OPERATING MANUALS FOR SYSTEM PROTOTYPE AND TOOLS

PROSBAS PROJECT

Prepared by:	GMV Team
Approved by:	M. Cueto (GMV)
Authorized by:	M. Cueto (GMV)

Code:	PROSBAS-GMV-D3.1.5
Version:	1.0
Date:	26/11/2013
Internal code:	GMV 22530/13 V1/13

GMV Aerospace and Defense S.A.U.
Isaac Newton 11, PTM Tres Cantos, 28760 Madrid
Tel. +34 918072100, Fax. +34 918072199, www.gmv.com

PROSBAS project is funded by the European Commission.

DOCUMENT STATUS SHEET

Version	Date	Pages	Changes
1.0	26/11/2013	77	First Version of the Document for MTR

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

1. INTRODUCTION

1.1. PURPOSE

This document is the *Operating Manuals for System Prototype and Tools* for the “Prototyping and Support to Standardisation of SBAS L1/L5 Multi-Constellation Receiver” project, for brevity herein referred to as PROSBAS.

The purpose of this document is to provide the Operating Manual of the software delivered for MTR, that is, PROSBAS Service Provider Prototype.

The present document is the first and definitive version of T3.1.5 output document, submitted in accordance with PROSBAS proposal.

1.2. SCOPE

The present document has been organized as follows:

- Chapter 1. gives an introduction to the document, including purpose and scope of the plan.
- Chapter 2. provides the list of project applicable and reference documents.
- Chapter 3. provides the list of definitions and acronyms used throughout the plan.
- Chapter 4. sketches the software overview.
- Chapter 5. describes the software environment.
- Chapter 6. describes the installation and execution of the software.
- Chapter 7. provides the user manual of the different software modules.

1.3. DOCUMENT CONTRIBUTIONS

Next table provides the details concerning the contributions to this document.

Table 1-1: Document Contributions

Section	Company
All sections	GMV

2. REFERENCES

2.1. APPLICABLE DOCUMENTS

The following documents, of the exact issue shown, form part of this document to the extent specified herein. Applicable documents are those referenced in the Contract or approved by the Approval Authority. They are referenced in this document in the form [AD.X]:

Table 2-1: Applicable documents.

Ref.	Title	Code	Version	Date
[AD.1]	PROSBAS Contract	157/PP/ENT/RCH/12/6373	N/A	29/08/2012

2.2. REFERENCE DOCUMENTS

The following documents, although not part of this document, amplify or clarify its contents. Reference documents are those not applicable and referenced within this document. They are referenced in this document in the form [RD.X]:

Table 2-2: Reference documents.

Ref.	Title	Code	Version	Date
[RD.1]	System Prototype Specification and Definition	PROSBAS-GMV-TN-D3.1.3	V2.0	26/11/2013
[RD.2]	Service Provider/User Receiver Prototype ICD	PROSBAS-GMV-TN-O3.1.1	V1.1	25/04/2013
[RD.3]	Operating Manuals For Receiver Prototype and Tools	PROSBAS-GMV-TN-D4.2.1	V1.0	26/11/2013
[RD.4]	Test Plan/Procedure/Criteria	PROSBAS-GMV-TN-D.3.1.2	V2.0	26/11/2013
[RD.5]	Test Reports For the SBAS L1/L5 ICD and User Standards Definition	PROSBAS-GMV-TN-D.3.3.1	V2.0	26/11/2013
[RD.6]	Test Tools specification and definition	PROSBAS-GMV-TN-D3.1.4	V2.0	26/11/2013
[RD.7]	Enhanced SBAS L1/L5 ICD: Analysis of Performances and Robustness	MSIL2-WO2-GMV-TN-011_SS07	V1.0	22/11/2013
[RD.8]	Minimum Operational Performance standards for global positioning system/wide area augmentation system airborne equipment	RTCA DO-229D	N/A	Dec 2006

3. TERMS, DEFINITIONS AND ABBREVIATED TERMS

3.1. DEFINITIONS

Concepts and terms used in this document and needing a definition are included in the following table:

Table 3-1 Definitions

Concept / Term	Definition

3.2. ACRONYMS

Acronyms used in this document and needing a definition are included in the following table:

Table 3-2 Acronyms

Acronym	Definition
CFI	Customer Furnished Items
DAL	Development Assurance Level
DDF	Design Definition File
DJF	Design Justification File
ECAC	European Civil Aviation Conference
ECSS	European Cooperation for Space Standardization
EGEP	European GNSS Evolution Programme
EGNOS	European Geostationary Navigation Overlay Service
ENP	European Neighbouring Policy
FDIR	Fault Detection, Isolation and Retrieval
FE	Feared Event
FEC	Forward Error Coding
FR	Final Review
GEO	Geostationary Satellite
GIVE	Grid Ionosphere Vertical Error
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HW	Hardware
ICD	Interface Control Document
ISC	Inter Signal Corrections
IWG	Interoperability Working Group
KO	Kick Off
KP	Key Point
MGD	Mission Guideline Document
MOPS	Minimum Operational Performance Standards
MRD	Mission Requirement Document
MTR	Mid Term Review
PM	Progress Meeting
PRN	Pseudo Random Noise
PROSBAS	Prototyping and Support to Standardisation of L1/L5 Multi-Constellation Receiver
RD	Reference Document

Acronym	Definition
RF	Radio Frequency
RP	Receiver Prototype
SBAS	Satellite Based Augmentation System
SIS	Signal In Space
SoL	Safety of Life
SPP	Service Provider Prototype
SoW	Statement of Work
SQM	Signal Quality Monitoring
SRD	System Requirement Document
SUGAST	Support to GNSS Aviation Standardisation
SW	Software
TBC	To Be Confirmed
TBD	To Be Determined
TN	Technical Note
TTA	Time To Alert
TC	Test Cases
WBS	Work Breakdown Structure
WPD	Work Package Description

4. SOFTWARE OVERVIEW

This section is devoted to briefly describe PROSBAS Prototype and in particular, the Service Provider Prototype.

4.1. PROSBAS PROTOTYPE OVERVIEW

This section provides a brief description of PROSBAS Prototype.

For further information on the architecture, requirements and description, as well as data flows, please refer to [RD.1] and [RD.2] respectively.

PROSBAS Prototype can be decomposed in Service Provider Prototype and Receiver Prototype as shown in the following Figure:

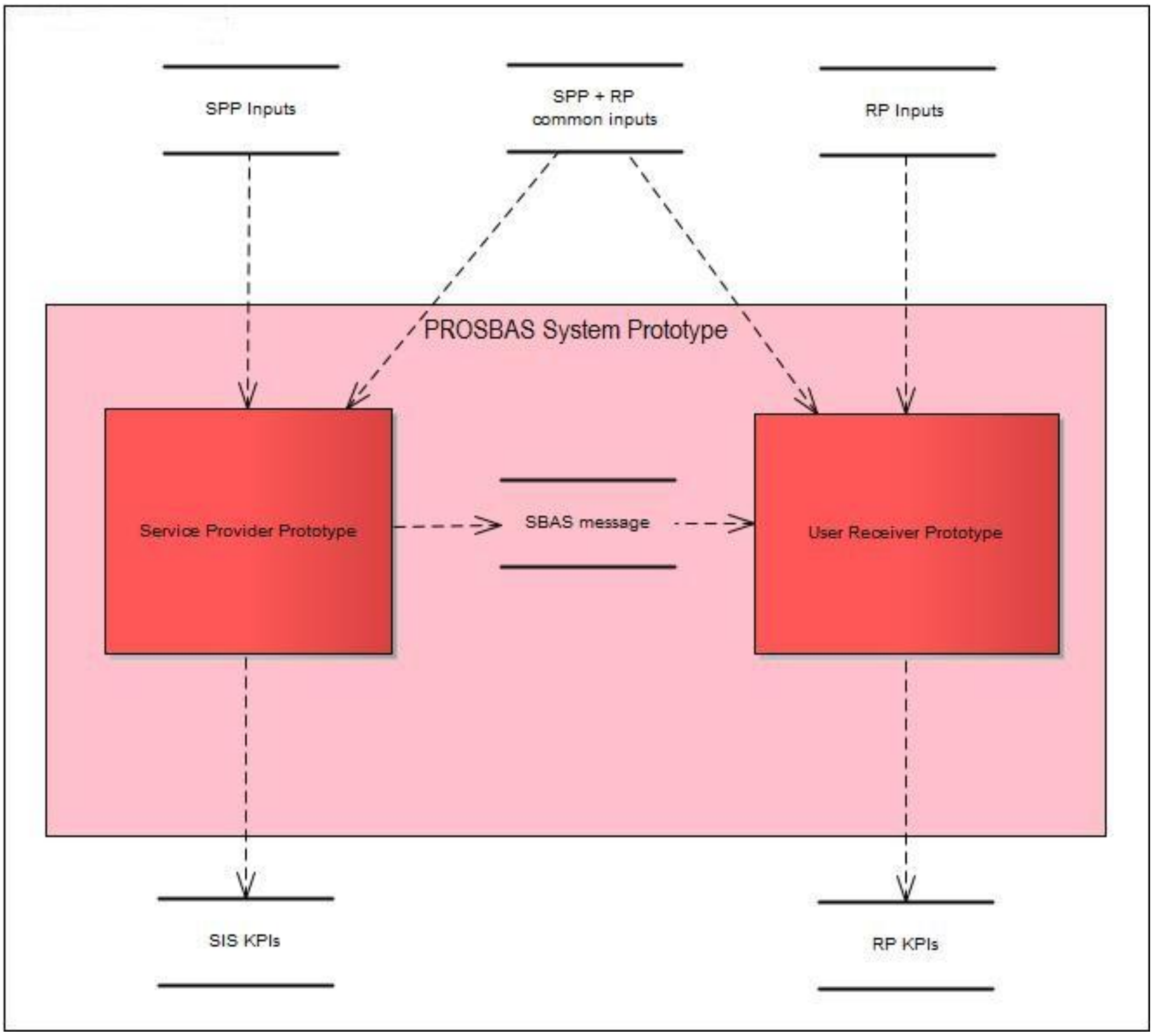

Figure 4-1: PROSBAS Prototype Architecture

4.2. SERVICE PROVIDER PROTOTYPE OVERVIEW

This section briefly describes the Service Provider Prototype.

In the following figure it is shown the architecture of SPP.

SPP consists of 5 modules:

- Message Sequence Generator (MSG)
- NOF Generator (NOFG)
- SBAS Message Encoder (SME)
- Message Loss Simulator (MLS)
- Message Sequence Analyser (MSA)

In addition, there is an independent module NavRinexReader that is used to compute the positions of the satellites from RINEX Navigation Files, generating a file SatPos.txt used as input by both SPP and RP.

MSG, NOFG and SME modules are integrated and then, only one executable is provided for these three modules. With a high-level perspective, MSG Module generates a message sequence, NOFG simulates the NOF and SME module encodes the NOF content in the message sequence for generating the SBAS Message.

Then, MLS Module simulates loss of messages at user level and MSA module analyses the message sequence.

There are two types of SPP Outputs: on the one hand, SIS KPIs outputs and on the other hand, the SBAS Messages that are used as input by the Receiver Prototype.

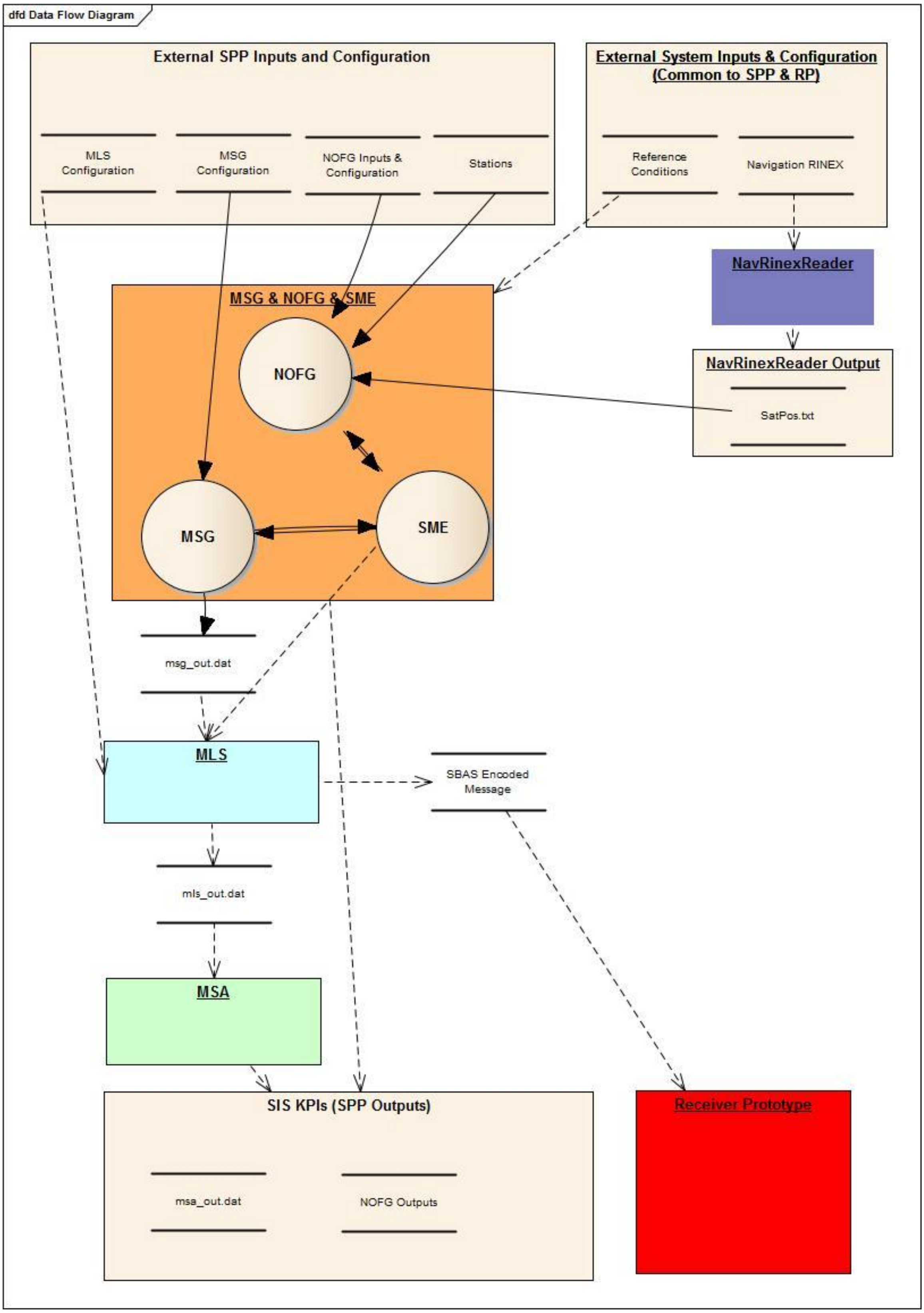

Figure 4-2: SPP Data Flow Diagram

For a detailed description of SPP architecture, please refer to [RD.1].

5. ENVIRONMENT

This section describes the environment where PROSBAS Prototype is allocated.

In particular, the HW characteristics where PROSBAS can be run will be described.

The typical performances in terms of execution time and disk usage are provided.

Finally, the directories structure provided in the HW deliverable will be explained.

5.1. ENVIRONMENT

Taget HW: 64-bit PC Intel(R) Core(TM)2 Duo CPU E8400 @ 3.00GHz

RAM: 3 Gb

Hard Disk: 450 Gb

OS: 32 bit running Linux Ubuntu 12.04

PROSBAS Prototype makes use of the following dynamic libraries (as provided by the linux ldd command):

- linux-gate.so.1 => (0xb7769000)
- libstdc++.so.6 => /usr/lib/i386-linux-gnu/libstdc++.so.6 (0xb7676000)
- libm.so.6 => /lib/i386-linux-gnu/libm.so.6 (0xb764a000)
- libgcc_s.so.1 => /lib/i386-linux-gnu/libgcc_s.so.1 (0xb762b000)
- libc.so.6 => /lib/i386-linux-gnu/libc.so.6 (0xb7482000)
- /lib/ld-linux.so.2 (0xb776a000)

In addition, MLS module depends on the following Ubuntu 12.04 features:

- /bin/bash
- /usr/bin/awk linked to /usr/bin/mawk

Visualization Module makes use of gnuplot Version 4.4 patchlevel 3 and GNU Octave, version 3.2.4.

5.2. TYPICAL PERFORMANCE

This section provides the typical performance expected with PROSBAS Prototype, in particular, the expected execution time and disk usage.

5.2.1. EXECUTION TIME

The execution time of PROSBAS Prototype depends on the configuration.

In particular, the execution time strongly depends on the Operational Mode, on the number of epochs to process, on the number of SVs configured in mask, on the broadcast or not of Ionospheric Messages, on the size of the service area and on the grid used to compute for example, the PLs.

The execution time then varies from several minutes to several hours. From all the scenarios executed before this writing, no scenario has been found with an execution time larger than 15 hours.

5.2.2. DISK USAGE

The disk usage of PROSBAS Prototype depends on the configuration.

In particular, the disk usage strongly depends on the number of epochs to process, on the number of SVs configured in mask, on the broadcast or not of Ionospheric Messages, on the size of the service area and on the grid used to compute for example, the PLs.

The disk usage is of the order of tens of Gb.

5.3. DIRECTORIES STRUCTURE

In this section the directories structure provided within the HW & SW delivered is explained.

This structure can be seen in the following Figure:

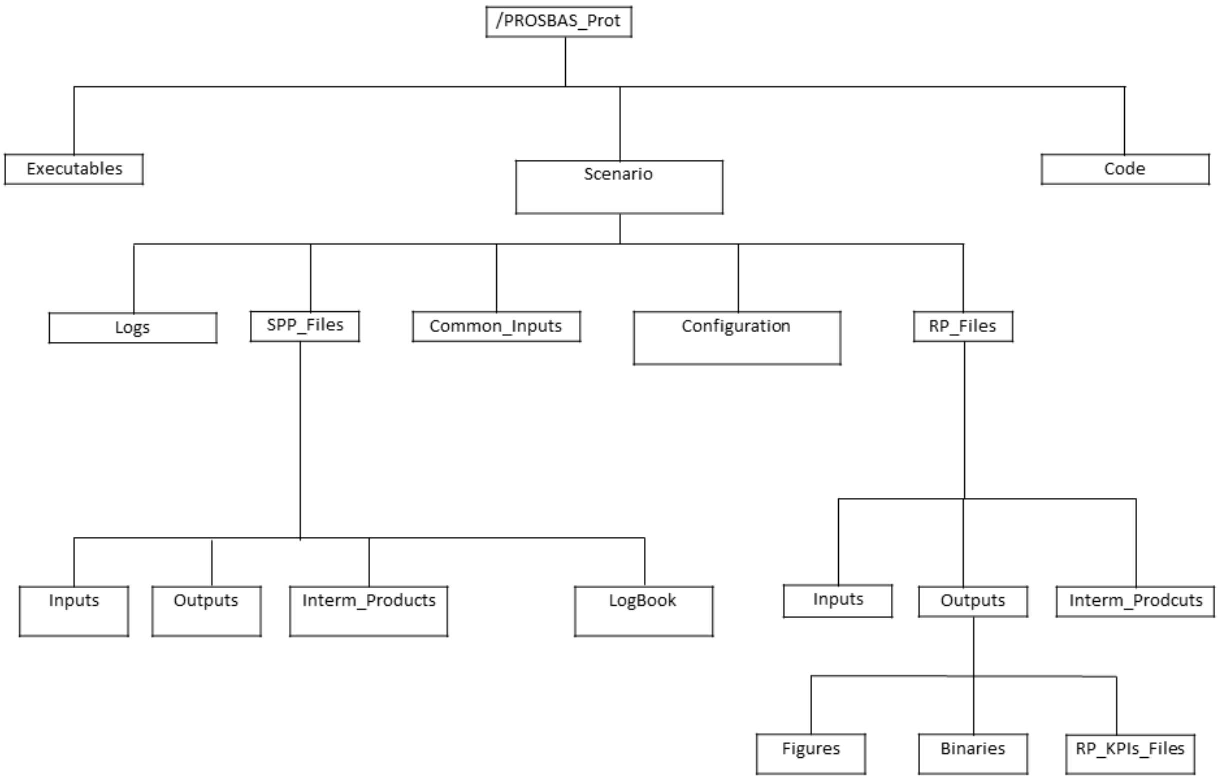

Figure 5-1: Directories structure provided in PROSBAS HW & SW delivery

The different modules present in PROSBAS Prototype can be run separately and sequentially using configuration files and the outputs of certain modules being the inputs of other modules. In that case all the executable files and configuration or input files could be placed in the same folder.

The commands to run an individual module and the needed configuration files can be checked in Section 7. for the case of SPP Prototype while for the RP modules, please refer to [RD.3].

Nevertheless, a full directories structure provided in PROSBAS HW & SW delivery is deployed with a driver that executes all PROSBAS Prototype.

The root directory is */PROSBAS_Prot*, where the different folders are located.

- In the *Executables* folder, there are placed the different executables of the different modules that composes PROSBAS Prototype. Also the driver of PROSBAS Prototype, with name *PROSBAS_driver.sh* is placed in this folder for executing PROSBAS Prototype.
  In addition, a compilation script of the code in also placed as well as a script for cleaning the Scenario folder.
- In the *Code* folder PROSBAS Prototype code is located.
- The *Scenario* folder contains a sample Scenario directory tree that will be used to run the PROSBAS Prototype. Every Scenario must contain:

- In the *Common_Inputs* folder, the common inputs to both SPP and RP Prototype are located.
- In the *Configuration* folder, the configuration files of the different PROSBAS Prototype Modules are located.
- In the *SPP_Files* folder the different inputs, outputs and intermediate product files of the SPP are located. In particular, the following directories substructure is provided:
  - *Inputs* folder with the input files to the SPP.
  - *Outputs* folder with the final outputs of the SPP.
  - *Interm_Products* folder with outputs of several modules of the SPP that are used as inputs for other modules.
  - *LogBook* folder where the LogBooks containing the SBAS Message generated by the SPP are located. These files are used as inputs for the RP.
- In *RP_Files* folder the different inputs, outputs and intermediate product files of the RP are located. In particular, the following directories substructure is provided:
  - *Inputs* folder with the input files to the RP.
  - *Interm_Products* folder with outpus of several modules of the RP that are used as inputs for other RP modules.
  - *Outputs* folder. In this folder, two subfolders are present:
    - *Figures* folder with plots displaying the computations performed by the RP.
    - *Binaries* folder with binary files generated by RP.
    - *RP_KPIs_Files* folder with files containing data computed by the RP.
- In the *Logs* folder, execution logs are generated by the PROSBAS Prototype modules.

In the following table it is shown the content of the different folders described above, that is, the files, executables or subfolders that are contained.

Table 5-1: Directories structure provided in PROSBAS HW & SW delivery

Directory	Purpose	Content
/PROSBAS_Prot/	Main directory	*./Executables/* *./Scenario* *./Code/*
/PROSBAS_Prot/Executables/	Directory where the different executables of the different modules of PROSBAS Prototype are located. Also, some scripts are available, as the driver to run all PROSBAS Prototype is placed in this folder.	*PROSBAS_driver.sh* *Install.sh* *clean_scenario.sh* *NavRinexReader* *MLS* *MSA* *SME (MSG, NOFG and SME modules)* *SBASdecoder_Enhanced_ICD* *UserLevelAnalysis* *Visualization* Compilation Logs
/PROSBAS_Prot/Code/	Directory where the source code of PROSBAS Prototype is located.	Source code of PROSBAS Prototype.
/PROSBAS_Prot/Scenario	Directory where the configuration and input files of PROSBAS Prototype are located	*Common_Inputs/* *Configuration/* *Logs/*

	and where the outputs and Intermediate products are generated.	*RP_Files/* *SPP_Files/*
/PROSBAS_Prot/Scenario/Common_Inputs/	Directory where the inputs common to both SPP and RP are located. In particular, there are placed the RINEX Navigation files and once the *NavRinexReader* module is executed, a file *SatPos.txt* with the positions of the SVs in WGS84 is generated.	RINEX Navigation files *SatPos.txt*
/PROSBAS_Prot/Scenario/Configuration/	Directory where the different configuration files of the different PROSBAS Prototype modules are located. In particular, the configuration file *ref_cond_cfg.dat* that is a common configuration file to both SPP and RP is placed.	*deg_params_cfg.dat* *gen_L1_NOF_DFRE_cfg.dat* *IGP_world_ecac.cfg* *mls_cfg.dat* *msg_cfg.dat* *ref_cond_cfg.dat* *rp_conf.txt* *stations.cfg*
/PROSBAS_Prot/Scenario/Logs	Directory where the log files of the different modules are generated	*NavRinexReader_err.log* *NavRinexReader.log* *SME_err.log* *SME.log* *MLS_err.log* *MLS.log* *SBASdecoder_Enhanced_ICD_err.log* *SBASdecoder_Enhanced_ICD.log* *UserLevelAnalysis_err.log* *UserLevelAnalysis.log* *Visualization_err.log* *Visualization.log*
/PROSBAS_Prot/Scenario/SPP_Files/	Directory containing subfolders where the different input, output and intermediate files of SPP are located.	*./Inputs/* *./Outputs/* *./Interm_Products/* *./LogBook*
/PROSBAS_Prot/Scenario/SPP_Files/Inputs/	Directory where the inputs of the SPP are located. In particular, in case of Operational Mode 2, the files containing the L1 SBAS	Files with L1 SBAS Messages in case Operational Mode 2. *SV_Mask.dat* *MT2_5.dat* *MT6.dat*

	Messages are placed.	*MT10.dat* *MT12.dat* *MT18.dat* *MT24.dat* *MT25_VelCode0.dat* *MT26.dat* *MT28.dat*
/PROSBAS_Prot/Scenario/SPP_Files/Outputs/	Directory where the final output of SPP is generated (that is, for example, the output of *MSA* module with bandwidth, UI & TO fulfillment… analyses).	*give_l5* *Iode* *msa_out.dat* (if MLS execution is not activated) *msa_out.dat_SPP* (if MLS execution is activated) *msa_out.dat_RP* (if MLS execution is activated) *stations_latitude_longitude_heigh* *udre_l1* *udre_l5* *IODE_PRNx.png* *NumSVvsTime_SPP.png* *NumsSVvsTime_SPPvsRP.png*
/PROSBAS_Prot/Scenario/SPP_Files/Interm_Products/	Directory where the intermediate files of the SPP are generated. These files are outputs from a certain SPP module that are inputs of other SPP modules.	*msg_out.dat* *mls_out.dat*
/PROSBAS_Prot/Scenario/SPP_Files/LogBook/	Directory where the LogBooks containg the L1/L5 SBAS Messages are generated. These files are a final output of the SPP that is used as input for the RP. Those files are copied into */PROSBAS_Prot/RP_Files/Inputs/* since they are the inputs for the RP.	LogBooks with SBAS L1/L5 messages. Name of LogBook files: LogBook_year_month_day_hour.dat Example: *LogBook_1999_09_03_00.dat* *LogBook_1999_09_03_01.dat* …
/PROSBAS_Prot/Scenario/RP_Files/	Directory containing subfolders where the different input, output and intermediate files of RP are located.	*./Inputs/* *./Outputs/* *./Interm_Products/*
/PROSBAS_Prot/Scenario/RP_Files/Inputs/	Directory where the LogBooks with the SBAS L1/L5 messages generated by the SPP	LogBooks with SBAS L1/L5 messages. *IGP_world_ecac.cfg* *SatPos.txt*

	are copied and used as inputs for the RP. The IGP definition file and the satellite positions files are also located in this directory	
/PROSBAS_Prot/Scenario/RP_Files/Interm_Products/	Directory where the intermediate files of the RP are generated. These files are outputs from a certain RP module that are inputs of other RP modules.	*Udre.dat* *GIVE_XXXXX.dat*
/PROSBAS_Prot/Scenario/RP_Files/Outputs/	Directory containing subfolders where the final output of the RP are placed.	*./Figures/* *./RP_KPIs_Files/* *./Binaries/*
/PROSBAS_Prot/Scenario /RP_Files/Outputs/Figures	Directory containing plots generated as outputs of the RP.	*ContinuityBreaks.png* *ContinuityRisk.png* *HDOPdeviation.png* *HDOPmax.png* *HDOPmean.png* *HDOPpercentile95.png* *HDOPpercentile99.png* *HDOPrms.png* *HPEdeviation.png* *HPEintoHPLdeviation.png* *HPEintoHPLmax.png* *HPEintoHPLmean.png* *HPEintoHPLpercentile95.png* *HPEintoHPLpercentile99.png* *HPEintoHPLrms.png* *HPEmax.png* *HPEmean.png* *HPEpercentile95.png* *HPEpercentile99.png* *HPErms.png* *HPEvsHPLandVPEvsVPL_Measured.png* *HPEvsHPL_Measured.png* *HPLdeviation.png* *HPLmax.png* *HPLmean.png* *HPLpercentile95.png* *HPLpercentile99.png* *HPLrms.png* *HPLvsHALandVPLvsVAL_Measur*

		ed.png *HPLvsHALandVPLvsVAL_SignalInSpace.png* *HPLvsHAL_Measured.png* *HPLvsHAL_SignalInSpace.png* *PDOPdeviation.png* *PDOPmax.png* *PDOPmean.png* *PDOPpercentile95.png* *PDOPpercentile99.png* *PDOPrms.png* *UsedSatellitesdeviation.png* *UsedSatellitesmax.png* *UsedSatellitesmean.png* *UsedSatellitespercentile95.png* *UsedSatellitespercentile99.png* *UsedSatellitesrms.png* *VDOPdeviation.png* *VDOPmax.png* *VDOPmean.png* *VDOPpercentile95.png* *VDOPpercentile99.png* *VDOPrms.png* *VPEdeviation.png* *VPEintoVPLdeviation.png* *VPEintoVPLmax.png* *VPEintoVPLmean.png* *VPEintoVPLpercentile95.png* *VPEintoVPLpercentile99.png* *VPEintoVPLrms.png* *VPEmax.png* *VPEmean.png* *VPEpercentile95.png* *VPEpercentile99.png* *VPErms.png* *VPEvsVPL_Measured.png* *VPLdeviation.png* *VPLmax.png* *VPLmean.png* *VPLpercentile95.png* *VPLpercentile99.png* *VPLrms.png* *VPLvsVAL_Measured.png* *VPLvsVAL_SignalInSpace.png* *xPLvsTime_XX_YY.png* *NumSVvsTime_RP.png*

		NumSVvsTime_SPPvsRP.png
/PROSBAS_Prot/Scenario/RP_Files/Outputs/RP_KPIs_Files	Directory containing files with data generated as outputs of the RP.	*ContinuityRisk.dat* *GridPLs_XXXXX.dat* *HDOPdeviation.dat* *HDOPmax.dat* *HDOPmean.dat* *HDOPmonitoredEpochs.dat* *HDOPpercentile95.dat* *HDOPpercentile99.dat* *HDOPrms.dat* *HPEdeviation.dat* *HPEintoHPLdeviation.dat* *HPEintoHPLmax.dat* *HPEintoHPLmean.dat* *HPEintoHPLmonitoredEpochs.dat* *HPEintoHPLpercentile95.dat* *HPEintoHPLpercentile99.dat* *HPEintoHPLrms.dat* *HPEmax.dat* *HPEmean.dat* *HPEmonitoredEpochs.dat* *HPEpercentile95.dat* *HPEpercentile99.dat* *HPErms.dat* *HPEvsHPLandVPEvsVPL.dat* *HPEvsHPL.dat* *HPLdeviation.dat* *HPLmax.dat* *HPLmean.dat* *HPLmonitoredEpochs.dat* *HPLpercentile95.dat* *HPLpercentile99.dat* *HPLrms.dat* *HPLvsHALandVPLvsVAL.dat* *HPLvsHAL.dat* *OS_AccuracyAvailability.dat* *OS_GeometryAvailability.dat* *PDOPdeviation.dat* *PDOPmax.dat* *PDOPmean.dat* *PDOPmonitoredEpochs.dat* *PDOPpercentile95.dat* *PDOPpercentile99.dat* *PDOPrms.dat*

		Sigma_dfc_XXXXX.dat *UsedSatellitesdeviation.dat* *UsedSatellitesmax.dat* *UsedSatellitesmean.dat* *UsedSatellitesmonitoredEpochs.dat* *UsedSatellitespercentile95.dat* *UsedSatellitespercentile99.dat* *UsedSatellitesrms.dat* *VDOPdeviation.dat* *VDOPmax.dat* *VDOPmean.dat* *VDOPmonitoredEpochs.dat* *VDOPpercentile95.dat* *VDOPpercentile99.dat* *VDOPrms.dat* *VPEdeviation.dat* *VPEintoVPLdeviation.dat* *VPEintoVPLmax.dat* *VPEintoVPLmean.dat* *VPEintoVPLmonitoredEpochs.dat* *VPEintoVPLpercentile95.dat* *VPEintoVPLpercentile99.dat* *VPEintoVPLrms.dat* *VPEmax.dat* *VPEmean.dat* *VPEmonitoredEpochs.dat* *VPEpercentile95.dat* *VPEpercentile99.dat* *VPErms.dat* *VPEvsVPL.dat* *VPLdeviation.dat* *VPLmax.dat* *VPLmean.dat* *VPLmonitoredEpochs.dat* *VPLpercentile95.dat* *VPLpercentile99.dat* *VPLrms.dat* - *VPLvsVAL.dat*
/PROSBAS_Prot/Scenario /RP_Files/Outputs/Binaries	Directory containing binary files with data generated as intermediate outputs of the RP.	*HDOP* *HPE* *HPEintoHPL* *HPEvsHPL* *HPL* *HPLvsHAL*

		OS_Accuracy *OS_Geometry* *PDOP* *UsedSatellites* *ValidEpochs* *ValidIntegEpochs* VDOP VPE VPEintoVPL VPEvsVPL VPL VPLvsVAL

6. GETTING STARTED

In this section, the installation, execution and de-installation of PROSBAS Prototype is explained.

6.1. INSTALLATION

PROSBAS Prototype will be fully installed in the machine delivered in MTR. The directories structure described in Section 5.3 will be deployed, together with the executables of the different PROSBAS Prototype modules and PROSBAS driver *PROSBAS_driver.sh.*

Also all the configuration files and inputs necessary to run PROSBAS Prototype will be placed in the adequate folders. In this way, PROSBAS prototype can be run with a default configuration using the driver.

In order to run PROSBAS Prototype with another configuration, just change the configuration options in the configuration files.

Nevertheless, if it is desired to use PROSBAS Prototype in another machine, one should create a directory structure as the one shown in Section 5.3 using *mkdir* commands or the script *clean_scenario.sh* with –n option and then place the executables in /PROSBAS_Prot/Executables folder and the configuration and input files in the adequate folders. Let us note that the executables can be created from the compilation of the source code. Let us also remark that default configuration files can be generated using *PROSBAS_driver.sh* script with –e option.

Note that it is not compulsory to have this directories structure. PROSBAS Prototype can be used running the different modules sequentially. Nevertheless, it is recommended to have this directories structure in order to be able to run PROSBAS Prototype using the driver.

It is also important to remark that default configuration files can be generated with PROSBAS driver, as it will be explained in the section 6.2.1.

6.1.1. SOURCE CODE COMPILATION

For the case where source is to be compiled, an installation script is provided. It is also placed in the Executables directory, and it automatically compiles all the modules and places the compiled executables in the Executables folder.

The help command of this 'Install.sh' script shows the following information:

./Install.sh –h

Table 6-1: *Install.sh* usage

```
#################################
###              Install.sh              ###
#################################
#  brief: PROSBAS Prototype installation script
#
#  Copyright: GMV & European Commission
#
#  Usage:
#     ./Install.sh [-h]
#
#  File description:
#     This script compiles all the modules within the PROSBAS Prototype,
#     and prepares the Executables folder to be able to run the prototype.
```

```
#
#     Options:
#         -h       Show help and exit.
#
#  Common modules
#      * Satellite Position computation
#  SPP modules
#      * MSG (Message Sequence Generator)
#      * MLS (Message Loss Simulator)
#      * NOFG (NOF Generator).
#      * SME (SBAS Message Emulator)
#      * MSA (Message Sequence Analyzer)
#  RP modules
#      * SBAS_DECODER (SBAS Message decoder)
#      * USER (User Level Analysis)
#      * VISUALIZATION (Visualization module)
#
################################
```

Then, typing

./Install.sh

The following executables are generated: *NavRinexReader, SME, MLS, MSA, SBASdecoder_Enhanced_ICD, UserLevelAnalysis and Visualization*.

In addition, logs of the compilation of the different modules are generated: *compilation_libConfiguration.log, compilation_MLS.log, compilation_MSA.log, compilation_MSG.log, compilation_NavRinexReader.log, compilation_nofg.log, compilation_SBASdecoder_Enhanced_ICD.log, compilation_SME.log, compilation_UserCommonLib.log, compilation_UserLevelAnalysis.log, compilation_Visualization.log.*

6.2. EXECUTION

The PROSBAS Prototype can either be run manually, module by module, or using the 'PROSBAS_driver.sh' script provided. The latter option is highly recommended.

6.2.1. PROSBAS PROTOTYPE EXECUTION USING PROSBAS DRIVER

PROSBAS Prototype can be run using a script with name *PROSBAS_driver.sh* placed in Executables folder.

For doing that, all configuration and input files should be placed in the adequate folders into *Scenario* folder. The outputs will also be generated in the corresponding folders into *Scenario* folder.

Configuration and input files for running PROSBAS Prototype will be fully provided in PROSBAS MTR delivery. Nevertheless, one can generate default configuration files that will be placed in */Scenario/Configuration* folder.

For seeing the usage of PROSBAS driver one should go to *Executables* folder and type:

./PROSBAS_driver.sh –h

This command will show the driver help, which is:

Table 6-2: *PROSBAS_driver.sh* usage

```
###########################################
###              PROSBAS_driver.sh             ###
###########################################
#  brief: PROSBAS driver
#
#  Copyright: GMV & European Commission
#
#  Usage:
#      ./PROSBAS_driver.sh [-h] [-d] [-e scenario_path] [-c PROSBAS_driver.cfg]
#
#  File description:
#      This script launches sequentially the configured modules that form
#      PROSBAS prototype.
#
#      Options:
#          -h       Show help and exit.
#
#          -d       Print a default driver configuration in stdout and exit. This output may
#                   be redirected to a file to generate a default configuration file PROSBAS_driver.cfg.
#
#          -c file  Run PROSBAS Prototype. Specify the configuration file.
#
#          -e scenario_path
#                   Generate default configuration files for the different PROSBAS
#                   Prototype Modules and place them in the adequate directories of Scenario folder.
#
#  Common modules
#      * Satellite Position computation
#  SPP modules
#      * MSG (Message Sequence Generator). See note below
#      * MLS (Message Loss Simulator)
#      * NOFG (NOF Generator). See note below
#      * SME (SBAS Message Emulator)
#      * MSA (Message Sequence Analyzer)
#  RP modules
#      * SBAS_DECODER (SBAS Message decoder)
#      * USER (User Level Analysis)
#      * VISUALIZATION (Visualization module)
#
# NOTE: The NOFG and MSG modules are launched inside the SME module
# NOTE: The MSA module can be launched with several options that can
#       be configured in PROSBAS_driver.cfg
###############################################
```

For generating default configuration files, go to *Executables* directory and type:

./PROSBAS_driver.sh –e scenario_path

For running PROSBAS Prototype with the driver, one should have a configuration file *PROSBAS_driver.cfg* placed in *Executables* folder. This configuration file will be deployed by installation. Nevertheless, one can generate a default PROSBAS driver configuration file going to *Executables* directory and typing:

./PROSBAS_driver.sh –d > PROSBAS_driver.cfg

Finally, for running PROSBAS Prototype with PROSBAS driver according to the configuration in PROSBAS_driver.cfg file, one should go to *Executables* folder and type:

./PROSBAS_driver.sh –c PROSBAS_driver.cfg

In the following table, there is an example of PROSBAS_driver.cfg file:

Table 6-3: example of *PROSBAS_driver.cfg* file

```
########################################
###              PROSBAS driver - Configuration file              ###
########################################

########################################
# Paths configuration                                            #
########################################
# Note that relative paths will be relative to the directory from where
# the PROSBAS_driver is executed.
ExecutableFolder="./"
SCENARIO_PATH="../Scenario"

#######################################
# Modules configuration: 1=Execute, 0=Do NOT execute          #
#######################################
# COMMON modules
EXEC_NAVRINEX=1
# SPP modules
EXEC_MLS=1
EXEC_SME=1
EXEC_MSA=1
# RP modules
EXEC_SBAS_DECODER=1
EXEC_USER=1
EXEC_VISUALIZATION=1
# Option to run MSA Module. If no option is selected, the output of MSA is the standard.
# Several options can be selected at the same time.
# The following options can be used:
# -a                    Toggle to show detailed data on TTA violations (default: do not).
# -m mx_epoch           Maximum epoch to process.
```

```
# -t                    Toggle to show detailed data on timeout violations (default: do not).
# -u                    Toggle to show detailed data on update time violations (default: do not).
# -A                    Toggle to show summary data on TTA violations (default: do it).
# -B                    Toggle to perform bandwidth static analysis (default: do it).
# -T                    Toggle to show summary on update time and timeout violations (default: do
it).
# -s sh_epochs    Number of epochs to skip before counting violations (default: zero).
EXEC_MSA_OPTION=""
```

The options to run PROSBAS Prototype with the configuration present in *PROSBAS_driver.cfg* file are briefly described below:

- ExecutableFolder is the path where the executables of the different PROSBAS Prototype module are placed. In principle, they can be placed in the same folder that *PROSBAS_driver.sh* file, that is, *Executables* folder.
- SCENARIO_PATH is the path of the Scenario folder, where the input and configuration files should be allocated and where the outputs will be generated. This allows to have several scenario folders in the directory /PROSBAS_Prot, or in any other directory.
- EXEC_NAVRINEX, EXEC_MLS, EXEC_SME, EXEC_MSA, EXEC_SBAS_DECODER, EXEC_USER and EXEC_VISUALIZATION are Boolean options to run (1) or not (0) the different PROSBAS Prototype modules.
- EXEC_MSA_OPTION is a parameter that is used to generate more or less detailed information on the analysis of the message sequence performed with MSA module.

In *Executables* folder there is also a script that allows to clean the *Scenario* folder either removing all the configuration, input, intermediate products and output files, or either removing only the intermediate an output files.

It may also be used to generate clean_scenario directory trees. All the information about this script is shown with the help command:

./clean_scenario.sh –h

Table 6-4: *clean_scenario.sh* usage

```
########################################
###               clean_scenario.sh                ###
########################################
#  brief: Clean Scenario
#
#  Copyright: GMV & European Commission
#
#  Usage:
#      ./clean_scenario.sh [-h] :
#                  Usage
#
#      ./clean_scenario.sh [-a Scenario_path] :
#                  Remove all files from Scenario folder
#                  (inputs, configuration, outputs, intermediate products).
#
#      ./clean_scenario.sh [-o Scenario_path] :
#                  Remove outputs and intermediate products
```

```
#                    from Scenario folder. Inputs and configuration are kept.
#
#     ./clean_scenario.sh [-c PROSBAS_driver_config_file] :
#                    Remove outputs and intermediate products
#                    from the configured Scenario folder.
#                    Inputs and configuration are kept.
#
#     ./clean_scenario.sh [-n Scenario_path] :
#                    Create a NEW full scenario directory tree.
#                    Any previous content is deleted (asks for user confirmation).
#
#  File description:
#     This script manages the Scenario directory tree.
#
#     -It removes files within the scenario but does not remove any folder.
#     Depending on the option, the inputs and configuration files can be
#     removed or not. The outputs and intermediate products are always removed.
#
#     -It may also create a new scenario directory tree.
#
##########################################
```

For cleaning the scenario (before the new execution of PRROSBAS prototype) one should go to *Executables* folder an type:

./clean_scenario.sh –a Scenario_path

for cleaning all the configuration, inputs, intermediate products and output files or:

./clean_scenario.sh –o Scenario_path

for removing only the intermediate products and output files.

There is an additional option, which performs the same actions than the '-o', but uses as input a PROSBAS_driver.cfg file, cleaning the configured Scenario within:

./clean_scenario.sh –c PROSBAS_driver.cfg

6.2.2. PROSBAS PROTOTYPE EXECUTION WITHOUT PROSBAS DRIVER

The different PROSBAS Prototype modules can be run separately if they are fed with the adequate inputs and configuration files.

Please, refer to Section 7. to learn how to run separately the SPP PROSBAS Prototype modules. The RP modules execution is described in [RD.3].

Let us only remark here the order of the sequential execution of the different PROSBAS Prototype Modules:

1. NavRinexReader
2. SME (MSG & NOFG & SME)
3. MLS
4. MSA
5. SBAS Decoder
6. User Level Analysis
7. Visualization

6.3. DE-INSTALLATION

For de-installation of PROSBAS Prototype just remove */PROSBAS_Prot* folder where the code, executables and *Scenario* folder are located.

For that, just type:

rm –r /PROSBAS_Prot

7. USER MANUAL

In this section, the different modules of the SPP will be reviewed. In particular, the invocation, configuration, input format and output format will be explained.

The same issues concerning RP modules are explained in [RD.3].

The design of PROSBAS Prototype, where more information is provided concerning the different modules and data flows can be seen in [RD.1] and [RD.2].

7.1. COMMON INPUTS AND CONFIGURATION

In this section the common inputs and configuration files shared by both SPP and RP are described.

The common configuration file is *ref_cond_cfg.dat* file that is placed in */PROSBAS_Prot/Scenario/Configuration* folder.

The common inputs to both SPP and RP are the RINEX Navigation files that are used by NavRinexReader module to generate the *SatPos.txt* file with the positions of the SVs.

7.1.1. REFERENCE CONDITIONS

ref_cond_cfg.dat file is a file that contains configuration parameters used by both SPP and RP modules.

In the following, there is shown an example of *ref_cond_cfg.dat* file.

Table 7-1: Example of *ref_cond_cfg.dat* file

```
# Reference Conditions Configuration file common to SPP and RP
#
# Service area configuration. Latitude and longitude in deg.
# Lat_Num and Lon_Num define the service area grid (min num = 1).
# delta_Lat is defined as (Lat_Max - Lat_Min)/Lat_Num . delta_Lon is equivalent.
# Lat_Num and Lon_Num must have values that imply an integer number of delta_lat and delta_lon degrees in each step
Service_Area_Lat_Min=20
Service_Area_Lat_Max=70
Service_Area_Lat_Num=5
Service_Area_Lon_Min=-40
Service_Area_Lon_Max=40
Service_Area_Lon_Num=5
#
# Start time in the simulation. Format: YYYY/MM/DD-HH:MM:SS.
startTime=2012/06/11-12:01:00
#
# Number of epochs in the simulation.
Number_Of_Epochs=86400
#
# Constellation TM. 0: No Constellation TM  1: Constellation TM
Constellation_TM=0
#
# Initial Epoch of Loss of Constellation TM
```

```
Initial_Epoch_Loss_TM_Constellation=3000
#
# Final Epoch of Loss of Constellation TM
Final_Epoch_Loss_TM_Constellation=3001
#
# Costellation for TM. 1: GPS 2: Galileo 3: Glonass 4: Compass
Constellation_for_TM=1
#
# TM_Satellites. 1 if TM Satellites or 0 if not
TM_Satellites=0
#
# Initial Epoch of loss for SV TM.
Initial_Epoch_Loss_TM_Sat=3000
#
# Final Epoch of loss for SV TM.
Final_Epoch_Loss_TM_Sat=3001
#
# PRN for TM. From 1 to 210
PRN_for_TM=5
#
# Operational Mode. 1 for simulated L1 NOF. 2 for Real L1 NOF
Operational_Mode=1
#
# Frequency. 2 for L1/L5. 5 for L5-only
Frequency=2
# Configuration of number of IGPs.
# This parameter is used to configure the number of MT18 and MT26 to be broadcast in L5-only back
# mode
# In the case of L1/L5 execution, this parameter does not apply (not ionosphere messages are #
broadcast)
# ECAC = 1 ECAC+AFI = 2 ECAC+ENP = 3
# ECAC: 5 bands 22 blocks. ECAC+AFI: 5 bands 41 blocks. ECAC+ENP: 5 bands 30 blocks
IGP_Zone=1
#
# Use MT_6_1/2 instead of MT_C if more than 7 DFREs change at once
Use_MT_6_1_MT_6_2_Asynchronous=1
#
# Update rates of specific messages, in seconds.Enhanced ICD
ENHANCED_ICD_MT_PRN_Mask_Update_Rate=120
ENHANCED_ICD_MT_6_1/2_Update_Rate=300
ENHANCED_ICD_MT_Degradation_Params_Update_Rate=120
ENHANCED_ICD_MT_SBAS_Almanac_Update_Rate=120
ENHANCED_ICD_MT_SBAS_Ephemeris_Update_Rate=60
ENHANCED_ICD_MT_ISC_Message_Update_Rate=300
ENHANCED_ICD_MT_12_Update_Rate=300
# Update rates of ENHANCED ICD MT18 and MT26. These messages are only broadcast in L5-only
```

```
#backup mode.
ENHANCED_ICD_MT_18_Update_Rate=300
ENHANCED_ICD_MT_26_Update_Rate=300
# Update rates of SV Correction Messages, in seconds.ENHANCED ICD
# In order to fulfill 6 seconds of UI in Integrity Message it is recommended to set MT SV Corrections
UI as function of SVs in Mask not larger than:
ENHANCED_ICD_MT_SV_Correction_Update_Rate=50
# MT_C with integrity flags and 7 DFREIs UI
ENHANCED_ICD_MT_Integrity_Flag_Update_Rate=6
#
# Timeout of specific messages in ENHANCED ICD, in seconds.
# (For information: these parameters do not affect the message sequence.)
ENHANCED_ICD_MT_PRN_Mask_Timeout=600
ENHANCED_ICD_MT_6_1/2_Timeout=600
ENHANCED_ICD_MT_Degradation_Params_Timeout=600
ENHANCED_ICD_MT_SBAS_Ephemeris_Timeout=120
ENHANCED_ICD_MT_ISC_Message_Timeout=600
ENHANCED_ICD_MT_12_Timeout=86400
# Timeouts of ENHANCED ICD MT18 and MT26. These messages are only broadcast in L5-only backup
#mode.
ENHANCED_ICD_MT_18_Timeout=1200
ENHANCED_ICD_MT_26_Timeout=600
# Timeout of SV Correction Messages, in seconds.Enhanced ICD
# In order to refresh in less than 6 seconds the Integrity Message it is recommended to set MT SV
#Corrections TO in following ranges as function of SVs in Mask:
ENHANCED_ICD_MT_SV_Correction_Timeout=100
# MT_C with integrity flags and 7 DFREIs TO
ENHANCED_ICD_MT_Integrity_Flag_Timeout=12
#
# Forced mask:
#     Configured as a line of type:
#          Forced_Mask=prn1 prn2 prn3 ...
Forced_Mask=1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
# seed for the pseudo-random number generator used for both Alert Generator of MSG module and
#Gauss-Markov processes of NOFG module
seed=2
```

In the following table, the different *ref_cond_cfg.dat* parameters are described.

Table 7-2: *ref_cond_cfg.dat* parameters

Parameter	Description	Range
Service_Area_Lat_Min	Configuration of the service area, as a rectangle in latitude and longitude.	[-90,90]
Service_Area_Lat_Max		[-90,90]
Service_Area_Lat_Num		Integer > 0
Service_Area_Lon_Min		[-180,180]

Parameter	Description	Range
Service_Area_Lon_Max	"Min" and "Max" are set in degrees. "Num" refers to the number of points in the corresponding dimension to define a grid of users. The grid of users is used to determine the visibility of the satellites. It is noted that the number of messages to provide ionosphere mask or ionosphere corrections is to be configured separately with IGP_Zone parameter.	[-180,180]
Service_Area_Lon_Num		Integer > 0
startTime	Start Time of the simulation (GPS Time in Calendar format). Format: YYYY/MM/DD-HH:MM:SS Example: startTime=2012/06/11-12:01:00	N/A
Number_Of_Epochs	Number of epochs to simulate.	[1,86400]
Constellation_TM	Boolean parameter indicating if a constellation transition mode is to be simulated	{0,1}
Initial_Epoch_Loss_TM_Constellation	Epoch where a constellation is lost	[1,Number_Of_Epochs]
Final_Epoch_Loss_TM_Constellation	Epoch where a constellation is recovered	[Initial_Epoch_Loss_TM_Constellation, Number_Of_Epochs]
Constellation_for_TM	Constellation for which the TM applies	1: GPS 2: Galileo 3: Glonass 4: Compass
TM_Satellites	Boolean parameter indicating if a satellite transition	{0,1}

Parameter	Description	Range
	mode is to be simulated	
Initial_Epoch_Loss_TM_Sat	Epoch where some SVs are lost	[1,Number_Of_Epochs]
Final_Epoch_Loss_TM_Sat	Epoch where some SVs are recovered	[Initial_Epoch_Loss_TM_Sat, Number_Of_Epochs]
PRN_for_TM	SV for which the TM applies. It can be repeated to simulate various simultaneous SVs transition modes	Integer: 1 to 210
Operational_Mode	Operational Mode for generating L1 NOF. If Operational Mode 2 is selected, several files with L1 NOF data should feed the SPP	1: Simulated L1 NOF 2: Real L1 NOF
Frequency	SBAS Frequency Mode. This determines the use or not of MT18, MT26 and MT30	2: L1/L5 5: L5-only back-up mode
IGP_Zone	Zone of IGPs. This is also related with the service area. It determines the number of submessages of MT18 and MT26.	1: ECAC 2: ECAC+AFI 3: ECAC+ENP
Use_MT_6_1_MT_6_2_Asynchronous	Boolean parameter indicating if SBAS L1/L5 SBAS ICD MT_6_1/2 should be used in asynchronous events. If it is set to 1, if in a certain epoch where MT_C is to be broadcast it happens that more than 7 DFREs are changing at once, MT_C is substituted by MT_6_1 followed by MT_6_2 if more than 51 SVs in	{0,1}

Parameter	Description	Range
	mask. If it is set to 0, MT_C is not substituted and the SVs with changing DFRECI whose DFREI cannot be allocated in MT_C are set to Not Monitored.	
ENHANCED_ICD_MT_PRN_Mask_Update_Rate	Update Intervals of the different MTs in Enhanced ICD. They should be configured carefully to have an adequate bandwidth usage.	Positive Integer
ENHANCED_ICD_MT_6_1/2_Update_Rate		Positive Integer
ENHANCED_ICD_MT_Degradation_Params_Update_Rate		Positive Integer
ENHANCED_ICD_MT_SBAS_Almanac_Update_Rate		Positive Integer
ENHANCED_ICD_MT_SBAS_Ephemeris_Update_Rate		Positive Integer
ENHANCED_ICD_MT_ISC_Message_Update_Rate		Positive Integer
ENHANCED_ICD_MT_12_Update_Rate		Positive Integer
ENHANCED_ICD_MT_18_Update_Rate		Positive Integer
ENHANCED_ICD_MT_26_Update_Rate		Positive Integer
ENHANCED_ICD_MT_SV_Correction_Update_Rate		Positive Integer
ENHANCED_ICD_MT_Integrity_Flag_Update_Rate		Positive Integer
ENHANCED_ICD_MT_PRN_Mask_Timeout	Time-outs of the different MTs in Enhanced ICD.	Positive Integer
ENHANCED_ICD_MT_6_1/2_Timeout		Positive Integer
ENHANCED_ICD_MT_Degradation_Params_Timeout		Positive Integer
ENHANCED_ICD_MT_SBAS_Ephemeris_Timeout		Positive Integer
ENHANCED_ICD_MT_ISC_Message_Timeout		Positive Integer
ENHANCED_ICD_MT_12_Timeout		Positive Integer
ENHANCED_ICD_MT_18_Timeout		Positive Integer
ENHANCED_ICD_MT_26_Timeout		Positive Integer
ENHANCED_ICD_MT_SV_Correction_Timeout		Positive Integer
ENHANCED_ICD_MT_Integrity_Flag_Timeout		Positive Integer
Forced_Mask	PRN Mask.	Minimum number of SVs: 16 Maximum number of SVs: 91 Example:

Parameter	Description	Range
		Forced_Mask=1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
seed	Seed for the pseudo-random number generator used for both Alert Generator of MSG module and Gauss-Markov processes of NOFG module	Positive Integer

7.2. NAVIGATION RINEX READER

The '*NavRinexReader'* module is the one in charge of processing the navigation RINEX files and generating the SatPos.txt file, which contains the positions of the configured satellites in each epoch.

This module is common to both the SPP and the RP, as they need to know the position of the satellites.

7.2.1. INVOCATION

The invocation information is shown using the '*help'* command:

./NavRinexReader –h

Table 7-3: *NavRinexReader* usage

```
NAME
    ./NavRinexReader -  PROSBAS Navigation RINEX Reader

SYNOPSIS
    ./NavRinexReader [-h] [-c ref_cond_cfg.dat] [-i input_path] [-o output_path] [-d]

DESCRIPTION
Generates the file 'SatPos.txt' with the position of every configured satellite, per epoch.
    -h              Print usage in stdout and exit.
    -c ref_cond_cfg.dat    Set configuration file. Default files is ref_cond_cfg.dat.
    -i input_path   Set input path containing the RINEX files. Default path is ./.
    -o output_path  Set output path where SatPos.txt file will be created. Default path is ./.
    -d              Print default configuration file ref_cond_cfg.txt in stdout and exit.

COPYRIGHT
    GMV & European Commission 2013, GMV & European Commission property; all rights reserved.
```

7.2.2. CONFIGURATION

The '*NavRinexReader'* uses the reference conditions configuration file, which is common to most of the PROSBAS Prototype modules.

The satellites and epochs for which the positions should be computed are read from the reference conditions file.

It is important to point out that the RINEX Navigation files that feed NavRinexReader module should be in accordance with the parameters in *ref_cond_cfg.dat* file startTime and Number_Of_Epochs, that is, NavRinexReader module should be feed with navigation files corresponding to the epochs to simulate.

A comprehensive description of the reference conditions file is given in Section 7.1.1.

7.2.3. INPUT FORMAT

The '*NavRinexReader'* module accepts navigation RINEX files as inputs. There are several constrains for the RINEX files:

- Rinex version: 2 or 3
- Single constellation file name: **brdc**DDDF.YYC
    - DDD is the day of the year
    - F is the file sequence number within day (use 0 for daily RINEX files)
    - YY is the year
    - C is the constellation, where:
        - 'n' is GPS
        - 'g' is GLONASS
        - 'x' is GALILEO
        - 'l' is COMPASS
- Multiple constellation name: **brmix**DDDG.YYC

The module computes the filenames according to the dates configured in the reference conditions file, and look for the RINEX files in the input folder. It will always try to open first the single constellation file (brdc) and, in case it does not exist, it will try to open the multiple constellation file (brmix). If none of them exist for any of the configured constellations, the module execution fails.

GPS RINEX Navigation files can be downloaded from the following FTP:

ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily/

For example, to download the GPS Navigation of 2th January 2012, one should download the following file: ftp://cddis.gsfc.nasa.gov/pub/gps/data/daily/2012/brdc/brdc0020.12n.Z

GLONASS RINEX Navigation files can be downloaded from the following FTP:

ftp://ftp.glonass-ianc.rsa.ru:/MCC/BRDC

For example, to download GLONASS Navigation of 2th January 2012, one should download the following file: ftp://ftp.glonass-ianc.rsa.ru:/MCC/BRDC/2012/MCCK0020.12G

The GALILEO and COMPASS RINEX file contents have been assumed to be exactly the same as the GPS ones, and so is the satellite position computation. RINEX Navigation files for Galileo and Compass have been generated with GMV tools to be able to run PROSBAS Prototype with 4 constellations.

7.2.4. OUTPUT FORMAT

The '*NavRinexModule'* has only one output: the *SatPos.txt* file. This contains an epoch-by-epoch record of the satellites' information provided in the navigation RINEX files.

This file has several columns, which are described in Table 7-4.

Table 7-4: SatPos.txt file format

Column	Description	Range
1: GPS_TIME	The GPS_TIME epoch. Integer value. Unit: seconds.	[-inf, inf]
2: PRN	The satellite's PRN Integer value. Unit: N/A	[1, 210]
3: Status	The status of the record. If any error occurs, or the satellite is unhealthy, the data for that epoch is invalidated. Boolean.	0: Invalid 1: Valid
4: IODE	Issue Of Data: Ephemeris. The IODE with which the data has been computed. Integer value.	[0, inf]
5: X_WGS84	Satellite's X coordinate in WGS84 reference frame. Floating point value. Unit: meters	[-inf, inf]
6: Y_WGS84	Satellite's Y coordinate in WGS84 reference frame. Floating point value. Unit: meters	[-inf, inf]
7: Z_WGS84	Satellite's Z coordinate in WGS84 reference frame. Floating point value. Unit: meters	[-inf, inf]
6: Clock	Satellite's clock correction. Floating point value. Unit: meters	[-inf, inf]

7.3. MSG, NOFG AND SME MODULES

The MSG, NOFG and SME modules have been integrated in a single module. It involves the generation of the message sequence, the generation of the L5 NOF and the codification of that information to generate the SBAS messages that will be processed in the RP.

7.3.1. INVOCATION

Although PROSBAS Prototype is expected to be run entirely with PROSBAS driver, it is also possible to execute the different sub-modules separately.

It is important to point out that the order of execution of the different SPP modules is the following:

- NavRinexReader
- MSG & NOFG & SME
- MLS
- MSA
- Receiver Prototype (see [RD.3])

To execute the MSG &NOFG & SME modules alone, the user should go to the directory where the executable is placed and type the following instruction in Linux terminal:

./SME –c config_path/ –n SPP_Input_path/ –o output_SME_path/ -s output_MSG_path

If the –c, -n, -o and –s options are not specified, the default values are used. These default values are shown in the –h option.

To receive detailed instructions related with the usage of the NOFG & SME sub-module, the user should be placed in the folder where the executable is placed and type:

./SME –h

This command prints in the screen the following information:

Table 7-5: *SME* usage

```
NAME
    ./SME -  PROSBAS SBASMessageEmulator

SYNOPSIS
    ./SME [-h] [-c configuration_path] [-s sequence_file_path] [-n nofg_input_path] [-o output_path]
[-d file_ID]

DESCRIPTION
Generates the NOF and encodes the SBAS Messages in the Logbook format.
     -h                    Print usage in stdout and exit.
     -c configuration_path   Set configuration files path. Default path is ./.
      -s sequence_file_path   Set output file path for the SBAS Message Sequence. Default path is
./msg_out.dat
    -n nofg_input_path      Set path containing the NOFG input data. Default path is ./.
     -o output_path          Set output path where the Logbook files will be generated. Default path is
./
     -d file_ID              Print in stdout the default configuration file indicated by file_ID parameter
and exit.
                              - file_ID 0: Reference conditions file (ref_cond_cfg.dat)
                              - file_ID 1: Degradations parameters file (deg_params_cfg.dat)
                              - file_ID 2: File of configuration parameters of the generation of the NOF
in L1 (gen_L1_NOF_DFRE_cfg.dat)
                              - file_ID 3: Message sequence generator configuration file (msg_cfg.dat)
COPYRIGHT
        GMV & European Commission 2013, GMV & European Commission property; all rights
reserved.
```

It is important to remark that the configuration and input files should be placed in the adequate folders.

7.3.2. CONFIGURATION

The MSG, NOFG & SME module uses five configuration files:

- the common configuration file to all the prototype: *ref_cond_cfg.dat* that is described in Section 7.1.1.
- the MSG configuration file *msg_cfg.dat*
- the Degradations parameters configuration file: *deg_params_cfg.dat*
- File of configuration parameters of the generation of the NOF in L1: *gen_L1_NOf_DFRE_cfg.dat*
- The stations positions file: *stations.cfg*

The first one is common to all the Service Provider Prototype sub-modules. Please refer to section 7.1.

7.3.2.1. MSG Configuration

msg_cfg.dat file is used to configure MSG module to generate the message sequence.
In the following it is shown an example of *msg_cfg.dat* file.

Table 7-6: example of *msg_cfg.dat* file

```
# MSG Configuration
#
#
# Number Of GEOs
Number_Of_GEOs=4
#
# Scheduler configuration.
# Use 0 for false, 1 for true.
# ISC Messages can only be active in L5-only back up mode.
Use_ISC_Messages=0
#
# Configuration for UDRE Alert Generation.
Offset_For_User_Event_Epoch=2
#    Add a line per each PRN to use
#      PRN_For_Udre_Alerts=prn
# Random UDRE alert generation
Generate_Random_Udre_Alerts=0
#It is recommended to set a probability of alert smaller than 0.01
Average_Ratio_Of_Random_Udre_Alerts=0.001
# Periodic UDRE alert generation
Generate_Periodic_Udre_Alerts=0
Module_For_Periodic_Udre_Alerts=1000
Offset_For_Periodic_Udre_Alerts=0
```

In the following table, the different parameters contained in msg_cfg.dat file are described.

Table 7-7: *msg_cfg.dat* parameters

Parameter	Description	Range
Number_Of_GEOs	Number of GEOs in the SBAS system. This determines the number of MT_G and MT_E	Integer larger than 0
Use_ISC_Messages	Boolean parameter that determines if MT30 is to be used in the case L5-only back up mode (configured with Frequency parameter in *ref_cond_cfg.dat* file)	{0,1}
PRN_For_Udre_Alerts	Configuration of alert generation. In order to generate alerts, either Generate_Random_Udre_Alerts or Generate_Periodic_Udre_Alerts must be set to	[1,210]
Offset_For_User_Event_Epoch		Integer >=0

Parameter	Description	Range
Generate_Random_Udre_Alerts	true. If random alerts are set, there will be alerts according to the seed configured, as an integer, seed (in ref_cond_cfg.dat file), and with the probability given by Average_Ratio_Of_Random_Udre_Alerts (float). Periodic alerts are generated with the offset and module given by the parameters Module_For_Periodic_Udre_Alerts and Offset_For_Periodic_Udre_Alerts, both integer. Alerts will be generated for all the satellites provided with the lines PRN_For_Udre_Alerts. Satellites that are not in the mask when the alert is to be provided will be ignored. The parameter Offset_For_User_Event_Epoch is used as an additional offset to account for time-to-alert, so that, as regards the analysis, the alert event will be considered to have taken place the given number of seconds before the SBAS system took notice of it.	{0,1}
Average_Ratio_Of_Random_Udre_Alerts		[0,1]
Generate_Periodic_Udre_Alerts		{0,1}
Module_For_Periodic_Udre_Alerts		>0
Offset_For_Periodic_Udre_Alerts		Integer >=0

7.3.2.2. Configuration of the degradation parameters

The second configuration file, *deg_params_cfg.dat*, contains all the configuration parameters that are needed to define the degradation from L1 to L5 and also the degradation factors associated to the Degradation Parameters SBAS Message (MT-F).

Configuration values are assigned to parameters using a '=' symbol. Note that no blanks are allowed around the '=' symbol, nor at the end of each line. Comment lines are allowed, starting with the character '#'.

The file contents are detailed in Table 7-8.

Table 7-8: deg_params_cfg.dat file format

Parameter	Description	Range [1]
deltaT_IP	Half the length of the forward time interval used to estimate the UDRE border effect in UDRE degradation from L1 to L5. Integer value. Unit: seconds	[0, ∞)
estimation_arc_Delta_FC	Length of the backward time interval used to estimate the clock in UDRE degradation from L1 to L5. Integer value. Unit: seconds	[0, ∞)
slope_Delta_FC	First order coefficient of the estimation of the clock used to degrade the UDRE from L1 to L5. Floating point value. Unit: meter^2/sec	[0, ∞)

[1] Some parameters included in this table are the same ones that are encoded in the MT-F SBAS Message. For this reason, in some cases it is specified a precision value which corresponds to the Scale factor (LSB). In case the user configures a value not allowed by that precision number, the tool will internally trunk the parameter and issue an alert message to the user.

Parameter	Description	Range [1]
timeout_interval	Time-out interval for the satellite correction and SBAS ephemeris messages. Integer value. Units: seconds	[60, 432] Scale: 6
i_corr	Update time interval between the satellite correction and ephemeris messages. Integer value. Units: seconds	[30, 216] Scale: 6
c_corr	Bump in the confidence value to older messages. Floating point value. Units: meters	[0, 2.55] Scale: 0.01
r_corr	Correction rate in the confidence value to older messages. Floating point value. Unit: mm/sec	[0, 51] Scale: 0.2
a_corr	Acceleration uncertainty in the confidence value to older messages. Floating point value. Unit: mm/sec^2	[0, 5.1] Scale: 0.02
sigma_dfrei_0	Greatest σ_{DFRE} value for DFREI=0 Floating point value. Unit: meters	[0.125, 1.0625] Scale: 0.0625
sigma_dfrei_1	Greatest σ_{DFRE} value for DFREI=1 Floating point value. Unit: meters	[0.25, 2.125] Scale: 0.125
sigma_dfrei_2	Greatest σ_{DFRE} value for DFREI=2 Floating point value. Unit: meters	[0.375, 2.25] Scale: 0.125
sigma_dfrei_3	Greatest σ_{DFRE} value for DFREI=3 Floating point value. Unit: meters	[0.5, 2.375] Scale: 0.125
sigma_dfrei_4	Greatest σ_{DFRE} value for DFREI=4 Floating point value. Unit: meters	[0.625, 2.5] Scale: 0.125
sigma_dfrei_5	Greatest σ_{DFRE} value for DFREI=5 Floating point value. Unit: meters	[0.75, 4.5] Scale: 0.25
sigma_dfrei_6	Greatest σ_{DFRE} value for DFREI=6 Floating point value. Unit: meters	[1, 4.75] Scale: 0.25
sigma_dfrei_7	Greatest σ_{DFRE} value for DFREI=7 Floating point value. Unit: meters	[1.25, 5] Scale: 0.25
sigma_dfrei_8	Greatest σ_{DFRE} value for DFREI=8 Floating point value. Unit: meters	[1.5, 5.25] Scale: 0.25
sigma_dfrei_9	Greatest σ_{DFRE} value for DFREI=9 Floating point value. Unit: meters	[1.75, 5.5] Scale: 0.25
sigma_dfrei_10	Greatest σ_{DFRE} value for DFREI=10 Floating point value. Unit: meters	[2, 9.5] Scale: 0.5
sigma_dfrei_11	Greatest σ_{DFRE} value for DFREI=11 Floating point value. Unit: meters	[2.5, 10] Scale: 0.5
sigma_dfrei_12	Greatest σ_{DFRE} value for DFREI=12 Floating point value. Unit: meters	[3, 18] Scale: 1

Parameter	Description	Range [1]
sigma_dfrei_13	Greatest σ_{DFRE} value for DFREI=13 Floating point value. Unit: meters	[4, 49] Scale: 3
rss_dfc	Flag that determines how the elements of the bounding of the dual frequency are combined. Integer value. Unitless	0 or 1
cer	Degradation parameter for En Route through NPA. Floating point value. Unit: meters	[0, 31.5] Scale: 0.5
ccovariance	Parameter to compensate quantization in the covariance matrix. Floating point value. Unitless	[0, 12.7] Scale: 0.1
ciono_step	Bound of the difference between successive ionospheric grid delay values. Floating point value. Unit: meters	[0, 1.023] Scale: 0.001
iiono	Minimum update interval for ionospheric corrections. Integer value. Unit: seconds	[0, 511] Scale: 1
ciono_ramp	Rate of change of the ionospheric corrections. Floating point value. Unit: m/s	[0, 0.005115] Scale: 0.000005
rss_iono	Root-sum-square flag for ionospheric variance. Integer value. Unitless	0 or 1

A sample *deg_params_cfg.dat* is shown below:

Table 7-9: example of *deg_params_cfg.dat*

```
########################################
# UDRE degradation from L1 to L5
########################################
DeltaT_IP=6
estimation_arc_Delta_FC=5
# slope to estimate Delta_FC measured in m*m/s:
slope_Delta_FC=0.1

########################################
# Degradation factors and parameters
########################################
timeout_interval=240
i_corr=120
c_corr=0
r_corr=0
a_corr=0
# upper bound of the standard deviation of the
# UDRE in L5 for each index measured in m:
```

```
sigma_dfrei_0=0.625
sigma_dfrei_1=1.25
sigma_dfrei_2=1.375
sigma_dfrei_3=1.5
sigma_dfrei_4=1.625
sigma_dfrei_5=2.75
sigma_dfrei_6=3
sigma_dfrei_7=3.25
sigma_dfrei_8=3.5
sigma_dfrei_9=3.75
sigma_dfrei_10=6
sigma_dfrei_11=6.5
sigma_dfrei_12=11
sigma_dfrei_13=28
rss_dfc=0
cer=0
ccovariance=0.5
ciono_step=0
iiono=0
ciono_ramp=0
rss_iono=0
```

7.3.2.3. Configuration of the generation of the NOF in L1

The third file, *gen_L1_NOf_DFRE_cfg.dat*, contains all the configuration parameters that are needed to generate the NOF in L1.

Configuration values are assigned to parameters using a '=' symbol. Note that no blanks are allowed around the '=' symbol, nor at the end of each line. Comment lines are allowed, starting with the character '#'.

The file contents are detailed in Table 7-10.

Table 7-10: *gen_L1_NOf_DFRE_cfg.dat* file format

Parameter	Description	Range
fast_correction_k	Parameter k associated to the Gauss-Markov process used in the generation of the fast corrections. Floating point value. Unit: unitless	(0, 1)
fast_correction_mean	Mean associated to the Gauss-Markov process used in the generation of the fast corrections. Floating point value. Unit: meters	[0, ∞)
fast_correction_sigma	Standard deviation associated to the Gauss-Markov process used in the generation of the fast corrections. Floating point value. Unit: meters	[0, ∞)

Parameter	Description	Range
minimum_stations_not_monitored	Minimum of the interval of visible stations from a SV for which that SV is considered not monitored. Integer value. Unit: stations	[0, ∞)
maximum_stations_not_monitored	Maximum of the interval of visible stations from a SV for which that SV is considered not monitored. Integer value. Unit: stations	[0, ∞)
minimum_stations [2]	Minimum of the interval of visible stations from a SV for which its UDRE is simulated with the following Gauss-Markov process. Integer value. Unit: stations	[0, ∞)
maximum_stations[2]	Maximum of the interval of visible stations from a SV for which its UDRE is simulated with the following Gauss-Markov process. Integer value. Unit: stations	[0, ∞)
udre_k[2]	Parameter k associated to the Gauss-Markov process used in the generation of the UDRE for the previously defined interval. Floating point value. Unit: unitless	(0, 1)
udre_mean[2]	Mean associated to the Gauss-Markov process used in the generation of the UDRE for the previously defined interval. Floating point value. Unit: **meter^2**	[0, ∞)
udre_sigma[2]	Standard deviation associated to the Gauss-Markov process used in the generation of the UDRE for the previously defined interval. Floating point value. Unit: **meter^2**	[0, ∞)
udre_k_rest	Parameter k associated to the Gauss-Markov process used in the generation of the UDRE for the rest of the cases. Floating point value. Unit: unitless	(0, 1)
udre_mean_rest	Mean associated to the Gauss-Markov process used in the generation of the UDRE for the rest of the cases. Floating point value. Unit: **meter^2**	[0, ∞)

[2] The blocks of parameters

- minimum_stations, maximum_stations, udre_k, udre_mean, udre_sigma;
- minimum_ipps, maximum_ipps, give_k, give_mean, give_sigma

can be repeated as many times as necessary to define the required intervals.

Parameter	Description	Range
udre_sigma_rest	Standard deviation associated to the Gauss-Markov process used in the generation of the UDRE for the rest of the cases. Floating point value. Unit: **meter^2**	[0, ∞)
minimum_ipps_not_monitored	Minimum of the interval of IPPs close to an IGP for which that IGP is considered not monitored. Integer value. Unit: IPPs	[0, ∞)
maximum_ipps_not_monitored	Maximum of the interval of IPPs close to an IGP for which that IGP is considered not monitored. Integer value. Unit: IPPs	[0, ∞)
minimum_ipps^2	Minimum of the interval of IPPs close to an IGP for which its GIVE is simulated with the following Gauss-Markov process. Integer value. Unit: IPPs	[0, ∞)
maximum_ipps2	Maximum of the interval of IPPs close to an IGP for which its GIVE is simulated with the following Gauss-Markov process. Integer value. Unit: IPPs	[0, ∞)
give_k^2	Parameter k associated to the Gauss-Markov process used in the generation of the GIVE for the previously defined interval. Floating point value. Unit: unitless	(0, 1)
give_mean^2	Mean associated to the Gauss-Markov process used in the generation of the GIVE for the previously defined interval. Floating point value. Unit: meter^2	[0, ∞)
give_sigma^2	Standard deviation associated to the Gauss-Markov process used in the generation of the GIVE for the previously defined interval. Floating point value. Unit: meter^2	[0, ∞)
give_k_rest	Parameter k associated to the Gauss-Markov process used in the generation of the GIVE for the rest of the cases. Floating point value. Unit: unitless	(0, 1)
give_mean_rest	Mean associated to the Gauss-Markov process used in the generation of the GIVE for the rest of the cases. Floating point value. Unit: meter^2	[0, ∞)

Parameter	Description	Range
give_sigma_rest	Standard deviation associated to the Gauss-Markov process used in the generation of the GIVE for the rest of the cases. Floating point value. Unit: meter^2	[0, ∞)
clock_correction_k	Parameter k associated to the Gauss-Markov process used in the generation of the slow corrections. Floating point value. Unit: unitless	(0, 1)
clock_correction_mean	Mean associated to the Gauss-Markov process used in the generation of the slow corrections. Floating point value. Unit: meters	[0, ∞)
clock_correcyion_sigma	Standard deviation associated to the Gauss-Markov process used in the generation of the slow corrections. Floating point value. Unit: meters	[0, ∞)
orbit_correction_k	Parameter k associated to the Gauss-Markov process used in the generation of the orbit corrections. Floating point value. Unit: unitless	(0, 1)
orbit_correction_mean	Mean associated to the Gauss-Markov process used in the generation of the orbit corrections. Floating point value. Unit: meters	[0, ∞)
orbit_correction_sigma	Standard deviation associated to the Gauss-Markov process used in the generation of the orbit corrections. Floating point value. Unit: meters	[0, ∞)
iodp_mask	Initial value for the IODP in operational mode 1. Integer value. Unit: unitless	[0, 3]
fast_correction_initial	Initial value for the fast corrections in operational mode 1. Floating point value. Unit: meters	[0, ∞)
brrc	Initial value for B_{rrc} in operational mode 1. Floating point value. Unit: meters	[0, 2.046]
cltc_lsb	Initial value for C_{ltc_lsb} in operational mode 1. Floating point value. Unit: meters	[0, 2.046]
cltc_v1	Initial value for C_{ltc_v1} in operational mode 1. Floating point value. Unit:	[0, 0.05115]

Parameter	Description	Range
	meter/sec	
iltc_v1	Initial value for I_{ltc_v1} in operational mode 1. Integer value. Unit: seconds	[0, 511]
cltc_v0	Initial value for C_{ltc_v0} in operational mode 1. Floating point value. Unit: meters	[0, 2.046]
iltc_v0	Initial value for I_{ltc_v0} in operational mode 1. Integer value. Unit: seconds	[0, 511]
cgeo_lsb	Initial value for C_{geo_lsb} in operational mode 1. Floating point value. Unit: meters	[0, 0.5115]
cgeo_v	Initial value for C_{geo_v} in operational mode 1. Floating point value. Unit: meter/sec	[0, 0.05115]
igeo	Initial value for I_{geo} in operational mode 1. Integer value. Unit: seconds	[0, 511]
a0wnt	Initial value for A_{0WNT} in operational mode 1. Floating point value. Unit: sec/sec	$\pm 7.45\times10^{-9}$
a1wnt	Initial value for A_{1WNT} in operational mode 1. Floating point value. Unit: seconds	±1
t0t	Initial value for T_{0t} in operational mode 1. Integer value. Unit: seconds	[0, 602112]
wnt	Initial value for WN_t in operational mode 1. Integer value. Unit: weeks	[0, 255]
dtls	Initial value for Δt_{LS} in operational mode 1. Integer value. Unit: seconds	±127
wnlsf	Initial value for WN_{LSF} in operational mode 1. Integer value. Unit: weeks	[0, 255]
dn	Initial value for DN in operational mode 1. Integer value. Unit: days	[1, 7]
dtlsf	Initial value for Δt_{LSF} in operational mode 1. Integer value. Unit: seconds	±127

Parameter	Description	Range
utc_id	Initial value for UTC Standard Identifier in operational mode 1. Integer value. Unit: unitless	[0, 7]
gps_tow	Initial value for GPS time of week in operational mode 1. Integer value. Unit: seconds	[0, 604799]
gps_wn	Initial value for GPS week number in operational mode 1. Integer value. Unit: weeks	[0, 1023]
glo_ind	Initial value for GLONASS indicator in operational mode 1. Integer value. Unit: unitless	[0, 1]
ent_glo_offset	Initial value for GLONASS time offset in operational mode 1. Floating point value. Unit: seconds	$\pm 2^{-8}$
gal_ind	Initial value for Galileo indicator in operational mode 1. Integer value. Unit: unitless	[0, 1]
ent_gal_offset	Initial value for Galileo time offset in operational mode 1. Floating point value. Unit: seconds	$\pm 2^{-8}$
compass_ind	Initial value for Compass indicator in operational mode 1. Integer value. Unit: unitless	[0, 1]
ent_compass_offset	Initial value for Compass time offset in operational mode 1. Floating point value. Unit: seconds	$\pm 2^{-8}$
iodi18	Initial value for the IODI if the IGP mask in operational mode 1. Integer value. Unit: unitless	[0, 3]
orbit_correction_initial	Initial value for the orbit corrections in operational mode 1 formatted as δx δy δz. List of floating point values. Unit: meters	[0, ∞) for the three orbit corrections
clock_correction_initial	Initial value for the slow corrections in operational mode 1. Floating point value. Unit: meters	[0, ∞)
iodi26	Initial value for the IODI associated to GIVD and GIVE in operational mode 1. Integer value. Unit: unitless	[0, 3]
givd_initial	Initial value for the IGP delay estimate (GIVD) in operational	[0, ∞)

Parameter	Description	Range
	mode 1. Floating point value. Unit: meters	
givei_initial	Initial value for the GIVE indicator in operational mode 1. Integer value. Unit: unitless	[0, 15]
scale_exponent	Initial value for the covariance scale exponent in operational mode 1. Integer value. Unit: unitless	[0, 7]
covariance_matrix	Initial value for the covariance matrix in operational mode 1 formatted as E_{11} E_{12} E_{13} E_{14} E_{22} E_{23} E_{24} E_{33} E_{34} E_{44}. List of integer values. Unit: unitless	[0, 511] on the diagonal and [-512, 511] elsewhere
debug_outputs	Flag to generate additional output information (1) or not (0). Integer value. Unit: unitless	[0, 1]

A sample *gen_L1_NOf_DFRE_cfg.dat* is shown below:

Table 7-11: example of *gen_L1_NOF_DFRE.cfg*

```
##################################################################
# Gauss-Markov parameters for fast corrections
##################################################################
fast_correction_k=0.2
fast_correction_mean=1
fast_correction_sigma=1

##################################################################
# Range where the UDREs are always "Not monitored"
##################################################################
minimum_stations_not_monitored=0
maximum_stations_not_monitored=0

##################################################################
# Gauss-Markov parameters for UDRE
##################################################################
minimum_stations=1
maximum_stations=1
udre_k=0.2
# mean and standard deviation of the UDRE in L1 measured in m*m:
udre_mean=100
udre_sigma=10
```

```
################################################################
# Gauss-Markov parameters for UDRE for the rest of the cases
################################################################
udre_k_rest=0.2
# mean and standard deviation of the UDRE in L1 measured in m*m:
udre_mean_rest=1
udre_sigma_rest=1

################################################################
# Range where the GIVEs are always "Not monitored"
################################################################
minimum_ipps_not_monitored=0
maximum_ipps_not_monitored=0

################################################################
# Gauss-Markov parameters for GIVE
################################################################
minimum_ipps=1
maximum_ipps=1
give_k=0.2
give_mean=100
give_sigma=10

################################################################
# Gauss-Markov parameters for GIVE for the rest of the cases
################################################################
give_k_rest=0.2
give_mean_rest=1
give_sigma_rest=1

################################################################
# Gauss-Markov parameters for clock corrections
################################################################
clock_correction_k=0.2
clock_correction_mean=1
clock_correction_sigma=1

################################################################
# Gauss-Markov parameters for orbit corrections
################################################################
orbit_correction_k=0.2
orbit_correction_mean=1
orbit_correction_sigma=1

################################################################
# Initial values
```

```
################################################################
iodp_mask=0
fast_correction_initial=0
brrc=0
cltc_lsb=0
cltc_v1=0
iltc_v1=0
cltc_v0=0
iltc_v0=1
cgeo_lsb=0
cgeo_v=0
igeo=0
a0wnt=0
a1wnt=0
t0t=0
wnt=0
dtls=0
wnlsf=0
dn=1
dtlsf=0
utc_id=0
gps_tow=0
gps_wn=0
glo_ind=0
ent_glo_offset=0
gal_ind=0
ent_gal_offset=0
compass_ind=0
ent_compass_offset=0
iodi18=0
orbit_correction_initial=0 0 0
clock_correction_initial=0
iodi26=0
givd_initial=0
givei_initial=15
scale_exponent=0
covariance_matrix=23 0 0 0 23 0 0 23 0 23

################################################################
# Debug outputs
################################################################
debug_outputs=0
```

7.3.2.4. Stations Positions

stations.cfg file contains the positions of the stations.

In the following table it is shown a sample of an example of *stations.cfg* file.

Table 7-12: Sample of an example of *stations.cfg*

```
[RIMS_CONFIGURATION]
NumberRIMS=36

[RIMS_1_CONFIGURATION]
RIMS_type=Q
RIMS_name=ALBA
RIMS_ID=1
Position_L1_X=3421468.383665045258
Position_L1_Y=596107.469931780943
Position_L1_Z=5331772.181737492792
DeltaPosition_L2_X=0.0
DeltaPosition_L2_Y=0.0
DeltaPosition_L2_Z=0.0

[RIMS_2_CONFIGURATION]
RIMS_type=P
RIMS_name=ACRA
RIMS_ID=2
Position_L1_X=4386269.978103646077
Position_L1_Y=-2393737.594014056958
Position_L1_Z=3950775.793667550199
DeltaPosition_L2_X=0.0
DeltaPosition_L2_Y=0.0
DeltaPosition_L2_Z=0.0

[RIMS_3_CONFIGURATION]
RIMS_type=P
RIMS_name=BRNA
RIMS_ID=3
Position_L1_X=3795125.322680848651
Position_L1_Y=914007.980817505741
Position_L1_Z=5027224.313796168193
DeltaPosition_L2_X=0.0
DeltaPosition_L2_Y=0.0
DeltaPosition_L2_Z=0.0
...
```

The only fields of stations.cfg file used by PROSBAS Prototype are the fields:

Position_L1_X

Position_L1_Y

Position_L1_Z

For each of the configured station, these three fields provide the XYZ coordinates in WGS84 and the units are meters.

7.3.3. INPUT FORMAT

In Operational Mode 2 (real L1 NOF as input), several files are required to simulate the messages from which the L1/L5 NOF is generated.

In what follows, a breakdown of the aforementioned files together with a detailed explanation of their format will be presented.

For further details as to the parameters included within these files, please refer to [RD.8].

7.3.3.1. Satellite mask file

The satellite mask has to be written in a file called *SV_Mask.dat*. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Epoch IODP nsat prn[1] … prn[nsat]

where

- *Epoch* is the epoch within the simulation;
- *IODP* is the satellite mask IODP;
- *nsat* is the number of satellites in the mask;
- *prn[1] … prn[nsat]* are the PRNs of the satellites of the mask (trailing zeros up to 51 values are allowed but not required).

Example:

0 0 33 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 23 24 25 26 27 28 29 30 31 32 120 124 126 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

7.3.3.2. Fast corrections and UDRE values files (optional)

The fast corrections and UDRE values have to be written in files called *MT2_5.dat* or *MT24_fast.dat*. The files are made up of lines limited by line feeds and, in turn, each line of such files is a list of space separated fields with the following format:

Epoch fast_type IODP IODF prn[0] fast_corr[0] UDREi[0] … prn[n] fast_corr[n] UDREi[n]

where

- *Epoch* is the epoch within the simulation;
- *fast_type* is the type of fast correction;
- *IODP* is IODP;
- *IODF* is IODF;
- *prn[0] … prn[n]* are the n+1 PRNs referred to by the fast corrections and UDREs;
- *fast_corr[0] … fast_corr[n]* are the n+1 fast corrections in meters;
- *UDREi[0] … UDREi[n]* are the n+1 UDRE indicators.

Example:

75 3 0 1 15 3.750000e-01 7 16 -8.750000e-01 6 17 0.000000e+00 14 18 -7.500000e-01 6 19 -5.000000e-01 6 20 0.000000e+00 14 21 -3.750000e-01 7 23 0.000000e+00 14 24 0.000000e+00 14 25 0.000000e+00 14 26 0.000000e+00 14 27 0.000000e+00 14 28 0.000000e+00 14

7.3.3.3. Integrity file (optional)

The UDRE values have to be written in a file called *MT6.dat*. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Epoch IODF2 IODF3 IODF4 IODF5 UDREi[1] … UDREi[nsat]

where

- *Epoch* is the epoch within the simulation;
- *IODF2* is $IODF_2$;
- *IODF3* is $IODF_3$;
- *IODF4* is $IODF_4$;
- *IODF5* is $IODF_5$;
- *UDREi[1] … UDREi[nsat]* are the nsat UDRE indicators referred to the satellites at the same positions of the satellite mask.

Example:

0 3 3 3 3 14 7 14 7 7 9 10 14 11 14 14 6 14 8 14 14 14 14 14 15 14 14 14 8 8 9 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14

7.3.3.4. Degradation parameters file

The degradation parameters have to be written in a file called *MT10.dat*. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Epoch Brrc Cltc_lsb Cltc_v1 Iltc_v1 Cltc_v0 Iltc_v0 Cgeo_lsb Cgeo_v Igeo Cer Ciono_step Iiono Ciono_ramp Rss_udre Rss_iono Ccov

where

- *Epoch* is the epoch within the simulation;
- *Brrc* is B_{rrc};
- *Cltc_lsb* is C_{ltc_lsb};
- *Cltc_v1* is C_{ltc_v1};
- *Iltc_v1* is I_{ltc_v1};
- *Cltc_v0* is C_{ltc_v0};
- *Iltc_v0* is I_{ltc_v0};
- *Cgeo_lsb* is C_{geo_lsb};
- *Cgeo_v* is C_{geo_v};
- *Igeo* is I_{geo};
- *Cer* is C_{er};
- *Ciono_step* is C_{iono_step};
- *Iiono* is I_{iono};
- *Ciono_ramp* is C_{iono_ramp};
- *Rss_udre* is RSS_{UDRE};
- *Rss_iono* is RSS_{iono};
- *Ccov* is $C_{covariance}$.

Example:

0 0.000000e+00 3.580000e-01 7.800000e-03 120 9.580000e-01 120 1.770000e-01 2.000000e-02 120 0.000000e+00 6.570000e-01 300 0.000000e+00 0 0 0.000000e+00

7.3.3.5. Time offset parameters file

The time offset parameters have to be written in a file called *MT12.dat*. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Epoch A0wnt A1wnt T0t Wnt Dtls Wnlsf Dn Dtlsf Utc_id Gps_tow Gps_wn Glo_ind Ent_glo_offset

where

- *Epoch* is the epoch within the simulation;
- *A0wnt* is A_{1WNT};
- *A1wnt* is A_{0WNT};

- *T0t* is T_{0t};
- *Wnt* is WN_t;
- *Dtls* is Δt_{LS};
- *Wnlsf* is WN_{LSF};
- *Dn* is DN;
- *Dtlsf* is Δt_{LSF};
- *Utc_id* is the UTC standard identifier;
- *Gps_tow* is the GPS time of week;
- *Gps_wn* is the GPS week number;
- *Glo_ind* is the GLONASS indicator;
- *Ent_glo_offset* is the GLONASS time offset.

Example:

0 0.000000e+00 0.000000e+00 0 0 0 0 0 0 7 0 0 0 0.000000e+00

7.3.3.6. Ionospheric mask file

The ionospheric mask has to be written in a file called *MT18.dat* if the simulation is run in L5-only mode. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Num_bands IODI Band nigp igp[1] … igp[nigp]

where

- *Num_bands* is the total number of IGP bands;
- *IODI* is IODI;
- *Band* is the number of the band whose mask is in this line;
- *nigp* is the total number of IGPs present in the mask for this band;
- *igp[1] … igp[nigp]* are the IGPs of this band that make up the mask (trailing zeros up to 201 values are allowed but not required).

Example:

0 5 0 9 80 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 184 185 186 187 188 189 190 0

7.3.3.7. Ephemeris and clock corrections files (optional)

The slowly varying satellite ephemeris and clock corrections have to be written in files called *MT25_VelCode0.dat* or *MT24_slow.dat*. The files are made up of lines limited by line feeds and, in turn, each line of such files is a list of space separated fields with the following format:

Epoch Vel_code IODP PRN IOD dX dY dZ daf0

where

- *Epoch* is the epoch within the simulation;
- *Vel_code* is the velocity code (it must be 0);
- *IODP* is the IODP of the satellite mask;
- *PRN* is the PRN of the satellite;
- *IOD* is the IODE (of the ephemeris);
- *dX,dY,dZ* are the three components of the orbit correction in meters;
- *daf0* is the clock slow correction in meters.

Example:

9 0 0 0 116 0.000000 0.000000 0.000000 1.814823

7.3.3.8. Ionospheric delay corrections file

The ionospheric delay corrections have to be written in a file called *MT26.dat* if the simulation is run in L5-only mode. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Epoch IODI Band Block IGP Givd Givei

where

- *Epoch* is the epoch within the simulation;
- *IODI* is IODI (of the ionospheric mask);
- *Band* is the band to which the corrections are referred;
- *Block* is the block within the band to which the corrections are referred;
- *IGP* is the IGP within the band and block to which the corrections are referred;
- *Givd* is the IGP delay estimate (GIVD);
- *Givei* is the GIVE indicator.

Example:

94 0 9 2 90 0.8750 13

7.3.3.9. Relative Covariance Matrix for clock and ephemeris file

The relative covariance matrix for clock and ephemeris has to be written in a file called *MT28.dat*. The file is made up of lines limited by line feeds and, in turn, each line of such file is a list of space separated fields with the following format:

Epoch IODP PRN Scale_Exponent E11 E12 E13 E14 E21 E22 E23 E24 E31 E32 E33 E34 E41 E42 E43 E44

where

- *Epoch* is the epoch within the simulation;
- *IODP* is the satellite mask IODP;
- *PRN* is the PRN of the satellite;
- *Scale_Exponent* is the scale exponent of the covariance matrix;
- *Eij (i, j=1, 2, 3, 4)* are the elements of the Cholesky decomposition of the covariance matrix (notice that this is an upper triangular matrix).

Example:

3 0 0 2 258.0 -196.0 -188.0 374.0 0.0 71.0 104.0 -93.0 0.0 0.0 62.0 -32.0 0.0 0.0 0.0 8.0

7.3.4. OUTPUT FORMAT

The MSG & NOFG & SME modules produce two kinds of outputs:

- Logbooks. Contains the SBAS Message Data to be sent to the RP Prototype.
- MSG output: *msg_out.dat* file with the message sequence.
- NOFG output information: *give_15, iode, stations_latitude_longitude_height, udre_l1, udre_l5, IODE_PRNx.png*[3], *NumSVvsTime_SPP.png*[3], *NumsSVvsTime_SPPvsRP.png*[3]*.*

7.3.4.1. Logbook files

The data files containing the SBAS messages are in Logbook format with one SBAS message per epoch encoded in hexadecimal format as can be seen in the following table.

The Logbook files are generated hourly with name "LogBook_DATE.dat" (DATE stands for the date in YYYY_MM_DD_HH format).

[3] Please note that these plots are generated by the Visualization module (please refer to [RD.3].) but using as inputs some of the files generated by the MSG & NOFG & SME module.

Table 7-13: Example of Logbook file with SBAS messages in hexadecimal format

```
2012/04/13 03:01:19. L=700
00000000000000000000000000000000000000000000000000000000000000000000009A06FFFFEFFC000000000000000
000000000000000000000000000002599768078000000000000000000000000000000000000000
2012/04/13 03:01:20. L=700
0000000000000000000000000000000000000000000000000000000000000000000000C6083C000000000000000000000
000002E170B85C0000000000000026EA874C07800000000000000000000000000000000000000
2012/04/13 03:01:21. L=700
00000000000000000000000000000000000000000000000000000000000000000000005308405B0000000001180000000
000002E170B85C000000000000001DF9466807800000000000000000000000000000000000000
2012/04/13 03:01:22. L=700
00000000000000000000000000000000000000000000000000000000000000000000009A1E8A000000000021000024 8CD
BD348CB000000000000000000000015F7E8C07800000000000000000000000000000000000000
2012/04/13 03:01:23. L=700
0000000000000000000000000000000000000000000000000000000000000000000000C60844000000000000000000000
000140201008040000000000000039F9CB7807800000000000000000000000000000000000000
2012/04/13 03:01:24. L=700
00000000000000000000000000000000000000000000000000000000000000000000005344000000000000000000000000
000000000000000000000000000003CD434407800000000000000000000000000000000000000
```

In case of L1 SBAS messages, there are several tools that decode the SBAS messages and analyze the performances of the system as for example, GMV's tool ***eclayr*** [RD.6].

In case L1 SBAS messages, one can also decode a specific SBAS message with the help of the tool SBAS TACHER (see http://www.egnos-pro.esa.int/sbasteacher/index.html for more information about this tool).

For SBAS L1/L5 messages, there is not any operational performance analysis tool developed for performing the analysis. It will be precisely the PROSBAS User Receiver Prototype the tool that will allow the decoding of the SBAS messages and the analysis of the performances obtained with the system. The LogBook files containing the SBAS messages are the interface between Service Provider Prototype and Receiver Prototype.

The following information encoded in the L1/L5 Enhanced ICD SBAS message is relevant for Receiver Prototype UDRE Module:

Table 7-14: RP UDRE Module external inputs (concerning SBAS Message)

Data Flow	I/O	Description
Message_type_B	I	Message included in SBAS encoded file.
Message_type_C	I	Message included in SBAS encoded file.
Message_type_6_1/2	I	Message included in SBAS encoded file.
Message_type_D	I	Message included in SBAS encoded file.
Message_type_E	I	Message included in SBAS encoded file.
Message_type_F	I	Message included in SBAS encoded file.
Message_type_G	I	Message included in SBAS encoded file.
Message_type_12	I	Message included in SBAS encoded file.
Message_type_30	I	Message included in SBAS encoded file.

The following information encoded in the L1/L5 Enhanced ICD SBAS message is relevant for Receiver Prototype GIVE Module (in case L5-only back-up mode):

Table 7-15: RP GIVE Module external inputs (concerning SBAS Message)

Data Flow	I/O	Description
Message_type_10	I	Message included in SBAS encoded file.
Message_type_18	I	Message included in SBAS encoded file.
Message_type_26	I	Message included in SBAS encoded file.

For further information on the content of Enhanced ICD SBAS Messages, please refer to [RD.7].

7.3.4.2. MSG output information

MSG module generates a file *msg_out.dat* containing the message sequence.
In the following it is shown a sample of an example of msg_out.dat file.

Table 7-16: example of *msg_out.dat* file

```
#####
# Begin information on messages configured
MSGCFG:  MT: MTL12_ENHANCED  UI: 300 TO: 86400  NUM_MSG: 1
MSGCFG:  MT: MTL6_ENHANCED  UI: 300 TO: 600  NUM_MSG: 2
MSGCFG:  MT: MTLUdreOneSat_ENHANCED  UI: 3000000 TO: 6000000  NUM_MSG: 91
MSGCFG:  MT: MTL_B  UI: 120 TO: 600  NUM_MSG: 1
MSGCFG:  MT: MTL_C  UI: 6 TO: 12  NUM_MSG: 1
MSGCFG:  MT: MTL_D  UI: 204 TO: 408  NUM_MSG: 91
MSGCFG:  MT: MTL_E  UI: 60 TO: 120  NUM_MSG: 2
MSGCFG:  MT: MTL_F  UI: 120 TO: 600  NUM_MSG: 1
MSGCFG:  MT: MTL_G  UI: 120 TO: 21600  NUM_MSG: 2
# Other configuration parameters
MSGCFG: NUM_SATS: 91 PRN_LIST: 1 2 3 4 5 6 7 9 14 15 16 17 18 19 21 22 23 25 26 27 28 29 31
32 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 75 76 77 78 79 80 81
83 88 89 90 91 92 93 95 96 97 98 99 100 101 102 103 105 174 175 176 177 178 179 180 182 187
188 189 190 191 192 194 195 196 197 198
MSGCFG: NUM_SATS_MONITORED_INITIALLY: 0
MSGCFG: NUM_SATS_IN_INITIAL_MASK: 91
# End information on messages configured
#####

#####
# Begin Service Area
Latitude: 20 70 10 Longitude: -40 40 20 (Min_deg/Max_deg/Num)
# End Service Area
```

```
…
AUX_INFO Epoch: 845  Epoch_Last_Alert: 0 IDOP: 0 0  Mask: NumSats: 91 MkIx: 0 PRN: 1 MkIx: 1
PRN: 2 MkIx: 2 PRN: 3 MkIx: 3 PRN: 4 MkIx: 4 PRN: 5 MkIx: 5 PRN: 6 MkIx: 6 PRN: 7 MkIx: 7 PRN: 9
MkIx: 8 PRN: 14 MkIx: 9 PRN: 15 MkIx: 10 PRN: 16 MkIx: 11 PRN: 17 MkIx: 12 PRN: 18 MkIx: 13
PRN: 19 MkIx: 14 PRN: 21 MkIx: 15 PRN: 22 MkIx: 16 PRN: 23 MkIx: 17 PRN: 25 MkIx: 18 PRN: 26
MkIx: 19 PRN: 27 MkIx: 20 PRN: 28 MkIx: 21 PRN: 29 MkIx: 22 PRN: 31 MkIx: 23 PRN: 32 MkIx: 24
PRN: 38 MkIx: 25 PRN: 39 MkIx: 26 PRN: 40 MkIx: 27 PRN: 41 MkIx: 28 PRN: 42 MkIx: 29 PRN: 43
MkIx: 30 PRN: 44 MkIx: 31 PRN: 45 MkIx: 32 PRN: 46 MkIx: 33 PRN: 47 MkIx: 34 PRN: 48 MkIx: 35
PRN: 49 MkIx: 36 PRN: 50 MkIx: 37 PRN: 51 MkIx: 38 PRN: 52 MkIx: 39 PRN: 53 MkIx: 40 PRN: 54
MkIx: 41 PRN: 55 MkIx: 42 PRN: 56 MkIx: 43 PRN: 57 MkIx: 44 PRN: 58 MkIx: 45 PRN: 59 MkIx: 46
PRN: 60 MkIx: 47 PRN: 61 MkIx: 48 PRN: 75 MkIx: 49 PRN: 76 MkIx: 50 PRN: 77 MkIx: 51 PRN: 78
MkIx: 52 PRN: 79 MkIx: 53 PRN: 80 MkIx: 54 PRN: 81 MkIx: 55 PRN: 83 MkIx: 56 PRN: 88 MkIx: 57
PRN: 89 MkIx: 58 PRN: 90 MkIx: 59 PRN: 91 MkIx: 60 PRN: 92 MkIx: 61 PRN: 93 MkIx: 62 PRN: 95
MkIx: 63 PRN: 96 MkIx: 64 PRN: 97 MkIx: 65 PRN: 98 MkIx: 66 PRN: 99 MkIx: 67 PRN: 100 MkIx: 68
PRN: 101 MkIx: 69 PRN: 102 MkIx: 70 PRN: 103 MkIx: 71 PRN: 105 MkIx: 72 PRN: 174 MkIx: 73
PRN: 175 MkIx: 74 PRN: 176 MkIx: 75 PRN: 177 MkIx: 76 PRN: 178 MkIx: 77 PRN: 179 MkIx: 78
```

```
PRN: 180 MkIx: 79 PRN: 182 MkIx: 80 PRN: 187 MkIx: 81 PRN: 188 MkIx: 82 PRN: 189 MkIx: 83
PRN: 190 MkIx: 84 PRN: 191 MkIx: 85 PRN: 192 MkIx: 86 PRN: 194 MkIx: 87 PRN: 195 MkIx: 88
PRN: 196 MkIx: 89 PRN: 197 MkIx: 90 PRN: 198 NumSatsInUdreAlert: 0 PRNs/Age/Msg:
NumSatsInISCAlert: 0 PRNs/Age/Msg:

Epoch: 845  Skip: No  MTL_D  SubMT: 71 MaskIx: 71 PRN: 105 DFRE_Status: UDREiMonitored

AUX_INFO Epoch: 846  Epoch_Last_Alert: 0 IDOP: 0 0  Mask: NumSats: 91 MkIx: 0 PRN: 1 MkIx: 1
PRN: 2 MkIx: 2 PRN: 3 MkIx: 3 PRN: 4 MkIx: 4 PRN: 5 MkIx: 5 PRN: 6 MkIx: 6 PRN: 7 MkIx: 7 PRN: 9
MkIx: 8 PRN: 14 MkIx: 9 PRN: 15 MkIx: 10 PRN: 16 MkIx: 11 PRN: 17 MkIx: 12 PRN: 18 MkIx: 13
PRN: 19 MkIx: 14 PRN: 21 MkIx: 15 PRN: 22 MkIx: 16 PRN: 23 MkIx: 17 PRN: 25 MkIx: 18 PRN: 26
MkIx: 19 PRN: 27 MkIx: 20 PRN: 28 MkIx: 21 PRN: 29 MkIx: 22 PRN: 31 MkIx: 23 PRN: 32 MkIx: 24
PRN: 38 MkIx: 25 PRN: 39 MkIx: 26 PRN: 40 MkIx: 27 PRN: 41 MkIx: 28 PRN: 42 MkIx: 29 PRN: 43
MkIx: 30 PRN: 44 MkIx: 31 PRN: 45 MkIx: 32 PRN: 46 MkIx: 33 PRN: 47 MkIx: 34 PRN: 48 MkIx: 35
PRN: 49 MkIx: 36 PRN: 50 MkIx: 37 PRN: 51 MkIx: 38 PRN: 52 MkIx: 39 PRN: 53 MkIx: 40 PRN: 54
MkIx: 41 PRN: 55 MkIx: 42 PRN: 56 MkIx: 43 PRN: 57 MkIx: 44 PRN: 58 MkIx: 45 PRN: 59 MkIx: 46
PRN: 60 MkIx: 47 PRN: 61 MkIx: 48 PRN: 75 MkIx: 49 PRN: 76 MkIx: 50 PRN: 77 MkIx: 51 PRN: 78
MkIx: 52 PRN: 79 MkIx: 53 PRN: 80 MkIx: 54 PRN: 81 MkIx: 55 PRN: 83 MkIx: 56 PRN: 88 MkIx: 57
PRN: 89 MkIx: 58 PRN: 90 MkIx: 59 PRN: 91 MkIx: 60 PRN: 92 MkIx: 61 PRN: 93 MkIx: 62 PRN: 95
MkIx: 63 PRN: 96 MkIx: 64 PRN: 97 MkIx: 65 PRN: 98 MkIx: 66 PRN: 99 MkIx: 67 PRN: 100 MkIx: 68
PRN: 101 MkIx: 69 PRN: 102 MkIx: 70 PRN: 103 MkIx: 71 PRN: 105 MkIx: 72 PRN: 174 MkIx: 73
PRN: 175 MkIx: 74 PRN: 176 MkIx: 75 PRN: 177 MkIx: 76 PRN: 178 MkIx: 77 PRN: 179 MkIx: 78
PRN: 180 MkIx: 79 PRN: 182 MkIx: 80 PRN: 187 MkIx: 81 PRN: 188 MkIx: 82 PRN: 189 MkIx: 83
PRN: 190 MkIx: 84 PRN: 191 MkIx: 85 PRN: 192 MkIx: 86 PRN: 194 MkIx: 87 PRN: 195 MkIx: 88
PRN: 196 MkIx: 89 PRN: 197 MkIx: 90 PRN: 198 NumSatsInUdreAlert: 0 PRNs/Age/Msg:
NumSatsInISCAlert: 0 PRNs/Age/Msg:

Epoch: 846  Skip: No  MTL_D  SubMT: 72 MaskIx: 72 PRN: 174 DFRE_Status: UDREiMonitored

AUX_INFO Epoch: 847  Epoch_Last_Alert: 0 IDOP: 0 0  Mask: NumSats: 91 MkIx: 0 PRN: 1 MkIx: 1
PRN: 2 MkIx: 2 PRN: 3 MkIx: 3 PRN: 4 MkIx: 4 PRN: 5 MkIx: 5 PRN: 6 MkIx: 6 PRN: 7 MkIx: 7 PRN: 9
MkIx: 8 PRN: 14 MkIx: 9 PRN: 15 MkIx: 10 PRN: 16 MkIx: 11 PRN: 17 MkIx: 12 PRN: 18 MkIx: 13
PRN: 19 MkIx: 14 PRN: 21 MkIx: 15 PRN: 22 MkIx: 16 PRN: 23 MkIx: 17 PRN: 25 MkIx: 18 PRN: 26
MkIx: 19 PRN: 27 MkIx: 20 PRN: 28 MkIx: 21 PRN: 29 MkIx: 22 PRN: 31 MkIx: 23 PRN: 32 MkIx: 24
PRN: 38 MkIx: 25 PRN: 39 MkIx: 26 PRN: 40 MkIx: 27 PRN: 41 MkIx: 28 PRN: 42 MkIx: 29 PRN: 43
MkIx: 30 PRN: 44 MkIx: 31 PRN: 45 MkIx: 32 PRN: 46 MkIx: 33 PRN: 47 MkIx: 34 PRN: 48 MkIx: 35
PRN: 49 MkIx: 36 PRN: 50 MkIx: 37 PRN: 51 MkIx: 38 PRN: 52 MkIx: 39 PRN: 53 MkIx: 40 PRN: 54
MkIx: 41 PRN: 55 MkIx: 42 PRN: 56 MkIx: 43 PRN: 57 MkIx: 44 PRN: 58 MkIx: 45 PRN: 59 MkIx: 46
PRN: 60 MkIx: 47 PRN: 61 MkIx: 48 PRN: 75 MkIx: 49 PRN: 76 MkIx: 50 PRN: 77 MkIx: 51 PRN: 78
MkIx: 52 PRN: 79 MkIx: 53 PRN: 80 MkIx: 54 PRN: 81 MkIx: 55 PRN: 83 MkIx: 56 PRN: 88 MkIx: 57
PRN: 89 MkIx: 58 PRN: 90 MkIx: 59 PRN: 91 MkIx: 60 PRN: 92 MkIx: 61 PRN: 93 MkIx: 62 PRN: 95
MkIx: 63 PRN: 96 MkIx: 64 PRN: 97 MkIx: 65 PRN: 98 MkIx: 66 PRN: 99 MkIx: 67 PRN: 100 MkIx: 68
PRN: 101 MkIx: 69 PRN: 102 MkIx: 70 PRN: 103 MkIx: 71 PRN: 105 MkIx: 72 PRN: 174 MkIx: 73
PRN: 175 MkIx: 74 PRN: 176 MkIx: 75 PRN: 177 MkIx: 76 PRN: 178 MkIx: 77 PRN: 179 MkIx: 78
PRN: 180 MkIx: 79 PRN: 182 MkIx: 80 PRN: 187 MkIx: 81 PRN: 188 MkIx: 82 PRN: 189 MkIx: 83
PRN: 190 MkIx: 84 PRN: 191 MkIx: 85 PRN: 192 MkIx: 86 PRN: 194 MkIx: 87 PRN: 195 MkIx: 88
PRN: 196 MkIx: 89 PRN: 197 MkIx: 90 PRN: 198 NumSatsInUdreAlert: 0 PRNs/Age/Msg:
NumSatsInISCAlert: 0 PRNs/Age/Msg:

Epoch: 847  Skip: No  MTL_C  SubMT: 0
…
```

In the section:

Begin information on messages configured

There are shown the different MTs configured, together with the configured Update Interval, Time-out and number of submessages.

For example, in this case, the line:

MSGCFG: MT: MTL_D UI: 204 TO: 408 NUM_MSG: 91

Means that MT_D SVs correction has been configured with an Update Interval of 204 seconds, a Time-Out of 408 seconds and there are 91 sub-messages since in the corresponding execution, 91 SVs were configured in mask.

In section:

Other configuration parameters

There is shown the PRN mask configured.

In section:

Begin Service Area

It is shown the service area configured.

Then, for each epoch simulated, there are two sections.

In section:

AUX_INFO it is shown the following information:

- Epoch
- Epoch_Last_Alert: the epoch when the last alert took place.
- IODP
- Mask. NumSats (total number of SVs in mask), and for each SV configured in mask: MkIx (Mask Index) and PRN.
- NumSatsInUdreAlert: Number of SVs in UDRE Alert.
- PRNs/Age/Msg: three numbers for each satellite in UDRE Alert (if any): the PRN, the Age (number of epochs that was before in alert) and the number of messages in alert which have already reported this specific alert.

In section:

Epoch:

it is shown the following information:

- Epoch
- Skip: field that is generated always by MSG module with "No" value and is used by MLS module for indicating that a certain message has been lost at user level changing the field to "Yes" value.
- Message Type: for example, MTL_C.
- SubMT: index of the number of submessage, beginning by 0. For example, for MTL6_ENHANCED, SubMT=0 means MT_6_1 while SubMT=1 means MT_6_2.

7.3.4.3. NOFG output information

In this section several output files generated by NOFG module will be described.

7.3.4.3.1. UDREs in L1 file

The output file *udre_l1* includes the UDRE in L1 for each epoch and each PRN. Please, note that this file is only generated if "debug_outputs" parameter in gen_L1_NOF_DFRE_cfg.dat file is set to 1. It has an initial line with the text "epoch PRN UDREI_L1 UDRE_L1" and then lines with the format shown in the following table.

Table 7-17: parameters in *udre_l1* file

Parameter	Description	Range
epoch	GPS time of the observation.	[0, ∞)

Parameter	Description	Range
	Integer value. Unit: seconds	
PRN	PRN of the observed satellite. Integer value. Unit: unitless	[1, 210]
UDREI_L1	UDRE in L1 indicator for that PRN at that epoch. Integer value. Unit: unitless	[0, 15]
UDRE_L1	NOF's UDRE in L1 value for that PRN at that epoch. Floating point value. Unit: $meter^2$	ℝ[4]

Here it is shown an example of one such file:

Table 7-18: example of *udre_l1* file

epoch PRN UDREI_L1 UDRE_L1
943671659 2 14 -1
943671659 3 14 -1
943671659 4 14 -1
943671659 5 7 1.8709
943671659 6 14 -1
943671659 7 14 -1

7.3.4.3.2. UDREs in L5 file

The output file *udre_l5* includes the UDRE indicator in L5 for each epoch and each PRN. It has an initial line with the text "epoch PRN UDREI_5" and then lines with the format shown in the following table.

Table 7-19: parameters in *udre_l5* file

Parameter	Description	Range
epoch	GPS time of the observation. Integer value. Unit: seconds	[0, ∞)
PRN	PRN of the observed satellite. Integer value. Unit: unitless	[1, 210]
UDREI_L5	UDRE indicator in L5 for that PRN at that epoch. Integer value. Unit: unitless	[0, 15]

In the following table it is shown an excerpt of one such file.

Table 7-20: example of *udre_l5* file

epoch PRN UDREI_L5
943671660 24 14
943671660 26 14
943671660 27 6
943671660 28 6

[4] Negative UDREs might be obtained throughout the simulation performed in operational mode 1 due to the stochastic nature of the Gauss-Markov process. Nevertheless, they are forced into range in NOFG module. Moreover, NOFG module uses the value -1 to represent "Not monitored" UDREs internally; analogously, it uses the value -2 to represent "Do not use" UDREs.

943671660 29 14
943671660 30 14

Please, note that this file is generated from SME module, but not from NOFG module, because alerts are processed in the former —outside the latter— and they are taken into account when computing the final UDRE in L5.

7.3.4.3.3. IODEs file

The output file *iode* includes the IODE for each epoch and each PRN. Please, note that this file is only generated if "debug_outputs" parameter in gen_L1_NOF_DFRE_cfg.dat file is set to 1. It has an initial line with the text "epoch PRN IODE" and then lines with the format shown in the following table:

Table 7-21: parameters in *iode* file

Parameter	Description	Range
epoch	GPS time of the observation. Integer value. Unit: seconds	[0, ∞)
PRN	PRN of the observed satellite. Integer value. Unit: unitless	[1, 210]
IODE	IODE for that PRN at that epoch. Integer value. Unit: unitless	[0, 255]

In the following it is shown an example of such file:

Table 7-22: example of *iode* file

epoch PRN IODE
943671616 30 87
943671616 31 87
943671616 32 14
943671617 2 119
943671617 3 87
943671617 4 85

7.3.4.3.4. GIVEs in L5 file

The output file *give* includes the GIVE indicators in L5 for each epoch, each band and each block. Please, note that this file is only generated if "debug_outputs" parameter in gen_L1_NOF_DFRE_cfg.dat file is set to 1. It has an initial line with the text "epoch band block GIVEIs_L5" and then lines with the format shown in the following table.

Table 7-23: parameters in *give* file

Parameter	Description	Range
Epoch	GPS time of the observation. Integer value. Unit: seconds	[0, ∞)
Band	Band of the observed IGP.	[0, 10]

Parameter	Description	Range
	Integer value. Unit: unitless	
Block	Block in the band of the observed IGP. Integer value. Unit: unitless	[0, 13]
GIVEIs_L5	GIVE indicator for that the IGPs in the block. List of up to 15 integer values. Unit: unitless	[0, 15]

In the following it is shown an example of such file.

Table 7-24: example of *give* file

```
epoch band block GIVEIs_L5
1018321438 4 0 15 15 15 15 11 9 9 11 12 12 15 15 15 15 10
1018321438 4 1 9 9 10 11 11 15 15 15 15 10 9 9 10 10 10
1018321438 4 2 15 15 15 15 10 9 9 9 10 10 15 15 15 13 9
1018321438 4 3 9 10 10 10 15 15 15 15 10 9 10 10 10 15 15
1018321438 4 4 15 15 11 10 10 10 10 15 15 15 15 14 10 10 10
1018321438 4 5 10
```

7.3.4.3.5. Stations location file

The output file stations_latitude_longitude_height includes the latitude, longitude and height of each configured station. It has an initial line with the text "latitude longitude height" and then lines with the format shown in the following table:

Table 7-25: parameters in *stations_latitude_longitude_height* file

Parameter	Description	Range
latitude	Latitude of the station. Floating point value. Unit: degree	[-90, 90]
longitude	Longitude of the station. Floating point value. Unit: degree	[-180, 180]
height	Height of the station. Floating point value. Unit: meter	[0, ∞)

In the following it is shown an example of such file.

Table 7-26: example of *stations_latitude_longitude_height* file

```
latitude longitude height
57.0964 9.88319 42.3925
38.5187 -28.6228 198.549
52.3569 13.5411 83.4461
27.9454 -15.3794 51.0082
37.4636 15.062 47.1638
51.8447 -8.49621 201.713
52.2148 21.0676 124.49
```

7.3.4.3.6. SVs monitorization plot at SPP level

NumSVvsTime_SPP.png represents the number of Monitored, Not Monitored and Don't Use satellites versus time at SPP level.

In the following figure it is shown an example of such a file.

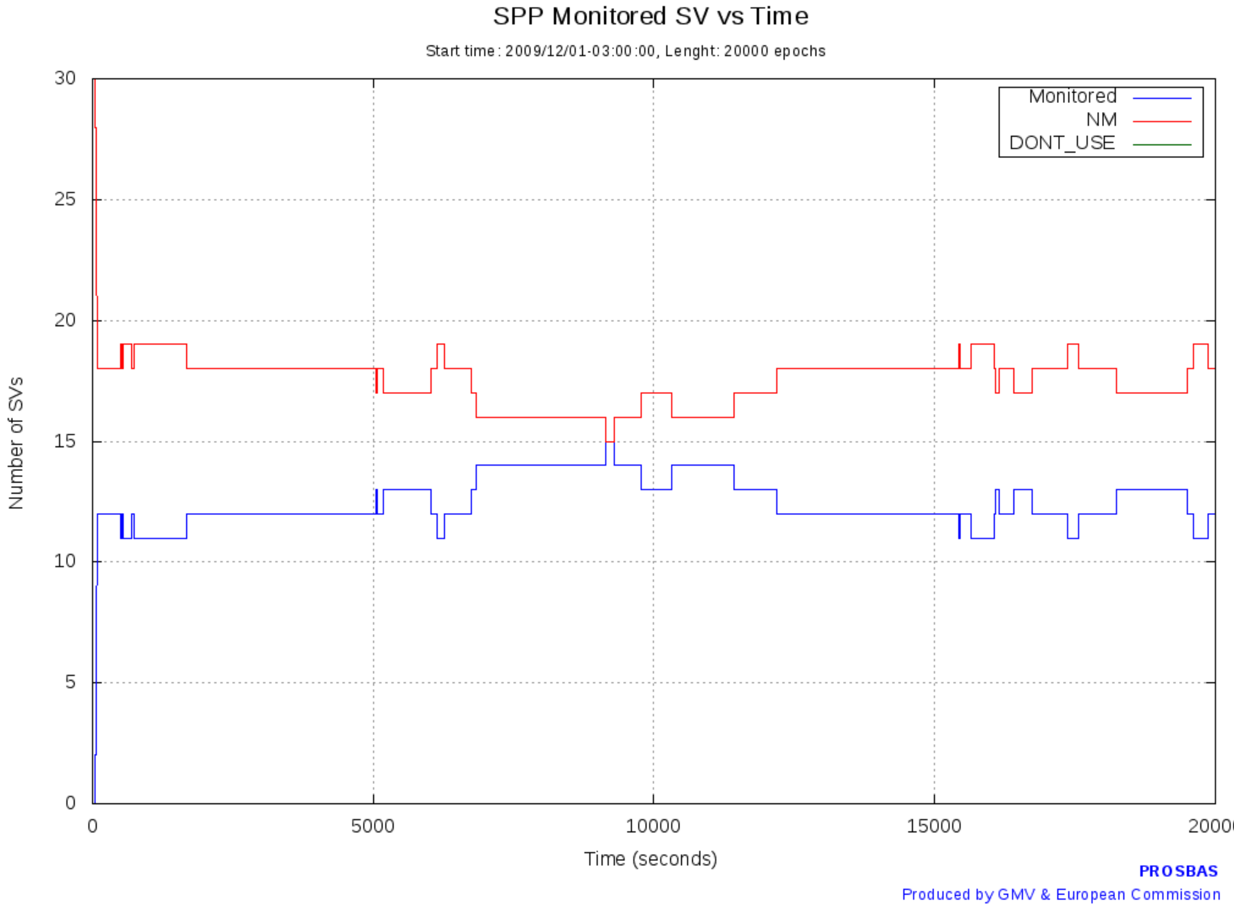

Figure 7-1: example of *NumSVvsTime_SPP.png* file

Please note that this plot is generated by Visualization Module using as input NOFG outputs. Then, in order to generate this plot, the RP should also be run.

7.3.4.3.7. SVs monitorization plot at SPP and RP level

NumSVvsTime_SPPvsRP.png represents the number of Monitored SVs at SPP level, at RP level and the difference between them.

In the following figure it is shown an example of such a file.

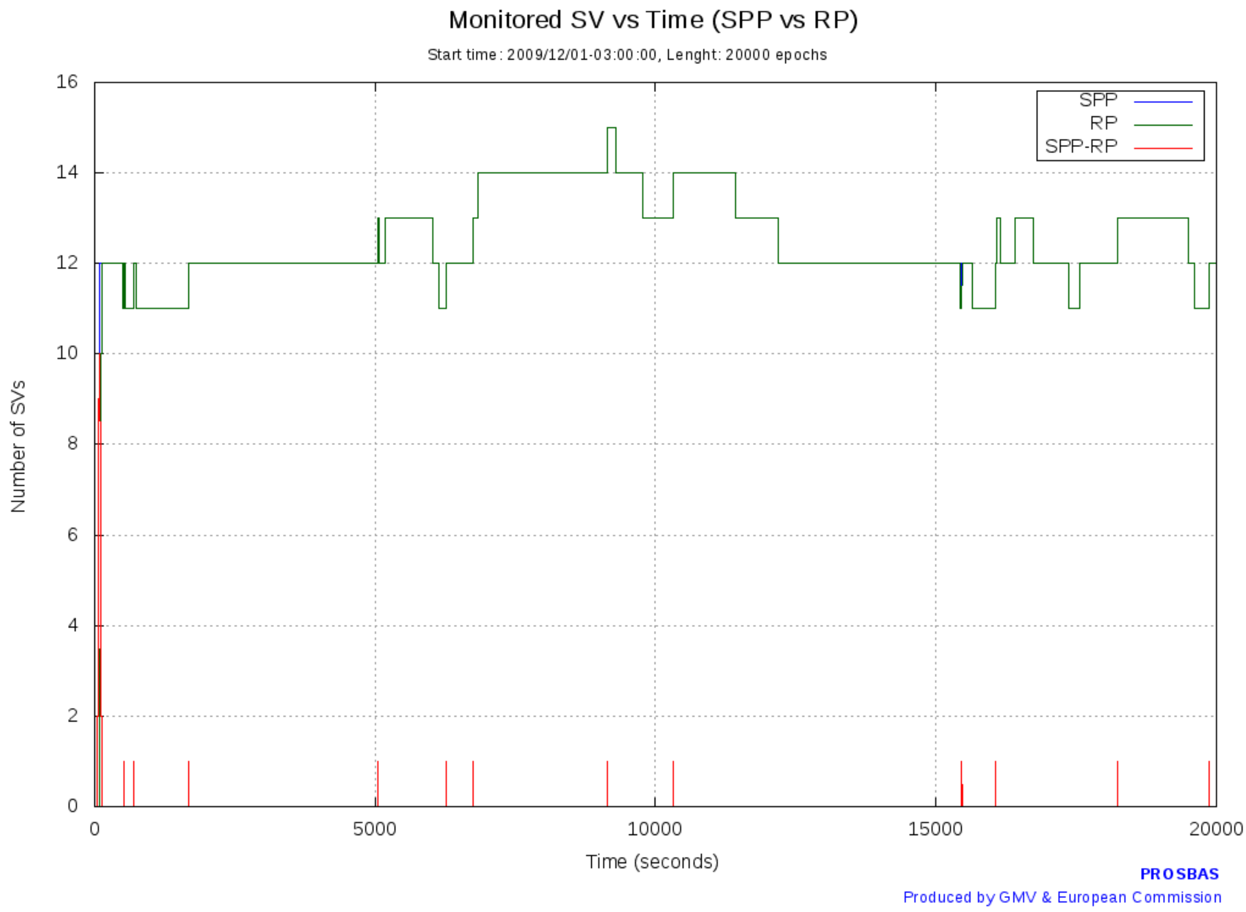

Figure 7-2: example of *NumSVvsTime_SPPvsRP.png* file

Please note that this plot is generated by Visualization Module using as input NOFG outputs and RP outputs. Then, in order to generate this plot, the RP should also be run.

7.3.4.3.8. IODEs plots

IODE_PRNx.png are plots (one for each PRN configured in mask) showing the evolution of IODE with time.

In the following figure it is shown an example of such a file.

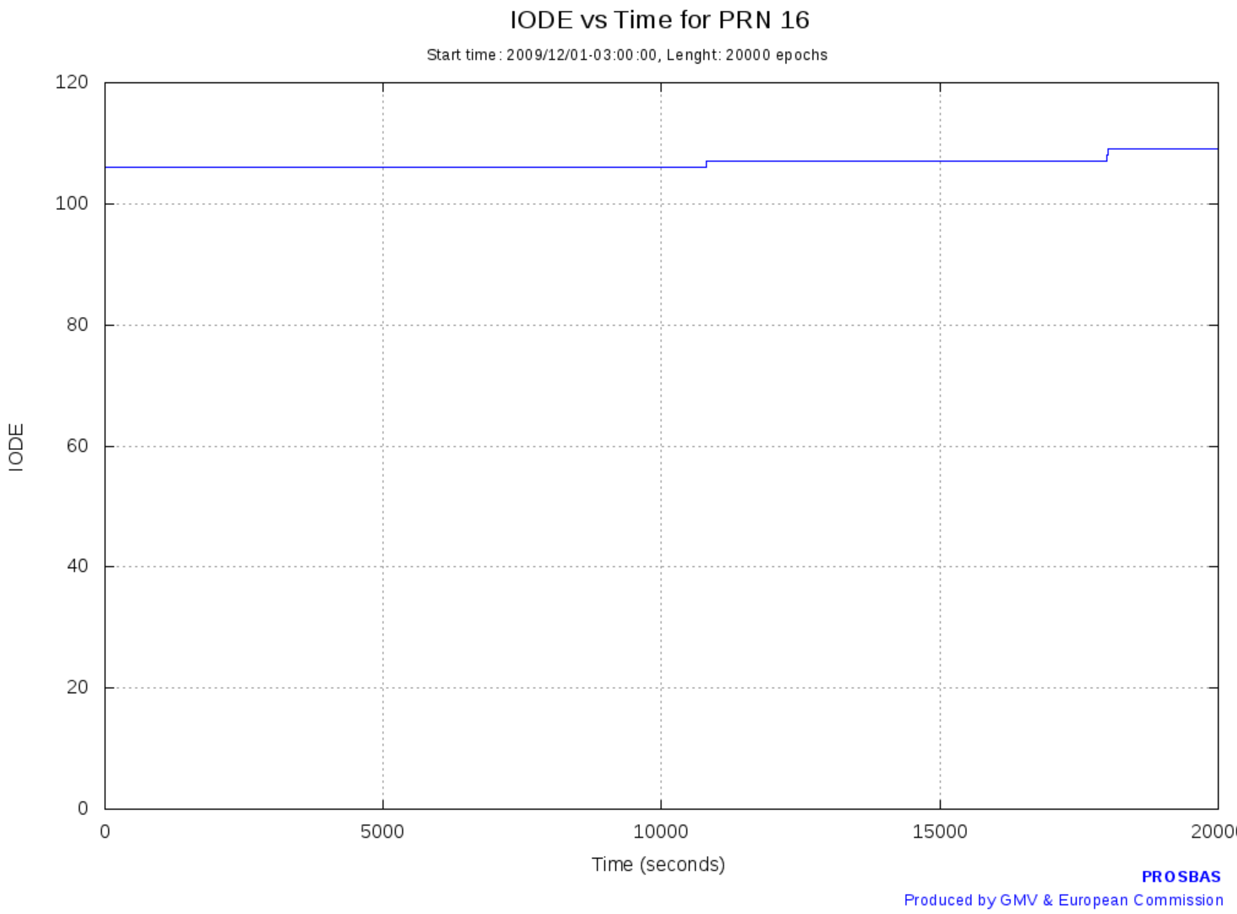

Figure 7-3: example of *IODE_PRN16.png* file

Please note that this plot is generated by Visualization Module using as input NOFG outputs. Then, in order to generate this plot, the RP should also be run.

7.4. MLS MODULE

7.4.1. INVOCATION

MLS module uses as input the following files:

- the MLS module specific configuration file: *mls_cfg.dat*
- the *ref_cond_cfg.dat* file
- the output of MSG module: *msg_out.dat*
- the output of SME module, that is, the LogBook files.

For getting instructions related with the usage of MLS module, the user should be placed in the folder where the executable is placed and type:

./MLS –h

In the following there is displayed the result of executing the usage option of MLS module:

Table 7-27: *MLS* usage

```
NAME
\t./MLS: Message Loss Simulator

SYNOPSIS
\t./MLS [-h] [-d] [-c cfgfile] [-l LogBook_file] [input [output]]

DESCRIPTION
\tSimulate message loss on SBAS messages.
\tIf input or output are not provided, standard input and output are used instead.

\t-h\t Help.
\t-d\t Show default configuration parameters and exit. The output can be used as a default
configuration file.
\t-c cfgfile\t Change configuration file name. Default configuration file is mspa_mls.cfg.
\t-l LogBook_file\t Logbook or folder of logbooks to modify
\t-r ref_cond_cfg Read the general configuration file

VERSION
\tV1.1.1

COPYRIGHT
\tCopyright GMV, 2013
```

For generating default configuration file for MLS module, the user should be placed in the folder where the executable is placed and type:

./MLS –d > mls_cfg.dat

For the execution of MLS module alone, assuming that the input files are in the same folder than the executable, the user should go to the directory where the executable is placed and type the following instruction in Linux terminal:

./MLS –c mls_config_path -l < LogBook_path –r ref_cond_path msg_out.dat_path output_mls_path

7.4.2. CONFIGURATION

MLS module accepts as inputs his specific configuration file: *mls_cfg.dat*

In the following, an example of *mls_cfg.dat* file is provided and the different configuration parameters are explained.

Table 7-28: example of *mls_cfg.dat* file

```
# Seed for random number generation
RAND_SEED=1

# Probability of loss per message. General case.
# More than three consecutive message losses are not allowed.
PROB_GEN_LOSS=0
```

```
# Minimum epoch before message losses are allowed.
MIN_EPOCH_FOR_LOSSES=0

# Losses generated upon alerts.
PROB_ALERT_LOSS=0
# PROSBAS Prototype Receiver does not process LogBooks
# with four or more consecutive loss messages since the system would not be available.
# If the number of alert message losses configured is equal or larger than four,
# it will be set internally to three.
NUM_ALERT_LOSSES=3
```

The configuration parameters are explained in the following:

Table 7-29: MLS module configuration parameters present in *mls_cfg.dat*

Parameter	Description
`# Seed for random number generation` `RAND_SEED=1`	Seed for random number generation. This is an integer.

`# Probability of loss per message.`
`General case.`
`PROB_GEN_LOSS=0`
	Probability of loss per message, given as a float between 0 (no loss) and 1 (total loss).

`# Minimum epoch before message losses`
`# are allowed.`
`MIN_EPOCH_FOR_LOSSES=0`
	No loss will be introduced before the indicated epoch.

`# Losses generated upon alerts.`
`# The offset refers to the first message`
`with an alert in a alert sequence.`
`PROB_ALERT_LOSS=0`
`NUM_ALERT_LOSSES=3`
	Probability of loss for alert messages. In addition to the generic loss probability, the loss simulator may assign a higher loss probability to alert messages.
	
	If the loss is activated, the given number of messages will be lost in a row, starting from the first one.
	
	Let us note that the maximum number of alert losses is 3

In addition, MLS also needs *ref_cond_cfg.dat* file whose description can be found in Section 7.1.1.

7.4.3. INPUT FORMAT

MLS module uses as input the output generated by MSG module: *msg_out.dat* file.
For further information on the format of these files, please refer to Section 7.3.4.2.
In addition, it also uses the LogBooks generated by SME module as inputs.
For further information on the format of these files, please refer to Section 7.3.4.1.

7.4.4. OUTPUT FORMAT

There are two kind of outputs of MLS module.

On the one hand, the file *mls_out.dat* file has a format identical to *msg_out.dat* file but with the "Skip" field set to "Yes" in the epochs when a loss of the message has been simulated at user level.

For further details on *mls_out.dat* file, please check Section 7.3.4.2.

On the other hand, MLS generates again the same LogBooks as SME but delaying the SBAS messages in the epochs where a message loss has been simulated.

For further information on LogBook files, please refer to 7.3.4.1.

7.5. MSA MODULE

7.5.1. INVOCATION

MSA module uses as input the output file from either MSG module *msg_out.dat* (if MLS module, that is not compulsory, is not executed) or either MLS module *mls_out*.dat.

MSA module does not need a configuration file as input. Nevertheless, it can be executed with several options that will be described in the following.

For getting instructions related with the usage of MSA module, the user should be placed in the folder where the executable is placed and type:

./MSA –h

In the following it is shown the usage displayed executing MSA with usage option:

Table 7-30: *MSA* usage

```
NAME
        ./MSA -  Message Sequence Analyzer

SYNOPSIS
        ./MSA [-h] [-a] [-i] [-l n_sats] [-m mx_epoch] [-n ui_mask] [-p] [-t] [-u] [-v] [-A] [-B] [-I] [M]
[-P] [-T] [-s sk_epochs]

DESCRIPTION
        -h              Print usage in stdout and exit.
        -a              Toggle to show detailed data on TTA violations (default: do not).
        -m mx_epoch     Maximum epoch to process.
        -t              Toggle to show detailed data on timeout violations (default: do not).
        -u              Toggle to show detailed data on update time violations (default: do not).
        -A              Toggle to show summary data on TTA violations (default: do it).
        -B              Toggle to perform bandwidth static analysis (default: do it).
        -T              Toggle to show summary on update time and timeout violations (default: do it).
        -s sk_epochs    Number of epochs to skip before counting violations (default: zero).

VERSION
        V1.1.1 evolved

COPYRIGHT
        Copyright GMV, 2013.
```

For the execution of MSA module alone, assuming that the input file is in the same folder than the executable, the user should go to the directory where the executable is placed and type the following instruction in Linux terminal:

./MSA –OPTION < msg_out.dat > msa_out.dat

or ./MSA –OPTION < mls_out.dat > msa_out.dat

Note that MSA module uses the standard output and it is redirected to the file *msa_out.dat.*

If MSA module is executed without any option, then the output will be the standard one.

For a more detailed analysis on a certain issue, one can run MSA module with the following options:

DESCRIPTION
-h Print usage in stdout and exit.
-a Toggle to show detailed data on TTA violations (default: do not).

-m mx_epoch Maximum epoch to process.
-t Toggle to show detailed data on timeout violations (default: do not).
-u Toggle to show detailed data on update time violations (default: do not).
-A Toggle to show summary data on TTA violations (default: do it).
-B Toggle to perform bandwidth static analysis (default: do it).
-T Toggle to show summary on update time and timeout violations (default: do it).
-s sk_epochs Number of epochs to skip before counting violations (default: zero).

7.5.2. CONFIGURATION

As stated in the previous section, MSA module does not use any configuration file.

Nevertheless, it can be executed with several options in order to obtain more or less detailed information.

For more information on MSA executing options, please refer to the previous section.

7.5.3. INPUT FORMAT

The input file of MSA module is the output of MLS module (*mls_out.dat*) or MSG module (*msg_out.dat*) depending on if MLS is executed or not.

For further information on the format of these files, please refer to sections 7.3.4.2 and 7.4.4.

7.5.4. OUTPUT FORMAT

As stated in Section 7.5.1, MSA output file (*msa_out.dat*) generates a less or more detailed information depending on the options with which the module is called.

Please note that in case MLS execution is activated, two files are generated from MSA module: *msa_out.dat_SPP* and *msa_out.dat_RP.* In this way, Update Interval and Time-out fulfillment analysis is performed both at SPP and at RP level. In the later case, if a message is lost at receiver level, MSA might account of an Update Interval violation. Please note that the format of *msa_out.dat_SPP* and *msa_out.dat_RP* is identical to the one of *msa_out.dat* described in the following.

In the following Figure it is shown an example of msa_out.dat output in the case the module has been executed without any option.

Table 7-31: Example of *msa_out.dat* file

```
#### BANDWIDTH STATIC ANALYSIS SUMMARY
Num_Sats: 24   Num_SatsInMask: 24
Msg: MTL12_ENHANCED 300 1 0.00333333 0.00333333 0.00333333
Msg: MTL18_ENHANCED 300 5 0.00333333 0.0166667 0.02
Msg: MTL26_ENHANCED 300 22 0.00333333 0.0733333 0.0933333
Msg: MTL6_ENHANCED 6 1 0.166667 0.166667 0.26
Msg: MTLISCValueOneSat_ENHANCED 120 1 0.00833333 0.00833333 0.268333
Msg: MTL_B 120 1 0.00833333 0.00833333 0.276667
Msg: MTL_C 6000 1 0.000166667 0.000166667 0.276833
Msg: MTL_D 52 24 0.0192308 0.461538 0.738372
Msg: MTL_E 60 2 0.0166667 0.0333333 0.771705
Msg: MTL_F 120 1 0.00833333 0.00833333 0.780038
Msg: MTL_G 120 1 0.00833333 0.00833333 0.788372
Total_Static_Bandwidth: 0.788372
####


#### TIMEOUT AND UPDATE TIME FULFILMENT SUMMARY
UPDATE_INTERVAL_FULFILMENT  NUM_EPOCHS_WITH_VIOLATIONS: 2 / 71994  LAST_EPOCH: 301
TIMEOUT_FULFILMENT  NUM_EPOCHS_WITH_VIOLATIONS: 0 / 72000  LAST_EPOCH: 0
####


#### MISLEADING INFORMATION
NUM_ALERT_MESSAGES: 0
NUM_MESSAGES_LOST: 0
NUM_MESSAGES_WITH_ALERT_AND_LOST: 0
####
```

In the following table there are explained the different outputs (for an execution of MSA without options, that is, with the basic information).

Table 7-32: *msa_out.dat* parameters

Format	Comment
#### BANDWIDTH STATIC ANALYSIS SUMMARY Num_Sats: *num* Num_SatsInMask: *num* *{* Msg: *msgType updateInterval numMsg msgBw bwContrib cumulatedBw* *}numMsgs* Total_Static_Bandwidth: *bw* ####	Static bandwidth analysis. For each of the messages types to be broadcast, it is provided a line with the message type, the update interval, the needed number of instances of the message, the contribution to the bandwidth of each of the messages, the contribution considering the number of instances, and the cumulated bandwidth considering this message and the previous messages listed so far. The bandwidth is provided as a number between 0 and 1.
#### TIMEOUT AND UPDATE TIME FULFILMENT SUMMARY UPDATE_INTERVAL_FULFILMENT NUM_EPOCHS_WITH_VIOLATIONS: *num* / *numAnalyzed* LAST_EPOCH: *epoch* TIMEOUT_FULFILMENT NUM_EPOCHS_WITH_VIOLATIONS: *num* / *numAnalyzed* LAST_EPOCH: *epoch* ####	Statistics on timeout and update interval violations. It is shown the number of violations versus the total number of epochs analysed and the last epoch when a violation took place.
#### MISLEADING INFORMATION NUM_ALERT_MESSAGES: *num* NUM_MESSAGES_LOST: *num* NUM_MESSAGES_WITH_ALERT_AND_LOST: *num* EPOCHS_WITH_TTA_VIOLATIONS Epochs*: num* ####	Misleading information is provided the number of alert messages, the number of messages lost at user level, the number of alert messages that have been lost and the number of epochs with a TTA violation.

END OF DOCUMENT