



Project
EPS-SG Local Processors
Re-engineering Processor Prototypes

MWS User Manual
L1b_LP-MA-ISARD-EPS_SG-0209

isardSAT Ref.:	ISARD_EUM_EPS-SG-L1b-MWS_SUM_1251
Version:	3.1
Date:	15/07/2025
Author:	isardSAT Team

	<i>Function</i>	<i>Date</i>	<i>Signature</i>
<i>Verified by</i>	Technical Manager	15/07/2025	
<i>Approved by</i>	Project Manager	15/07/2025	

Copyright and confidentiality

This document is the exclusive property of the company isardSAT and cannot be reproduced, even in partial form, without the written authorization of the companies themselves.

Use is limited to all those who, for various reasons, produce company quality management system documentation and / or environmental management system documentation.

Any changes to this document are subject to the provisions of the SGQ and SGQ Documentation Management Procedure (QPRC001S) in the version in force.

Configuration Control

Title: MWS User Manual
Created by: isardSAT Team
Editing Tool: Microsoft WORD

Document Change Record

Version	Date	Author/Revision	Reason for Change	Changed Sections/Pages
1.0	20/02/2023	isardSAT Team	Draft Issue	-
2.0	05/12/2023	isardSAT Team	Issue for AR1	All
3.0	10/12/2024	isardSAT Team	Version for MWS V2 TRR2#1 Add gtest in the table of external librairies used Replace NetCDF cxx4 by NetCDF C++ Remove ./configure as it is not needed before calling the rpmbuild script. Update the description of MWS L0 product selection. Add new AUX_CNF auxiliary package Update the list of Error Codes and Messages	§5.1 §5.3 §6.2.1 §6.2.2, §6.2.3 Appendix A
3.1	15/07/2025	isardSAT Team	Indicate the need for xml2 and gfortran external libraries. Indicate the need to copy manually the schemas, in case of installation directly from the sources. MWS L1b LP input validation schemas are now relocatable together with the software (fix Redmine-3170). Indicate the schemas are installed with the software when doing the installation from RPM. Describe the schemas in the installed content structure.	§5.1 §5.3.1 §5.3.3 §5.4

Version	Date	Author/Revision	Reason for Change	Changed Sections/Pages
			Provide the description of the different validation schemas.	§6.5 (added)

Distribution List

Internal Distribution Name	External Distribution Name
Maria Jose Escorihuela (isardSAT)	Michela Sunda (EUM)
Stéphanie Urien (isardSAT)	Davide Castellazzi (EUM)
Gorka Moyano (isardSAT)	Luca Galli (EXP)
Mark Pattie (isardSAT)	Alessandra Giustiniani (EXP)
Roger Almasqué (isardSAT)	
Berta Moreno (isardSAT)	

Table of Content

TABLE OF CONTENT	5
LIST OF FIGURES.....	6
LIST OF TABLES.....	6
1. INTRODUCTION	7
1.1. DRD IDENTIFICATION	7
1.2. PURPOSE AND OBJECTIVE	7
2. APPLICABLE AND REFERENCE DOCUMENTS	8
2.1. APPLICABLE DOCUMENTS.....	8
2.2. REFERENCE DOCUMENTS	8
2.3. ACRONYMS	8
3. CONVENTIONS.....	10
4. PURPOSE AND EXTERNAL VIEW OF THE SOFTWARE.....	11
5. INSTALLATION PROCEDURES	12
5.1. HARDWARE AND SOFTWARE CONFIGURATION REQUIREMENTS	12
5.2. LP DELIVERY PRESENTATION	13
5.3. LP PACKAGES INSTALLATION.....	14
5.3.1. <i>Building executable from the source code</i>	<i>14</i>
5.3.2. <i>Building RPM from the source code.....</i>	<i>15</i>
5.3.3. <i>Installation from the RPM.....</i>	<i>16</i>
5.4. INSTALLED CONTENT STRUCTURE.....	17
5.5. UNIT TESTS EXECUTION	17
6. OPERATIONS BASICS	19
6.1. LAUNCHING EXECUTABLE	19
6.2. SELECTION AND CONTROL	19
6.2.1. <i>Input MWS L0 products.....</i>	<i>19</i>
6.2.2. <i>Additional L0 product and Auxiliary files</i>	<i>19</i>
6.2.3. <i>Configuration files.....</i>	<i>20</i>
6.2.4. <i>Job Order/Task Table.....</i>	<i>21</i>
6.3. NORMAL OPERATION.....	21
6.4. ERROR CONDITIONS.....	21
6.5. VALIDATING INPUTS USING SCHEMAS	21
APPENDIX A MWS L1B LP ERROR CODES AND MESSAGES	23

List of Figures

None

List of Tables

Table 1-1. ECSS clauses.	7
Table 5-1. External libraries used by the MWS L1b LP.....	13
Table 5-2. MWS L1b LP delivery files.	13
Table 6-1. MWS L1b LP Additional L0 and Auxiliary files.	20
Table 6-2. MWS L1b Local processor input validation schemas.	22
Table A-1. Warning and error codes list.	23
Table A-2. Return codes list.	24

1. Introduction

This document, the EPS-SG MWS L1b LP User Manual (MWS-UM) as defined in ECSS-E-ST-40C.

1.1. DRD identification

The Software User Manual (SUM) is called from the normative provisions summarized in Table H-1 of ECSS – ECSS-M-ST-40 (06/03/2009).

This DRD is called from ECSS-E-ST-40 following clauses.

Table 1-1. ECSS clauses.

ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.5.2.8	All
	5.6.3.3	All
	5.6.4.3	All

1.2. Purpose and Objective

This document is the Software User Manual (SUM) for the EPS-SG MWS L1b LP. Its purpose is to provide instructions for the users of the software.

2. Applicable and Reference Documents

2.1. Applicable Documents

Document Name	Document Identifier	Internal Reference
Statement of Work for re-engineering EPS-SG Local Mission L1 Processor	EUM/LEO-EPSSG/SOW/21/1225668 V1, 26 October 2021	[SOW]
EPS-SG Local Mission L1 Processor Technical Requirements	EUM/LEO-EPSSG/REQ/21/1227040 v1 e-signed, 30 September 2021	[L1-AD]
EPS-SG Local Processor Project Management Plan	L1b_LP-PL-XPR-EPS_SG-0101	[PMP]
Computer Hardware, Operating System and COTS Products Reference Guide	EUM/OPS/TEN/07/2373	[COTS]
EPS-SG MWS Level 1B Auxiliary Data Specification (ADS)	EUM/LEO-EPSSG/SPE/14/777588	[MWS-L1B-ADS]
EPS-SG MWS L1b Local Processor Re-engineering and Development Plan	L1b_LP-PL-ISARD-EPS_SG-0201	[MWS-RDP]
EPS-SG MWS L1b Local Processor Interface Control Document	L1b_LP-ID-ISARD-EPS_SG-0206	[MWS-ICD]
EPS-SG MWS L1b Local Processor Software Design Document	L1b_LP-ID-ISARD-EPS_SG-0207	[MWS-SDD]
EPS-SG MWS L1b LP Software package	L1b_LP-SW-ISARD-EPS_SG-0211	[MWS-SW]
EPS-SG MWS L1b LP Test Data package	L1b_LP-SW-ISARD-EPS_SG-0212	[MWS-TD]

2.2. Reference Documents

None

2.3. Acronyms

Terms, Definitions and Abbreviated Terms

APT	Anomaly Processing Tool
AR	Acceptance Review
ARB	Anomalies Review Board
BPMN	Business Process Model Notation
BRC	Basic Repeat Cycle
CEFR	Common European Framework of Reference for Languages
DBA	Direct Broadcast Acquisition
CFI	Customer Furnished Item
COTS	Commercial Off-the-Shelf
DFDA	Digital Factory Aerospace & Defence
EO	Earth Observation

EPS-SG	EUMETSAT Polar System-Second Generation
FOV	Field-Of-View
HW	HardWare
ICI	Ice Cloud Imaging
ICD	Interface Control Document
IPR	Intellectual Property Right
KOM	Kick-off Meeting
L1	Level 1
LP	Local Processor
MWI	Microwave Imaging for Precipitation
MWS	Microwave Sounding
NetCDF	Network Common Data Form
NAVATT	NAVigation and ATTitude
NRB	Non-Conformance Review Board
NRT	Near Real Time
N/A	Not applicable
OS	Operating System
OSS	Open Source Software
PDGS	Payload Data Ground Segment
PFS	Product Format Specification
PGF	Product Generation Function
PGS	Product Generation Specification
PS	Processing Specification
QA	Quality Assurance
RCP	Requirement Check Point
REQ	REquirement
SAF	Satellite Application Facility
SCA	Scatterometer
SOW	Statement of Work
SW	SoftWare
TBC	To Be Confirmed
TBD	To Be Defined
TCE	Technical Computing Environment
TDS	Test Data Set
TN	Technical Note
TRB	Test Review Board
TRR	Test Readiness Review
VCD	Verification Control Document
VII	Visible-Infrared Imaging
WBS	Work Breakdown Structure
WP	Work Package
XPR	Exprivia S.p.A.

3. Conventions

The commands are indicated in *Italic* and framed.

mkdir \$MWS_TMP_DIR

The text to be replaced is surrounded by <>.

<version_number>

4. Purpose and External View of the Software

The **MWS L1b LP** SW contains just one main processing chain: the **MWS L1b** processor.

The MWS L0 input products are processed by the **MWS L1b** processor, with the help of the auxiliary files, to generate the corresponding MWS L1b products, which are the main output products. In addition, it generates the auxiliary Context Data files.

The definition of the algorithms of the processing chain is given in [MWS-L1B-PGS] and [INR-SPEC-SATA]. The definition of the input L0 files is given in [L0-PFS], of the auxiliary files in [MWS-L1B-ADS], and of the output L1 files in [MWS-L1B-PFS].

The **MWS L1b LP** SW is designed to be executed via command line or integrated in a Processing Framework based on job order interface. The Processing Framework will be able to invoke the MWS L1b LP processor to generate the output MWS L1b products and the auxiliary Context Data files, starting from the inputs data according to the interface defined in [MWS-ICD]. In addition, the MWS L1b LP processor can be executed standalone by command line, with all the input data (products and auxiliary data compliant with EUMETSAT PFSSs/ADSs) available in a file system.

5. Installation procedures

This section contains the description of the procedures the operator must use to install the EPS-SG MWS L1b LP.

5.1. Hardware and software configuration requirements

The installation and execution of the **MWS L1b LP SW** requires to have the following:

System:

- Hardware as defined in [COTS].
- Memory of 8 GB or higher.

Environment:

- Red Hat Enterprise Linux Server release 9.0 Operating System.
- g++ compiler version v11.3.

External libraries:

The list of external libraries needed in compilation and run-time is provided in Table **5-1**. The external libraries shall be deployed in the OS prior to the installation of MWS L1b LP.

The EO CFI RPM is delivered as an RPM package along with the MWS L1b LP. The other libraries shall be directly deployed using the OS package manager.

Table 5-1. External libraries used by the MWS L1b LP.

SW	Version	Description
EOCFI Library	4.23	The Earth Observation CFI software is a collection of precompiled C++ libraries for timing, coordinate conversions, orbit propagation, satellite pointing calculations, and target visibility calculations
NetCDF	4.3.2	NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data
Xerces	3.2	Xerces-C++ is a validating XML parser written in a portable subset of C++
xml2	2.9.13	xml2 is an XML processing library written in C for use in C/C++ applications
gfortran	11.4.1	Fortran compiler
EIGEN	3.4.0	Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms
spdLog	1.10.0	Very fast, C++ logging library
gtest	1.11.0	C++ unit test framework

5.2. LP delivery presentation

The **MWS L1b LP** software is a self-contained delivery that provides the nominal version of the **MWS L1b LP SW**. It consists of two packages, the software package [MWS-SW] and the Test Data package [MWS-TD]:

Table 5-2. MWS L1b LP delivery files.

SW package	File	Description
L1b_LP-SW-ISARD- EPS_SG-0211- v<x.y.z>	eps-sg-mws-l1-lp_src- v<x.y.z>.tar.gz	MWS L1 LP processor complete source code, including makefiles
	eps-sg-mws-l1-lp_utdata- v<x.y.z>.tar.gz	MWS L1 LP processor unitary test data
	eps-sg-mws-l1-lp-<x.y.z>- <x.y.z>-1.el9.x86_64.rpm	MWS L1 LP processor compiled, ready-to-install RPM executable package for Linux
L1b_LP-TD-ISARD- EPS_SG-0212- v<x.y.z>	eps-sg-mws-l1-lp_tds- v<x.y.z>.tar.gz	MWS L1 LP processor Test Data Sets as used and produced during validation testing

Where x.y.z represents the 3 digits of the SW version number.

5.3. LP packages installation

5.3.1. Building executable from the source code

Starting from the delivered source code compressed package, the user can build the application executable. In order to do so, the user shall follow the next steps:

- Create a local temporary directory to extract and build the application, e.g. "~/my_tmp/mws_inst/", defined by \$MWS_TMP_DIR.

```
mkdir $MWS_TMP_DIR
```

- Uncompress the software package [MWS-SW] into the temporary directory.

```
cd $MWS_TMP_DIR
tar xzf /path/to/L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>.tar.gz
```

- Uncompress the source code folder from the eps-sg-mws-l1-lp_src-v<x.y.z>.tar.gz package.

```
cd L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>
tar xzf eps-sg-mws-l1-lp_src-v<x.y.z>.tar.gz
```

- Build the application indicating the desired installation directory, defined by \$MWS_INST_DIR_V, typically "/opt/mws/eps-sg-mws-l1-lp-<x.y.z>".

```
cd eps-sg-mws-l1-lp-<x.y.z>
export RELCOMPFLAG=true
./configure --prefix=$MWS_INST_DIR_V
make -j4
make install
```

where *x.y.z* is the version number of the package, e.g. 1.1.0.

- Copy the schemas needed for the validation of inputs.

```
cd eps-sg-mws-l1-lp-<x.y.z>
mkdir $MWS_INST_DIR_V/schemas
cp schemas/mws/* $MWS_INST_DIR_V/schemas
```

If no installation folder has been defined *when running the command "./configure" the schemas shall be copied under a folder "/usr/local/schemas"*.

- Activate the version via symbolic links. Note the link will be a generic project name without the version: "/opt/mws/eps-sg-mws-l1-lp", defined by \$MWS_INST_DIR.

```
ln -sfh $MWS_INST_DIR_V $MWS_INST_DIR
```

Three new directories will be created in the installation directory (**bin/**, **lib/**, and **schemas/**), containing the application executables and application libraries, respectively. The contents of the \$MWS_TMP_DIR directory can be removed, if desired.

- Expand the LD_LIBRARY with the path to the NetCDF C++ shared library, e.g., "/netcdf/netcdf-cxx4/build/cxx4/", if this library has not been installed in one of the directories on the system search path:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH: </path/to/>libnetcdf-c++4.so
```

5.3.2. Building RPM from the source code

Starting from the delivered source code compressed package, the user can build the application executable RPM. To do so, the user shall follow the next steps:

- Create a local temporary directory to extract and build the application, e.g. "~/my_tmp/mws_inst/", defined by \$MWS_TMP_DIR.

```
mkdir $MWS_TMP_DIR
```

- Uncompress the software package [MWS-SW] into the temporary directory.

```
cd $MWS_TMP_DIR
tar xzf </path/to/>L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>.tar.gz
```

- Uncompress the source code folder from the eps-sg-mws-l1-lp_src-vx.y.z.tar.gz package.

```
cd L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>
tar xzf eps-sg-mws-l1-lp_src-v<x.y.z>.tar.gz
```

- Build the executable RPM.

```
cd eps-sg-mws-l1-lp_src-v<x.y.z>/rpmbuild
./build_rpm_mws.sh <x.y.z> true
```

The installation directory is defined by the “_Prefix” variable inside the “eps-sg-mws-l1-lp.spec” file available in the directory “rpmbuild”. By default, the “_Prefix” variable is set to “/opt/mws”; the user can modify it before building the RPM.

```
(base) surien@dl20:/data/dl380-data/EPS-SG/eps-sg-mws-l1-lp$ more rpmbuild/eps-sg-mws-l1-lp.spec | grep Prefix
%define _Prefix /opt/mws
%define install_dir %{_Prefix}
Prefix: %{_Prefix}
```

The executable RPM will be generated under the directory “rpmbuild”.

Once the executable RPM is built the installation can be performed from the RPM following the instruction from 5.3.3.

5.3.3. Installation from the RPM

The user can install the RPM contents into the system, either the RPM in the delivery package or the one generated by the user from the source code, by running the following:

- Create a local temporary directory to extract and build the application, e.g. “~/my_tmp/mws_inst/”, defined by \$MWS_TMP_DIR.

```
mkdir $MWS_TMP_DIR
```

- Uncompress the software package [MWS-SW] into the temporary directory.

```
cd $MWS_TMP_DIR
tar xzf /path/toL1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>.tar.gz
```

- Install the executable RPM package.

```
cd L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>
sudo rpm -i eps-sg-mws-l1-lp-<x.y.z>-<x.y.z>-1.el9.x86_64.rpm
```

The previous command installs the **MWS L1b LP** executables, libraries and schemas in the following path:

```
/opt/mws/eps-sg-mws-l1-lp-<x.y.z>
```

Where x.y.z is the version number of the package, e.g. 2.2.1.

This new directory is automatically linked to the generic directory:

```
/opt/mws/eps-sg-mws-l1-lp
```

- Expand the LD_LIBRARY with the path to the NetCDF C++ shared library, if this library has not been installed in one of the directories on the system search path:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH: </path/to/>libnetcdf-c++4.so
```

5.4. Installed content structure

After the software installation is successfully performed, the software executables are available in the directory /opt/mws/eps-sg-mws-l1-lp/bin.

The installation package deployment tree is:

```

eps-sg-mws-l1-lp
├── bin
│   ├── mws_l1_lp.bin
│   ├── l1_lp_data_objects_test.bin
│   └── mws_l1_lp_algorithms_test.bin
├── lib
│   ├── libepssglpcommon.la
│   ├── libepssglpcommon.so
│   ├── libepssglpcommon.so.0
│   └── libepssglpcommon.so.<x.y.z>
└── schemas
    ├── mws_ccdb_reference.nheader
    ├── mws_cfi_reference.nheader
    ├── mws_cnf0_reference.xsd
    ├── mws_cnf1_reference.xsd
    ├── mws_cnfc_reference.xsd
    ├── mws_cnfp_reference.xsd
    ├── mws_cof_reference.nheader
    ├── mws_dem_reference.nheader
    ├── mws_joborder_reference.xsd
    ├── mws_joborder_reference_app1.xsd
    ├── mws_l0_reference.nheader
    ├── mws_lsm_reference.nheader
    └── mws_navatt_reference.nheader
    
```

The MWS L1b LP processor executable has the following name:

"mws_l1_lp.bin"

The MWS L1b UT executables have the following names:

"mws_l1_lp_algorithms_test.bin"

"l1_lp_data_objects_test.bin"

5.5. Unit Tests execution

The user can execute the Unit Tests to ensure the SW runs correctly. The Unit Test executables are present in the installation directory, and are the following:

- l1_lp_data_objects_test.bin – Unit Test covering the read/write of input and output data objects.
- mws_l1_lp_algorithms_test.bin – Unit Test covering the MWS algorithm functions.

The Unit Test data is provided in the SW package and shall be uncompressed. In order to do so, the user shall follow the next steps:

- Create a local temporary directory to extract and build the application, e.g. "~/my_tmp/mws_inst/", defined by \$MWS_TMP_DIR.

```
mkdir $MWS_TMP_DIR
```

- Uncompress the software package [MWS-SW] into the temporary directory.

```
cd $MWS_TMP_DIR  
tar xzf /path/to/L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>.tar.gz
```

- Uncompress the Unit Test data folder from the eps-sg-mws-l1-lp_utdata-v<x.y.z>.tar.gz package.

```
cd L1b_LP-SW-ISARD-EPS_SG-0211-v<x.y.z>  
tar xzf eps-sg-mws-l1-lp_utdata-v<x.y.z>.tar.gz
```

- Execute the unit tests.

```
/opt/mws/eps-sg-mws-l1-lp/bin/l1_lp_data_objects_test.bin  
  
/opt/mws/eps-sg-mws-l1-lp/bin/mws_l1_lp_algorithms_test.bin
```

Both tests shall finish with "PASSED" result.

6. Operations Basics

6.1. Launching executable

The **MWS L1b LP** processor is executed launching a binary, using as only input a job order. This job order can be automatically generated by the Processing Framework or can be written by the user.

In both cases, the interface executable is started by a single command:

```
<executable_name> <joborder_file.xml>
```

Where:

- *executable_name* is the name of the executable to be run, i.e., *mws_l1_lp.bin*
- *joborder_file.xml* is the XML file containing the input/output files information, as defined in [MWS-ICD].

6.2. Selection and control

6.2.1. Input MWS L0 products

The **MWS L1b LP** is designed to process one or several MWS L0 file(s), which cover a time-period equal or greater than a minimum time span. As per local processor requirement, any length equal or greater than 2.5 minutes can be processed; the actual minimum length of input data depends on the several processing averaging windows defined in the auxiliary input calibration file.

In the context of local processing nominal conditions, assuming that L0 data do not overlap, a sequential processing of the MWS L0 files is mandatory, and the reception of MWS L0 product is triggering the processing. The two MWS L0 granules surrounding the triggering MWS L0 product are also passed as input to ensure nominal processing at the edges of the triggering product, if they are available. The time interval [T1, T2] to process, defined by the start sensing time and stop sensing time parameters in the Job Order, corresponds to the start and stop of the triggering MWS L0 product filename.

The output of the **MWS L1B LP** will be a product spanning the time interval [T1I0, T2I0] corresponding to the actual start and stop of the triggering MWS L0 product. This product will be continuous in terms of content, metadata, and quality, with no degradations and discontinuities at the boundaries, provided the MWS L0 granules surrounding the triggering MWS L0 product are passed as input.

6.2.2. Additional L0 product and Auxiliary files

From MWS L1b LP utilisation point of view, all input files other than the MWS L0 products can be considered as Auxiliary data (static or dynamic). These files are listed in Table 6-1 and deeper described in [MWS-ICD].

When several types of files can be used, the priority is indicated in the column "Priority".

Note that, according to the MWS L1b LP Task Table, the Processing Framework will select:

- All MWS L0 and NAVATT L0 files that cover partly the time interval specified in the Job Order (ValIntersect selection rule).
- The Static auxiliary files with the latest creation date (LatestValCover selection rule), for each type of file.

Table 6-1. MWS L1b LP Additional L0 and Auxiliary files.

File	Description	Priority
NAV_00_SRC	NAVATT Level 0 product. It contains SGA Navigation and Attitude information generated by the Satellite	Orbit: 1 st choice
AUX_POFD	The predicted orbit file contains orbit state vectors for a given Metop-SG satellite with a time step of 1 minute and in Earth-Fixed reference frame	Orbit: 2 nd choice Time correlation: 1 st choice
AUX_IBA	The IERS bulletin A provides information on Earth's orientation in the IERS Reference System	Time correlation: 2 nd choice
MWS_1B_CCDB	The MWS L1B Configuration and Calibration Parameters file contains all information necessary for the MWS geolocation and calibration	
MWS_1B_CNF	The MWS L1B LP Configuration Parameters file contains: • The MWS L1b LP processors switches, which can be modified without the need of recompiling the s/w. • The definition of the ISP structure of a MWS L0 product. • The definition of NetCDF variables of a MWS L1b product • The definition of NetCDF variables of a MWS Context data file	
MWS_1B_LSM	The MWS Land-Sea Mask provides the information on whether the pixel is land, sea, or coast	
MWS_1B_DEM	The MWS Digital Elevation Model provides information on the terrain elevation at pixel locations	

6.2.3. Configuration files

The **MWS L1b LP** processor is configurable by means of an external auxiliary data file, the MWS L1B Configuration and Calibration Parameters package, which is specified in [MWS-ICD] and in [MWS-L1B-ADS]. The user can modify the configuration parameters in that file and, without the need of re-compiling the software, the processor will have a different behaviour. These configuration parameters are typically switches and selectors.

The MWS L1B LP Configuration Parameters package is composed of 5 files:

- Manifest.xml: The manifest describing the package content, which format and content are defined in [MWS-L1B-ADS], updated to include the additional configuration files for MWS L1b LP SW.
- MWS_L1B_AUX_CNFP_vxxyy.xml: The MWS L1b LP configuration parameters file.
- MWS_L1B_AUX_CNFO_vxxyy.xml: The MWS L0 Product configuration file.
- MWS_L1B_AUX_CNF1_vxxyy.xml: The MWS L1b Product configuration file.
- MWS_L1B_AUX_CNFC_vxxyy.xml: The MWS Context Data configuration file.

The configuration parameters which are intended to be modified by the user for its specific needs are gathered in the MWS L1b LP configuration parameters file (MWS_L1B_AUX_CNFP). Most of the configuration parameters stored inside the other files of the MWS L1B LP Configuration Parameters package are static and frozen unless the MWS instrument, [MWS-L1B-ADS], or [MWS-L1B-PFS] specifications evolve.

6.2.4. Job Order/Task Table

The Job Order is written in ASCII-based XML-format specified in [MWS-ICD] and in [L1-AD].

The parameters read from the Job Order files are dynamical ones, in the sense that each time the Job Order is generated by the Processing Framework, it can be different. When using the Processing Framework, the Job Order is filled automatically from the information available in the Task Table. Alternatively, if the processor is executed manually at prompt level, it can be fed with a hand-made Job Order file.

6.3. Normal Operation

In the nominal case, the processing ends successfully with return code set to 0 (success).

6.4. Error Conditions

In case of ending with an error, the processing terminates with a return code set to a value between 128 and 255. Messages of level ERROR are written in the log file to explain the root cause of the failure.

Error and Warning messages output by the **MWS L1b LP** are listed in Appendix A Table A-1. Exit Codes returned by the **MWS L1b LP** are listed in Appendix A Table A-2.

6.5. Validating inputs using schemas

Prior to the processing the **MWS L1b LP** performs a validation of the correctness of the filename and the format of each input files provided as input with respect to their specifications. This validation is performed towards templates provided together with the software.

The templates are listed in Table **6-2**.

Table 6-2. MWS L1b Local processor input validation schemas.

Schema	Validated input file	File Format
mws_jobborder_reference_app1.xsd mws_jobborder_reference.xsd	Job Order	xml
mws_ccdb_reference.ncheader	MWS_1B_AUX_CCDB file MWS_L1B_AUX_CCDB	netCDF
mws_cfi_reference.ncheader	MWS_1B_AUX_CCDB file MWS_L1B_AUX_CFI	netCDF
mws_cnf0_reference.xsd	MWS_1B_AUX_CNF_ file MWS_L1B_AUX_CNF0	xml
mws_cnf1_reference.xsd	MWS_1B_AUX_CNF_ file MWS_L1B_AUX_CNF1	xml
mws_cnfc_reference.xsd	MWS_1B_AUX_CNF_ file MWS_L1B_AUX_CNFC	xml
mws_cnfp_reference.xsd	MWS_1B_AUX_CNF_ file MWS_L1B_AUX_CNFP	xml
mws_cof_reference.ncheader	MWS_1B_AUX_COF_	netCDF
mws_dem_reference.ncheader	MWS_1B_AUX_DEM_	netCDF
mws_l0_reference.ncheader	MWS-00-SRC	netCDF
mws_lsm_reference.ncheader	MWS_1B_AUX_LSM_	netCDF
mws_navatt_reference.ncheader	NAV-00-SRC	netCDF

Appendix A MWS L1b LP Error Codes and Messages

Table A-1 presents the list of warning and error codes used in the **MWS L1b LP**.

Table A-1. Warning and error codes list.

Code number	Severity	Description
10	Error	Error launching the processing: command line argument is not ok.
20	Error	Error initialising a mandatory file from the job order
21	Warning	Filename validation failed
22	Warning	NetCDF validation failed
23	Warning	XML validation failed
30	Error	Error opening or reading the L0 Schema XML file
40	Warning	Optional input LSM or DEM static auxiliary file could not be initialised
41	Warning	Optional input INR dynamic auxiliary file could not be initialised
42	Warning	Optional input L1 COF file could not be initialised
47	Warning	Optional output L1 COF file could not be opened
51	Warning	Error reading ISP record
52	Warning	Error reading instrument L0 NetCDF variable
60	Warning	Input file could not be opened
61	Warning	Input file could not be read
62	Warning	Input file could not be initialised/stored
63	Warning	Unexpected NetCDF type used
65	Warning	Error creating one output directory
70	Error	Output directory does not exist
72	Warning	Error writing the output COF data file (bad quality flag)
80	Error	Error executing the processing algorithms

Table A-2 presents the list of return codes used in the **MWS L1b LP**.

Table A-2. Return codes list.

Code number	Severity	Description
0	Ok	Successful termination
128	Error	Generic error (unexpected situations)
130	Error	Job Order file missing
131	Error	Job Order file corrupted
132	Error	MWS CCDB input file invalid
133	Error	MWS CNF input file invalid
135	Error	MWS L0 input file missing or corrupted
137	Error	MWS L1b or COF output folder invalid
140	Error	Error detected during the validation of the input files

End of Document