


The EUMETSAT Network of Satellite Application Facilities	 NWP SAF Numerical Weather Prediction	AAPP v8 Top-level Design	Doc ID :NWPSAF- MO-DS-034 Version : 1.0 Date : 06/09/2017
---	--	---------------------------------	---

AAPP v8 Top-level Design


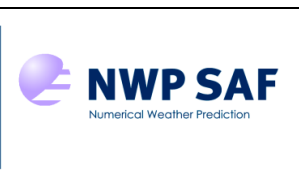
This documentation was developed within the context of the EUMETSAT Satellite Application Facility on Numerical Weather Prediction (NWP SAF), under the Cooperation Agreement dated 7 December 2016, between EUMETSAT and the Met Office, UK, by one or more partners within the NWP SAF. The partners in the NWP SAF are the Met Office, ECMWF, DWD and Météo France.

COPYRIGHT 2017, EUMETSAT, ALL RIGHTS RESERVED.

Change record			
Version	Date	Author / changed by	Remarks
1.0	06/09/2017	N C Atkinson	

Table of Contents

1.	INTRODUCTION	3
1.1	Reference documents	3
1.2	Design Drivers	3
2.	MAIA V4	3
3.	IMAGER CLUSTERS GENERATION	7
4.	ECCODES INTERFACES FOR BUFR ENCODE/DECODE	8
5.	OTHER CHANGES INTRODUCED SINCE RELEASE OF AAPP V7.1 ..	10

		<h1>AAPP v8 Top-level Design</h1>	Doc ID :NWPSAF- MO-DS-034 Version : 1.0 Date : 06/09/2017
--	--	-----------------------------------	---

1. INTRODUCTION

This document defines the top level design for AAPP version 8, in accordance with the requirements of the NWP SAF.

1.1 Reference documents¹

[RD-1]	NWPSAF-MF-UD-001, AAPP Documentation Scientific Description
[RD-2]	NWPSAF-MF-UD-002, AAPP Documentation Software Description
[RD-3]	NWPSAF-MF-UD-003, AAPP Documentation Data Formats
[RD-4]	NWPSAF-MO-UD-004, AAPP Overview
[RD-5]	NWPSAF-MO-UD-005, AAPP Installation Guide
[RD-6]	NWPSAF-MO-UD-036, AAPP User Guide
[RD-7]	NWPSAF-MF-UD-003, OPS-LRS User Manual
[RD-8]	NWPSAF-MO-UD-027, Annex to AAPP scientific documentation: Pre-processing of ATMS and CrIS
[RD-9]	NWPSAF-MO-DS-014, AAPP Version 7 Top Level Design
[RD-10]	NWPSAF-MO-DS-033, AAPP Version 8 Product Specification

1.2 Design Drivers

The main changes in AAPP v8, compared with v7 are detailed in the Product Specification (RD-10):

- Updated MAIA v4 replacing MAIA2.1, MAIA3 and the initial MAIA4.
- Imager clusters generation for hyperspectral sounders (to be introduced later as an update release)
- BUFR encode/decode routines based on ecCodes. The existing routines based on BUFRDC will be retained.

These are detailed in the following sections.

Existing functions of AAPP will be unchanged at top level: for the design of these, see RD-9, in conjunction with the list of changes that were introduced as v7 updates (section 5).

2. MAIA V4



MAIA4 is the current version of the maia cloud mask for VIIRS (introduced with AAPP v7.5), this version will be extended to AVHRR and will replace the MAIA3 cloud mask.

The file format for MAIA4 Cloud files is hdf5. MAIA4 VIIRS files are called “viiCT” and MAIA4 AVHRR “avhCT”.

In this document, the avh11c file converted into hdf5 file format is called “avh5”.

a) Obsolete code

¹ See <https://nwpsaf.eu/site/software/aapp/documentation/>

		<h1 style="text-align: center;">AAPP v8 Top-level Design</h1>	Doc ID :NWPSAF- MO-DS-034 Version : 1.0 Date : 06/09/2017
---	---	---	---

AAPP/src/preproc/libmaia_2.1
AAPP/src/preproc/libavh2hirs_maia_2.1
AAPP/src/maia3
metop-tools/src/bin/aapp1d2pfs.ksh
metop-tools/src/bin/aapp1d2pfs.c
src/maia3/bin/avhrrin.F is kept and moved into the src/maia4/bin directory.

b) Obsolete data files

AAPP/data_maia3

c) New source code

AAPP/src/maia4/bin/MAIA4_RUN_AVHRR.ksh: top level script for AVHRR cloud mask on AVHRR grid

AAPP/src/maia4/bin/maia_Avhrr.F90: generation of hdf5 MAIA4 file for AVHRR
input : avh5: AVHRR level1c in hdf5 format (generated with avhl1c_to_hd5.exe from avhl1c file)
output: MAIA4 hdf5 file for AVHRR

AAPP/src/maia4/bin/maia4_Avhrr.ksh: script which executes maia_Avhrr.exe

AAPP/src/maia4/bin/maia2pfs.c : replacement for aapp1d2pfs.c
input : pfsAVH1B Level 1b PFS file and MAIA4 (hdf5)
output : pfsAVH1Bc Level 1b PFS file with CLOUD_INFORMATION updated

AAPP/src/maia4/libmaia4/maia_wrapper.F90 : high level subroutine for calling MAIA
input : filename or fortran file unit
output : maia_par(nbparam,nbpixel,nblines) : array of parameters computed by MAIA.

AAPP/src/maia4/libmai4IO/maia_IO_AvhCT_h5.F90
input/output subroutines for avhCT (MAIA4 AVHRR)

AAPP/src/maia4/libmai4IO/maia_read_Avh1c_h5.F90
input subroutines for AVHRR (for avh1c hdf5 file format)

AAPP/src/maia4/libmaia4IO/avhrr_maia4_data_def.F90
AAPP/src/maia4/libmaia4IO/avhrr_maia4_data_mem.F90
AAPP/src/maia4/libmaia4IO/avhrr_maia4_data_io.F90
AAPP/src/maia4/libmaia4IO/avhrr_maia4_header_def.F90
AAPP/src/maia4/libmaia4IO/avhrr_maia4_header_mem.F90
AAPP/src/maia4/libmaia4IO/avhrr_maia4_header_io.F90
AAPP/src/maia4/libmaia4IO/mod_maia_att.F90

d) updated code source

AAPP/src/preproc/bin/avh2hirs_atovs.F

AAPP/src/preproc/bin/avh2hirs_atovs.ksh
AAPP/src/preproc/bin/avh2hirs.F
AAPP/src/preproc/bin/avh2hirs.ksh

source code updated from libmaia_2.1:

AAPP/src/preproc/libavh2hirs/av_map_maia.F
AAPP/src/preproc/libavh2hirs/ppellip.F
AAPP/src/preproc/libavh2hirs/xavg.F
AAPP/src/preproc/libavh2hirs/fill_maia.F (updated from maia.F)
AAPP/include/avh2hirs.h (updated from maia.h)

avh2hirs.F and avh2hirs_atovs.F will call the maia_wrapper subroutine and will use the cloud information from the maia_par array

MAIA4 needs forecasts unlike maia2 which could work with a climatology, therefore there will be an option to deactivate the call to MAIA for users who do not have available forecasts.

e) Data files

Thresholds for MAIA4 for all AVHRR/3 satellites of the NOAA series.

f) Data flow diagram

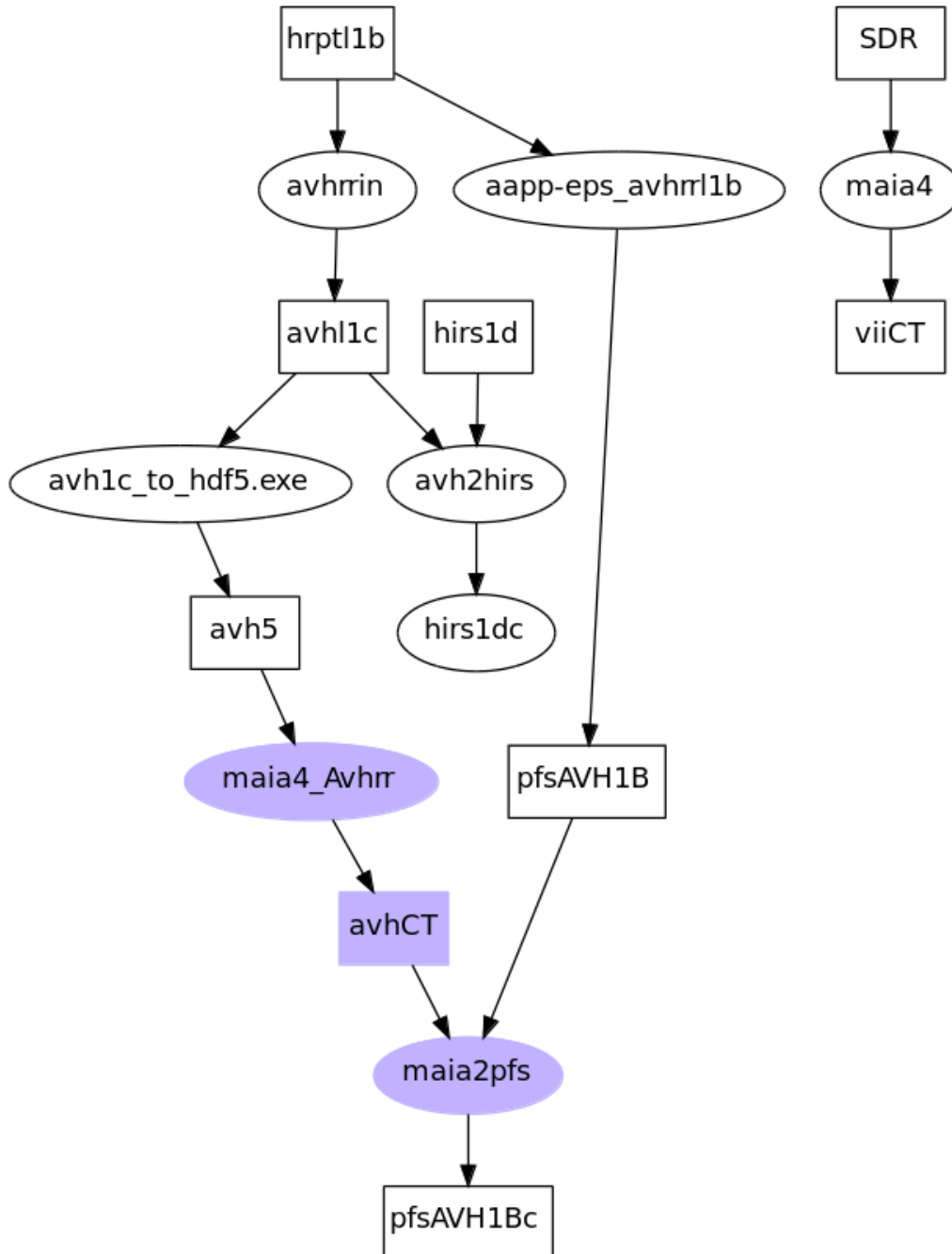


Figure 1: data flow diagram for maia4

avh5: AVHRR level 1c in hdf5 file format

avhCT: MAIA4 file for AVHRR (hdf5 file format)

pfsAVH1Bc: PFS AVHRR 1B with updated CLOUD_INFORMATION

viiCT: MAIA4 file for VIIRS (hdf5 file format)

hirs1dc : updated hirs1d

3. IMAGER CLUSTERS GENERATION

In order to follow the ITSC-XX recommendation DA/NWP-17 (“Use the AVHRR cluster algorithm available in AAPP for all hyperspectral sounders.”), the relevant code will be extracted from the OPS software to create an independent tool which will allow the cluster generation with the “nuées dynamiques” method, for both VIIRS/CRIS and AVHRR/IASI sensors.

The **clustergen routine** will work for each field of view of the hyper-spectral sounder:

input:

- radiances/reflectance/brightness temperatures arrays for every channels
- sounder field of view print / sounder weights

output:

- For each imager channel :
 - Number of clusters
 - mean and standard deviation (RMS)
 - cluster mapping

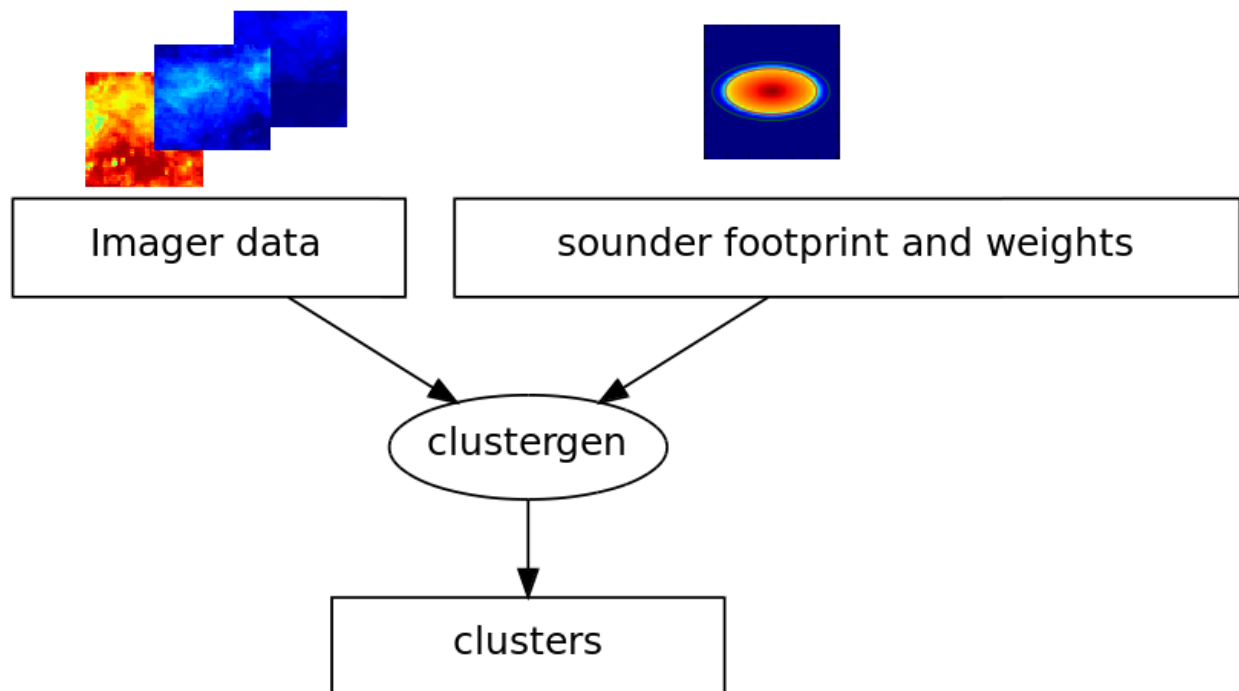


Figure 2: data flow diagram for imager clusters generation

4. ECCODES INTERFACES FOR BUFR ENCODE/DECODE

BUFR facilities were introduced into AAPP with v6.1 (released 2006), using ECMWF’s BUFRDC² library. The top-level scripts in AAPP are *aapp_encodebufr_1c* and *aapp_decodebufr_1c*; these convert between AAPP “level 1c” (.11c) files and BUFR. In some cases, level 1d (.11d) is also supported. For details, see the AAPP User Guide³.

For AAPP v8, the plan is to support both BUFRDC and ecCodes⁴, to allow users to gradually migrate.

There are a number of advantages to migration to ecCodes:

- Long-term support by ECMWF (BUFRDC likely to be phased out eventually)
- Potential for the calling code to be more intuitive and more readable
- Reduces the number of external libraries (same library used for BUFR and GRIB)

There are some significant differences between ecCodes and BUFRDC, as indicated in Table 1.

Table 1: Some differences between BUFRDC and ecCodes


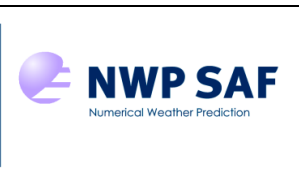
Characteristic	BUFRDC	ecCodes
Language	Mostly FORTRAN77. Some C for I/O	Mostly C. With C, Fortran90 and python interfaces.
Build process	Shell scripts <i>build_library</i> and <i>install</i>	<i>cmake</i> → <i>make</i> → <i>make install</i> Separate build directory
BUFR tables	Approx 7000 BUFR table files all in one directory \$BUFR_TABLES. The file name defines the table version, e.g. B00000000000254019001.TXT	Different versions are in different directories. Each directory contains 3 files: codetables , element.table , sequence.def
Descriptors	Referred to by number, e.g. 012163	Referred to by mnemonic, e.g. <i>brightnessTemperature</i>
Encoding strategy	Fill up a big array with all the data to encode (have to get the order correct), then, for each message, make a call to BUFRN	Each descriptor is treated separately. Do not need to know the order, e.g. <code>call codes_set(ibufr, 'latitude', rvalues)</code> Then at the end we create the message using: <code>call codes_set(ibufr, 'pack', 1)</code>

The different encoding strategy means that AAPP cannot simply loop over input records, filling up the “values” array. Instead, data for all the scans that are to be encoded into a BUFR message need to be held in memory, in order to call the ecCodes interface (*codes_set*) for each descriptor. This necessitates a re-write of all the encode/decode routines.

² <https://software.ecmwf.int/wiki/display/BUFR/BUFRDC+Home>

³ https://nwpsaf.eu/site/software/aapp/documentation/userguide/#BUFR_conversion_tools

⁴ <https://software.ecmwf.int/wiki/display/ECC/ecCodes+Home>

		AAPP v8 Top-level Design	Doc ID :NWPSAF- MO-DS-034 Version : 1.0 Date : 06/09/2017
--	--	---------------------------------	---

The new routines will be held in separate directories from the old:

AAPP/src/tools_eccodes/bin


```
eccodes_encodebufr_1c.ksh
eccodes_encodebufr_1c.F
eccodes_decodebufr_1c.ksh
eccodes_decodebufr_1c.F
```

AAPP/src/tools_eccodes/libecbufr

```
eccodes_get_1c_amsua.F
eccodes_get_1c_amsub.F
eccodes_get_1c_atms.F
eccodes_get_1c_cris.F
eccodes_get_1c_hirs.F
eccodes_get_1c_iasi.F
eccodes_get_1c_pciasi.F
eccodes_get_1c_iras.F
eccodes_get_1c_mwhs2.F
eccodes_get_1c_mwhs.F
eccodes_get_1c_mwts2.F
eccodes_get_1c_mwri.F
eccodes_put_1c_amsua.F
eccodes_put_1c_amsub.F
eccodes_put_1c_atms.F
eccodes_put_1c_cris.F
eccodes_put_1c_hirs.F
eccodes_put_1c_iasi.F
eccodes_put_1c_pciasi.F
eccodes_put_1c_iras.F
eccodes_put_1c_mwhs2.F
eccodes_put_1c_mwhs.F
eccodes_put_1c_mwts2.F
eccodes_put_1c_mwri.F
```

The Fortran routines will use Fortran90 constructs, but will have fixed-formatting in order to interface with the existing AAPP “include” files.

For the user, the use of ecCodes in AAPP will be optional: the user will specify at the *configure* step whether or not AAPP is be linked to the ecCodes dynamic libraries. Each routine will make use of pre-processor directives (e.g. `#ifdef HAS_LIBECCODES`) to ensure that compilation will not fail if the user has chosen not to link ecCodes.

The EUMETSAT Network of Satellite Application Facilities	 NWP SAF Numerical Weather Prediction	AAPP v8 Top-level Design	Doc ID :NWPSAF- MO-DS-034 Version : 1.0 Date : 06/09/2017
---	--	--------------------------	---

5. OTHER CHANGES INTRODUCED SINCE RELEASE OF AAPP V7.1

The top-level design document for AAPP v7 (RD-9) reflects the capabilities of AAPP v7.1. This section summarises the top-level changes that have been made, via update releases, since that release was issued. These functions are not planned to be changed in the initial release of AAPP v8, but are listed for completeness.

- Ingest of TOVS 1b data from NOAA CLASS archive. Top-level script **noaa_class_to_aapp.ksh**. Handles Tiros-N to NOAA-14; MSU, HIRS/2 and AVHRR.
- Conversion of AVHRR I1b data to NOAA format (16-bit). Top-level script **avhrr_aapp_to_class.ksh**.
- Ingest for sensors on the FY-3 satellite series: MWHS, MWHS-2, MWTS, MWTS-2, IRAS, MWRI. Top-level scripts **mwhs_sdr.ksh**, **mwts_sdr.ksh**, etc.
- FY-3 re-mapping tools: **mwts2_to_mwhs2.ksh**, **mwhs2_to_mwts2.ksh**, **mwhs2_to_iras.ksh**.
- Conversion of level 1c datasets to hdf5. All AAPP .l1c files can be converted to hdf5 format, using a script **convert_to_hdf5.ksh**. The structure of the hdf5 files closely resembles that of the original .l1c files.
- Tools for handling Multi-Mission-Administration-Messages, for Metop satellites (**mmam-main.exe**, **print-mmam-obt-utc.pl**, **patch-level0-from-mmam.exe**).
- A tool to convert CrIS full-spectral-resolution (FSR) I1c files to normal-spectral-resolution (NSR): **cris_degrade_fsr.ksh**.