM6-MPIOM_rcp45.xml' --plot='QCL2_images_CMIP5_output_MPI-M_ECHAM6-MPIOM_rcp45.pdf'

3. Implementation of QC L2 into the CMIP5 data publication process

1. Run QC at ESGF partner on agreed subset of CMIP5 data
2. Error case:
   a. Inform modeling center about the error(s) and receive new data
   b. replicate new data among ESGF partners (if old data was replicated)
   c. delete old data
   d. Rerun QC on altered data:
   - new version assigned: rerun wrapper
   - same version assigned: Activate option REDO=t1/t2 in configure file in Wrapper rerun, e.g. REDO=20100801/20101015 to replace QC results between 01.08.2010 and 15.10.2010.
3. Assign QC L2 and exchange QC results after passed QC (CIM example suggested to metafor group in E. CIM Quality Document)

3.1 Open Issues

How to ingest quality result in CIM repository?

How to receive an email address for the modelling center contact? CIM simulationRun objects does not assure that.

How to set quality flag to “subjective quality control passed”? Suggestion via CIM, so that experiment log and quality flag are sent to the same location (implicit set for QC L2 assignment). Alternative: ESG publish only of metadata.

Do we set the quality flag for the whole experiment or for a model_realm?

In case of partly passing QC L2 checks: Generally, we would have to wait until all data (belonging to a certain model_realm or experiment) has passed the QC L2 checks, before doing so. Do we need to give all WGs access to the part of data passing the QC checks? If so, when do we set the quality flag to passed for the whole model_realm?

In case of errors: Is the corrected data of the modelling center to be sent directly to the ESGF partner responsible for the QC and ESG published there? Or via PCMDI? It might be that during the contact with the modelling center a new single file might be received by email. How to replicate it?

In case the modelling center withdraws data already replicated. How do the responsible ESGF partner gets to know that?

Specifications of QC L2 assignment criteria...

REFERENCES

Lautenschager, M., M. Stockhause, F. Toussaint, B. Lawrence, S. Kindermann (2010): The CMIP5/AR5 Model Data Quality Control – Discussion Paper, Draft May 15th, 2010.

CMIP5 QC, http://purl.org/org/cmip5/qc

A. QC netCDF Result Example

```
netcdf qc_ta_Amon_ECHAM6-MPIOM-LR_rcp45_r1 {
dimensions:
        time = UNLIMITED ; // (480 currently)
        cycle = 1 ;
        size = 4 ;
variables:
        double time(time) ;
                time:bounds = "time_bnds" ;
                time:units = "days since 1955-1-1" ;
                time:calendar = "gregorian" ;
                time:axis = "T" ;
                time:long_name = "time" ;
                time:standard_name = "time" ;
        float ta ;
                ta:about = "original attributes of the checked variable" ;
                ta:original_dimensions = "(time, plev, lat, lon)" ;
                ta:standard_name = "air_temperature" ;
                ta:long_name = "Air Temperature" ;
                ta:units = "K" ;
                ta:cell_methods = "time: mean" ;
                ta:history = "2010-03-09T07:40:44Z altered by CMOR: replaced missing value flag
(-9e+33) with standard missing value (1e+20)." ;
                ta:missing_value = 1.e+20f ;
                ta:_FillValue = 1.e+20f ;
                ta:associated_files = "baseURL:http://cmip-pcmdi.llnl.gov/CMIP5/dataLocation
gridspecFile:gridspec_Amon_ECHAM6-MPIOM-LR_rcp45_r0_fx.nc cellAreaFile:cellArea_Amon_ECHAM6-
MPIOM-LR_rcp45_r0_fx.nc cellVolumeFile:cellVolume_Amon_ECHAM6-MPIOM-LR_rcp45_r0_fx.nc" ;
        double time_step(time) ;
                time_step:long_name = "time_step" ;
                time_step:units = "days" ;
        double min(time) ;
                min:long_name = "global_minimum" ;
                min:units = "K" ;
                min:_FillValue = 1.e+20 ;
                min:valid_range = 186.315444946289, 186.872787475586 ;
        double max(time) ;
                max:long_name = "global_maximum" ;
                max:units = "K" ;
                max:_FillValue = 1.e+20 ;
                max:valid_range = 305.719482421875, 313.203796386719 ;
        double ave(time) ;
                ave:long_name = "global_average" ;
                ave:units = "K" ;
                ave:_FillValue = 1.e+20 ;
                ave:valid_range = 239.587732429839, 242.371260493529 ;
        double std_dev(time) ;
                std_dev:long_name = "standard_deviation" ;
                std_dev:units = "K" ;
                std_dev:_FillValue = 1.e+20 ;
                std_dev:valid_range = 28.4119355141472, 30.2377536487199 ;
        int fill_count(time) ;
                fill_count:long_name = "number of cells with _FillValue" ;
                fill_count:units = "1" ;
                fill_count:_FillValue = -1 ;
        int flag(time) ;
                flag:long_name = "error_flag" ;
                flag:units = "1" ;
                flag:_FillValue = -1 ;
                flag:coding = "Note: code numbers accumulate" ;
                flag:code_-1 = "not checked" ;
                flag:code_0 = "ok" ;
                flag:code_1 = "error: negative time step" ;
                flag:code_2 = "error: missing time step" ;
                flag:code_4 = "error: identical time step" ;
                flag:code_8 = "error: negative time-bounds range" ;
```

```
                flag:code_16 = "error: overlapping time-bounds ranges" ;
                flag:code_32 = "error: gap between adjacent time-bounds ranges" ;
                flag:code_100 = "warning: record with entire field of filling value" ;
                flag:code_200 = "warning: record with entire field of a constant value" ;
                flag:code_400 = "warning: equivocal minimum" ;
                flag:code_800 = "warning: equivocal maximum" ;
                flag:code_1600 = "warning: undefined standard deviation" ;
        double internal(cycle, size) ;
                internal:long_name = "resuming_session_props" ;
                internal:units = "1" ;
                internal:_FillValue = 1.e+20 ;
                internal:record_components(rc) = "values for each cycle step:" ;
                internal:rc_0 = "time" ;
                internal:rc_1 = "time bounds (begin)" ;
                internal:rc_2 = "time bounds (end)" ;
                internal:rc_3 = "time step (t_n - t_n-1)" ;
                internal:first_record_date = "1955-01-16T12:00:00" ;
                internal:last_record_date = "1994-12-16T12:00:00" ;
                internal:cycle_steps = 1. ;
                internal:isTimeBoundsTest = 1. ;
                internal:sumOfMaxCounter = 480. ;
                internal:sumOfMax = 147840.211181641 ;
                internal:sumOfMinCounter = 480. ;
                internal:sumOfMin = 89545.6695556641 ;
                internal:tBoundsRangeScale = 1. ;

// global attributes:
                :project = "CMIP5" ;
                :product = "quality check of CMIP5 data set" ;
                :version = "QC package version in /scratch/local2/QC_results2/CMIP5/output/MPI-
M/ECHAM6-MPIOM-LR/rcp45/version/version_2010-05-25_14:20:11.txt" ;
                :contact = "heinz-dieter.hollweg@zmaw.de" ;
                :standard_table = "modified csv version of standard_output_3May2010.xlsx" ;
                :creation_date = "2010-05-25T14:21:15CEST" ;
                :institution = "MPI (MPIfM,Hamburg, Germany)" ;
                :institute_id = "MPI-M" ;
                :experiment_id = "rcp45" ;
                :source = "ECHAM5 T31 Millenium run19" ;
                :model_id = "ECHAM6-MPIOM-LR" ;
                :references = "ECHAM6" ;
                :experiment = "RCP4.5" ;
                :frequency = "mon" ;
                :table_id = "Table Amon (19 February 2010)" ;
                :title = "ECHAM6-MPIOM-LR model output prepared for CMIP5 RCP4.5" ;
                :modeling_realm = "atmos" ;
                :realization = 1 ;
                :cmor_version = "2.0.0" ;
                :variable_attributes_check = "conforming to table; consistent throughout the
project" ;
                :dimension_attributes_check = "conforming to table; consistent throughout the
project" ;
                :dimension_layout_check = "conforming to table; consistent throughout the
project" ;
                :record_check = "no findings" ;
                :history = "2010-05-25T14:21:37CEST data set:
/scratch/local2/testdata/CMIP5/output/MPI-M/ECHAM6-MPIOM-
LR/rcp45/mon/atmos/ta/r1/v1/ta_Amon_ECHAM6-MPIOM-LR_rcp45_r1_195501-199412.nc" ;
}
```

B. Error Codes of QC Tool

B1. Error flag value in netCDF result file

```
List of Checks conducted in qC.x
================================

Program-unit
Exit-code
      Member()
           Description

Note: ^ indicates footnote(s)
_____________________________________________________
Error Flags thrown for affected records and
indicated in file qc_'file-name'.nc by variable 'flag'
_____________________________________________________

  -1  --
          Not checked

   0  --
          No error found

   1  testTimeStep() ^0, ^5
          Error: negative time step

   2  testTimeStep() ^0, ^5
          Error: missing time step

   4  testTimeStep() ^0, ^5
          Error: identical time step

   8  testCalendarTimeBounds()
          Error: negative/zero time bounds range

  16  testCalendarTimeBounds() ^0
          Error: overlapping time bounds ranges

  32  testCalendarTimeBounds() ^0
          Warning: gap between time bounds ranges

 100  testData()
          Warning: found a record entirely with filling value

 200  testData()
          Warning: found a record entirely with constant value

 400  testData()
          Warning: suspect minimum
          Note: no table comparison; inferred from the data

 800  testData()
          Warning: suspect maximum
          Note: no table comparison; inferred from the data

1600  testData()
          Warning: undefined standard deviation

3200  testData()
          Warning: suspecting a replicated record
          Note: a record of min., max., ave., and std. dev.
                is identical to a previous one.
          Note: fields of constant or filling value excluded.
```

B2. Error Messages

```
_____________________________________________________
Checks between netCDF file and standard/project table;
any failure will issue a message.
_____________________________________________________

Dimensions: output dimension name
            checksum *) (Fletcher) of dimension-variable values
            standard name
            long name
            units **)
            bnds_name

            *) not for time
           **) time: reference date in the attributes may be
                     different, only string 'days since' is checked

Variable:   output variable name
            standard name
            long name
            units
            cell-method
            type (i.e. type must be of non-integer format)
```

B3. Program Errors and Warnings (partly with program abort)

```
_____________________________________________________
Error/Warning Codes (partly interrupting a process)
Note: For codes with a '_', only the figure in front
      of it is returned as exit code.
_____________________________________________________

qc_main.cpp
   1      --
          Internal errors indicating flaws in
          the scripts or qc_main.cpp.

   1_1    getFilename() ^1
            Invalid name.

   1_2    insertPointedObj() ^1
            String parsing: definition of object ... must not have references.

   1_3    getopt() ^1
            Undefined option ...

   1_4    setObjLinks() ^1
            ... must have a single reference.

   2      entry() ^1
            No QC object linked to InFile object.

Base.cpp
   5   setFillingValue()
          Undefined explicitly provided variable name.
          Note: not related to CMIP5.

   6   setVarPropsNoOperation()
          No rules to link alias to any variable in operation.
          Note: not related to CMIP5.

   7   setVarPropsNoOperation()
          No rules for finding a variable name.
          Note: not related to CMIP5.

   8   setVarPropsNoOperation()
```

```
          No rules for finding a variable name.
          Note: not related to CMIP5.

   9   getVarname(std::string &s, std::vector<std::string> &) ^1
          Invalid assignment statement of variable names.
          Note: not related to CMIP5.

InFile.cpp
  10  init() ^1
         Could not open netCDF file.
         Note: non-operational processing only.

  12  checkVarType()
         The type of the variable is not float.
         Note: CMIP5 requires always float.

  14  setGeoParam()
         Dimension rank of the field is higher than 3D.
         Note: the dimension of time is excluded.

  15  setGeoParam()
         Vertices or bounds of unknown format.

  16  scanNcVars() ^1
         Explicitly provided variable name not found in the netCDF file.
         Note: non-operational processing only.

  18  xtractNum()
         Dimension rank of the field is higher than 3D.
         Note: the dimension of time is excluded.

  19  init() ^1
         No QC object linked to the InFile object.
         Note: non-operational processing only.

  21  scanNcVars()
         Neither explicit variable name nor CMOR encoded filename.
         Note: search priority: clear excluded variables, time
         dependence, multi-dimensional variables, independent
         variables matched with filename.

  22  scanNcVars()
         Fixed field variable name from filename not found in the netCDF file.

Parse.cpp
  28  convertEmbeddedObj()
         Undefined specification in the command-line arguments.
         Note: non-operational processing only.

  29  convertEmbeddedObj()
         Syntax error in the command-line arguments.
         Note: non-operational processing only.

qcExecutor_FS (script)
  30_1  checkParent()
         Invalid path to parent: <path>
         Note: issues a warning.

  30_2  checkParent()
         No parent file found in path: <file>
         Note: issues a warning.

QC.cpp
  30_3  checkCMIP5_Filename()
         Missing netCDF attribute: parent_experiment_id.

  30_4  checkCMIP5_Filename()
         Missing netCDF attribute: parent_experiment_rip.

testParentChild.cpp
  30_5  main()
```

```
        Missing command-line parameter for the child experiment: -c file.

  30_6  main()
        Missing command-line parameter for the parent experiment: -p file.

  31_1  main()
        Could not open the netCDF file of the child experiment

  31_2  main()
        Could not open the netCDF file of the parent experiment

  32    getTimeProperties()
        Different calendar types in parent and child.
        Note: This could work. Thus, only a warning.

  33_1  getTimeProperties()
        No time units attribute in the child file.

  33_2  getTimeProperties()
        No time units attribute in the parent file.

  34_1  getTimeProperties()
        Time units attribute: different measuring in child and parent.

  34_2  getTimeProperties()
        Time units attribute: different reference dates in child and parent.

  35    sync()
        Child begins earlier than the parent
        Note: potentially different time units are taken into account. The last lag of the
parent is the one closest before the first one of the child.

  36    checkTime()
        The lag across files differs from the last lag in the parent.
        Note: potentially different time units are taken into account. The last lag of the
parent is the one closest before the first one of the child.

  37    checkTime()
        The last lag of the parent differs from the first lag of the child.
        Note: potentially different time units are taken into account. The last lag of the
parent is the one closest before the first one of the child.

  38_1 checkTime()
        No bounds attribute of time of the parent.

  38_2 checkTime()
        No bounds attribute of time of the child.

  38_3 checkTime()
        Time bound lags of the last parent lag and the first lag of the child overlap.

QC.cpp
  41 checkDimStandardTable() ^2
        Standard table not found.

  42   checkStandardTable() ^2
        MIP table name not found in the standard table.

  43   checkStandardTable() ^2
        Missing column(s) in the standard table.

  44   checkStandardTable() ^2
        Variable not found in the standard table.

  45_1 getDimMetaData()
        Variable time has no unit attribute.

  45_2 getDimMetaData()
        Time attribute declares time_bounds, but there is no such variable.

  45_3 getDimMetaData()
```

```
     Variable name in filename does not match any variable in the file.

46   checkCMIP5_Filename()
       Checks filename syntax and netCDF attributes.
       Note: Only CMIP5

46_1 checkCMIP5_Filename()
       Missing netCDF attribute: project_id.

46_2 checkCMIP5_Filename()
       Missing netCDF attribute: physics_version.

46_3 checkCMIP5_Filename()
       Missing netCDF attribute: initialization_method.

46_4 checkCMIP5_Filename()
       Missing netCDF attribute: realization.

46_5 checkCMIP5_Filename()
       Missing netCDF attribute: experiment_id.

46_6 checkCMIP5_Filename()
       Filename is inconsistent with CMOR encoding.

46_7 checkCMIP5_Filename()
       Filename does not match CMIP5 attributes.

46_8 getMIP_table()
       Invalid MIP table name in CMIP5 attributes.

47   checkDimTableEntry() ^4
       Table error for dimension(s) of a variable.

47_1 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: output name.

47_2 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: standard name.

47_3 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: long name.

47_4 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: axis.

47_5 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: checksum.
       Note: Indicates a change in the layering or model grid.

47_6 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: units.

47_7 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: bounds.
       a) Conflict in request for bounds.
       b) Bounds not available in the file.

47_8 checkDimTableEntry() ^4, ^6
       Standard/Project table: <variable>, <dimension>: size.

47_9 checkDimTableEntry() ^4, ^6
       Standard/Project table: time, units: missing key string
       in the units attribute 'PERIOD since'.
       Note: PERIOD indicates key: minutes, hours, days, etc.
       Note: Throws always an error flag, because the date at each
             time step cannot be determined later.

47_10 checkDimTableEntry() ^4, ^6
       Standard/Project table: time, units: different reference dates.
       Note: Throws always a warning flag, because the date at each
             time step might be found correct later.
```

```
47_11 checkStandardTableDimBounds() ^4, ^6
       Number of dim_bounds do not match those of the standard table.

47_12 checkStandardTableDimBounds() ^4, ^6
       Values of dim_bounds do not match those of the standard table.

48    checkDimStandardTable() ^2, ^3
       Dimension not found in the standard table.

49    checkDimStandardTable() ^2
       MIP table for dimensions not found in the standard table.

50    checkDimStandardTable() ^2
       Corrupt standard sub-table for dimensions: wrong number of columns.

51    checkDimStandardTable() ^2
       Missing value in MIP table 'dims' in column 'bounds'?.
       Note: Required is 'yes' or 'no'.

52    checkDimStandardTable() ^2, ^3
       Dimension from the table not found in the file.

54    checkProjectTable()
       Could not create project table.
       Note: Due to a faulty configguration file entry.
       Note: Signal to qcManager to shutdown.

55    checkProjectTable() ^1
       Corrupt project table: variable | dimension | auxiliary.

56    checkProjectTableAuxiliary()
       Project-table errors for auxiliaries.

56_1  checkProjectTableAuxiliary()
       Project-table: time independence check failed.

56_2 checkProjectTableAuxiliary()
       Project-table: number of dimensions of auxiliary changed.

56_3 checkProjectTableAuxiliary()
       Project-table: checksum of levels or grid values changed.

56_4 checkProjectTableAuxiliary()
       Project-table: missing auxiliary in the file.

56_5 checkProjectTableAuxiliary()
       Project-table: missing auxiliary in the table.

57 checkTables()
       No path to table; faulty config-file entry.
       Note: Signal to qcManager to shutdown.

58 checkVarTableEntry() ^4
       Table error for a variable.

58_1 checkVarTableEntry() ^4
       Standard/Project table: variable: standard name conflict.

58_2 checkVarTableEntry() ^4
       Standard/Project table: variable: long name conflict.

58_3 checkVarTableEntry() ^4
       Standard/Project table: variable: units conflict.
       a) variable has no units attribute, table requires units= or units=1.
       b) variable has no units attribute.
       c) variable has <v_units>, table requires <t_units>.

58_4 checkVarTableEntry() ^4
       Standard/Project table: variable: cell-method conflict.
```

```
  58_5 checkVarTableEntry() ^4
         Standard/Project table: variable: type conflict.

  56_6 checkSigmaPressureCoordinates()
         No variables a(k) or b(k) found for the hybrid sigma pressure coordinates.

  60 initResumeSession()
         Name of a variable has changed
         Note: Compared to previous chunks.

  63 sync()
         All records have previously been processed.
         Note: This is not an error: idle

  64 syncRedo()
         Indication of a renewal of the input file.

  65 syncRedo()
         Faulty argument in config-file for redo-option.

  66 syncRedo()
         REDO is selected, but the qc-file is empty.

  67 syncRedo()
         REDO requires at least one record in qc_<filename>.nc

  68 testTimeStep()
         Waiting for a closing temporal gap.
         Note: This is not an error.

  69 initResumeSession() ^1
         Sub-cycle of time step has changed.
         Note: For any kind of cycles within a time step (e.g. diurnal cycle).
         Note: Not for CMIP5

  70 (different methods)
         Any error in any record.

  71 testTimeStep()
         Error in a time record (Error Flags: 1, 2, 4) stops and blocks
         further attempts, if option STOP_AT_TIME_ERROR is enabled.

  95 testTimeStamp()
         Ambiguous or faulty first time-stamp in the filename.
         Note: Faulty, if the stamp is younger than 1st time-record.
         Ambig., if the stamp is too coarse to resolve time-records.

  96 testTimeStamp()
         Filename time-stamp error (2nd date).
         Note: Time record exceeds time-stamp of the filename.

  97 testTimeStamp()
         Invalid time-range in filename.

  99 finally()
         Status: apparently in progress.
         Note: This indicates that no error occurred. But, the end time
         of the input filename and the last time value in the data
         did not match (with the uncertainty of 0.75 of a time step).
         So, it is likely that the file is still in a stage of processing.

 111 qc-main
         The program exited without any exit code above and did not
         produce a single file. This is an error.

footnotes:
^0  also across from previous file
^1  Non-operative mode.
^2  for option TABLE_STANDARD=... set in the configuration
^3  for option TABLE_RELAXED_DIM_CONSTRAINT disabled
^4  a series of tests where a failure of one of a few of them aborts
```

```
    the program and leads to a blocking of further attempts.
^5  writes messages to a file qc_error_<variable...>.nc and blocks
    further attempts
^6  conflict is reported, if the attribute of the dimension is available
    in both table and file meta data. If one is missing, this is reported.
```

C Installation Instructions

The order of installation is first QC Tool and second QC Wrapper.

C1 Installation of QC Tool

Quality Control of CMIP5 netCDF files
==========

Please take note of the "Disclaimer of Warranty" in file DoW.txt!

A brief overview
==========

The CMIPT5 quality control (QC) execution flow is basically
controlled by the scripts 'qcManager' and 'qcExecutor'.
The virtual QC of the CMIP5 netCDF files is conducted by a
C++ program embedded in the qcExecutor_FS bash script.
Additionally, small C++ and plain C utilities are used in the scripts. This text explains the steps to conduct the QC.
There is an example and a test of the installation.

Requirements
==========

A) a few basic programs in the environmental PATH.
These are checked by executing

your-path/QC/scripts/qcManager -E_CHECK_TOOLS

This will display the result of the check.
(You should do this after C), or ensure that
eventually ncdump will be in PATH.)

B) a c/c++ compiler

C) a netCDF installation
Successfully tested on 32/64 bit machines:
1) Linux 2.6.18-6-686 (Debian 2.6.18.dfsg.1-26etch2)
gcc version 4.1.2 (pre-release)
Debian 4.1.1-21)) #1 SMP
2) Linux 2.6.18-164.15.1.el5
gcc version 4.1.2
Red Hat 4.1.2-46 #1 SMP
3) powerpc-ibm-aix5.3.0.0

i) netcdf-3.6.3 with configuration

```
export CC="/usr/bin/gcc"
export CFLAGS="-static -O2"
export CPPFLAGS="-DNDEBUG"
export CXX="/usr/bin/g++"
export CXXFLAGS="-static -O2"
export FC=""
export F90=""

./configure --prefix=path/netcdf-3.6.3  \
  --disable-examples --disable-docs-install --disable-fortran-compiler-check \
```

ii) netcdf-4.1.2-beta1 with configuration:
(requires: hdf5-1.8.5 and zlib)

```
export CC="/usr/bin/gcc"
export CFLAGS="-static -O2"
export CPPFLAGS="-DNDEBUG"
export CXX=""
export CXXFLAGS=""
export FC=""
export F90=""
export HDF5DIR="path/hdf5-1.8.4-patch1/hdf5"

./configure --prefix=path/netcdf-4.1.1 \
  --enable-netcdf-4  --enable-static \
  --disable-shared --disable-fortran-compiler-check --disable-f77 \
  --with-hdf5="path/hdf5-1.8.4-patch1/hdf5"  \
  --with-zlib="path/zlib-1.2.3/lib"
```

D) The standard table is stored as comma-separated-value file (csv).
Without a reference table, the QC will produce a table
from the netCDF files. In the directory 'QC/Project_tables'
the current table standard_output_....csv was derived
from the CMIP5 corresponding xlsx table with
the open-office tool by:
1) mark all sheets (shift-ctrl pageDown),
but 'general','CFMIP output', and 'other output'
2) replace all ',' characters by ';'
3) save all sheets as csv files (unfortunately this has to
be done separately for each sheet)
4) concatenate the dim-sheet as the first and the other
sub-tables. This can be done with the script qqqCSV
in path QC/bin. Note: the dim-sheet may not be the first
but, it will be read for each variable.

Components
==========

C++ program (in directory QC/src):
getNC_att.cpp
qc_main.cpp
syncFiles.cpp
testParentChild.cpp
testValidNC.cpp
Note: sources of Classes are included automatically at compile time

Bash scripts (in QC/scripts):
cpMod
qcConfigurator
qcExecutor_FS
qcManager
qc.cron (might be used as cron-job script to restart after a shutdown)
zg++

C programs (in QC/src):
fModTime.c
unixTime.c

Auxiliary (in QC/src):
dist.cpp
(for convenience, this auxiliary C++ program is supplied
to finalise frequency distributions)

Configuration file (in QC/scripts/Conf):
qc_CMIP5.conf

All scripts and executables must in deed be executable!
Starting the script 'qcManager' would try to compile programs, if no
executables are detected (except qC_main.cpp, respectively qC.x).
But, it is recommended to do this manually beforehand.

Configuration
=============
a) Compiler

Name convention for executables: replace extensions '.c' and
'.cpp', respectively, by '.x'. Store the binaries in
directory QC/bin .

Compilation of plain C programs does not need a special setting.

Compilation of netCDF dependent cpp programs:
(qc_main.cpp --> qC.x, syncFiles.cpp --> syncFiles.x)

To be independent on any environment, a script for compilation
(zg++) is provided. However, using this script is not required.
If it is used, then edit the script appropriately:
1) set path and name of your compiler
2) set path to the include directory of this package: path/QC/include
3) set path to your netCDF include
4) set path to your netCDF lib
The script must be executed in directory 'src'.
The script relays any compile-directives, e.g. -g,
but required is: -o ../bin/exec-name.x name.cpp

If netCDF-4.1.1 is used, change macro-option in zg++ to -DNC4
(or if the zg++ is not used, set the macro-option for your compilation).
(Note: the default -DNC3 is used for CMIP5)

b) Quality control configuration:
Edit the QC/scripts/Conf/qc_CMIP5.conf
Each entry of the configuration is sufficiently described, hopefully.

Command-line options (for debugging purpose) are displayed by:
path/QC-scripts/qcManager --help

Installation
==========

Put all executables into directory 'QC/bin.
If scripts and executables are accessible, you are done.

Help
====
1) about the scripts:
a) the configuration file in dir QC/scripts/Conf
b) path/QC/scripts/qcManager --help (on the command-line)
c) path/QC/scripts/qcManager -E_CHECK_TOOLS
d) path/QC/scripts/qcManager -f conf-file -E_SHOW_CONF
(display the selected configuration)
e) path/QC/scripts/qcManager -f conf-file -E_SHOW_EXP
(display the selected experiment name and
the selected sub-paths to the DRS directory tree)

c-e) recommended before starting a QC run.)

2) about qC_main.cpp

The documentation tool used is doxygen.
The documentation is found in directory QC/doc/qC.x_html, please browse there for 'index.html'.

Debugging the main program gives the most detailed
understanding of the program flow. This leads also
directly to the embedded comments.
You may inquire the command-line arguments for a
call of the cpp-program by invoking something similar to this:

cd path/QC/example
../scripts/qcManager -f qc_test.conf -E_SHOW_CALL -E_NEXT

(Note: the displayed call is a single line, although multiple lines are
feasible, see description of class Parse.cpp and the example QC/scripts/z.txt)

Start

=====

In a unix console, type:

path/QC/scripts/qcManager -f another-path/qc_CMIP5.conf

This would start the QC process based on the configuration in another-path/qc_CMIP5.conf.
Then, the script reports the name of the currently analysed file
in the console. The names are replaces continuously on the same line.
Because the quality control is expected to be running a rather long time,
it is recommended to send the process into the background.
The progress monitoring may be suppressed by one of the following:
  1) enable key-word QUIET in the configuration file.
  2) start the qcManager with option -q
  3) re-direct output to /dev/null

If processing is subject to any queueing procedure,
i.e. the entire process shall have a limited time to live,
then try the command-line option '-E_NEXT[=int]'.
But, the difficulty would be that a queueing-scheduler presumably inhibits background processing. In this case,
please get in touch
with the EMail addressee at the end of this text.

Example and Test
==========

This little test simulates two cases:
1) the CMIP5 file of a variable (here 'tas') is split
   into chunks with the time range in the filename
   according to the paper 'CMIP5 and AR5 Data Reference Syntax (DRS)',
   but, with some freedom left in defining the time range.
   Additionally, there is a file without range indicating the
   netCDF file currently built by the CMOR program.
   In fact, this simulates the multi-session features of the QC.
   You might want to conduct the script/program run one by one. Then,
   execute repeatedly on the command-line from directory QC/example:
      ../scripts/qcManager -f qc_test.conf -E_NEXT

   Each execution stipulates a session.

2) an atomic data set with a missing time step (variable 'pr').

Note: sub-paths are according to the DRS Directory Layout.

What to do?

cd into directory 'example'.

(Short-cut:
start the script: start_test.bash
This will insert your current path into the qc_test.conf
file and start the initial script call.
../scripts/qcManager -f qc_test.conf [args])

Or manually within a console window, cd to directory 'path/QC/examples',
edit file 'qc_test.conf', and replace
'your-path' by the real path where the QC package is installed
in the following lines:

  QC_DATA_ROOT=your-path/QC/example/test
  DATA_ROOT_FS=your-path/QC/example/data/CMIP5

Start the scripts by typing:

  qcManager -f qc_test.conf [args]

The results will be written to directory 'test'.
This may be compared to the results in directory 'sample'.

For debugging or just learning, you would like to use:

```
# Save execution command of qcManager
qcManager -f qc_test.conf -E_DEBUG_M &> any-temp-filename

# Save execution command of qcConfigurator
#(Note: combination with -E_DEBUG_M saves both to the same file.)
qcManager -f qc_test.conf -E_DEBUG_C &> any-temp-filename

# Executes the next file (-E_NEXT[=int] the next [int] files)
qcManager -f qc_test.conf -E_NEXT

# Shows the next qC.x call, i.e. for the executable of the C++ program
qcManager -f qc_test.conf -E_NEXT -E_SHOW_CALL

# shows all qC.x calls qcManager would stipulate
qcManager -f qc_test.conf -E_SHOW_CALL
```

Communication
==========

Questions, recommendations, or just comments please direct to

heinz-dieter.hollweg@zmaw.de

=========
Good Luck
=========

C2 Installation of QC Wrapper, QC Record Checks

QC Tool Wrapper

Introduction

The Wrapper supports a CMIP5 QC tool run on the DRS experiment level as part of the CMIP5 QC Level 2 and stores the QC results and logfiles in a db (QCDB). The production QCDB is Oracle, test DBs are available on postgres. Switch between the DBs by setting:

```
QCDB_TYPE="ORACLE"
QCDB_TYPE="POSTGRES"
```
(No environment variable QCDB_TYPE set is equivalent to QCDB_TYPE="POSTGRES")

The wrapper consists of four independent scripts:

qcWrapper - check and provide input for QC tool, and store or update results inthe QCDB.

qcDbselect - check QC results in QCDB for errors and warnings

qcDbdelete - delete QC results from QCDB and file system

qcAssignL2 - update QCDB with QC result plots and QC flag (sending CIM quality document to CIM repository will be added as soon as the interface to CIM is defined and the CIM qctool is available.)

For help on these scipts use the --help option. They are located in the src directory.

It is checked against QC Tool in svn revision: 2911 by author: heinz-dieter on date: 2011-03-14 18:48:07 +0100 (Mon, 14 Mar 2011)

Further information on the QC Wrapper package is given in the document (docu/CMIP5-AR5-QC-L2.odt) and at http://cmip5qc.wdc-climate.de.

Requirements
---

1. Installation of the QC tool (checkout from http://svn-mad.zmaw.de/svn/mad/Model/QualCheck/QC/branches/QC-0.x) according to Readme.txt. netcdf-3 library is sufficient, netcdf-4 library is not needed.

1a. oracle: oracle instant client packages "basic" and "sdk"
(http://www.oracle.com/technetwork/database/features/instant-client/index-097480.html)
- set softlinks for version independent libs, e.g.
  libclntsh.so -> libclntsh.so.11.1
  libocci.so -> libocci.so.11.1
- set or add to environment variables:
ORACLE_HOME=<instant client directory>
LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$ORACLE_HOME"
PATH="$PATH:$ORACLE_HOME

2. Python >= 2.4 + modules sqlalchemy,
2a. postgres: psycopg2
  oracle: cx_Oracle
 -> check successful module installation by importing it in python

3. openssl in path for password decryption of DB user

4. For use of qcDbselect.py 'ncdump' in path (QC netcdf result's readability checked by ncdump -h)

5. For use of qcDbselect --outcera option for netCDF plotting 'xmgrace' (incl. netcdf-3, ps and pdf support) and 'cdo' (with netcdf-3 support) in path

5a. postgres: Send the IP of your machine to us for our firewall adaptation

6. data and qc results in netcdf-3 format

Quick Start
---

1. Install QC tool:

 install netcdf library netcdf-3.6.3 with configuration

```
        export CC="/usr/bin/gcc"
        export CFLAGS="-static -O2"
        export CPPFLAGS="-DNDEBUG"
        export CXX="/usr/bin/g++"
        export CXXFLAGS="-static -O2"
        export FC=""
        export F90=""

        ./configure --prefix=path/netcdf-3.6.3  --disable-examples --disable-docs-install --disable-fortran-compiler-check \
```

```
 cd  QC/src
 edit compiler instruction file zg++
 run ./zg++
```

2. Ensure QC tool is installed an running by:

```
   qcManager -E_CHECK_TOOLS
```

3. Run test for QC tool:

```
 cd QC/example
 edit qc_test.conf
 run ../QC/scripts/qcManager -f qc_test.conf
```

3a. postgres: Check accessibility of qc.dkrz.de on port 5432 through your firewall by:

```
telnet qc.dkrz.de 5432
```

3b. oracle: Check accessibility of oracle db by

```
src/access_oracle.py
```

4. Ensure python modules sqlalchemy, psycopg2 and openssl are installed and running and you sent the IP of your machine to martina.stockhause@zmaw.de. Check QC DB access by:

qcDbselect.py --postgresdb=test1 --list

5. Try an QC DB ingest by editing qc.conf example and running:

qcWrapper.py --conf=qc.conf

6. Ensure for plotting (qcDbselect --outcera) that ncdump, xmgrace,psmerge and cdo are installed and working

Run (see http://redmine.dkrz.de/collaboration/projects/cmip5-qc/wiki/Qc_l2#Run-QC)

Problems
----

1. QCDB errors or internet connection problems:
   Make sure you send the IP of the machine on which the QC is installed to us.
   In case of error occurrece after finished QC Tool, run the qcWrapper again with the additional option --noqc to skip the QC tool checks for new data.

2. Statistical QC tool abort:
   Set additional options back to default values of the configure file and enable the option IGNORE_ERROR. Run qcWrapper again.

 a. no error message available:
   Check QCDB_LOGFILE for stderr messages from the QC tool.
   Start the QC tool standalone by getting the call from the QCDB_LOGFILE (grep 'qcManager') and adding the option -E_DEBUG_C

 b. session log with error message available:
   Check it for errors in the processing of the last listed data file

 c. session log without error message:
   Start the QC tool standalone by getting the call from the QCDB_LOGFILE (grep 'qcManager') and adding the option -E_DEBUG_C

3. Statistical QC tool runs without writing any result data
   Cancel qcWrapper.py and try again with enabled option IGNORE_ERROR in the configure file.
   Enable option NUM_EXEC_THREADS in the configure file and set it to a value > 1 and try again.
   Delete output branch for this experiment and try again.
   Update your QC tool and try again.
   Contact martina.stockhause@zmaw.de and send the QCDB_LOGFILE.

4. QC Wrapper reports an error though I exchanged the data
   Delete existing error and warning files in the output directory at dataset level and try again.

Additional Information: docu/CMIP5-AR5-QC-L2.odt; http://cmip5qc.wdc-climate.de
----

Contact: martina.stockhause@zmaw.de

D. QCDB Configure File

The Configure file for the QC tool is enlarged by a couple of options for the qcWrapper, which are start with “QCDB“.

```
# Configuration file for the Quality Control
# ==========================================================

# Note: Default values in []
```

```
# KEY WORDS controlling the application in the quality-control-run.
# NOTE: although the QC is designed to run on distributed computers,
#       this was not tested for a long time and is not garanteed.

# Detailed explanation of syntax rules at the end of this file.
# QC Wrapper options

# Contact information for the QC run (person who performed the QC).
# Required by QcDb.
# Will be stored in the QC result DB and used in case of questions about the QC checks
  QCDB_PERSON=Name
  QCDB_EMAIL=name@institute.com

# QcDb information (name of sqlite DB file and log file).
# Required by QcDb.
  QCDB_LOGFILE=/mypath/qcdb.log
  QCDB_POSTGRESDB=test1      # testing: test1; production-output1: qc1; production-output2: qc2;

# DRS syntax file (placed in directory QCWrapper/table)
# Required by QcDb.
#  QCDB_DRS=drsold.txt # drsnew.txt (DRS of oct. 2010); drsold.txt (DRS of 2010-04-08)
  QCDB_DRS=drsnew.txt # drsnew.txt (DRS of oct. 2010); drsold.txt (DRS of 2010-04-08)

# ESG dataset version to be checked (DRS syntax version v<date>)
# default: 'latest' version found in DATA_ROOT_FS directory
# QCDB_VERSION=latest  # other example 'v20100928' or shortened 'v201009'
  QCDB_VERSION=v20     # all data versions

# QC Tool directory.
# Required by QcDb.
  QCDB_QCDEST=/mypath/QC

# QC Tool options
# Path to QC's data and log dir. Required.
# Enter the local root for the CMIP5 directory, the DRS path is added automatically to grant path uniqueness.
  QC_DATA_ROOT=/mypath/QCResults

# Path to the file system based netCDF files, which are to be checked.
# Note: all other sub-paths are appended to QC_DATA_ROOT/data
  DATA_ROOT_FS=/drsroot/CMIP5/output/MPI-M/ECHAM6-MPIOM-LR/rcp45

# ---------  please DON'T change

#
# This option controls whether the second time-stamp in the filename
# expresses the sharp end (deadline) of the period or whether the
# period in the stamp has to be extended to the end,
# e.g. 2000-03 <=> 2000-03-31 for extended.
# The difference is explained best by examples.
# Sharp:     2000 - 2001 <=> 2000-01-01T00:00:00 - 2001-01-01T00:00:00
# Extended:  2000 - 2001 <=> 2000-01-01T00:00:00 - 2001-12-31T24:00:00
# Sharp is applied as default, i.e. disabled.
# If there is a mix for the components of the stamps, then use this
# syntax: 'yx-mx-dx-Hx-Mx-Sx' with x=s or x=e for sharp and extended,
# respectively. Omission of any component sets individually x=s.
# E.g. APPLY_MAXIMUM_DATE_RANGE=ye-ms <=> ...=ye-ms-ds-Hs-Ms-Ss
#
# CMIP5-QC: CMOR extracts the frequencies out of the netCDF file headers:
# E.g. 2000 stands for one value representing the whole year.
# Therefore the data interpretation should be set to extended for all data.
  APPLY_MAXIMUM_DATE_RANGE=ye-me-de-He-Me-Se #ye=ye-ms-ds-Hs-Ms-Ss

# The original directory tree structure of the source data is preserved,
# i.e. a symbolic link there causes making a corresponding symbolic link
# in the directory tree of the QC results, too. If symbolic links
# are to be dereferenced, i.e. genuine data are to replace the symbolic links
# in the QC result tree, enable this option.
  DEREFERENCE_SYM_LINKS # [disabled]

# If distributed computing is also involved with distributed file system,
```

```
# then enable this option. Bi-directional communication and data
# transfer is done by ssh and scp, respectively. IMPORTANT NOTE: this is
# not installed and not projected for CMIP5. However, the QC for CMIP5
# may have distributed computing with a shared file system.
# DISTRIBUTED_FS # [disabled]

# Detected errors/warnings due to meta data checks (variables and
# dimensions) are merged by default on a single line in the log-files.
# To have each on a line of its own, enable this option.
  FINE_META_DATA_CHECK_REPORT=all # [f]

# An error in any record causes blocking of further processing
# (after the one causing it has finished completely). This option
# ignores any locks for coming chunk-files.
# Irrelevant for atomic datasets.
#  IGNORE_LOCK_FILES=t   # [f]

# Usally, detecting an error in the netCDF layout in
# comparison to the project table lets stop the QC and an error-file is
# generated blocking any further analysis of the respective variable.
# This option converts error-files into warnings and thus, cancle
# the blocking. I hope, you have read the disclaimer-of-warranty!
# Note: Real run-time errors shall not be ignored.
# Note: Failed checks against a standard table issue always warnings.
#  IGNORE_PROJECT_TABLE_ERR  # [disabled]

# Write the svn version-numbers of each package file to
# a file for each session. Location: QC_DATA_ROOT/session_logs
# Name:qc_version_yyyy-mm-dd_hh:mm:ss.log
# Note: if not svn, then names and modification dates
  QC_VERSION  # [disabled]

# By default, errors in the time variable will not abort the QC
# of given variable and block further attempts.
# STOP_AT_TIME_ERROR  #[disabled]

# Usually, the table prescribes a direct dimensionality of
# the grid-layout. However, a wider scope is often available
# in models (e.g. parameterised dimensions in ocean models depending
# on multiple arbitrary dimensions). This option allows for such
# a wider scope and a dimension is only checked if it is defined
# with identical layout within both table and file.
  TABLE_RELAXED_DIM_CONSTRAINT # [disabled]

# Raise only warnings for errors in the attribute of a variable.
  TABLE_RELAXED_VAR_CONSTRAINT # [disabled]

# Variables must be declared in the standard table.
# The behaviour by default, i.e. disabled, is described below.
# If there is any variable in the model run that is not part
# of CMIP5, then disable this.
  TABLE_VARIABLE_CONSTRAINT # [disabled]

# ---------  please check
#
# A table providing definitions of names, units, type, dimensions, which
# must be given as comma-separated-values.
# Basically, the properties of each file in QC are checked against this
# table for the entire project. If the standard table is not given,
# then a project table is created with the properties found in the
# first attempt of checking. All tables are looked for in the path
# QC_SRC/PROJECT_TABLES. This default path may be changed by a leading
# path in front of the name of the standard table (and must then be
# provided for each experiment, i.e. scenario, of the project).
# If not specified, then "$QC_SRC/Project_tables" by default or the
# path preceding one of the tables; the latter with precedence.
#  TABLE_PATH=...  # [ $QC_SRC/Project_tables ]

#  Rules for naming the project table:
#  1) precedence of a fully qualified project name 'anything'.csv
#  2) incomplete project table name and standard table name:
```

```
#     TABLE_PROJECT="incompleteName"_standardTable
#  3) incomplete project name and no standard table:
#     TABLE_PROJECT="incompleteName_table.csv", however,
#     "_table" substring omitted if "table" is already available.
#  4) no project table, but standard table:
#     TABLE_PROJECT="project_"standardTable
#  5) no project table and no standard table:
#     TABLE_PROJECT="project_table.csv"
#  TABLE_PROJECT=pt  # rule 2)

# An 'officially' standardised table. Format: comma-separated-values
#  TABLE_STANDARD=standard_output_9Jul2010.csv # [none]
  TABLE_STANDARD=standard_output_13Apr2011.csv # [none]

# ---------  performance relevant options
# By default, calculation of cell averages over a geographical grid is
# performed by weighting with the corresponding grid-cell area. In
# particular for ocean basin cross-section, this is very time consuming
# due to the irregular shape of the basins taken into account by
# varying filling values (also for regular grids with varying effective
# areas, e.g. T of snow). If the precise value of the average is less
# important, this option reduces calculation time.
  ARITHMETIC_MEAN  # [disabled]

# Calculate the frequency distribution of the entire domain
# Note: generally, you will need post-processing with 'hist.x' to
# assemble the final freq dist, except you select specific
# debug options.
# Rules for selecting class widths and starting point (e.g. centering):
# A fd-build-file of the same variable at destination has precedence.
# Second choice is a fd-build-file or fd-prop-file (the header of
# a build-file) in a specified path.
# Third choice is an explict statement of properties.
# Default: automatic determination of fd properties.
# FREQ_DIST  # [disabled]

# Partitioning the total time span of an experiment into
# smaller ranges. FreqDists will be calculated separately for each
# window. Note: the residual time interval may be shorter.
# Note: if no unit-designator is appended (y, mo, d, h, mi, s)
# the unit from the time variable is used.
# FD_TIME_PART=5y # 30y [ default: the entire experiment]

# By default, freq. dists. adjust automaticly the bin (class) width
# while processing the first time window (or the total time span),
# which is then used in subsequent time windows for the entire experiment.
# Specifying this path enables to retain the properties of the same
# variable (if found) of another experiment, e.g. control run.
# Such a file must have extension .build or .prop
# FD_PROPERTY_PATH=  # [no path]

# Explicit properties (two quantities separated by '/'):
#    class-width (number) / init value (number)
# e.g. 1/0 will centre a bin (class) of the freq.dist around zero.
# Note: The alignment of the fd-bin borders is done automaticly.
#       It is not possible, do have a class boundary exactly
#       at say 273.15 with class-width of 1.
# This option is only reasonable for selecting a particular common type
# of variable, e.g. temperature, in a separate run. See priority rules.
# FD_EXPLICIT_PROPS=1.E-05/3.5E-05

# Output of the Freq Dist as ready-to-use. By default, the output
# will be in a format to be re-read later in order to resume a
# previous session (the output file gets extension '.build').
# File extension in plain mode: '.hist'. Attention: this starts
# and completes a FD calculation. Makes little sense for
# multi-sessions. Purpose: debugging
# FD_PLAIN  # [disabled]

# Like FREQ_DIST_PLAIN, but output of the Freq Dist shows complete
# shapes of bars. Purpose: debugging
```

```
# FD_BARS  # [disabled]

# List of number(s) of simultaneous execution processes per host.
# Note: there is always a single qcManager process.
# Special: If a single number is specified, this is assigned to
# all hosts in the QC_EXEC_HOSTS list.
# Fine tuning: each positional number in the list corresponds to a
# position in the QC_EXEC_HOSTS list. If less positions are given
# than in QC_EXEC_HOSTS, then the last position is assigned to
# the omitted positions.
#  NUM_EXEC_THREADS=2 #1,0  # [1]

# ---------  adapt if necessary, defaults set
# Facing trillions of files to be checked, the logging of executions
# statements of successful checks would allocate a lot of disk space.
# With this option, the QC result log-tree would only contain logfiles
# for abnormally exited qC.x runs. Note: experiment and session log-files
# are always written.
  DISCARD_LOG_OF_GOOD_EXEC # [disabled]

# If distributed computing is also involved with distributed file system,
# then enable this option. Bi-directional communication and data
# transfer is done by ssh and scp, respectively. IMPORTANT NOTE: this is
# not installed and not projected for CMIP5. However, the QC for CMIP5
# may have distributed computing with a shared file system.
# DISTRIBUTED_FS # [disabled]

# Exclude specific variables globally from a check.
#  EXCLUDE_VARIABLE=average_T1,average_T2,average_DT

# For CMIP5 DRS compliant filnames variable name and sub-table names
# are found in the filename by default. However for non-compliant
# filenames, the two can be supplied by pattern matching with regular
# expressions.
# IMPORTANT: to check non-compliant, non-time dependent variables,
# so-called fixed fields, specification of FILENAME_PATTERN_VAR is
# the only way.
# The default for a non-compliant sub-table is 'Any'.
# Here is an example for a file 'yxc.asdf.pr_A1_1.nc' with
# variable 'pr' and sub-table 'A1'.
#  FILENAME_PATTERN_SUBTABLE='.*_\(.*\)_.*.nc'
#  FILENAME_PATTERN_VAR='.*\.\(.*\)_.*_.*.nc'


# A directory acting as communication-bench by semaphore files
# between qcManager and qcExecutor.
# PROC_POOL=$QC_SRC/ProcPool_CMPI5   #  [$QC_SRC/ProcPool]

# Path to the C compiler. A few c/c++ -progs will be compiled, if the
# executables do not exist (see function 'checkTools' in qcConfigurator).
# C/c++ -sources reside in QC/src.
# CC=   # [/usr/bin/gcc]
# CFLAGS=  # [disabled]

# The QC checks meta data, time data and data (of variables), where
# for the latter two the records are inspected.
# Modes can be enabled by key-words: meta, time, data.
# Note: data includes meta and time; time includes meta.
# CHECK_MODE=meta  #[data]

# Path to the C++ compiler. Must be identical to the one
# used to build netCDF4.
# CXX=/usr/bin/g++   # [/usr/bin/g++]
# CPPFLAGS=    # [disabled]

# Calculate checksum of every file that passed successfully the QC.
# If md5 or sha1 are assigned, then a seperate text file is generated
# where the checksum is stored with appended extension md5 or sha1,
# respectively, to the filename, i.e. filename.nc.md5.
# Else, a named function/program is executed.
# Such a function must be available in the user's search path for
```

```
# commands or the path has to be prefixed.
# The function is not part of the QC package. If thefunction outputs two
# elements to standard output (suppose: filename and a kind-of-checksum),
# then these are written to a table in the project path named
# according to the experiment-log-name prefixed by 'cs_'.
# Select the checksum algorithm ( md5 | sha1 | function_in_path)
# CHECKSUM=checkSum  # [disabled]

# If time-stamps of a file extend during the process of
# file creation and CHECKSUM is enabled, then the detection of
# a completed sub-temporal data set is not sufficiently given by
# comparing the time-stamp period with the time values in the file.
# This option will identify a sub-temporal data set as complete, when
# there is another file available with adjacent time-stamp.
  CHECKSUM_DYNAMIC_TIME_STAMP # [enabled]

# Checksums are only calculated, if the QC detects no error AND a
# range of dates is appended to the filename that fits with defined
# uncertainty to the time range in the netCDF file. This option here
# applies to files without range of dates. A checksum is calculated
# if the QC detected no further modification to the file and there
# is no error-file. Note: this option should only be switched on
# when the model has reached the final end of an experiment. This
# option is also available via the command-line; set -E_CHECKSUM_FINAL
# CHECKSUM_FINAL  # [disabled]

# Remove all former results corresponding to the paths and variables
# selected before re-doing the QC. However, corresponding log-files
# are not affected. If key-word 'only' is assigned, i.e. CLEAR=only,
# then no re-doing will take place. CLEAR=resume: start over QC check after error abort
# CLEAR=resume # [disabled]

# Generally, the QC will check every netCDF file found in the
# sub-directory tree under DATA_ROOT_FS (considering SELECT/LOCK!).
# However, from the directory layout of the Data Reference Syntax (DRS)
#--------------------------------
# <activity>/<institute>/<model>/<experiment>/<frequenc>/...
# <modeling realm>/<variable_name>/<ensemble_member>/<version>
#--------------------------------
# the tokens for <version> [if any], <ensemble_member>, <experiment>,
# <institute>, and <activity> are used to compose a unique
# log-file name for collecting all execution status output.
# A different DRS syntax may be provided by a
# comma-separated list by two alternative methods.
# 1) Random fulls specified sequence:
#    e<r_pos1>,m<r_pos2>,v<r_pos3>,i<r_pos4>,a<r_pos5>,b<r_pos6>
# (Note: m := model, b:= ensemble member, others as they match 1st char.)
# where the tokens r_posN have to be replaced by the real positional
# indexes counted from the back!!! starting with 1;
# e.g. the DRS syntax above has: m7,e6,a9,i8,b2,v1.
# The constructed filename is written into the
# experiment_log directory in the output path.
# 2) Pure comma-separated integers may be specified (any count). Then,
# the sequence of tokens is given by the sequence of integers.
#
# If no string can be composed, because the sub-path appended at
# DATA_ROOT_FS path has less items, then the default experiment_log file
# name is 'unknownExp.log'.
# NOTE: the user is responsible for setting DRS_TREE_INDEX corresponding
# to the sub path appended to the path in DATA_ROOT_FS (use SELECT/LOCK).
# SPECIAL: <version> is the right most item in the DRS sub-path and
# no index was provided for <version>: then, all <version>s to be found
# are taken into account automatically for constructing an experiment
# name, even if there is a mix with sup-paths without any <version>
# (those without a <version> token are also considered). In that case
# the index of <version> has to be ignored as if no <version> would be
# available, e.g. there is b1 for the trailing sequence <member>/<version>
# (the <version> name must have 'v' as first character). If there is a
# <version> index specified and it does not really fit the DRS/sub-path,
# then you will potentially generate a lot of experiment log-files
# (may-be for each variable; it depends).
```

```
# IMPORTANT: This does not select any path for checking.
# DRS_TREE_INDEX=i7,m6,e5,b1  # [i7,m6,e5,b1]

# In case of error: send e-mails to this comma-separated-list.
# Note: Be attuned to be flooded. In a testing phase you would
#       appreciate to set EMERGENCY_STOP=t.
# If empty, nobody will be notified.
# EMAIL_TO=your-email-adress@x.y,...    #[]

# Notify the finish of a session by e-mail.
# EMAIL_END_OF_SESSION  # [disabled]

# Stop all after encounting any error which
# causes notification by email.
# EMERGENCY_STOP  # [disabled]

# Enable a flowtrace analysis of the main loop of qcManager
# FLOW_TRACE   # [disabled]

# Set group name. This is necessary, if the QC is also operated
# by users who are not in the default group of the user who initially
# checked out from the svn repository. This will automatically also
# set the SGID-bit, i.e. grant permissions to all group members.
# The current setting of user permissions of each file is duplicated
# to the respective group permissions.
# Note: this can only be done by the owner of the QC directory.
# GROUP_NAME=  # [none]

# In order to enable trapping signals, long sleeping period are
# subdivided into smaller intervals of consecutive sleep commands.
# HARD_SLEEP_PERIOD=10  # [10]s

# The Hans-Luthardt-Extreme-Value-Detection (see code in qcExecutor).
# The HL extreme value is expressed as deviation range normalised to
# the units of standard deviation: (max-min)/stdDev,
# i.e. is dimensionless for each grid-cell.
# If a threshold is exceeded, then by default, a notification
# is written to the SESSION_LOG.
# Makes not really sense for applying to 3D.
# Derived by and therefore requires CDO.
# HLEVD  # [disabled]

# Delete netCDF file with HLEV in any case;
# only notification takes place.
# HLEVD_DELETE_FILE  # [disabled]

# To detrend a time series takes some time.
# If a trend of a variable does not boost exponentially,
# detrending should not affect this kind of extreme value
# detection (which is anyway rather vague).
# HLEVD_DETREND  # [disabled]

# Keep netCDF file with HLEV if threshold was exceeded.
# HLEVD_KEEP_THRESHOLD_FILE  # [disabled]

# Threshold: (max-min)/stdDev > threshold
# HLEVD_THRESHOLD=9  # [ 1 ]

# Locking of variables (LOCK takes precedence over SELECT)
# LOCK .*/Z=   # [disabled]

# Avoid requesting only a few blobs, if the data base
# is in the state of growing and/or the number of recs
# is smaller. Then, the requested number is enlarged.
# MAXIMUM_RECORD_SIZE=1000     # [1000]

# Maximum of free disk space required. Note: the real
# consumption will be definitely smaller. Units:
# none == KB, MB, GB, TB, PT
# MAX_REQUIRED_DISK_SPACE=10GB    #   [10GB]
```

```
# Avoid requesting only a few blobs, if the data base
# is in the state of growing and/or the number of requested
# records is small. If there is a small number of exceeding
# records, then MINIMUM_RECORD_SIZE is enlarged for the given file.
# MINIMUM_RECORD_SIZE=1000     # [1]

# Only progress by the given number of records at each check.
# Purpose: processing analyses.
# NEXT_RECORDS=10

# Apply linux 'nice' command to executables on
# guest machines (see 'man nice').
# NICE=15  # [0]

# Suspend processes between 8 - 19 o'clock on guest hosts and
# allow only a single process on the qcManager host.
# NIGHT_SHIFT # [disabled]

# The QC checks for each first sub-temporal file of each variable
# the continuation from a corresponding file of a parent experiment
# before a regular check of the child. This option disables any
# regular QC. Thus, only the parent-child crossing is checked.
# PARENT_CHILD_CHECK_ONLY  # [disabled]

# Usually, the QC checks for continuity between parent and derived
# experiments by using the global attribute 'parent_experiment_id'
# from the netCDF header. However, knowing the parent experiment is not
# sufficient for a child to infer a unique path to its ancestor.
# The algorithm applies the following rules:
# 1)  PARENT_EXP_ID=none skips any parent-child relation test.
# 2)  From global attributes of the netCDF file:
#     parent_experiment_id and parent_experiment_rip.
# 3a) If 2) fails, then PARENT_EXP_ID and/or PARENT_ENSEMBLE_MEMBER
#       are used.
# 3b) PARENT_ENSEMBLE_MEMBER not specified: same as of the child.
# 4)  PARENT_VERSION as specified;    else: same as of the child.
# 5)  Parent and child must be located in the same DRS directory tree
#     with similar setting, but experiment_id and version.
# 6)  The last sub-temporal file of the parent must be available.
#     If this fails, then the QC will throw an error flag.
# 7)  The QC checks the validity of the path to the parent file.
#     If invalid, then a notification is given as usual.
# It is due to the user that the rules apply.
  PARENT_EXP_ID=none  # [from the child]

# This parameter should be defined in the global attributes of
# the netCDF file as 'parent_experiment_rip'. This is looked for as
# the default. If absent, then the ensemble member of the current
# experiment is tried. Set option, if nothing fits (e.g. r1i1p1).
# PARENT_ENSEMBLE_MEMBER=

# PARENT_VERSION=          # [same as of the child]

# Not enabled at present.
# PARENT_PATH_PREFIX=path

# Switch on a quality control (on by default). It is possible to
# switch it off, for instance to calculate only frequency distributions.
# (not a good example, because both can be run simultaneously).
# QC=off  # [on]

# Executables reside in QC/bin by default. For the purpose of running
# the QC from another machine on the same file system, there may be
# a different bin directory specified. If the path is relative, the
# alternative bin directory has to be a sub-dir of QC.
# You have to compile all executables into this directory.
# QC_BIN=bin2 #[bin]

# A list of machines executing asynchronous jobs.
# QC_EXEC_HOSTS=comma-separated-list-of-hosts  # [$HOSTNAME of qcManager]
```

```
# The machine where a session is started, usually the user's pc.
# QC_MANAGER_HOST=your-hostname # [$HOSTNAME]

# If a process cannot be run, because e.g. the data base is not available
# or a server is absent, then, after a sleep, retry. But only
# for the below specified number.
# REATTEMPT_LIMIT=5  # [5]

# Recalculate a quality check for all selected variables,
# unless calculating the checksum, if enabled, reveals no changes.
# Any error-files, i.e. 'qc_error...' and 'qc_warning...' are removed.
# If a range is in terms of date, then the two dates are separated by '-';
# times have to be separared by '/'.
# Example for replacing chunks:
# REDO=20100801/20101015
# REDO        #[disabled; else redo all selected variables]
# REDO=t0/t1  #[disabled; else for a time range]
# REDO=d0-d1  #[disabled; else for the CMIP5 formatted range of dates]

# Selection of paths and variables (RegExp of the 'expr' command,
# i.e. full specification from the beginning of the word).
# Omission selects all.
# Paths and vars are given by '[path1[,path2,..]]=[var1[,var2,..]]'.
# Omission for path... selects the selected vars in all paths. Omission
# of var... selects all variables in the path(s).
# Notice the '=' sign at the end of the comma-sep list for paths.
# SELECT must precede each line for multiple lines.
# SELECT path=var [disabled]
# SELECT pr_*     # all variables named pr_What-Ever-You-Like
# SELECT .*/6hr,.*/mon/atmos/=  # example for a path selection

# Sleep period if waiting for a server or anything else. Note:
# particular ordinary processes have a sleep time of their own
# to distinguish them in the output of 'ps'.
# SLEEP_PERIOD=300 # [300] s

# Stop the QC at specified time (QC excluded).
# Use ISO-8601 format (yyyy-mm-ddThh:mm:ss). Truncations are accepted
# (1955-06 would process the last record before 1955-06-01 00:00:00).
# TIME_LIMIT=1955-08

# Enable checks for continuous data arrival not in temporal order;
# Note: If 'WAIT_FOR_MISSING_DATA' is disabled, i.e. the default,
# and the QC detects a gap caused by a missing file and additionally
# 'STOP_AT_TIME_ERROR' is disabled, also the default,
# then the qc-result.nc file contains a broken series of time values.
# WAIT_FOR_MISSING_DATA # [disabled]

# Start the analysis on a guest machine, if the average load from
# the uptime command is < 1.5 for 5 minutes and < 1 for 15 min.
# Note: the jblob request is startet immediately.
# WORK_AT_LOW_LOAD  # [disabled]

# End of Configuration Section
# ==============================

# Syntax Rules of the Configuration File
# --------------------------------------------------------

# a) There are key-words, assignments, comments. Key-words are always
# uppercase. Assignments of multiple values may be given as
# comma-separated lists. A comment is everything following the '#'
# character. Empty lines, and spaces are discarded.
# Notice that the spelling of key-words is not checked. Assignments may
# be anything. Unknown key-words (or typos) are ignored. Significant
# spaces (I hope there are none for CMIP5) have to be escaped by '\'.
# BUT, #this is not implemented. Will be done only on request!

# b) Assignment to a key-word is given by: key-word=value.

# c) Enabling/disabling features is done by assigning t/f to a key-word.
```

```
# A pure key-word on a text line defaults to true [t],
# e.g. ARITHMETIC_MEAN is equivalent to ARITHMETIC_MEAN=t
# Disabling by assigning 'f' is equivalent to commenting out.

# d) Multiple line assignments. An assignment spanning multiple lines
# must #begin with key-word := value. The last line of the block must be
# followed by an empty or comment line.

# f) SELECT/LOCK has a special syntax (with [] indicating optional):
# SELECT [path1[, path2, ...] = ] [var1[, var2, ...]]    # same for LOCK
# A path is a sub-path appended to the tree given by DATA_ROOT_FS.
# Several SELECT and LOCK statements, respectively, are cumulative.
# There must be no assignment character '=' following key-words SELECT or # LOCK.

# Each variable specification is applied to all paths.

# Assignments may contain (and usually do) regular expressions. The rules # of the RegExp must be those of the
'expr' command explained
# in 'man  grep'.
# This implies that each regExpr must begin from the first character with # the caret ^ omitted. E.g. SELECT
.*/historical/.*/Amon =

# Omitting path and variable defaults to all, respectively, i.e. '.*'

# Important: selecting one or more paths requires a '=' character behind # the last path. Otherwise, it is taken as
selection for variables.

# SELECT := starts a multiple line selection (same rules as for multiple # line assignment). Same for LOCK. Multiple
lines selections may be mixed # with such given by a single line.#

# Note: SELECT in a config-file and per command-line option '-S arg'
# are combined; arg (without key-word SELECT) is as a single-line.

# Examples:
# SELECT path=var # specifies a single path where to look for a single variable
# SELECT p1,p2=var  #  two paths to look for a variable
# SELECT p1,p2=  #  two paths with every variable, equivalent to p1,p2=.*
# SELECT p1=v1,v2  #   one path with two variables
# SELECT var[,v2,v3]  #   looks for variables in the entire DRS tree
# SELECT p1=,p2=v1...  --> error,  SELECT  p1,p2=v1...  --> ok.
```

E. CIM Quality Document

This is the actual example for a possible usage of the metafor quality document for publishing QC L2 results.

```
<CIMRecord xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:gco="http://www.isotc211.org/2005/gco"
xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns="http://www.metaforclimate.eu/schema/cim/1.5"
xsi:schemaLocation="http://www.metaforclimate.eu/schema/cim/1.5/cim.xsd">
<!-- Reference to simulation run document / DRS experiment to which this quality document belongs: How to get
Address and ID? -->
<reference xlink:href="http://???/simulationRun[id='???????']">
<CIMRecord>
  <quality>
    <scope>
      <level level="simulation"/>
          <levelDescription>subjective quality control passed</levelDescription><!-- Quality Flag; we could
alternatively add the QC Level number and use the field in key:value manner: QualityFlag:subjective quality control
passed QualityLevel:2 -->
    </scope>
    <report>
          <nameOfMeasure>WDCC Conformance and Subjective Controls</nameOfMeasure><!-- as named in
CMIP5 QC document -->
```

<measureDescription>WDCC automated technical quality checks of data (details in http://cmip-pcmdi.llnl.gov/cmip5/docs/CMIP5-AR5-QualityControl.pdf) and subjective data and metadata controls
Technical Quality Checks:
a) File consistency
1)in the end files will have the right number of records. The number is given in the metadata.
2)strictly regular time steps (ESG checker allows for time gaps)
b) Metadata consistency (check of consistency between metadata of the standard_output table and the metadata of the file headers)
c) Physical properties of variables
3) minimum and maximum are checked against specified ranges (default for an invalid current exterm value of a global field:
mean:
mextr(t) = 1 / N * sum_{i=0}^{N=t / (delta t) - 1} extr_{i}
N:= index over all time steps 'delta t' up to the actual time t
Default (case for all variables in the standard_output table of K. Taylor):
mextr_{N-1} - extr_{N} > order_of_magnitude(mextr_{N-1} * 10^{5}
4) time series are calculated for:
min
max
globally weighted mean (in case of no _FillValue)
area weighted mean (in case of existing _FillValue; reasonable, e.g., for temperature of snow)
standard deviation of the globally weighted mean.

Subjective Controls of data and metadata:
....?</measureDescription>
<dateTime>2010-08-17T09:30:47Z</dateTime>
<result>
<dQ_ConformanceResult pass="1" ><!-- "0" failed -->
<specification xlink:href="http://cmip-pcmdi.llnl.gov/cmip5/docs/CMIP5-AR5-QualityControl.pdf" xlink:title="The CMIP5/AR5 Model Data Quality Control" /><!-- CMIP5 QC document:explanation of checks performed -->
<gmd:CI_Citation>
<gmd:title><gco:CharacterString>The CMIP5/AR5 Model Data Quality Control (http://cmip-pcmdi.llnl.gov/cmip5/docs/CMIP5-AR5-QualityControl.pdf)</gco:CharacterString></gmd:title>
<gmd:date><gmd:CI_Date><gco:date><gco:Date>2010-06-29</gco:Date></gco:date></gmd:CI_Date></gmd:date>
<!-- person done the QC L2 -->
<gmd:citedResponsibleParty>
<gmd:CI_ResponsibleParty>
<gmd:individualName><gco:CharacterString>Name</gco:CharacterString></gmd:individualName>
<gmd:contactInfo><gmd:CI_Contact><gmd:address><gmd:CI_Address><gmd:electronicMailAddress><gco:CharacterString>name@neworg.edu</gco:CharacterString></gmd:electronicMailAddress></gmd:CI_Address></gmd:address></gmd:CI_Contact></gmd:contactInfo>
<gmd:role><CI_RoleCode codeList="http://www.isotc211.org/2005/resources/CodeList/gmxCodelists.xml#CI_RoleCode" codeListValue="author" /></gmd:role> <!-- or: Author -->
</gmd:CI_ResponsibleParty>
</gmd:citedResponsibleParty>
</gmd:CI_Citation>
</specification>
<explanation>CMIP5 Quality Level 2 (subjective quality control passed)

Quality Control Levels in CMIP5:
* Level 0: Spot checks on selected data
* Level 1: CMOR2 and ESG publisher conformance checks
* Level 2: Technical checks on the reliability of variable ranges and consistency checks between data and metadata
* Level 3: Data approved by author and published as STD-DOI</explanation><!-- what does result mean -->
</dQ_ConformanceResult>
</result>
</report>
</quality>

```
<!-- create UUID or is it done by the repository? -->
  <documentID>??</documentID>
  <documentVersion>1</documentVersion>
  <documentCreationDate>2010-06-20T00:00:00</documentCreationDate>
</CIMRecord>
</CIMRecord>
```