# ADM-Aeolus Mission
# Guide on applying L2B processed winds

Jos de Kloe
Ad Stoffelen
KNMI
Reference code: AE-SAF-KNMI_L2BP-003
Version: 1.1
Issue date: 20080520

# Contents

# Change log

| Version | Date | Comment |
|---|---|---|
| 1.0 | 05-Sep-2007 | first preliminary version, including internal comments from the L2B team. |
| 1.1 | 20-May-2008 | adapted to ESA comments dd. 21-Apr-2008 |

# 1 Introduction

A code base is being developed and maintained providing a unique and version-controlled L2B processing (L2Bp). From the code base a portable standalone L2B processor, pL2Bp, is being developed and put available. The output of this processor is well tested and the development team provides support in achieving the same outputs at the user site. A user manual and release note exist, and other documentation similar to that provided in the EUMETSAT NWP SAF.

The core L2Bp subroutine, called sL2Bp here, may be extracted and implemented in the user's integrated forecasting system (IFS). This document describes a few scenarios of how to do this and provides an estimate of the amount of interface development work needed. The user implementation will depend on the operational configuration at the user site. The development team will provide limited support for such implementations. However, a set of tests is provided with the pL2Bp that can be used for reference at the user's site after subsequent implementation of the pL2Bp.

A decision tree on the implementation scenarios provided in this guide is provided in figure 3. Based on the L2Bp development five NWP application scenarios are described:

- A: Use of externally provided L2B winds;
- B: Use of local pL2Bp output;
- C: Use of integrated sL2Bp, minimal number of interfaces ported;
- D: Use of integrated sL2Bp, L1B, L2B, AMD interfaces ported;
- E: Use of integrated sL2Bp, all interfaces ported.

The top two implementations provide little user work to implement, verify and maintain. The latter three implementations provide examples of sL2Bp, but with different user development cost. This is further elaborated in the five sections below, but first some general requirements are specified.

# 2 General requirements on observation operator

The L2Bp output is the L2B data file (for the pL2Bp) or datastructure in memory (for the sL2Bp), containing the Aeolus wind profiles, which is designed to be largely independent of NWP inputs. The L2Bp is using a reference NWP temperature and pressure profile, since the spectral shape of the molecular return depends on temperature T, and to a lesser extend on pressure p. This dependency is accounted for in the L2B output since the T and p references are given as well as the sensitivities of the retrieved winds to the local T and p, following a local linear Taylor expansion. The significance of these sensitivities is described in [RD15] and, if the user decides that these sensitivities are relevant, they may be represented in the observation operator of any data assimilation system in a straightforward way.

NWP profiles in the L2Bp are further used in the optical properties code that maybe used for the optical classification of the atmosphere. This classification discriminates scenes that potentially affect the quality of the computed wind profiles detrimentally. The wind error properties in each class are being determined and will be reported in the L2B output. Guidance will be provided to the user of how to combine, weigh and use the winds in the different classes. We do not expect and will verify the absence of a significant correlation between the reference NWP data used in the classification and the computed Aeolus winds, as part of the ongoing ADM L2Bp development.

Note that we use the terminology which is used for the ECMWF implementation (IFS, ODB, feedback file, etc.) since we are most familiar with that. Other users may have different names for systems with similar functionality. The user needs to port the L2Bp L2B reading routine or the in memory L2B datastructure to his Integrated Forcasting System (IFS), write interfaces to an internal operational data base (ODB), if present, present the data to the Aeolus observation operator routine, define data structures to capture the analysis process and a routine to export these. These activities will be needed for all implementation scenarios since it presents the core of the Aeolus wind exploitation in a Data Assimilation System (DAS). As such, the L2B output data needs to be understood. The most general observation operator needs NWP model wind, temperature and pressure as input, although the dependency on the latter two is very weak.

# 3 User decision tree

If the user decides to generate L2B data locally, using an sL2Bp setup, we recommend the additional installation of the pL2Bp in order to verify the performance of the software in the user configuration. This applies for all but the first two scenarios described below.

Which of the below presented scenarios is most suitable for a user depends on the requirements imposed by his local IFS. Arguments to consider are:

- is local processing needed, or is processing of externally delivered L2Bp winds acceptable (which will have been generated using T and p profiles from another forecasting model, and which will have some time delay compared to delivery of L1B product files)? If externally processed winds are acceptable we need to think about a dessimination scheme for the semi-real time product.
- is it possible to chain executables? If this is the case, use of the pL2Bp is the easiest option.
- is it allowed for a routine within the IFS to do its own filehandling (reading AND writing)? If so only a minimal effort is needed to build custom interfaces.
- is it allowed for a routine within the IFS to interface with a custom datastructure allocated, initialised, and filled with data in a setup routine outside the IFS main loop? Is it also allowed that this routine stores its result in a custom datastructure, that will be exported to file after the IFS main loop has ended? Also in this case only a minimal effort is needed to build custom interfaces.
- is the IFS, and the sL2Bp subroutine within it, required to do all its interfacing using the ODB system? In this case a large effort is needed to build (and maintain!) all required interfaces.

By following figure 3 while answering the above questions, the user is guided to a scenario most suitable for him.

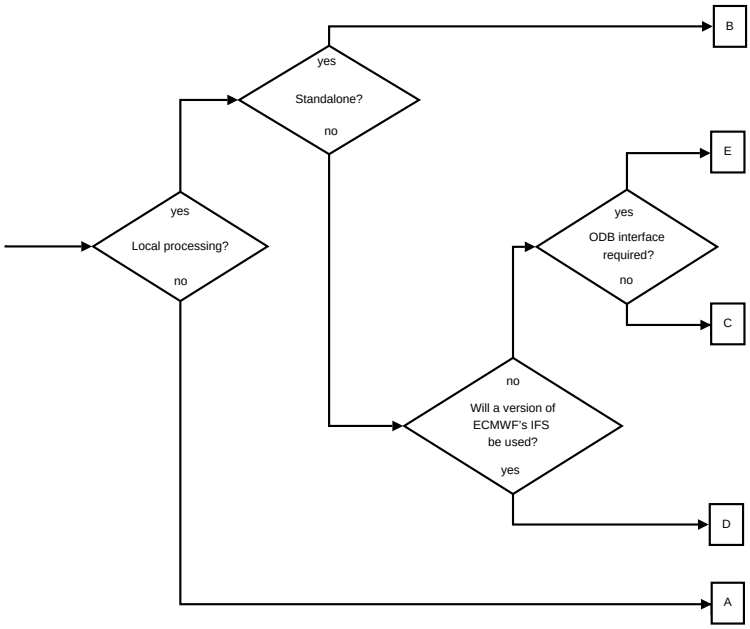


Figure 1: *Flow-diagram to illustrate the different possible configurations of using and/or generating the L2B wind product*

# 4 Dataflows

The content of the L1B (or L2B for scenario A) input file (in BUFR or EE format) is variable obviously. Therefore this file needs to be added to the user's system which handles incoming external datafiles. The stability of the other (auxiliary) input files is not yet precisely known.

Calibration scenarios and atmospheric interaction characteristics will be refined in due time. The current set of auxiliary input files will probably be maintained, which content may vary weekly and for which a timely non-critical update mechanism needs to be established.

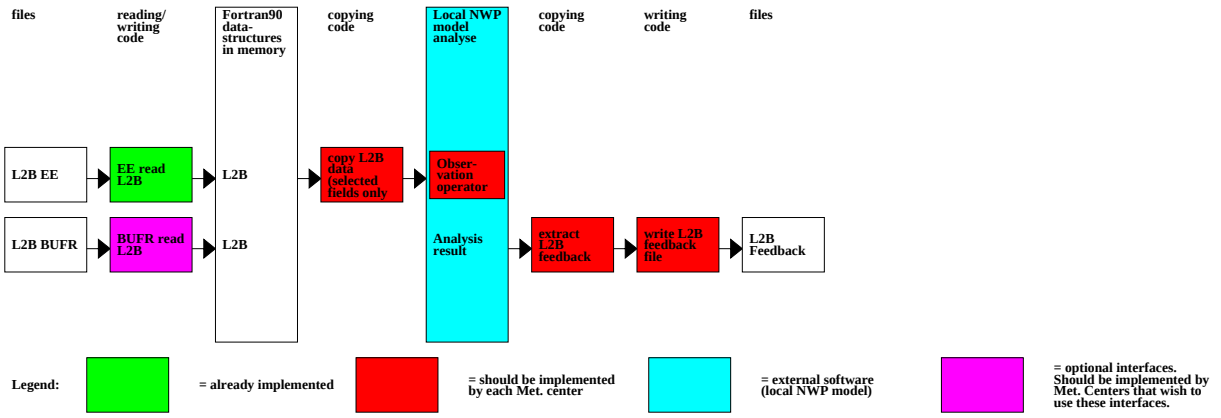


Figure 2: *Scenario A: Dataflow and program blocks for use of externally produced L2B datafiles.*

# 5 Implementation scenarios

## 5.1 A: External L2B winds

No knowledge of the L2Bp and its interfaces is required, except for interfacing to the available reading code of the L2B product file. An observation operator needs to be implemented, and a feedback mechanism is needed to assess the impact of this data in the assimilation.

It is currently foreseen that the ECMWF generates L2B data twice daily in their main runs. Unfortunately, these data will not be sufficiently timely for most NWP applications. No concrete project to centrally generate timely L2B data yet exists.

Required effort to implement this scenario (see figure 2 for a schematic representation):

1. interface the available L2Bp reading code to the user's IFS/ODB and implement compilation of the L2Bp reading code in the IFS make system. Note that a selection of the fields in this file is probably sufficient.

2. implement an observation operator for this datatype.

3. implement output of the analysis results for this datatype to a feedback file for debugging and monitoring purposes.

4. setup the incoming dataflow for the ADM Aeolus data.

5. optionally, if the input file is required to be in BUFR format, the user should implement a reading routine for this filetype[1] .

[1]Note that within the sL2Bp development software is available to read the L1B BUFR file, export it as a large list of real values to the ODB, import it from ODB, and copy the ODB results to the Fortran90 datastructures needed for further processing. Linking the 2 pieces of code together using a minimal dummy ODB interface would yield the required L1B-BUFR-to-Fortran90-Datastructures interface. Similar code will be developed for the L2B/L2C data.

## 5.2 B: Local L2B winds

Some knowledge of the L2Bp is required, in particular its input files. Reference input and output is provided with the pL2Bp, such that the compilation and installation may be verified at the users site.

An Auxiliary Meteorological Data (AMD) file needs to be produced for every L1B input file. Calibration scenarios (Auxiliary Calibration data) and atmospheric interaction characteristics (Auxiliary Climatological data) will be refined in due time. The current set of auxiliary input files will probably be maintained, but its content may vary weekly.

Required effort to implement this scenario (see figure 3 for a schematic representation):

1. all steps needed for scenario A mentioned above
2. implement an interface to export the L1B geolocation data read by the available L1B reading code, to the users ODB
3. implement an interface from the users ODB to the available writing code for the AMD file. This should export the T and p profiles from a short range forecast or first guess on the ADM Aeolus measurement locations, to enable the pL2Bp to use it.
4. compile and install the pL2Bp to the users system
5. Note that BUFR reading and writing routines for the L2B Product file are not being prepared within the current project. A user interested in using this fileformat is expected to implement it himself. See also footnote [1] on page 8

## 5.3 C: Minimal sL2Bp porting

Some knowledge of the L2Bp and knowledge of its main interface is required. Reference input and output is provided with the pL2Bp, such that the implementation of pL2Bp may be verified at the users site by generating a L2B output file with both pL2Bp and sL2Bp.

Note that this scenario assumes it is allowed to set up custom datastructures outside the users ODB, to be used by the sL2Bp routine within the IFS, or alternatively it assumes that this sL2Bp routine is allowed to open auxiliary datafiles for reading and writing.

Required effort to implement this scenario (see figure 4 for a schematic representation):

1. all steps needed for scenario B mentioned above (note that for testing/debugging purposes it is essential to have a version of the pL2Bp running on the same platform on which the sL2Bp is to be installed. So step 4 in scenario B is also needed here)
2. an AMD data set needs to be produced for every L1B input file inside the IFS and be presented to the sL2Bp core subroutine in the form of a specific Fortran90 datastructure. Note, however, that the possibility to export this AMD data to an EE formatted file is still required, because this is the only way to reproduce a problematic sL2Bp processing case with the pL2Bp. Therefore step 3 in scenario B is also needed here.
3. this scenario assumes preloading all L1B data in a custom datastructure is allowed in the users system, so the provided L1B reading code can be used directly.

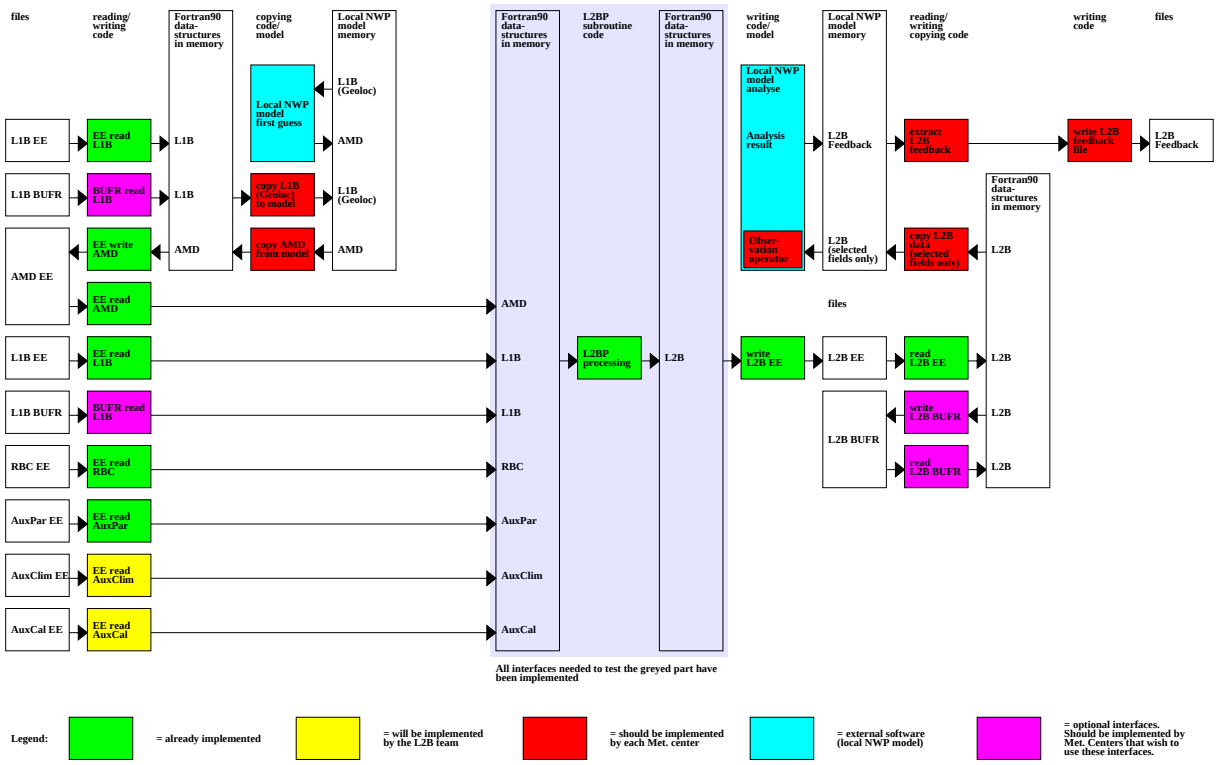


Figure 3: *Scenario B: Dataflow and program blocks for the L2BP standalone version (combined with for example Hirlam).*

4. implement an interface to copy the L2B processing results, available in a custom Fortran90 datastructure, to the users ODB. Note that this is essentially the same work as step 1 in scenario A above.

## 5.4 D: Partial sL2Bp porting

This scenario is the one currently being implemented at ECMWF. Sample code for this scenario will be available to all ECMWF members in the ECMWF IFS repository. It is basically identical to scenario C defined above, except for the L1B data handling.

Required effort to implement this scenario (see figure 5 for a schematic representation):

1. all steps needed for scenario C mentioned above
2. At ECMFW all L1B data will be copied to and from the ODB, so 2 interfaces need to be implemented, one from the L1B Fortran90 datastructure to ODB and one going back again. (this replaces step 3 in scenario C defined above)
3. on the output side the full L2B product is written to the ODB first. This is needed to allow parallel processing of several parts (BRC's) of the L2B input product, and it prevents the need to implement communication mechanisms inside the IFS main loop to collect all data before a complete output file or Fortran90 datastructure can

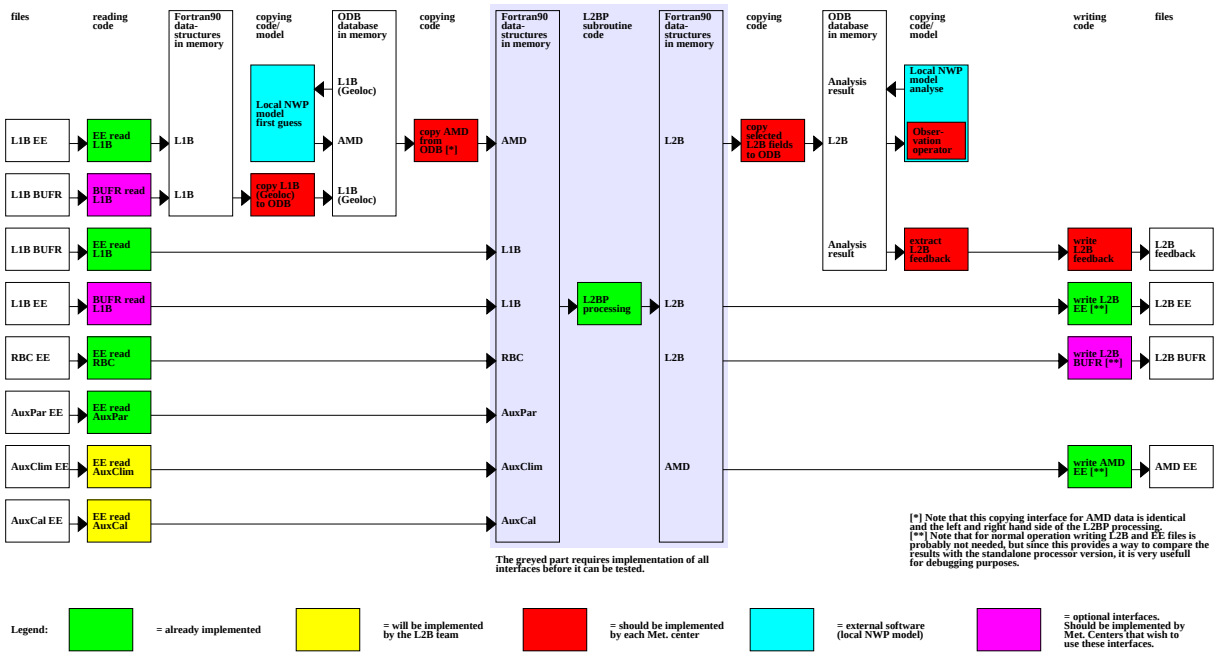


Figure 4: *Scenario C: Dataflow and program blocks for the L2BP subroutine version (minimal interface).*

be constructed. In a next post-processing step all this data is collected from the ODB and written to file. Therefore 2 interfaces need to be implemented on the output side as well, one from the L2B Fortran90 datastructure (at BRC level) to ODB and one going back again.

## 5.5 E: Full sL2Bp porting

Some knowledge of the L2Bp and full knowledge of its interfaces is required. Reference input and output is provided with the pL2Bp, such that the implementation of pL2Bp may be verified at the user's site by generating a L2B output file with both pL2Bp and sL2Bp.

This scenario assumes internal IFS subroutines are not allowed to do file handling AND are not allowed to import datastructures that should be setup/filled with data outside the IFS main loop.

An AMD dataset needs to be produced for every L1B input file inside the IFS and be presented to the sL2Bp core subroutine.

Also all additional auxiliary input datafiles need to be passed through the ODB to the sL2Bp in this scenario.

Required effort to implement this scenario (see figure 6 for a schematic representation):

1. all steps needed for scenario D mentioned above

2. implement an interface to copy the auxiliary RBC data from the custom Fortran90 datastructure to ODB and back again.

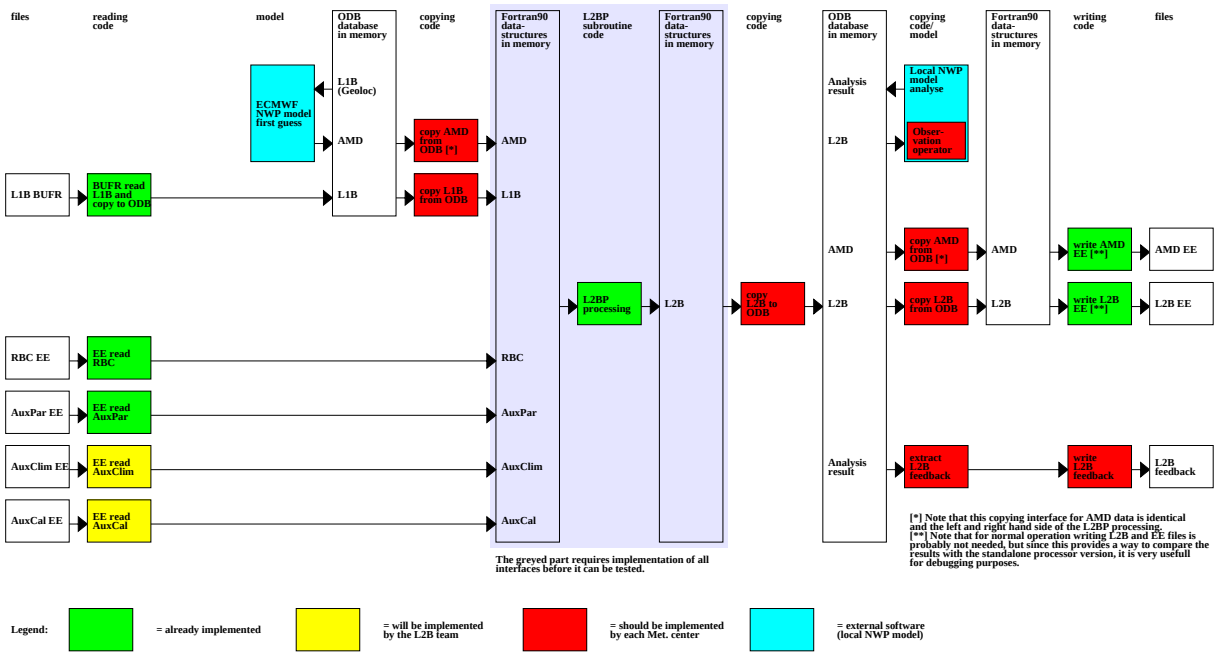


Figure 5: *Scenario D: Dataflow and program blocks for the L2BP subroutine version (as will be used at ECMWF).*

3. implement an interface to copy the auxiliary Parameter data from the custom Fortran90 datastructure to ODB and back again.

4. implement an interface to copy the auxiliary Climatological data from the custom Fortran90 datastructure to ODB and back again.

5. implement an interface to copy the auxiliary Calibration data from the custom Fortran90 datastructure to ODB and back again.

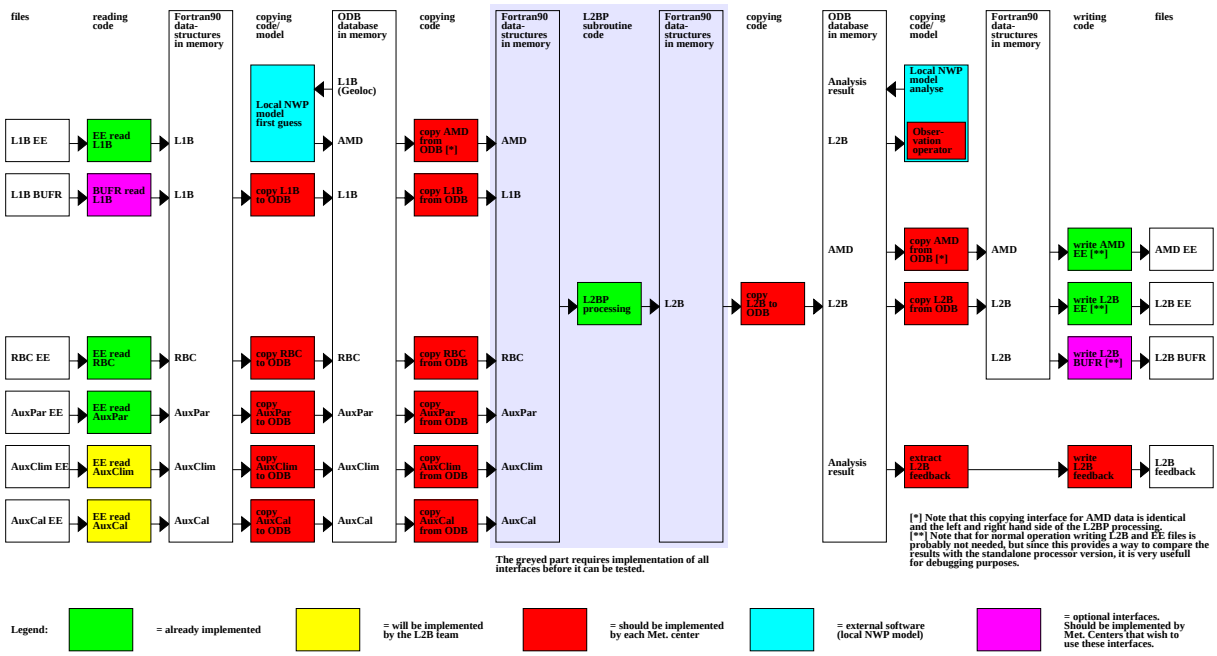


Figure 6: *Scenario E: Dataflow and program blocks for the L2BP subroutine version (all interfaces ported).*

# 6 Required effort

In order to estimate the effort needed to implement the different scenarios it is assumed that implementation of an interface requires an amount of work proportional to the amount of pages needed to describe the associated fileformat in the documentation. It is also assumed that the programmer doing the work is very familiar with the Fortran90 language and the user's IFS system, but has no advance knowledge at all on the L2Bp codes.

About 2 days of work have been estimated for the coding, implementing and testing of 5 pages of IODD definitions in Earth Explorer format. This estimate may be wrong by a factor of 2 maybe, but it should give at least a realistic order of magnitude. The thus calculated estimates are given in table 1.

Note that the BUFR description documentation generally is more compact, which clearly follows from the L2B product definition for which both formats should describe all fields of the product. So for BUFR 2 days of work have been estimated for each 2.5 pages of IODD.

Note also that the L1B BUFR product is a subset only of the L1B EE product, so the amount of elements and the required effort to code an interface might be smaller. However, looking at the number of IODD pages the difference will not be large (but this depends on the familiarity of the developer who implements it with the BUFR encoding/decoding software).

The content of a Feedback file is to be defined by the user, so no estimate can be given for that interface. Example code that will be available in the ECMWF IFS repository. Also estimates for the amount of work needed to implement an observation operator and to set up the dataflows in a user's IFS system are hard to estimate, and are not included in table 1.

Table 1: details of needed user interface development effort.

| **implementation** | Pages of IODD documentation | documentation | needed for scenario | **estimated work** |
|---|---|---|---|---|
| L2B product (EE) | 24 | L2BP IODD, sec. 4 [RD2] | A, B, C, D, E | 10 days |
| L2B product (BUFR) | 6+6 | BUFR Description, sec. 5+6 [RD5] | A, B, C, D, E | (10 days)* |
| L1B Selected Geolocation fields | 2 | L1B IODD, sec. 5.3.1 [RD1] | B, C | 1 day |
| L1B product (EE) | 5+30 | L1B IODD, sec. 5.3.1+sec.6 [RD1] | D, E | 14 days |
| L1B product (BUFR) | 12+6 | Bufr Description, sec. 3+4+6 [RD5] | D, E | (14 days)* |
| Aux. RBC file | 5 | RBC IODD, sec.4 [RD4] | E | 2 days |
| AMD file | 4 | L2BP IODD, sec. 5.1 [RD2] | B, C, D, E | 2 days |
| Aux. Par. file | 9 | L2BP IODD, sec. 5.4 [RD2] | E | 4 days |
| Aux. Clm. file | 4 | L2BP IODD, sec. 5.3 [RD2] | E | 2 days |
| Aux. Cal. file | 5 | L2AP IODD, sec. 4.3 [RD3] | E | 2 days |
| L2B feedback file | n.a. | the fields and formatting of this file are to be defined by the user | A, B, C, D, E | not estimated |

* note that the BUFR handling for the L2B product has not yet been implemented, and for the L1B file it is only partially implemented, so apart from interfacing, also the routines themselves need to be developed by the user. (tbc) This will increase the time needed for implementation by another 10 days or so.

# 7 Documentation

Because of the coded naming convention of the filenames of the documentation, and the fact that a number of documents have been produced according to the ECSS guidelines, with which the user (familiar with SAF documentation) might not be familiar, in this section a guide to the available documentation is given.

The following documentation in NWP-SAF style is available:

- product specification, see [RD6]
- top level design, see [RD7]

These documents define the several file types used by the processing:

- L1B IODD, see [RD1]
- L2A IODD, see [RD3]
- L2B IODD, see [RD2]
- RBC IODD, see [RD4]
- BUFR template definition, see [RD5]

These documents define the installation and interfaces of the L2BP software:

- software release notes, see [RD11]
- software user manual, see [RD13]
- input control file (JobOrder file), see [RD8]
- interfaces to all Fortran90 code needed to implement the sL2Bp inside an IFS like system, see [RD12]

These documents describe the design of the L2BP software:

- definition of all algorithms used by the software, see [RD14]
- design document, see [RD15]

These documents describe several plotting and other tools:

- Matlab reading software to read the several EE formatted product files, see [RD9]
- conversion software to convert EE formatted L1B product files into BUFR format, see [RD10]

# 8 Summary

Table 2 gives a summary of the effort needed to implement the different scenarios, based on the needed interfaces as detailed in sections 5.1 upto 5.5, and the numbers in table 1.

Table 2: overview of user development effort.

| **Implementation scenario** | **Needed work to implement the scenario [Days]** |
|---|---|
| A: External L2B* | 10 |
| B: local pL2Bp | 10+1+2 = 13 |
| C: Minimal sL2Bp | 10+1+2 = 13 |
| D: Partial sL2BP | 10+14+2 = 26 |
| E: Full sL2Bp | 10+14+2+2+4+2+2 = 36 |

* Note that the timely availability of external L2B winds for most NWP applications is not yet arranged.

The amount of maintenance will be proportional to the amount of code written by the user, so proportional to the amount of work stated above for implementation.

The following has been presented

- Data assimilation inputs are essentially L2B wind profiles, but L1B data may be implemented at user convenience.
- A stand-alone portable L2B processor is being developed including User Manual and Release Note to install, configure, test and run the L2Bp.
- The L2Bp is designed to provide largely NWP-model independent output. Auxillary Meteorolgical Data profiles can be replaced by the user in the L2Bp, but no substantial L2B output change is expected. A verification test will be provided for the user to test the AMD replacement.
- For L1B input, the core L2B subroutine and its inputs need to be ported and interfaced to the user system and subsequently tested.
- Standalone pL2Bp tests should be used as reference for testing the sL2Bp; guidance will be given for the user of the subroutine version, but no guarantees of correct implementation.
- Observation operator is expected identical for stand-alone and subroutine versions, i.e., HLOS profiles with basic weak T and p sensitivities for Rayleigh winds.

The L2Bp team solicits and appreciates feedback/suggestions on this development.

# 9 Reference documents

The following list gives the reference documents. All are available from the ESA eRoom.

[RD1] ADM-IC-52-1666_L1bP-IODD_Iss_3.3_070412.pdf

[RD2] AE-IF-ECMWF-L2BP-001_20070223_IODD_Iss1.3.pdf

[RD3] AE-IF-DLR-L2A-004_IODD_Iss_1.4_070312.pdf

[RD4] AE-TN-MFG-GS-0003-v1.1_20070312_RBCIODD.pdf

[RD5] AE-TN-ECMWF-L2P-0072_WMO-FM94-BUFR-description_Iss_0.3_060110.pdf

[RD6] AE-SAFPS-KNMI-L2BP-001

[RD7] AE-SAFTLD-KNMI-L2BP-001

[RD8] AE-IF-ECMWF-L2BP-002_20070223_ExtICD_Iss1.1.pdf

[RD9] MFG_readnplot_adm_aeolus_data_with_matlab.pdf

[RD10] AE-TN-ECMWF-L2BP-0072_20070223_EE2BUFRGuide_Iss1.1.pdf

[RD11] AE-RN-ECMWF-L2BP-001_20070622_SRN_Iss1.31.pdf

[RD12] MFG_L2BP_fortran_structures.doc

[RD13] AE-MA-ECMWF-L2BP-001_20070223_SUM_Iss1.3.pdf

[RD14] AE_TN_ECMWF_L2BP_0023_20070223_ATBD_V2.1.pdf

[RD15] AE-DD-ECMWF-L2BP-001_20070223_DD_Iss1.0.pdf

[RD15] "Impact of Line Shape on Aeolus-ADM Doppler Estimates", final report, nov. 2005, by Pierre H. Flamant et al.

# 10 Acronyms

AMD ..... Auxiliary Meteorological Data
ADM ..... Atmospheric Dynamics Mission
BUFR ..... Binary Universal Form for the Representation on meteorological data
DAS ..... Data Assimilation System
ECMWF ..... European Centre for Medium-Range Weather Forecasts
ECSS ..... European Cooperation for Space Standardization
EE ..... Earth Explorer (file format type)
EUMETSAT ..... European Organisation for the Exploitation of Meteorological Satellites
HiRLAM ..... High Resolution Local Area Model
IFS ..... Integrated Forecasting System
IODD ..... Input Output Definition Document
KNMI ..... Koninklijk Nederlands Meteorologisch Instituut
L2B ..... Level 2B
L2Bp ..... L2B processor
NWP ..... Numerical Weather Prediction
ODB ..... Operational Data Base
pL2Bp ..... portable L2Bp
RBC ..... Rayleigh Brillouin Correction
SAF ..... Satellite Application Facility
sL2Bp ..... subroutine L2Bp