SYSTEM PROTOTYPE SPECIFICATION AND DEFINITION

PROSBAS PROJECT

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

Code:	PROSBAS-GMV-TN-D3.1.3
Version:	2.1
Date:	20/01/2014
Internal code:	GMV 20901/14 V5/14

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. The results are the property of the European Commission. No distribution or copy is permitted unless prior authorization is given by the European Commission

DOCUMENT STATUS SHEET

Version	Date	Pages	Changes
1.0	18/04/2013	85	First internal version sent to RC for review.
1.1	25/04/2013	86	Document Version for PROSBAS KP1
1.2	28/06/2013	86	Document Version with PROSBAS KP1 Actions implemented: • RID PROSBAS-KP1-030: it has been remarked that PROSBAS Prototype will use HISTB outputs as inputs in page 24.
2.0	27/11/2013	75	Document Version for PROSBAS MTR. Main modifications consisting of: - Modifications of document and design for taking into account Enhanced ICD - Modifications of the design during the development
2.1	20/01/2014	75	Document Version with MTR Actions implemented. • RID PROSBAS-MTR-016: Renumbering of requirements in sections 4. and 6. • RID PROSBAS-MTR-063: Modification in Table 2-2 • RID PROSBAS-MTR-064: Reference to Acronyms and definitions TN added • RID PROSBAS-MTR-072: added a clarification on MLS module in page 33

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

1. INTRODUCTION

1.1. PURPOSE

This document is the *System Prototype Specification and Definition* 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 present the high-level design of PROSBAS System Prototype, to specify the prototype providing the requirements that should achieve and to trace the requirements to modules or interfaces.

The present document is the final version of T3.1.3 output document, submitted in accordance with PROSBAS proposal for MTR. The document has been updated with respect to the first version delivered at KP1 taking into account the design modifications that have been performed during the development of PROSBAS Prototype.

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 document.
- Chapter 2. provides the list of project applicable and reference documents.
- Chapter 3. provides the list of definitions and acronyms used throughout the document.
- Chapter 4. details the PROSBAS Prototype requirements.
- Chapter 5. presents the high-level design of PROSBAS prototype.
- Chapter 6. traces the requirements to modules.
- Chapter 7. (Annex A) presents both the detailed design of NOFG module and a theoretical overview on how to build L1/L5 NOF.
- Chapter 8. (Annex B) reviews L1/L5 SBAS Enhanced ICD implemented in PROSBAS Prototype.
- Chapter 9. (Annex C) presents an overview of MAST tool.

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
All sections review	RCUK

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]	SIS Message Definition	PROSBAS-GMV-TN-D2.2.1	2.1	03/2013
[RD.2]	Reference Conditions for the simulations (including justification)	PROSBAS-GMV-TN-D.1.2.1	3.0	2013
[RD.3]	Service Provider/User Receiver Prototype ICD	PROSBAS-GMV-TN-O3.1.1	1.1	02/2013
[RD.4]	Message Analysis Support Tools (MAST) User Manual	SUGAST-GMV-D-2330-RE	1.0	11/2011
[RD.5]	Test Tools specification and definition	GMV-PROSBAS-TN-D3.1.4	2.0	26/11/2013
[RD.6]	Operating Manuals for Prototype and Tools	PROSBAS-GMV-TN-D3.1.5	1.0	26/11/2013
[RD.7]	Minimum Operational Performance Standards for GPS/WAAS Airborne Equipment	RTCA/DO-229D	N/A	Jul 2006
[RD.8]	Walter, T., J. Blanch and P. Enge, "L1/L5 SBAS MOPS to Support Multiple Constellations" ION GNSS 2012	N/A	N/A	20/09/2012
[RD.9]	Qualitative Trade-Off on SBAS L1/L5 ICD Models	MSIL2-GMV-TN-004_SS06_v1.2	1.2	26/07/2012
[RD.10]	Receiver Contribution to the User Algorithm Definition	PROSBAS-GMV-O2.4.1	1.3	01/02/2013
[RD.11]	Memo: Robustness Analysis of UDRE ICD Alternative	GMV-PROSBAS-MEM-004	1.1	05/03/2013
[RD.12]	Support to EC for Analysis of Multi-GNSS Service Performance Assessment	MSIL2-GMV-TN-008-SS12	1.3	01/2013
[RD.13]	Enhanced SBAS L1/L5 ICD: Analysis of Performances and Robustness	MSIL2-WO2-GMV-TN--11_SS07	1.0	22/11/2013
[RD.14]	Operating manuals for receiver prototype and tools	PROSBAS-GMV-TN-D4.2.1	1.0	26/11/2013
[RD.15]	Standard User Algorithms specification/definition	PROSBAS-GMV-D2.3.1	2.0	26/11/2013
[RD.16]	List of acronyms and definitions in PROSBAS Project	N/A	1.0	20/01/2014

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. In [RD.16] all the acronyms used in PROSBAS Project are compiled.

Table 3-2: Acronyms

Acronym	Definition
APV	Approach with Vertical Guidance
CCC	Code Carrier Coherence
CFI	Customer Furnished Items
DAL	Development Assurance Level
DDF	Design Definition File
DJF	Design Justification File
EC	European Comission
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
HISTB	High Integrity System Test Bed
HW	Hardware
ICD	Interface Control Document
ISC	Inter Signal Corrections
IWG	Interoperability Working Group
KO	Kick Off
KP	Key Point
KPI	Key Performance Indicator
LPV	Lateral Precision with Vertical Guidance
MAST	Message Analysis Support Tools
MGD	Mission Guideline Document

Acronym	Definition
MLS	Message Loss Simulator (PROSBAS Prototype Module)
MOPS	Minimum Operational Performance Standards
MRD	Mission Requirement Document
MSA	Message Sequence Analyser (PROSBAS Prototype Module)
MSG	Message Sequence Generator (PROSBAS Prototype Module)
MTR	Mid Term Review
NOF	Navigation Overlay Frame
NOFG	NOF Generator (PROSBAS Prototype Module)
NPA	Non Precision Approach
PM	Progress Meeting
PRN	Pseudo Random Noise
PROSBAS	Prototyping and Support to Standardisation of SBAS L1/L5 Multi-Constellation Receiver
RD	Reference Document
RF	Radio Frequency
RP	Receiver Prototype
SBAS	Satellite Based Augmentation System
SEM	System Effectiveness Model
SIS	Signal In Space
SME	SBAS Message Emulator (PROSBAS Prototype Module)
SoL	Safety of Life
SoW	Statement of Work
SPP	Service Provider Prototype
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
TTFF	Time To First Fix
TC	Test Cases
WBS	Work Breakdown Structure
WPD	Work Package Description

PROSBAS project is funded by the European Commission.

4. PROSBAS PROTOTYPE REQUIREMENTS

In this section we will provide the requirements of PROSBAS System Prototype.

4.1. GENERAL

REQ-PROSBAS-1-01-1A ***Configuration***

PROSBAS System Prototype shall accept as a minimum the following configuration data:

- *Number of Epochs of Simulation.*
- *Message types, subtypes and update intervals and associated parameters. (1, 2)*
- *Satellite constellation and associated parameters.*
- *Alert Generation Model and associated parameters.*
- *Satellite Monitoring Model and associated parameters (1).*
- *User Message Loss Model and associated parameters.*
- *Time-to-alert and associated parameters (3).*
- *User location (1).*
- *Degradation models and associated parameters (clock and integrity).*
- *Transition Modes configuration.*

Note 1: The service area is not a direct configuration parameter, but it will affect the number of ionosphere messages needed, the monitoring model and the user location.

Note 2: In principle, velocity code is assumed to be hard coded to 0.

Note 3: The associated parameters could include the latencies since the satellite is in alert until the system is able to report the event and the user receives the notification.

REQ-PROSBAS-1-02-1A ***Reproducibility***

PROSBAS System Prototype shall have a reproducible behaviour according to its configuration.

Note 1: If any of the models includes random generation, the seed(s) should be configurable so that the behaviour can be reproduced unless the user permits a random initialization.

Note 2: Reproducibility is not required among tool versions or for executions in different SW environments; for instance, PROSBAS System Prototype could use embedded random number generation SW that could depend on such SW environment.

REQ-PROSBAS-1-03-1A ***Implementation of SBAS ICD***

PROSBAS System Prototype shall implement the SBAS Enhanced ICD to be specified in the document "SIS Message Definition" [RD.1].

REQ-PROSBAS-1-04-1A ***SBAS ICD Flexibility***

PROSBAS System Prototype shall implement the flexibility required in order to analyse open points in the document "SIS Message Definition" [RD.1].

Note: Examples of flexibility required: use of determined message types as allowed by the ICD (use of MT6_1/2 or not), definition of degradation parameters, configuration of update intervals and timeouts.

REQ-PROSBAS-1-05-1A ***Reference Conditions Flexibility***

PROSBAS System Prototype shall implement the required reference conditions in order to perform experiment in accordance to the Reference Conditions established in the document "Reference Condition for the simulation" [RD.2].

Note: Flexibility is understood as the capability to cope at least with reference conditions identified in [RD.2].

REQ-PROSBAS-1-06-1A ***Configurability of key parameters characterisation***

PROSBAS System Prototype shall admit as part of its configuration key performance parameters and/or key implementation parameters that could affect the performance of the SBAS L1/L5 standard.

Note: Examples of these key parameters could be: UDREi statistics, dependency of UDREi with respect to observability of satellites, typical covariance matrix for MT28 or equivalent, potential degradation of satellite corrections versus latency and time, statistics versus latency and time, statistics on UERE, GIVEi statistics, degradation of GIVE… These parameters could be provided for certain combinations of constellations or frequencies.

REQ-PROSBAS-1-07-1A ***Nominal and degraded scenarios***

PROSBAS System Prototype shall generate and analyse Nominal and Degraded Scenarios.

Note: examples: satellite alerts, satellite loss and recovery, message loss, loss and recovery of a constellation, loss and recovery of a frequency.

REQ-PROSBAS-1-08-1A ***Expandability up to four constellations***

PROSBAS System Prototype shall process up to four GNSS constellations.

Note: examples: GPS, Glonass, Galileo, Compass

REQ-PROSBAS-1-09-1A ***L1 mode***

PROSBAS System Prototype shall accept a L1 back-up mode.

Note: This requirement has been deleted since it was part of former ICD_1 as in the case of GPS the ICD_1 was equivalent to SBAS L1 mode. It was agreed to implement only Enhanced ICD and the equirement has been removed.

REQ-PROSBAS-1-10-1A ***L5 mode***

PROSBAS System Prototype shall accept a L5 back-up only mode.

REQ-PROSBAS-1-11-1A ***Hosting Platform***

PROSBAS System Prototype shall be capable to run in a platform with the minimum specification detailed in [RD.6].

Note: It is foreseen to use a PC running a given distribution of Linux. The delivery would include all the SW and license with the appropriate configuration.

4.2. SERVICE PROVIDER PROTOTYPE

4.2.1. GENERAL

REQ-PROSBAS-2-01-1A ***Interface with Receiver Prototype***

PROSBAS Service Provider Prototype shall implement an interface with the Receiver Prototype as defined in the document "Service Provider / User Receiver Prototype ICD" [RD.3].

Note: This ICD shall specify the output from the Service Provider Prototype to the Receiver Prototype and the output from the Receiver Prototype to the System Prototype for final analyses. The flow from the Service Provider Prototype to the Receiver Prototype will typically include the SBAS messages in LogBook format. For more information on LogBook format, please refer to [RD.6].

4.2.2. MESSAGE SEQUENCE GENERATION

REQ-PROSBAS-3-01-1A ***SBAS Message Sequence Generation***

PROSBAS Service Provider Prototype shall generate a SBAS Message Sequence according to its configuration.

REQ-PROSBAS-3-02-1A ***Enhanced ICD configuration***

PROSBAS Service Provider Prototype shall generate a SBAS Message Sequence according to SBAS L1/L5 Enhanced ICD.

REQ-PROSBAS-3-03-1A *Configurable Number of Simulation Epochs*

PROSBAS Service Provider Prototype shall generate a SBAS Message Sequence during a configurable Number of Simulation Epochs.

REQ-PROSBAS-3-04-1A *Message types and subtypes in SBAS Message Sequence*

PROSBAS Service Provider Prototype shall generate a SBAS Message Sequence including the configured message types and subtypes.

Note: The fact that a SBAS L5 or L1/L5 is experimented will be then determined by the configuration of the messages to generate.

REQ-PROSBAS-3-05-1A *Message types update interval*

PROSBAS Service Provider Prototype shall generate SBAS Message Sequences according to the configured update intervals if possible.

Note 1: This requirement does not imply that the update intervals will be fulfilled; this will depend on the bandwidth.

REQ-PROSBAS-3-06-1A *Issue of Data in SBAS Message Sequence*

PROSBAS Service Provider Prototype shall include Issue of Data in the messages composing the SBAS Message Sequence.

REQ-PROSBAS-3-07-1A *IODE out of Satellite Monitoring Model*

PROSBAS Service Provider Prototype shall simulate the evolution of satellites IODEs.

REQ-PROSBAS-3-08-1A *PRN Mask*

PROSBAS Service Provider Prototype shall generate the SBAS Message Sequence according to the configured PRN Mask.

REQ-PROSBAS-3-09-1A *Alert Generation Model*

PROSBAS Service Provider Prototype shall insert alerts in the SBAS Message Sequence according to the configured Alert Generation Model.

4.2.3. SIMULATION OF USER MESSAGE LOSSES

REQ-PROSBAS-4-01-1A *Generation of Message Sequence with User Message Losses*

PROSBAS Service Provider Prototype shall generate User Message Sequences based on SBAS Message Sequence with simulation of User Message Losses depending on its configuration.

REQ-PROSBAS-4-02-1A ***User Message Losses in Alerts***

PROSBAS Service Provider Prototype shall generate User Message Sequences with message losses during alerts depending on its configuration.

4.2.4. ANALYSIS OF MESSAGE SEQUENCE

REQ-PROSBAS-5-01-1A ***Analysis of SIS Key Performance Indicators***

PROSBAS Service Provider Prototype shall analyse the defined SIS Key Performance Indicators.

REQ-PROSBAS-5-02-1A ***Static Bandwidth***

PROSBAS Service Provider Prototype shall determine the minimum bandwidth for the configured message sequence from a static point of view.

REQ-PROSBAS-5-03-1A ***Analysis of Message Sequences***

PROSBAS Service Provider Prototype shall analyse SBAS Message Sequences and User Message Sequences.

REQ-PROSBAS-5-04-1A ***Update Intervals***

PROSBAS Service Provider Prototype shall report statistics on the achieved Update Intervals in comparison with configured Update Intervals for a given Message Sequence.

REQ-PROSBAS-5-05-1A ***Schedulability***

PROSBAS Service Provider Prototype shall determine whether or not a Message Sequence has fulfilled the target Update Intervals.

Note: The Dynamic Bandwidth is obtained by configuring a filling message in a scenario without alerts so that its update interval is as big as possible still generating a schedulable message sequence. The dynamic bandwidth is the complement to the bandwidth occupied by the filling message.

REQ-PROSBAS-5-06-1A ***Message Timeouts***

PROSBAS Service Provider Prototype shall report statistics on the fulfilment of message timeouts in a Message Sequence.

Note: No timeouts are to be reported for messages linked to satellites that are not monitored at SBAS level, according to the reference information.

REQ-PROSBAS-5-07-1A ***Global Satellite Monitoring***

PROSBAS Service Provider Prototype shall report the monitorizarion status of satellites.

REQ-PROSBAS-5-08-1A ***Issue of Data Ephemeris evolution***

PROSBAS Service Provider Prototype shall report the evolution of IODEs with time

REQ-PROSBAS-5-09-1A ***Misleading Information***

PROSBAS Service Provider Prototype shall report events of monitored satellites at user level that were in alert at SBAS level after the Time-to-Alert.

Note: The origin of misleading information is twofold: the SBAS ICD model could be unable to report the alert on time or, having the alert been broadcast, the user failed to receive the messages or to match correctly the alert to the corresponding satellite.

4.2.5. GENERATION OF THE NOF

REQ-PROSBAS-6-01-1A ***L1/L5 NOF generation***

PROSBAS Service Provider Prototype shall generate a L1/L5 NOF with information on the following parameters based on specific models:

- *Clock corrections.*
- *Fast and slow clock corrections.*
- *Slow orbital corrections.*
- *Integrity parameters (UDRE or DFRE).*
- *Integrity Covariance parameters.*

REQ-PROSBAS-6-02-1A ***Operational Modes***

PROSBAS Service Provider Prototype shall be configured according to two operational modes.

REQ-PROSBAS-6-03-1A ***Operational Mode 1***

PROSBAS Service Provider Prototype shall accept as input L1 NOF simulated values and degradation model parameters for building a pseudo-realistic L1/L5 NOF

REQ-PROSBAS-6-04-1A ***Operational Mode 2***

PROSBAS Service Provider Prototype shall accept as input L1 NOF real values and degradation model parameters for building a realistic L1/L5 NOF.

REQ-PROSBAS-6-05-1A ***NOFG configuration parameters***

PROSBAS Service Provider Prototype shall accept the following configuration parameters and inputs:

- *In Operational Mode 1:*
  - *Configuration values for L1 parameters without time evolution.*
  - *Configuration values for performing a Gauss-Markov process to simulate an evolution with time for the corresponding L1 parameters.*
  - *Degradation parameters to evolve to L1/L5 parameters.*
- *In Operational Mode 2:*
  - *Real L1 SBAS data.*
  - *Degradation parameters to evolve to L1/L5 parameters*

Note: A detailed description of NOFG module can be found in Section 7.

4.2.6. ENCODIFICATION OF SBAS MESSAGE

REQ-PROSBAS-7-01-1A *SBAS Message generation*

PROSBAS Service Provider Prototype shall encode a SBAS-like message in hexadecimal format.

Note: The SBAS Message will be in accordance with the message sequence generated by MSG module and with the NOF generated by NOFG module.

Note 2: The format of the SBAS message data file will be identical to the LogBook.

4.3. USER RECEIVER PROTOTYPE

4.3.1. GENERAL REQUIREMENTS

REQ-PROSBAS-8-01-1A – *Interface with Service Provider Prototype*

PROSBAS Receiver Prototype shall implement an interface with the Service Provider Prototype as defined in Section 5.

4.3.2. INPUTS REQUIREMENTS

REQ-PROSBAS-9-01-1A – *Navigation Files*

PROSBAS Receiver Prototype shall accept as input RINEX navigation files for GPS, GLONASS, GALILEO and COMPASS constellations.

REQ-PROSBAS-9-02-1A – *SBAS Message Files*

PROSBAS Receiver Prototype shall accept as input SBAS-like messages in LogBook format (hexadecimal).

4.3.3. CONFIGURATION REQUIREMENTS

REQ-PROSBAS-10-01-1A – *Configuration File*

PROSBAS Receiver Prototype shall be configurable according to section 5.3.3.

REQ-PROSBAS-10-02-1A – *Reference Condition File*

PROSBAS Receiver Prototype shall use a Reference Condition file containing at least the following information:

- *Service Area*
- *Number of Epochs to simulate*
- *Satellites mask*
- *Number of frequencies*
- *Transition modes*

4.3.4. OUTPUTS REQUIREMENTS

REQ-PROSBAS-11-01-1A – *Udre.dat File*

PROSBAS Receiver Prototype shall generate the udre.dat file within the SBAS decoder (UDRE sub-module).

REQ-PROSBAS-11-02-1A – *Give.dat File*

PROSBAS Receiver Prototype shall generate the Give_XXXXX.dat files within the SBAS decoder (GIVE sub-module) when L5 mode is selected. These are hourly files where XXXXX is the scenario hour index, from 00000 to 99999.

REQ-PROSBAS-11-03-1A – *KPI Output Files*

PROSBAS Receiver Prototype shall generate the key performance indicator files within the User Level Analysis module containing at least the following parameters:

- *Protection Levels*
- *Position Errors*
- *xDOPs*
- *Continuity Risk*
- *Integrity*
- *Availability*

REQ-PROSBAS-11-04-1A – *KPI Figures of Merit*

PROSBAS Receiver Prototype shall generate at least the following figures of merit for the evaluation user performance:

- *Snapshot Protection Levels figures (xPLs vs Time)*
- *Statistics information of Protection Levels (max, mean, std. deviation, rms etc.)*
- *Position Error figures and statistics*
- *xDOPs figures and statistics*
- *Availability figures/maps*
- *Continuity figures/maps*

4.3.5. MAIN FUNCTIONALITIES REQUIREMENTS

REQ-PROSBAS-12-01-1A – *SBAS L1/L5 ICD implementation*

PROSBAS Receiver Prototype shall implement the Enhanced SBAS L1/L5 ICD.

REQ-PROSBAS-12-02-1A – *Protection Levels*

PROSBAS Receiver Prototype shall compute, from the broadcast SBAS message, the protection levels over a grid defined by configuration and visualize them in form of map (and text file).

REQ-PROSBAS-12-03-1A – *Position Errors*

PROSBAS Receiver Prototype shall compute the user position accuracy (xPEs) over a grid of users and visualize them in form of map (and text file).

REQ-PROSBAS-12-04-1A – *Availability*

PROSBAS Receiver Prototype shall compute, from the broadcast SBAS message, the Availability percentage over a grid defined by configuration and visualize them in form of map (and text file).

REQ-PROSBAS-12-05-1A – *Integrity*

PROSBAS Receiver Prototype shall compute, from the broadcast SBAS message, the Integrity percentage over a grid defined by configuration and visualize them in form of map (and text file).

REQ-PROSBAS-12-06-1A – *Continuity*

PROSBAS Receiver Prototype shall estimate the Continuity risk over a grid defined by configuration and visualize them in form of map (and text file).

REQ-PROSBAS-12-07-1A – *Statistics*

PROSBAS Receiver Prototype shall compute, additionally to the user-domain maps, statistic values. These statistics shall include at least: Mean, RMS, MAX, Standard Deviation.

5. PROSBAS SYSTEM PROTOTYPE HIGH-LEVEL DESIGN

In this section, the high-level architecture of PROSBAS Prototype will be presented.

5.1. PROSBAS SYSTEM PROTOTYPE OVERVIEW

PROSBAS Prototype is a tool developed by GMV for the EC that will be used to study the performances obtained with the ICD under analysis in the context of multi-constellation and multi-frequency SBAS.

In our solution, the PROSBAS System Prototype can be decomposed in Service Provider Prototype (SPP) and User Receiver Prototype (RP), as it is shown in Figure 5-1.

The main objectives of both prototypes are briefly explained below:

- The main purpose of SPP is to mitigate the technical risks inherent to the definition of SBAS L1/L5 and its adequacy for Europe objectives in EGNOS V3. For that, the SPP shall be able to emulate the message sequence of the SIS ICD under analysis, and will include the flexibility required to test the open points in the SIS and to generate message sequence in the defined reference conditions.

- The main objective of the RP is to have a design engineering tool able to assess the performances associated to a given model for the new SBAS L1/L5 SIS ICD. In particular, new or modified algorithm for the new SBAS SIS shall be designed and its final performances assessed. In consequence, the Receiver Prototype development plan should ensure the tool simulation capabilities needed in order to compute the Key Performance Indicators (KPIs) included in the Receiver Test Plan, with the required system-receiver internal interfaces for proper integration, and with the tight scheduling needs allowing a reasonable test campaign duration.

The Service Provider Prototype shall include enough information in the generated message to allow the execution of the User Receiver Prototype and to integrate later its output for the end-to-end performance analysis.

Following the work done in SUGAST, the SPP should generate reference information that, not being part of the SIS, will simplify the analyses of the different Key Performance Indicators (KPIs). The prototype generated by GMV for the EC in the SUGAST project, "Message Analysis Support Tools" (MAST), is the perfect starting point for developing the SPP. However, it will have to be enhanced to cover not only the specificities of the SIS ICDs and reference conditions, but also to fill in the message sequence with figures needed to perform quantitative assessment of the SIS and end-to-end performance. In the MAST version generated in SUGAST the message sequence was populated essentially with IODs, satellite PRNs, and UDRE values focusing only in whether or not the visible satellites were monitored. With this information, it was possible to extract important conclusions of the ICD models analyzed in terms of bandwidth, fulfillment of update intervals and timeouts, overall monitored satellites, monitored satellites in the service area and worst user locations and robustness to degraded scenarios with alerts and message loss.

In the current activity, it will be necessary to include, as a minimum, quantitative values for the UDREs themselves, covariance matrices, and degradation coefficients for aged satellite clock corrections. Other parameters could be needed depending on the SIS ICD and the model for the user algorithms. In particular, it could be needed to feed the prototype and the user algorithm prototype with reference values obtained from EGNOS V3 activities.

The SPP will be complemented with the analysis tools required to assess the performance of the model.

The main missions of the RP are:

- To implement the SBAS L1/L5 ICDs and User Standard models (defined under Tasks T2.1 and T2.2 for RF, and under T2.3 for User algorithms definition).
- Analyse the performance required to implement the proposed algorithms, propose hardware/software implementation and verify that receiver can meet the performance requirements in nominal and degraded cases.
- Analyse and support the feasibility and design activities for the allocation of the user performances both from a global and individual components contributions (receiver noise, multipath, etc).

In addition to the Key Performance Indicators covered by SPP, it will be mandatory to consider SBAS end-to-end parameters, with special attention to availability, continuity and time to alert.

As it is shown in the following Figure 5-1, in our solution we distinguish SPP Level from RP Levels.

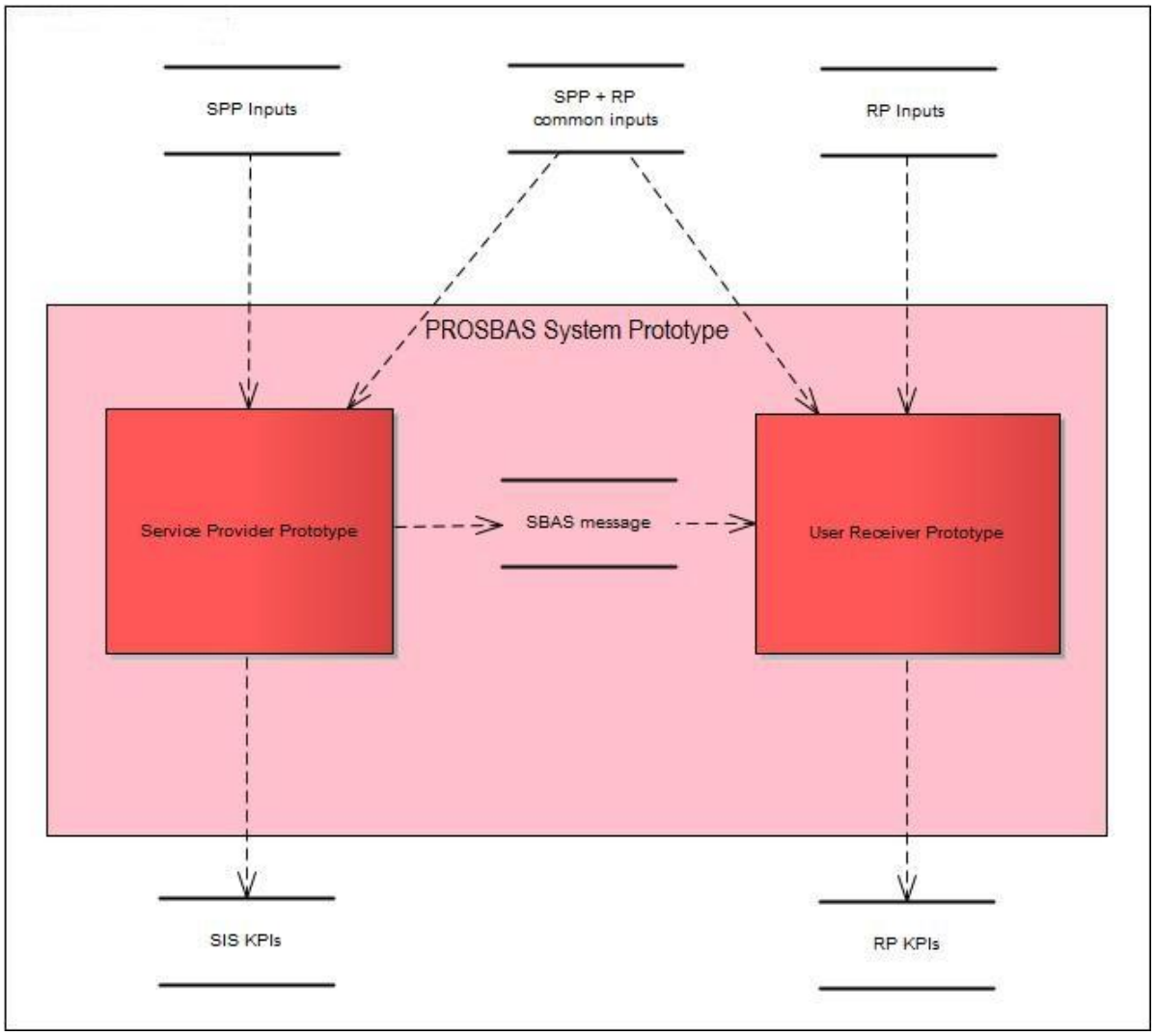


Figure 5-1: PROSBAS System Prototype Architecture

The end-to-end executions are based on three main functional blocks:

- End-to-End reference conditions. This block should provide all the configurability inputs and reference conditions needed for the processing to be performed in both Service Provider and Receiver prototypes.

- The message sequence generator and message builder, or what we refer to as Service Provider. This part of the prototype is in charge of computing sequence, format and content of the SBAS messages according to a given SBAS L1/L5 SIS ICD.
- The Receiver Prototype. It is in charge of testing the functionalities and objectives described above. There were basically two potential starting points for the development of the receiver prototype.

At very high level, the receiver prototype is composed by three sub-modules

- A module in charge of decoding and extracting the content of the SBAS messages;
- An integrity user processing module, in charge of computing/analyse the protection levels associated to the navigation solution, which are the base for performance parameters such as availability and continuity;
- A visualization module in order to produce the plots and dissemination needed for the analysis and reporting activities of the test receiver prototype.

The performance obtained by the system will be the combination of the performance of the Servicer Provider, generating the signal in space, and the Receiver, implementing the user algorithms for a given signal in space in their environment.

The experimentation that will be performed with PROSBAS Prototype will allow the study of the Key Performance Indicators (KPIs) in order to analyze the performances obtained with the ICD under analysis.

The SIS KPIs are as follows:

- SIS-KPI-01: Static bandwidth.
- SIS-KPI-02: Fulfillment of message updates intervals.
- SIS-KPI-03: Fulfillment of message timeout intervals.
- SIS-KPI-04: Number of monitored satellites in the service area.
- SIS-KPI-05: HDOP/VDOP in service area.
- SIS-KPI-06: Time to alert.
- SIS-KPI-07: Robustness upon alerts.
- SIS-KPI-08: Robustness upon message loss.
- SIS-KPI-09: Robustness/efficiency regarding interpretation of issues of data.
- SIS-KPI-10: Robustness upon transition modes.

The Receiver KPIs are summarized in the following:

- RP-KPI-01: Snapshot HPL and VPL.
- RP-KPI-02: HDOP/VDOP in service area and in the worst user location.
- RP-KPI-03: Fulfillment of SBAS L1/L5 mission Accuracy/Availability/Continuity/Integrity.
- RP-KPI-04: Fulfillment of EGNOS V3 mission in different service areas and configurations.

Let us remark that one can consider that the System KPIs as the union of the SPP KPIs and Receiver KPIs.

5.2. SERVICE PROVIDER PROTOTYPE

In this subsection the high-level design of the SPP will be described.

As explained before, the SPP will be based on GMV tool developed for the EC in SUGAST project: MAST [RD.4]. In Annex 9. MAST tool will be described. The architecture of SPP will be presented in the following.

5.2.1. SERVICE PROVIDER ARCHITECTURE

In this section the SPP architecture will be described.

In the first subsection, a general description will be provided, presenting the data flow diagram, the SPP modules decomposition, the configuration and inputs, the internal interfaces and the outputs of SPP. Then the different SPP modules, configuration and inputs, internal flows and outputs will be explained in more detail in different sections.

5.2.1.1. General description

The SPP consists of several modules, running in a Linux Operating System. Some of these modules will be executed sequentially and independently using input data stored in files and producing output data that is also stored in files that, depending on the case, could serve as input data files to other modules. Nevertheless, other modules are integrated and are run using a single executable.

The SPP will have two Operational Modes. As it will be explained, the three main tasks that perform the SPP are:

- Generate a message sequence according to the reference conditions as it is done with MAST.
- Fill this message sequence with pseudo-realistic/realistic data for transmitting the SBAS messages to the RP.
- Analyze the message sequence performances.

The data necessary to fill the message sequence in order to build the SBAS messages could be either simulated (with baseline values and evolution models) or it can be real, as an external (to PROSBAS Prototype) input (extracted externally from different means as EGNOS L1 SBAS Messages or using GMV tools as for example ***magicSBAS,*** or from EGEP test-beds executions as HISTB). These two options imply the two operational modes that will be called from herein as follows:

- **Operational Mode 1** in case of simulated L1 NOF (from baseline values and evolution models).
- **Operational Mode 2** in case of real L1 NOF (extracted externally to PROSBAS Prototype from e.g. HISTB).

It is worth noticing that ***magicSBAS*** is a state-of-the-art, multiconstellation, operational Satellite Based Augmentation System testbed developed by GMV to offer SBAS regional differential corrections and non-safety critical integrity augmentation to any interested region. For further information, please refer to [RD.5].

The data flow diagram of SPP can be checked in Figure 5-2.

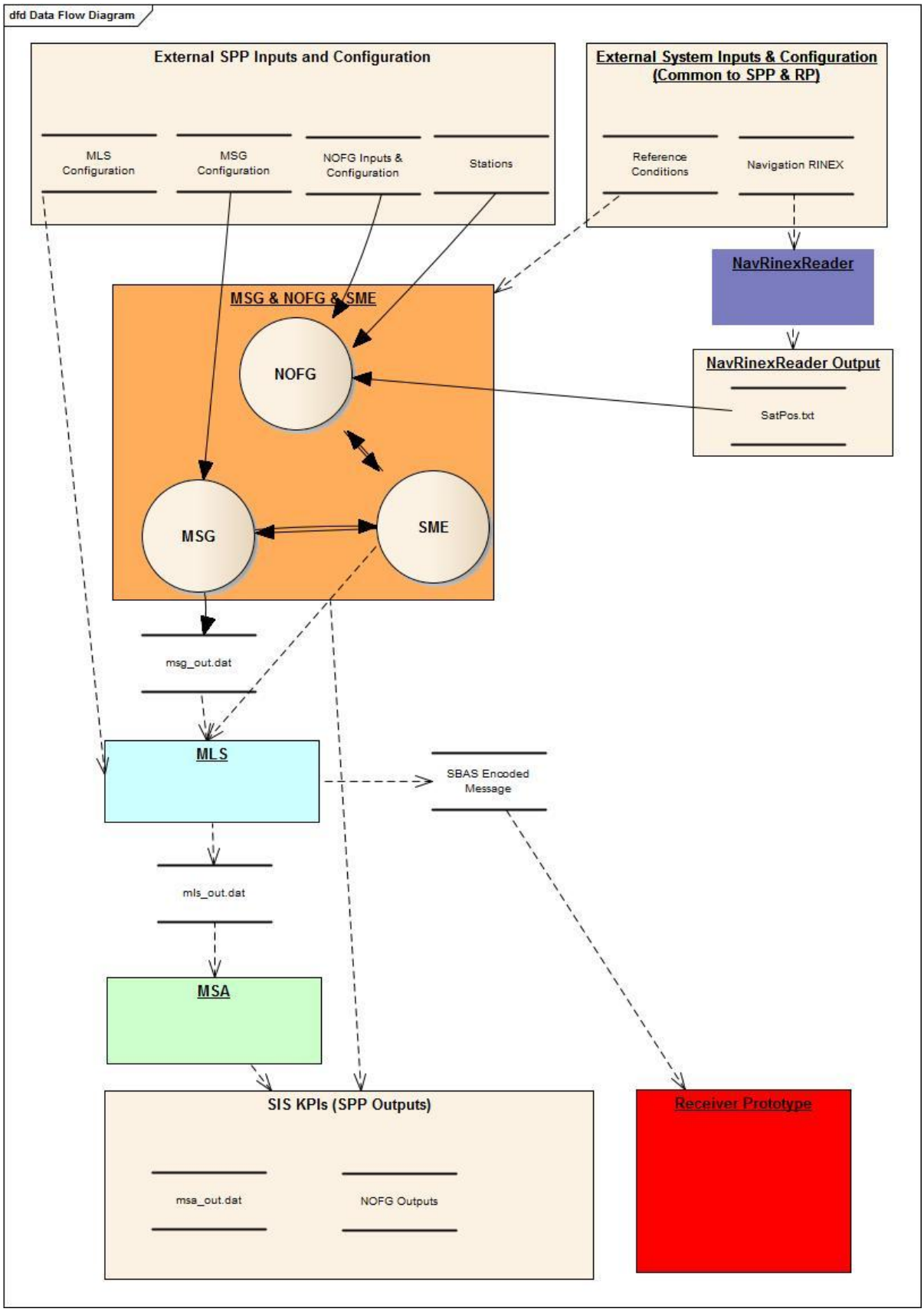


Figure 5-2: SPP High-level Architecture

The SPP architectural design will consist on 5 modules:

- Message Sequence Generator (**MSG**).
- Message Loss Simulator (**MLS**).
- Message Sequence Analyzer (**MSA**).
- NOF Generator (**NOFG**).
- SBAS Message Emulator (**SME**).

MSG, NOFG and SME modules are integrated and once compiled, there is only one executable called SME.

In addition, there is an ancillary stand-alone module, shared with the RP, which is in charge of processing the navigation RINEX files and generating the satellite position file:

- Navigation RINEX files reader (**NavRinexReader**).

A detailed description of the SPP interfaces can be found in [RD.3] (for an updated version of the inputs and outputs of PROSBAS Prototype, please refer to [RD.6]). Nevertheless, in the following we will summarize this issue.

At highest level, the inputs of the System Prototype could be divided into three categories: inputs to the SPP, inputs to the RP and inputs that are common to both SPP and RP.

In a lower level description, the inputs to the SPP could be decomposed into the following categories that correspond to either a single data file or to several data files:

- MSG Configuration: msg_cfg.dat.
- MLS Configuration: mls_cfg.dat.
- NOFG Inputs and configuration (Operational Mode 1 or Operational Mode 2): several data files.
- Reference Conditions: ref_cond_cfg.dat
- Ephemeris files: RINEX Navigation Files.
- Reference Stations: stations.cfg

In addition, there are several internal flows:

- Message Sequence without losses and SIS Reference Data: msg_out.dat
- Message Sequence with losses and SIS Reference Data: mls_out.dat

Finally, there are the following SPP outputs:

- SIS KPIs: msa_out.dat (if MLS is not activated) and msa_out.dat_SPP and msa_out.dat_RP (if MLS is activated).
- NOFG products
- SBAS Messages in LogBook format for the RP

Before entering in a more detailed description of the different modules and interfaces, it will be briefly described the sequential execution of the different SPP modules. Nevertheless, let us remark that PROSBAS Prototype can be run using a driver and the sequential execution of the different modules can be transparent to the user as it is explained in [RD.6].

- The first modules to be executed are the **MSG & NOFG & SME** modules.

  MSG module is inherited from MAST but several changes have been performed, as building the capacity to consider Enhanced ICD.
  MSG module generates a message sequence according to Enhanced ICD and reference conditions, simulates alerts etc…
  This module is fed with the following input files:

- MSG Configuration (with relevant information about the Alert Generator, information on the messages configured for Enhanced…): msg_cfg.dat.
- Reference Conditions: a file containing relevant information for both SPP and User Prototype as the service area, number of epochs of the simulation, Update Intervals and Time-outs of the different messages…: ref_cond_cfg.dat.

MSG module generates as output:

- Message Sequence without losses and SIS Reference Data with the message sequence and other relevant information: msg_out.dat

  In addition MSG module prepares the message sequence in order that SME fills it with NOF content to produce the SBAS Messages.

NOFG module is in charge of generating a pseudo-realistic/realistic NOF (depending on the operational mode) in order to fill the content of the message sequence from either baseline values and evolution models or from external Real L1 NOF and models to take into account L1/L5. NOFG module receives the following inputs:

- Satellite positions file (Provided by the NavRinexReader module): SatPos.txt
- Reference Stations positions: stations.cfg
- Reference Conditions: ref_cond_cfg.dat
- NOFG Inputs and Configuration (Operational Mode 1 or Operational Mode 2)

NOFG module prepares the NOF for SME module to fill the message sequence in order to generate the SBAS Messages.

In addition NOFG module generates several output files with relevant information concerning the NOF.

The detailed design of NOFG module will be presented in Section 7.

SME Module is in charge of driving the three modules integrated, communicating with MSG module and NOFG module. Very roughly, SME Module receives the message sequence from MSG module and receives the NOF content from NOFG module in order to fill the message sequence with NOF content in order to produce the SBAS Messages.

- The next module to be executed is **MLS** module.

  MLS module is an optional module in charge of simulating message losses at user level from the message sequence generated with MSG.

  This module takes as input outputs of MSG & NOFG & SME modules and a configuration files:
  - Message Sequence without losses and SIS Reference Data: msg_out.dat
  - LogBooks with SBAS Messages.
  - MLS Configuration, with configuration data to simulate the loss of messages at user level: mls_cfg.dat
  - Reference Conditions: ref_cond_cfg.dat

  MLS module generates as output a file mls_out.dat, very similar to the output of MSG module but changing a field in the message sequence indicating that the corresponding message has been simulated as lost at user level. In addition, it also generates again the same LogBooks as SME module but deleting the SBAS messages that have been simulated as lost at user level.

- The last SPP module to be executed is the **MSA** module.

  MSA module receives as input:
  - Message with/without losses and SIS Reference Data: mls_out.dat/msg_out.dat.

  Then, MSA analyzes the message sequence and computes the SIS KPIs, generating an output file:

- SIS KPIs: msa_out.dat (if MLS is not activated) and msa_out.dat_SPP and msa_out.dat_RP (if MLS is activated).

In Table 5-1 it is shown a summary of the High-Level architecture of the SPP.

Table 5-1: SPP High-Level architecture summary

Module	Service Provider Prototype
Type	Composite module
Function	Generate SBAS message sequences and reference data according to the configuration, for SBAS L1/L5 Enhanced ICD model, including generation of alerts. Generate a simulated or real L1 NOF that evolves to L1/L5 in order to fill the message sequence with content. Encode the NOF data in the message sequence to build realistic SBAS messages. Simulate user message losses in the SBAS message sequence to generate user message sequences. Analyse the performance of the message sequences.
Components	Message_Sequence_Generator (MSG) NOF_Generator (NOFG) SBAS_Message_Emulator (SME) Message_Loss_Simulator (MLS) Message_Sequence_Analyser (MSA)
Usage	SPP is a collection of tools. One of them, MSG, is in charge of generating SBAS message sequence with the needed configuration capabilities. The generation of L1/L5 NOF to fill the message sequence with content is performed with NOFG. The encoding of NOF data generated in message sequence to build realistic SBAS messages is performed with SME. Its output is optionally used by MLS to simulate the loss of messages at user level. The performance of the message sequence generated, either with or without simulated message loss, is determined by MSA.
Input	MSG Configuration: msg_cfg.dat MLS Configuration: mls_cfg.dat Reference Conditions: ref_cond_cfg.dat Satellite positions file: SatPos.txt Reference Stations positions: stations.cfg NOFG Inputs and Configuration (Operational Mode 1 or Operational Mode 2): several input data files
Output	SBAS Message Sequence and Reference Data: msg_out.dat User Message Sequence: mls_out.dat SIS KPI Data: msa_out.dat NOF data: several files containing NOF information SBAS messages: LogBooks

In the next figure it is shown the SPP first level decomposition.

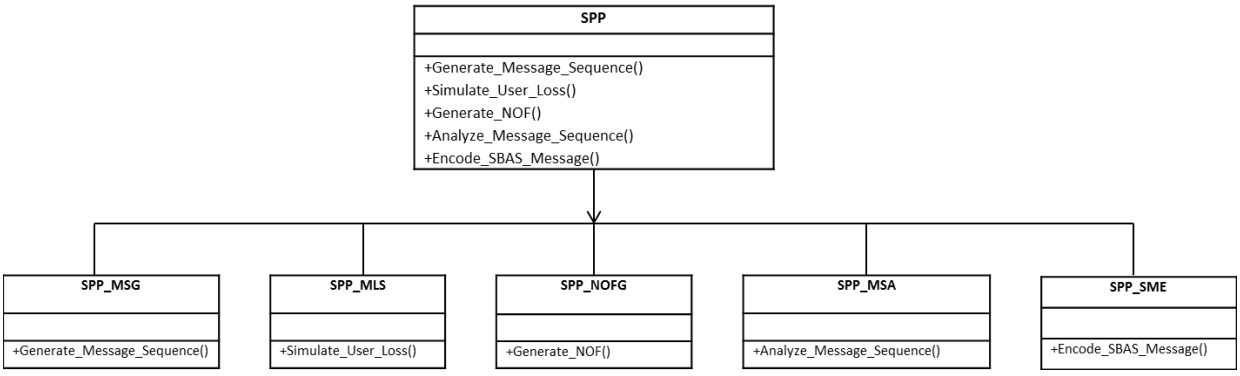


Figure 5-3: SPP first level decomposition.

5.2.1.2. Modules

As stated before, the SPP consists of five modules (MSG, MLS, NOFG, SME and MSA, three of them integrated) plus an ancillary stand-alone module to process navigation RINEX files (NavRinexReader).

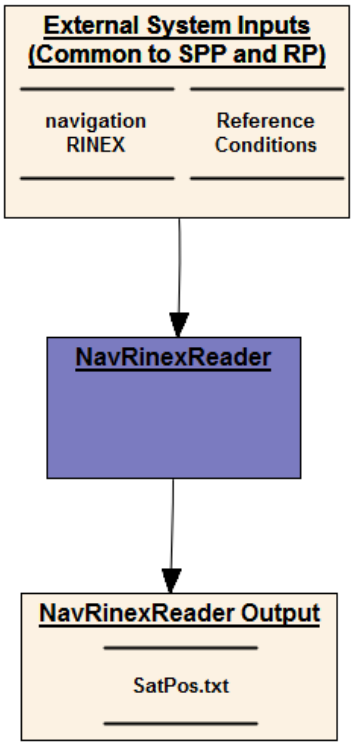
In the following subsections, it will be described in more detail the different modules.

5.2.1.2.1. Navigation RINEX files reader (NavRinexReader)

The NavRinexReader is a C++ module that runs independently of any other module, and may be used in either the SPP, in the RP or in both.

It processes navigation RINEX files and generates the satellite positions according to the satellites and dates specified in the reference conditions (ref_cond_cfg.dat) file, which is used as an input for the SPP and RP modules.

Figure 5-4: NavRinexReader data flow.



NavRinexReader module receives the following input files:

- Reference conditions (external input): ref_cond_cfg.dat
- Ephemeris Files (navigation RINEX files).

NavRinexReader module generates the following output:

- Satellite positions file: SatPos.txt

The input and output file format of the NavRinexReader module will be detailed in PROSBAS document "Operating Manuals for Prototype and Tools" [RD.6], delivered for MTR.

Table 5-2 summarizes the architecture of the NavRinexReader Module, and Figure 5-4 shows the basic data flow in which the module is involved.

Table 5-2: NavRinexReader architecture summary.

Module	Navigation RINEX Reader (NavRinexReader)
Type	Executable.
Function	Computes the satellite positions for every epoch, according to the reference conditions configuration. It also provides the IODE with which the position has been computed. The reference conditions configuration includes: • Initial time of the simulation • Number of Epochs of Simulation. • Satellite constellation and associated parameters.
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.
Input	Reference Conditions. Navigation RINEX files.
Output	Satellite positions file: SatPos.txt

5.2.1.2.2. Message Sequence Generator (MSG)

MSG is a module written in C++ language, based in MAST_MSG module.

As stated before, MSG module is in charge of generating a message sequence without NOF content and SIS reference data according to the messages configuration of Enhanced ICD and according to the reference conditions.

MSG module receives two external input data files:

- MSG configuration (external input): msg_cfg.dat
- Reference conditions (external input): ref_cond_cfg.dat

MSG module generates the following outputs:

- Message Sequence without losses and SIS Reference Data (to MLS/MSA depending on if optional module MLS is executed): msg_out.dat. In addition, it internally communicates the message sequence to SME module.

In Table 5-3 it is summarized the architecture of MSG module while in Figure 5-5 it is shown the MSG decomposition and in Figure 5-6 it is shown the MSG use diagram.

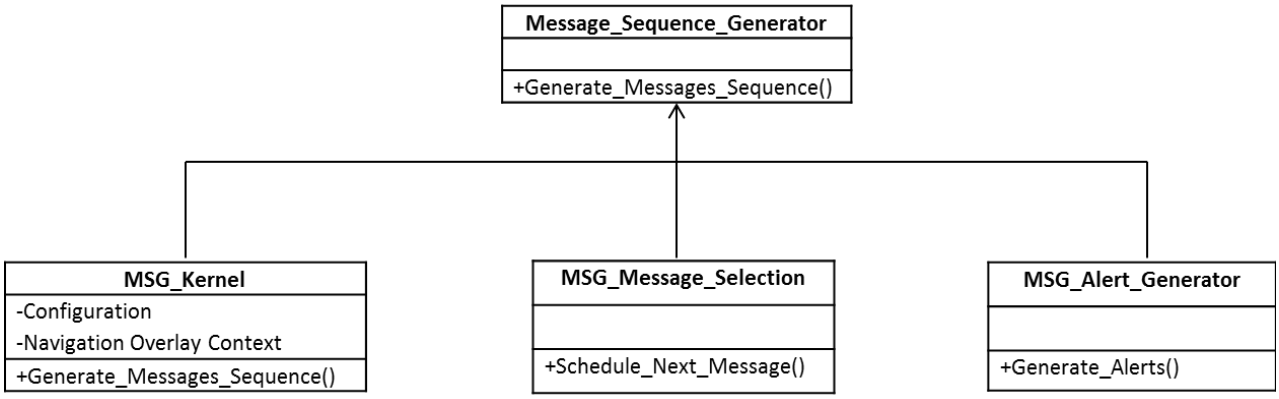


Figure 5-5: Message Sequence Generator decomposition.

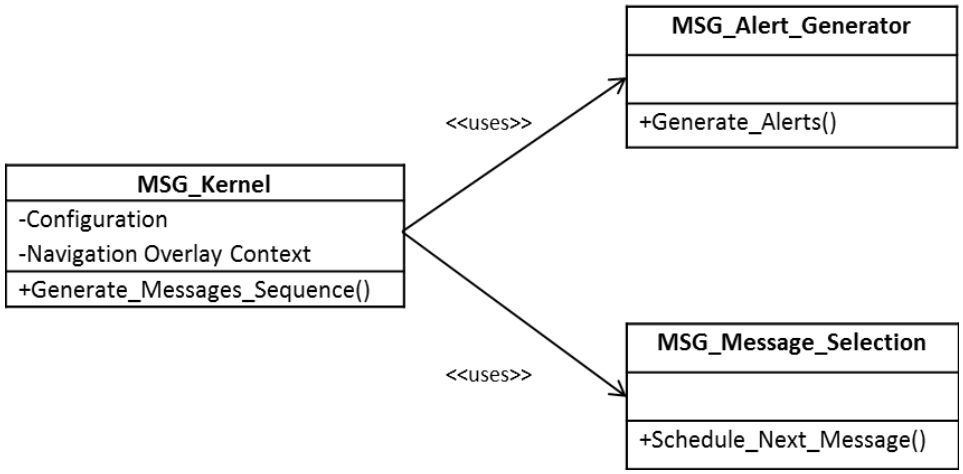


Figure 5-6: Message Sequence Generator use diagram

PROSBAS project is funded by the European Commission. The results are the property of the European Commission. No distribution or copy is permitted unless prior authorization is given by the European Commission.

Table 5-3: MSG architecture summary.

Module	Message_Sequence_Generator (MSG)
Type	Integrated module in MSG & NOFG & SME module. Composite module.
Function	Generate SBAS message sequences and reference data according to the configuration, for Enhanced ICD, including generation of alerts. The module will consider IODs and extended IODs as appropriate. PRNs will be associated to the message types as appropriate. The configuration includes: • Number of Epochs of Simulation. • Message types, subtypes and update intervals and associated parameters according to Enhanced ICD. • Satellite constellation and associated parameters. • Alert Generation Model and associated parameters. • Time-to-alert and associated parameters. • User location. • Seed for random-number generator (if any). The reference data includes: • Active IODP • Assignment mask to PRN. • Alerted UDRE satellites, if any, with age of the alert and indication of the number of messages broadcast with the alert. • Elapsed epochs since last alert broadcast.
Components	MSG_Kernel MSG_Message_Selection MSG_Alert_Generator

Module	Message_Sequence_Generator (MSG)
Usage	The MSG module is integrated with SME and NOFG modules that once compiled, produce a single executable called SME. It is composed of different classes to perform its function. The MSG module generates a message sequence and reference data according to its configuration and the function described above. The MSG_Kernel component is in charge of reading the configuration and generating the output, after using appropriately its sibling classes. At initialization, the MSG_Kernel read the configuration parameters. Every epoch, the MSG_Kernel will determine if an alert is to be broadcast. Then, it will call the MSG_Message_Selection module to determine the next message to be broadcast. Pseudo-random numbers, if any, will be generated depending on a configurable initial seed, so that the execution is repeatable. (This is particularly relevant for the MSG_Alert_Generator module.)
Input	Configuration. Reference Conditions.
Output	SBAS Message Sequence without losses and Reference Data.

MSG Invocation: N/A

5.2.1.2.3. Message Loss Simulator (MLS)

MLS module is a module in shell script *dash* language and using *mawk*. It was already implemented in MAST tool. This module is in charge of simulating message losses at user level to the message sequence generated with MSG according to configuration parameters. Note that the execution of this module is optional depending on whether the simulation of loss of messages at user level is desired to be studied.

Let us remark that Message Loss Simulator module simulates the loss of SBAS Messages on L5 GEO stream. MLS module is a part of Service Provider Prototype module since the design was simplified, acting over the SBAS Messages generated by Service Provider Prototype simulating the loss of some of them. The modified SBAS Messages sequence with gaps is the input of the Receiver Prototype.

MLS module receives the following inputs:

- Message Sequence without losses and SIS Reference Data (from MSG): msg_out.dat
- MLS configuration (external): mls_cfg.dat
- Reference conditions: ref_conf_cfg.dat
- LogBook files.

MLS module generates the following output data:

- Message Sequence with losses and SIS Reference Data (for MSA and SME modules): mls_out.dat

In the following table the MLS architecture is summarized:

Table 5-4: MLS architecture summary.

Module	Message_Loss_Simulator (MLS)
Type	Executable. Single module.
Function	Simulate message loss at user level upon SBAS message sequences according to the configuration. User message losses will be forced during alerts according to the configuration.
Components	N/A
Usage	MLS is an executable that takes as input the output generated by MSG and generates a similar output with flags indicating that specific message epochs are to be considered lost at user level. In addition, it receives LogBooks with SBAS Messages and simulates the loss of messages at user level generating LogBooks identical to the ones received but without the messages at the epochs when a loss has been simulated. MLS reads a configuration file with the probability of losing messages, including specific configuration parameters for the probability of loss on alert sequences. Pseudo-random numbers, if any, will be generated depending on a configurable initial seed, so that the execution is repeatable.
Input	Configuration. SBAS Message Sequence and SIS Reference Data. LogBook files
Output	User Message Sequence with losses.

MLS module is a simple script and is not decomposed in sub-modules.

MLS Invocation: The invocation of MLS module will be detailed in PROSBAS document "Operating Manuals for Prototype and Tools" [RD.6] that will be delivered for MTR.

5.2.1.2.4. NOF Generator (NOFG)

Due to the complexity of NOFG module, the detailed design of NOFG will be explained in Section 7.

NOFG module is a module not present in MAST that is in charge of generating an L1/L5 NOF in order to fill the message sequence with content to build the SBAS messages. Two operational modes are expected. The first operational mode consists of baseline parameters and evolution models to build the L1 NOF and then degrade to L1/L5 NOF. The second operational mode consists of real L1 NOF generated externally to PROSBAS Prototype from L1 SBAS messages and models to degrade to L1/L5 to take into account e.g. deltaFC effect or UDRE border effect. Note that in the second operational mode, the configuration of PROSBAS Prototype should be coherent with the external real L1 data.

NOFG module receives four kind of external inputs:

- Reference conditions (external input).
- RINEX Navigation Data Files (external input).
- Reference Stations positions (external input).
- NOFG Inputs and Configuration (external input. Operational Mode 1 or Operational Mode 2).

In addition, NOFG module receives an internal flow as input that is an output from MSG module:

- Message Sequence without losses and SIS Ref. Data

NOFG module generates the following output:

- NOF (to SME module).
- Several output data files containing NOF information.

NOFG architecture is summarized in the following table:

Table 5-5: NOFG architecture summary.

Module	NOF Generator (NOFG)
Type	Executable. Composite module.
Function	Build L1/L5 NOF to fill the message sequence with content in order to generate realistic SBAS messages. Depending on the Operational Mode, this module receives as inputs baseline parameters and evolution models or real L1 NOF and models. In addition, it also receives as input reference conditions, Navigation Files, Reference Stations positions and SV monitoring and Alert information.
Components	NOFG Detailed design will be discussed in Section 7.
Usage	NOFG Detailed design will be discussed in Section 7.
Input	Reference Conditions. Ephemeris. Reference Stations positions. Message Sequence without losses and SIS Reference Data NOFG Inputs and Configuration
Output	NOF

SPP_NOFG Invocation: N/A

It is worth noticing here that NOFG Module will also manage the transition modes configured in Reference Conditions file corresponding to the loss of an SV and the loss of an entire constellation.

5.2.1.2.5. SBAS Message Emulator

SME is a module not implemented in MAST. This module is in charge of driving MSG and NOFG executions and of encoding the NOF content in the message sequence in order to generate a SBAS message for the User Receiver Prototype.

SME module receives two internal flows as inputs:

- NOF (from NOFG module).
- Message sequence (from MSG module).

SME module generates the following output:

- SBAS messages encoded in LogBook format.

Information related with the messages format can be found in Section 5.3.2.2 for the SBAS Message Decoder Module of RP that can also be useful for the SME SPP module.

In the following table a summary of the architecture of SME module is provided:

Table 5-6: SME architecture summary.

Module	SBAS Message Emulator (SME)
Type	Executable. Composite module.
Function	To drive MSG and NOFG executions. To fill NOF content in message sequence to obtain realistic SBAS messages.
Components	SME_Kernel SME_Enhanced_ICD SME_Logbook_Writer
Usage	SME is an executable that drives the execution of MSG and NOFG modules. It takes as input the SBAS message sequence without losses and SIS Reference data, as well as the NOF, prepares the information and fills the message sequence with NOF content. Then, it writes the SBAS messages in a file in LogBook format for the Receiver Prototype. NAME ./SME - PROSBAS SBAS Message Emulator 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 ./. -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 ./ -s sequence_file_path Set output file path for the SBAS Message Sequence. Default path is ./msg_out.dat -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.
Input	Message Sequence with/without losses and SIS Reference data. NOF.

Module	SBAS Message Emulator (SME)
Output	SBAS Messages (LogBook format)

In Figure 5-7 it is shown the SME decomposition.

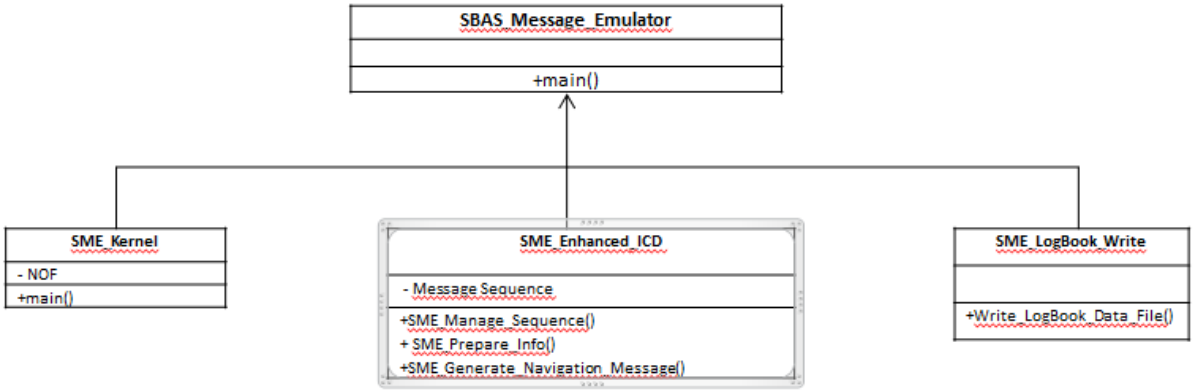


Figure 5-7: SBAS Message Emulator Decomposition.

In the following Figure it is shown the SME use diagram.

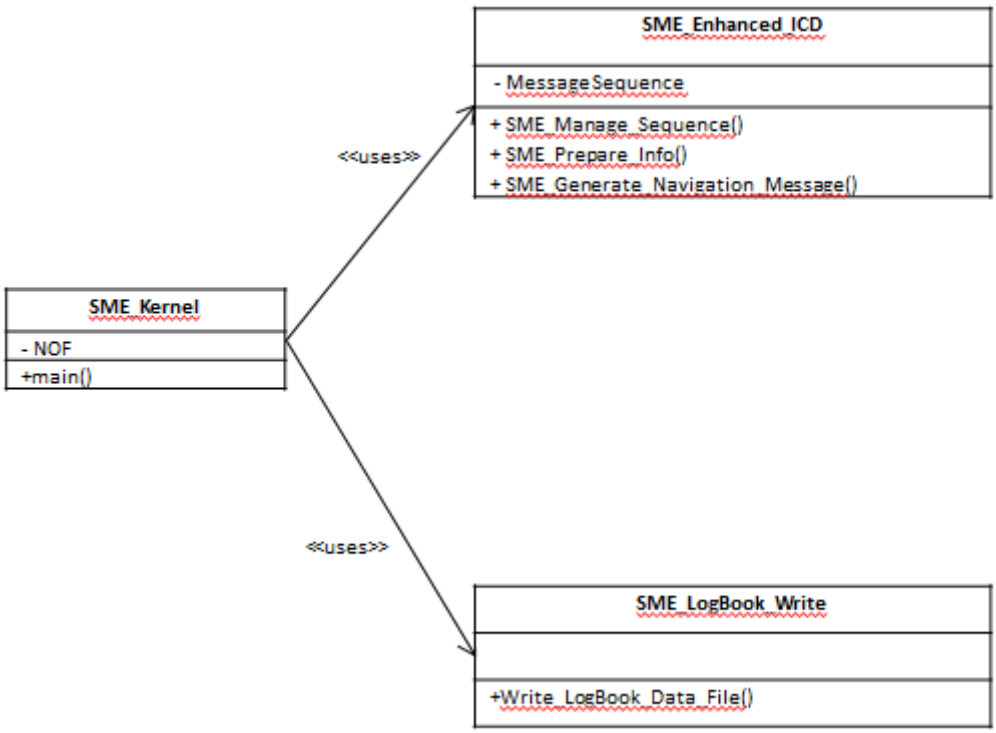


Figure 5-8: SBAS Message Emulator use diagram

SME module is decomposed in the following sub-modules:

- SME_Kernel
- SME_Enhanced_ICD: This sub-module is in charge of reading the message sequence generated with MSG and to manage the message type that should be written each epoch. It uses the functions SME_Prepare_Info (in charge of preparing information to fill SBAS messages from L1/L5 NOF generated by NOFG), and SME_Generate_Navigation_Message (in charge of encoding the bits of the SBAS message each epoch, according with the message sequence managed by SME_Manage_Message_Sequence sub-module and according to the information prepared by SME_Prepare_Info sub-module).
- SME_Logbook_Writer: This sub-module is in charge of writing the Logbook file (hexadecimal).

SME Invocation: The invocation of SME module is detailed in PROSBAS document "Operating Manuals for Prototype and Tools" [RD.6] that will be delivered for MTR.

5.2.1.2.6. Message Sequence Analyser (MSA)

MSA module is a C++ module already implemented in MAST. Nevertheless, certain modifications were performed (for example, removing the analysis concerning changes of mask). This module is in charge of analyzing the message sequence with or without losses generated with MLS or MSG and computing SIS KPIs data. Several options in the Linux executions are used to obtain more or less detailed output information.

MSA module receives the following inputs:

- Message Sequence with/without losses and SIS Reference Data (from MLS/MSG depending on if MLS module is executed or not): mls_out.dat/msg_out.dat

MSA module generates the following output:

- SIS KPIs (external final output): msa_out.dat (if MLS is not activated) and msa_out.dat_SPP and msa_out.dat_RP (if MLS is activated).

The MSA output contains the following information: Update/Timeout Report, Static BW Report, Robustness Report.

A summary on the MSA architecture is presented in next table:

Table 5-7: MSA architecture summary.

Module	Message_Sequence_Analyzer (MSA)
Type	Executable. Terminal module.

Function	Analyse a given message sequence and its accompanying reference data, according to the configuration.
	The analyses will be done either on SIS message sequences or user message sequences (i.e., SBAS message sequences after the user message losses have been simulated).
	The analysis will consist in the provision of SIS KPI data, including:
	• Static Bandwidth data (based on the SBAS L1/L5 Enhanced ICD configuration, not in message sequences).
	• Update and timeout data, including:
	  ○ Update intervals and comparison against timeouts.
	  ○ Schedulability.
	• Misleading information (monitored satellites that should be alerted or not monitored).

Components	N/A

Module	Message_Sequence_Analyzer (MSA)
Usage	MSA is an executable that takes the output provided by MSG or, optionally, by MLS and analyses the performance of the message sequence. SIS message sequences and user message sequences are analysed together with the configuration and the reference data. 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 GMV & European Commission 2013, GMV & European Commission property; all rights reserved.
Input	Configuration (option in the Linux execution for providing less or more information). Message Sequence with/without losses and SIS Reference Data.
Output	KPI Data.

MSA module is not decomposed in sub-modules.

MSA Invocation: Run MSA module with invocation options if necessary, using as input the output file from MSG or MLS (depending on if message losses have been simulated). The invocation of MSA module will be detailed in PROSBAS document "Operating Manuals for Prototype and Tools" [RD.6] that will be delivered for MTR.

Several options for invocation of MSA module produce less or more detailed information. For more details on the output that can be generated by MSA module and the invocation options, please refer to [RD.6].

5.2.1.3. Inputs and Configuration

In this section we will briefly describe the different SPP input and configuration data files. For a more detailed description of PROSBAS Prototype interfaces, please refer to [RD.6]. As stated before, the SPP prototype receives several data files as input that can be divided into different categories (corresponding to a single input data file or to several input data files).

5.2.1.3.1. MSG Configuration

MSG Configuration is a single input data file that is used to configure MSG module.

It is an external input data file used only by MSG module, with name "*msg_cfg.dat*".

The different input parameters contained in this file can be checked in [RD.6].

5.2.1.3.2. MLS Configuration

MLS Configuration is a single input data file that is used to configure MLS module.

It is an external input data file used only by MLS, with name "*mls_cfg.dat*".

The different input parameters contained in this file can be checked in [RD.6].

5.2.1.3.3. NOFG Inputs and Configuration in Operational Mode 1

NOFG Inputs and Configuration in the case Operational Mode 1 is selected consist of several input data files that are used as input for NOFG module.

They are external input data files used only by NOFG.

The different input parameters contained in these files can be checked in [RD.6].

5.2.1.3.4. NOFG Inputs and Configuration in Operational Mode 2

NOFG Inputs and Configuration in the case Operational Mode 2 is selected consist of several input data file that are used as input for NOFG module.

They are external input data files used only by NOFG.

The different input parameters contained in these files can be checked in [RD.6].

5.2.1.3.5. Reference Conditions

Reference Conditions consist of an input data file with configuration information relevant for both SPP and RP. It is a single external data file with name "*ref_cond.dat*" and at SPP level, it serves as input for MSG, MLS and NOFG.

The different input parameters contained in these files can be checked in [RD.6].

5.2.1.3.6. Ephemeris files

Ephemeris files consist of RINEX Navigation files with the ephemeris of the constellations used. They are used to compute the SVs positions during the simulation by NavRinexReader module.

These data files are used as inputs for the NOFG module as well as for the User Receiver Prototype.

For further information, please refer to [RD.6].

5.2.1.3.7. Reference Stations

Reference Stations consist of a single data file with name "*stations.cfg*" that contains the positions of the reference stations simulated. This file is used as input for NOFG module.

For further information, please refer to [RD.6].

PROSBAS project is funded by the European Commission.

5.2.1.4. Internal Flows

In this section we will describe the different SPP internal interfaces. For a more detailed description of PROSBAS Prototype interfaces, please refer to [RD.6]. As stated before, the SPP prototype has several data files that are internal flows (output from one module that serves as input to another module).

5.2.1.4.1. Satellite positions

Both the SPP and the RP receive navigation RINEX files as an input, and it is the NavRinexReader module the one in charge of processing these files and generating the "*SatPos.txt*" file, which contains the satellite positions for each epoch in the scenario.

This module only needs to be run once, and the SatPos.txt file may be used by both the SPP and the RP.

For further information on this internal flow, please refer to [RD.6].

5.2.1.4.2. Message Sequence without losses and SIS Reference Data

This internal flow is the output of MSG module with the message sequence without losses and SIS Reference data. It is also used as input for MLS or if no message losses are to be simulated, it is used as input for MSA. The name of the file is "*msg_out.dat*". In addition, there is an internal flow (not written in a file) from MSG to SME with the message sequence.

The information present in this file in the original MAST can be summarized as follows:

- Configuration Information Section. This is used for other modules that use configuration information without needing to use the external input files (e.g. MSA to analyze fulfillment of updates and timeouts…).
- Message Sequence Section. Two lines per epoch, one with auxiliary information and another one with the message selected: AUX_LINE and EPOCH_LINE.
- Message Queue Final Status Section. This section is just provided for debugging purposes, it is ignored in the analysis.

For further information on this internal flow, please refer to [RD.6].

5.2.1.4.3. Message Sequence with losses and SIS Reference Data

This internal flow is the output of MLS module with the message sequence with losses and SIS Reference data. It is an optional flow, depending if MLS is executed or not. It is used as input for MSA. The name of the file is "*mls_out.dat*".

This file is identical to the "Message Sequence without losses and SIS Ref Data" provided by MSG module except changing the "No" by "Yes" in "Skip" field for the epochs where a message is lost at user level.

For further information on this internal flow, please refer to [RD.6].

5.2.1.4.4. SBAS Message

This internal flow consists of LogBook files containing the SBAS messages. It is the output of SME module and serves as input for the User Receiver Prototype.

It contains the SBAS message (sequence + content) in hexadecimal format.

For further information on this internal flow, please refer to [RD.6].

5.2.1.5. Outputs

In this section we will describe the SPP outputs. For a more detailed description of PROSBAS Prototype interfaces, please refer to [RD.6].

5.2.1.5.1. SIS KPIs

This flow is the SPP external output. It contains the SIS KPIs data.

It is a single data file with name "msa_out.dat" that is the output of MSA module if MLS is not activated.

If MLS is activated, two files are generated: msa_out.dat_SPP and msa_out.dat_RP. In this way, Update Interval and Time-out fulfillment analysis is performed both at SPP level and at RP. In the later case, if a message is lost at user level, an Update Interval violation might occur.

MSA output is provided through the standard output channel, although it might be redirected to a file.

The SIS KPIs output has two sections:

MSA_DETAILED_INFORMATION_SECTION

MSA_SUMMARY_SECTION

The output information generated depends on the options in the execution of MSA module.

Please, refer to Section 5.2.1.2.6 and to [RD.6] for further information.

5.2.1.5.2. NOFG outputs

Several data files are generated by NOFG module containing NOF information, as *udre_l1, udre_l5, iode, give, stations_latitude_longitude_height, NumSVvsTIme_SPP.png, NumSVvsTime_SPPvsRP.png* and *IODRE_PRNx.png.*

Let us remark that in spite that the plots are built from outputs of NOFG module, there are generated by Visualization module. Please, refer to [RD.6] for further information.

5.3. USER RECEIVER PROTOTYPE

In this section the RP architecture will be described.

In the first subsection, a general description will be provided, presenting RP architecture.

Then the different RP modules, configuration and inputs, internal flows and outputs will be explained in more detail in different sections.

5.3.1. RECEIVER PROTOTYPE ARCHITECTURE

At high level, the Receiver Prototype architecture is composed by three main functional modules as it can be seen in Figure 5-9:

- **SBAS Decoder.** Module in charge of decoding the content of the SBAS messages according to SBAS L1/L5 Enhanced ICD.
- **User Analysis Module.** This module computes and analyses the protection levels associated to a navigation solution. These values are the base to compute and to analyse several performance parameters as integrity, accuracy, availability and continuity.
- **Visualization Module.** The last module is in charge of producing the plots and statistical figures needed for the analysis and reporting of the execution campaign.

For each of the above mentioned modules, a deeper view is presented in Section 5.3.2.

As happened in the SPP, there is one additional ancillary stand-alone module in the RP:

- **NavRinexModule**. Module in charge of processing the input navigation RINEX files and computing the satellite positions for the required epochs. This module is also used in the SPP (Please see section 5.2.1.2.1).

As observed in the diagram in Figure 5-9, two are the main input blocks that feed the PROSBAS RP:

- **SBAS Message** (SPP Output): this block represents the SBAS encoded message as coming to the RP from the SPP.

- **Satellite positions** (NavRinexReader output): The satellite positions file is generated by the NavRinexReader module (common to both SPP and RP).
- **External Inputs:** this block collects files the prototype needs to work and which have to be provided by the user in specified folders. This block can be divided in common inputs (both to SPP and RP) and RP inputs. It basically contains:
    - Satellite positions file, which is generated by the NavRinexReader module (common to both SPP and RP).
    - Reference Conditions (common to both SPP and RP).
    - RP Configuration (RP specific).

For further information on the inputs and configuration to the RP one can refer to Section 5.3.3

Finally, the RP will generate several output files with KPI data and KPI Figures of merit that will be described in Section 5.3.4.

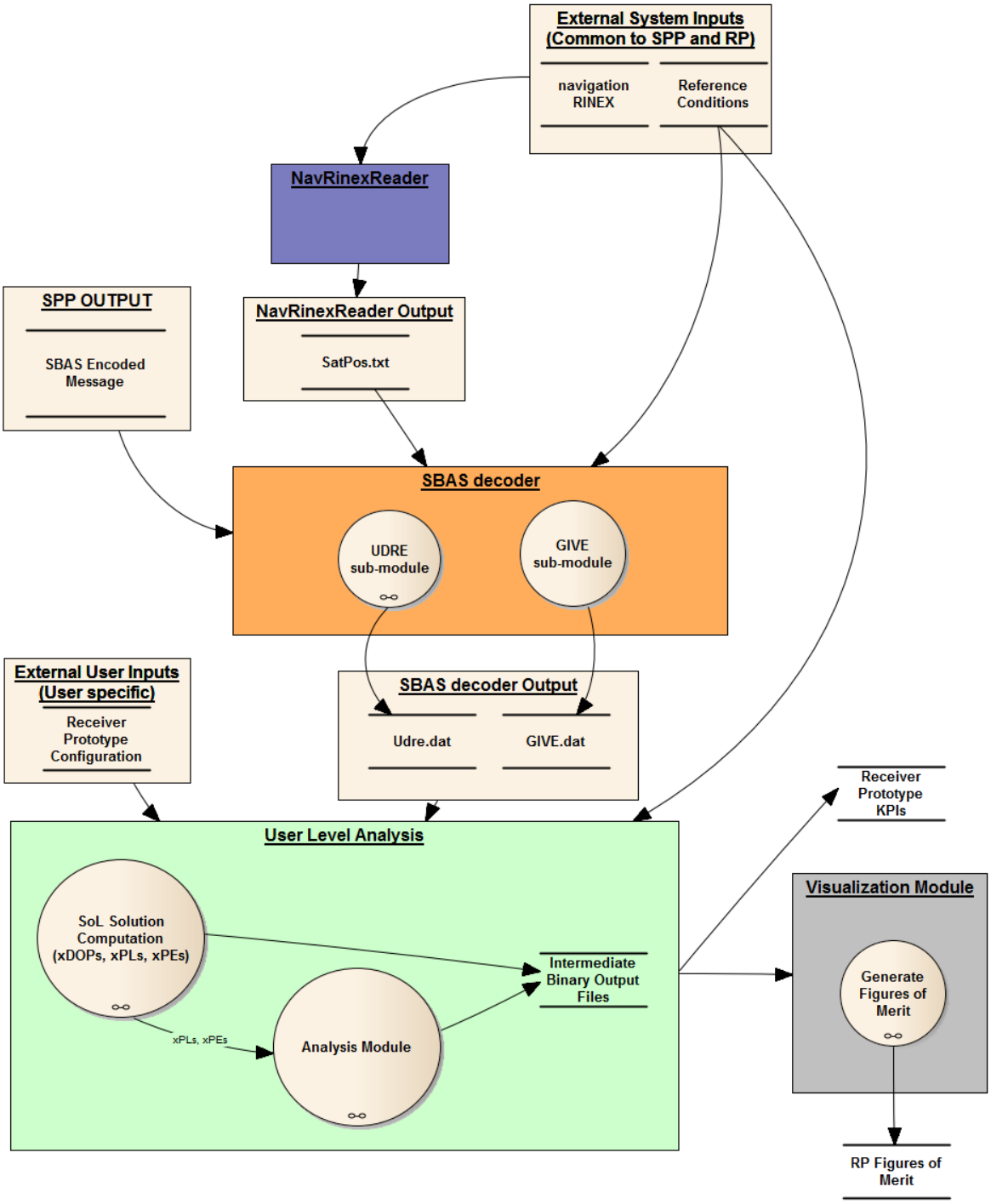


Figure 5-9: High-Level diagram of PROSBAS RP architecture.

Before entering in a more detailed description of the different RP modules, in the following table there is a summary of the main data flows circulating within the RP.

Table 5-8: Receiver Prototype main data flows

Data Flow	I/O	Description	From	To	Content
SBAS Message	**I**	File containing the SBAS messages generated by the Service Provider (hexadecimal format).	SPP	-SBAS Decoder	SBAS messages, epoch by epoch.
Navigation Files	**I**	Files containing the almanacs. One file per constellation. It should be coherent with input parameters as number of SVs and number of constellations.	External	-Satellite Position Computation sub-module	Satellite ephemeris
Receiver Prototype configuration file	**I**	File containing user defined parameter in order to customize simulations.	External	-SBAS Decoder - SoL Solution Comp. sub-module -Analysis sub-Module	-Alarm limits -iono-free error -sigma_flt_flag -etc.
Reference Conditions file	**I**	Configuration file containing parameters common to all the system (Service Provider+User)	External	-SBAS Decoder - SoL Solution Comp. sub-module -Analysis sub-Module	-Service Area -# of epochs -# of constellations -# of frequencies -Transition modes -etc.
SatPos.txt	**O**	File containing the satellite position for every satellite and every epoch	NavRinexReader	SBAS decoder	-Satellite position -IODE
udre.dat	**O**	File containing UDRE related information generated by the UDRE sub-module.	UDRE	- SoL Solution Comp. sub-module	File containing the decoded SBAS messages of types 1, 2 (0/2), 3, 4, 5, 6, 7, 24, 25 for each PRN.
GIVE_XXXX.dat	**O**	Files containing GIVE related information generated by the GIVE sub-module.	GIVE	- SoL Solution Comp. sub-module	File containing ionospheric user corrections related variables.
Intermediate Binary Output Files	**O**	These files are generated during the execution of the User Level Analysis modules in order to be processed at a later stage.	- SoL Solution sub-module -Analysis sub-Module		Binary files containing on-the-run computations (i.e. xPLs, xPEs, xDOPs,etc.)
KPI Output Files	**O**	Files containing key performance indicators with actual values.	- SoL Solution sub-module -Analysis sub-Module	-Visualization Module -External (user)	Text files containing the RP KPIs.

5.3.2. MODULES

In this section the RP Modules will be described: SBAS Decoder, User Level Analysis Module and Visualization Module.

5.3.2.1. NavRinexReader

The NavRinexReader is a C++ module that runs independently of any other module, and may be used in either the SPP, in the RP or in both.

It processes navigation RINEX files and generates the satellite positions according to the satellites and dates specified in the reference conditions (ref_cond_cfg.dat) file, which is used as an input for the SPP and RP modules.

For a comprehensive explanation about the NavRinexModule, please see section 5.2.1.2.1.

5.3.2.2. SBAS Decoder

The first operation performed by PROSBAS RP consists in the de-codification of the SBAS messages coming from the SPP in order to obtain the SBAS corrections needed to augment the navigation solution of one or more core constellations.

The SBAS decoder's architectural structure is depicted in Figure 5-10. Two sub-modules are shown within the SBAS_decoder:

- UDRE sub-module
- GIVE sub-module

These sub-modules are explained in the following sections.

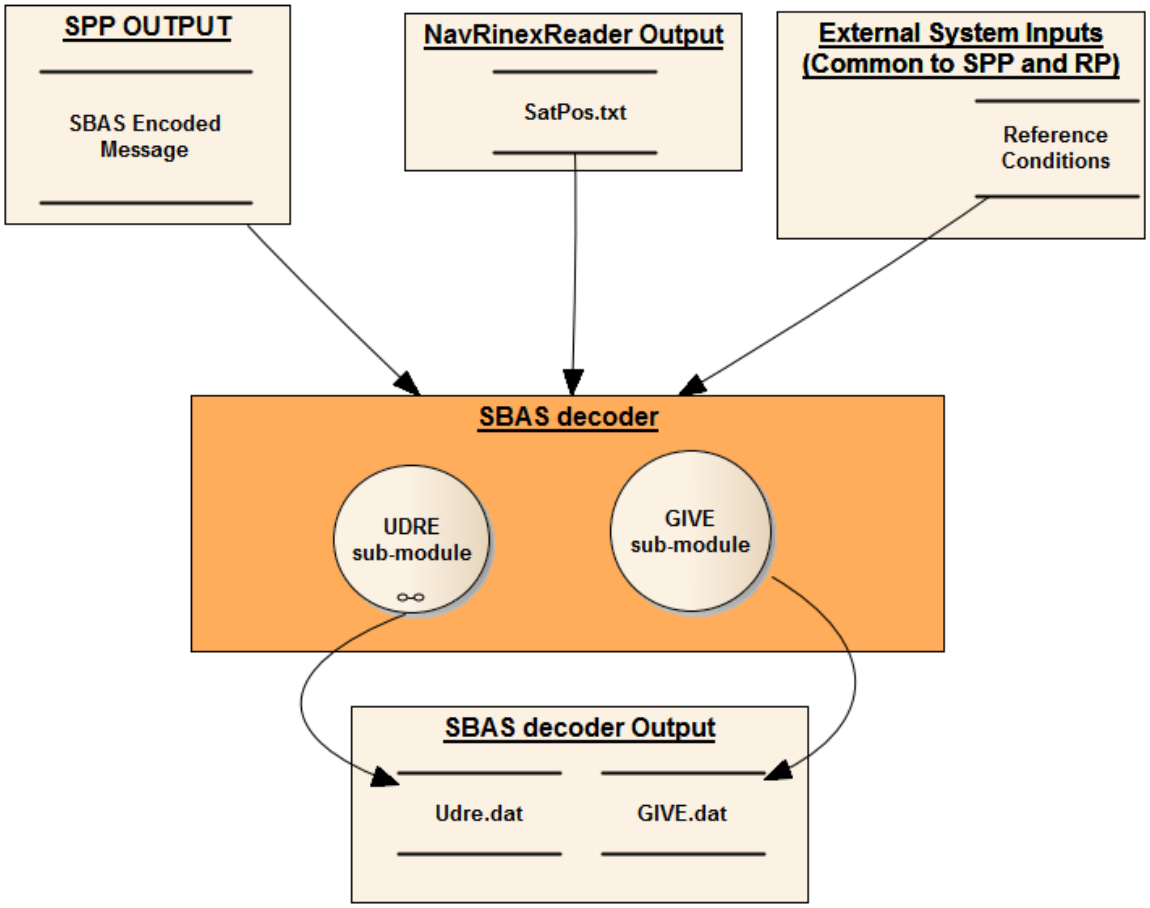


Figure 5-10: High-Level architecture of SBAS Decoder

5.3.2.2.1. UDRE sub-module

This module is in charge of extracting the Satellite Corrections (SV clocks and orbits) and their related confidence bounds (UDREI) performing the following steps:

- It decodes first several SBAS messages stored in the LogBook files. Messages to be decoded in this module are presented in [RD.13].
- Compute the UDRE related information (i.e.: sigma terms) as instructed in [RD.15].
- It writes UDRE related information of the SBAS messages in the *Udre.dat* file.

A detailed description of *udre.dat* file is provided in [RD.14].

5.3.2.2.2. GIVE sub-module

GIVE concept is defined as the upper bound of the GIVD error. This module is in charge of extracting the GIVE and GIVD values from the LogBooks, preforming the following steps:

- It decodes first several SBAS messages stored in the LogBook files. Messages to be decoded in this module are the ones related to the ionosphere, its corrections and its degradation parameters. The different message types are presented in [RD.13].
- Compute the GIVE related information (i.e.: sigma terms) as instructed in Appendix A of [RD.7].

- It writes GIVE related information of the SBAS messages in the *GIVE_XXXXX.dat* files. These are hourly files where XXXXX is the scenario hour index, from 00000 to 99999.

Please note that, as in Figure 5-10, *GIVE.dat* might be used to refer to the whole set of *GIVE_XXXXX.dat* files. A detailed description of *GIVE_XXXXX.dat* files is provided in [RD.14].

5.3.2.3. User Level Analysis Module

At a high level there are two functional modules in charge of computing the different receiver performance indicators, as can be seen in Figure 5-11: "Safety of Life (SoL) Solution Computation Module" and "Analysis Module"

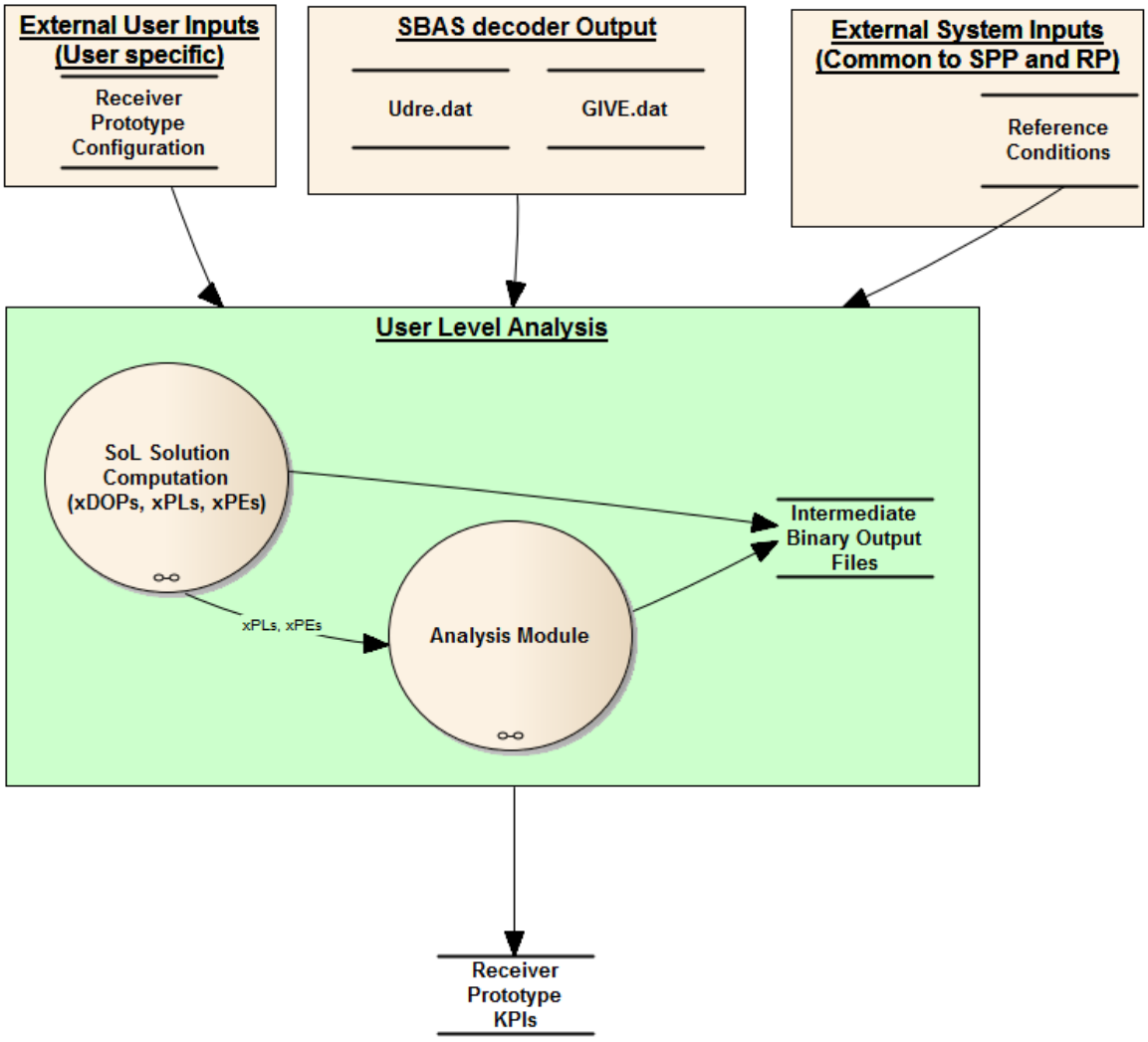


Figure 5-11: High level architecture of the User Level Analysis module

The first module called ***Safety of Life (SoL) Solution Computation*** is composed by the following sub-modules (as can be seen in the following figure).

1. **xDOPs**: this sub-module computes HDOP and VDOP values for each epoch and user location (either fixed or defined over a grid);
2. **xPLs Computation**: computes Horizontal and Vertical protection levels (xPLs) for each epoch and user location;
3. **xPEs Computation**: computes the horizontal and vertical position errors (accuracy) for each epoch and user location.

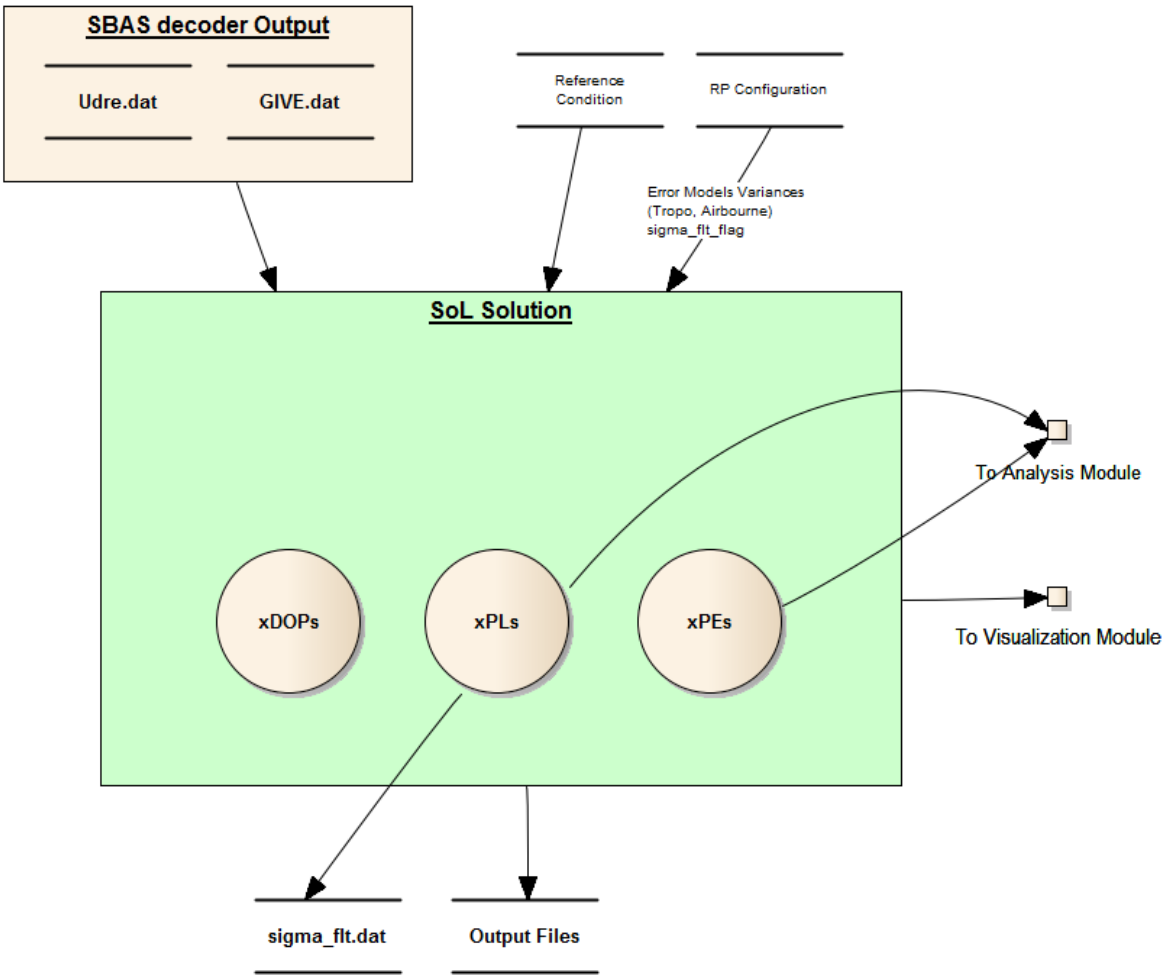


Figure 5-12: High level architecture of Safety of Life Solution Computation module

All these sub-modules write their results in intermediate binary files (transparent to the user) in order to allow a graphical representation at a later stage. Moreover, the xPLs and the xPEs values are also passed to the next functional block called *Analysis Module*, which is in charge of computing continuity, integrity, availability and accuracy related information.

Additionally, output files containing actual protection levels, accuracy and DOP values are generated and they are available to the final user for further analysis. We will indicate them as "*KPI Output Files*".

For information concerning the outputs of this sub-module one can refer to [RD.14].

The ***Analysis Module*** is composed by the following sub-modules (see next figure):

1. **Continuity Risk**: continuity risk is defined as the number of continuity breaks divided by the total number of epochs without alerts. For PA phases of flight a *continuity break* happens when a certain epoch, the xPL is greater than the xAL.

2. **Integrity**: integrity is defined as the number of integer epochs divided by the number of valid epochs. *Integer epoch* is when xPEs are below their corresponding xPLs and the position solution exists. *Valid epochs* is defined as the number of epochs where at least 4 satellites with a valid PL are on sight and monitored.
3. **Availability**: availability is defined as the number of available epochs divided by the total number of epochs (that is the simulation duration). *Available epochs* is the number of epochs for which the xPLs are below their corresponding xALs and the position solution exists.

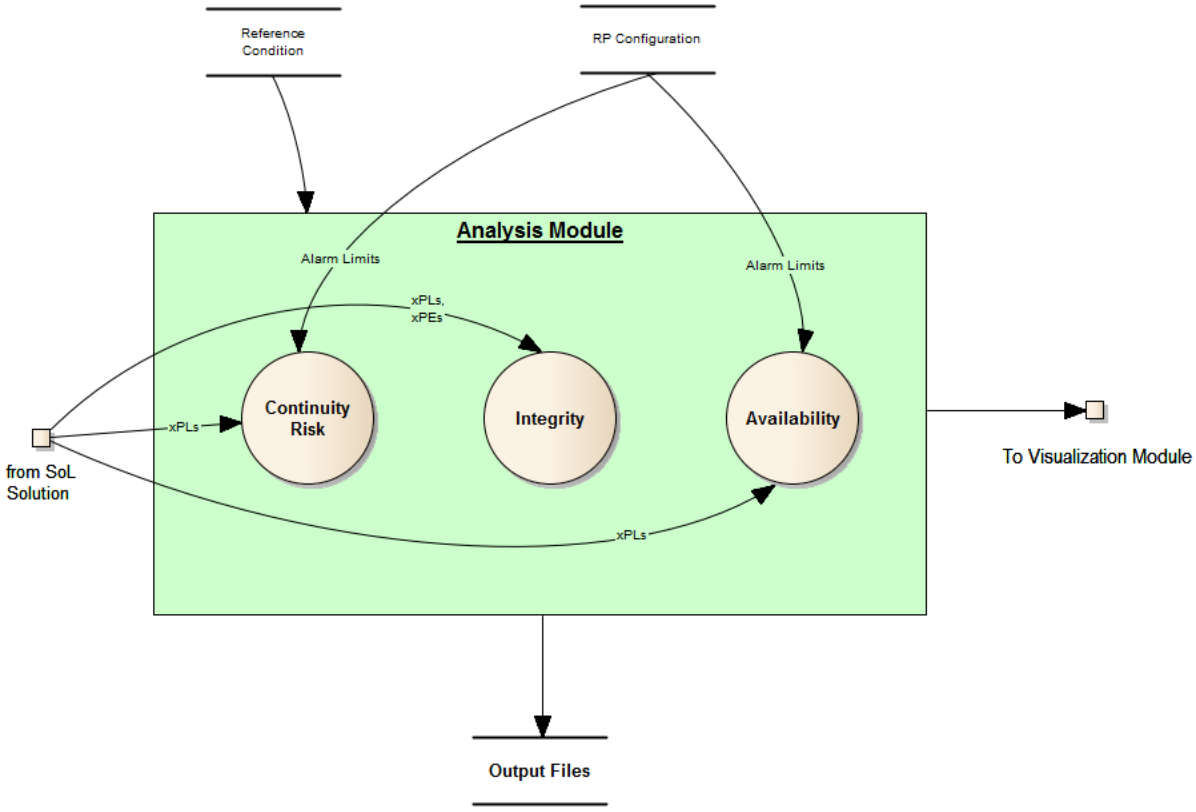


Figure 5-13: High-level architecture of Analysis module

All these sub-modules write their outputs in intermediate files as well, to allow a graphical representation at a later stage. Additionally, output files containing actual continuity risks, integrity (i.e. xPEs vs xPLs vs xALs) and availability values are generated and they are available to the final user for further analysis. We will indicate them as "*KPI Output Files*".

For information on the outputs generated by this module one can refer to [RD.14].

5.3.2.4. Visualization Module

This module is in charge of producing a variety of plots and maps to graphically represent to the user the results obtained with the *User Analysis* tools. Inputs of this module will be the intermediary files written at the previous step and the output will mainly consist in the generation of a set of figures:

- Snapshot xPLs figures (xPLs vs Time)
- Statistic xPLs information (max, mean, std. deviation, rms etc.)
- Accuracy (xPEs) figures and statistics
- xDOPs figures and statistics

- Availability figures/maps
- Continuity figures/maps

In addition, Visualization module is also in charge of generating plots from outputs of NOFG module. Figure 5-14 shows the high level architecture design of the Visualization Module.

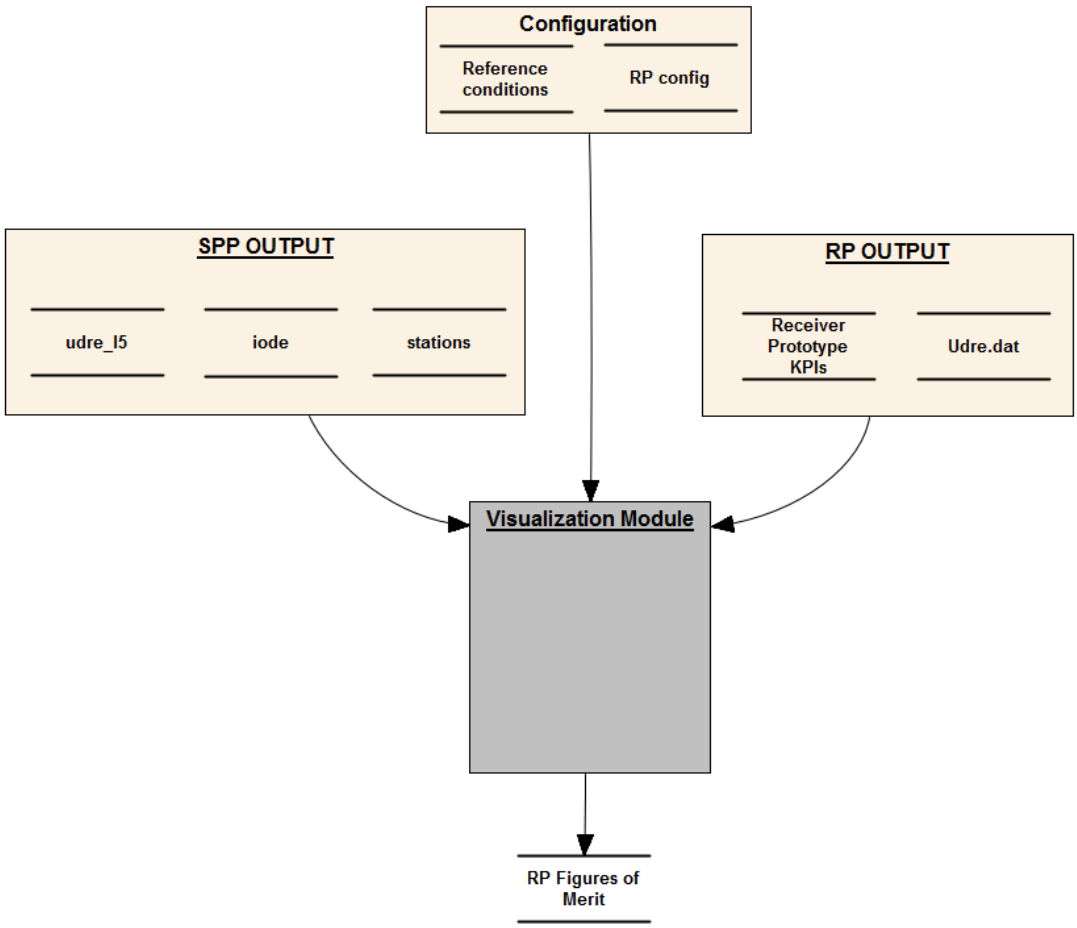


Figure 5-14: High-level architecture of Visualization module

5.3.3. INPUTS AND CONFIGURATION

The inputs file needed to run PROSBAS RP are collected in predefined folders and they can be grouped in the following types:

1. SBAS Message: encoded information as coming from the Service Provider. Further details on the message types and content of the ICD can be found in [RD.13].
2. Navigation Files: RINEX navigation format, containing various information (i.e. ephemeris data, clock etc.). The detailed information about the navigation RINEX files accepted by this module is detailed in [RD.6].

To make the RP as much as flexible as possible, some specific configuration parameters will be provided to the RP as configuration files placed by the user in appropriate folders. The goal of these files is to allow a certain user to customize each simulation and try out different scenarios according to what is needed. Two main kinds of configuration files have been identified:

1. RP Configuration File: in this file the user shall set the following parameters,
   - **Iono-free combination residual error**. This parameter allows the user to set a fixed ionospheric residual error and bounding common to all constellations which will be considered when the receiver is used in dual frequency mode. According to [RD.10], vertical measurements errors and bounding contributions are assumed to be around **3 and 10 cm** respectively. Slant contributions are estimated using MOPS Iono mapping function [RD.7].
   - **Tropospheric error**. This parameter allows the user to set a fixed tropospheric error and bounding common to all constellations. According to [RD.10], vertical tropospheric errors and bounding contributions are assumed to be around **12 cm**. This error can be estimated for other elevations using the mapping function provided in [RD.10].
   - **Noise Variance + Iono Divergence**. In the current L1 MOPS [RD.7], the expression $(\sigma^2_{noise}[i] + \sigma^2_{dvg}[i])^{1/2}$ which takes part in the computation of the $\sigma^2_{air}[i]$ variance, can assume six different values according to which Accuracy designator (e.g. A or B) and GPS/SBAS satellites signal levels (e.g. min or max) are considered. In RP PROSBAS context we allow the user to define any value through this configurable parameter.
   - **Alarm Limits.** Here the user can set the Alarm limits values to be considered for different phases of flight. Current values for LPV-200 are 40 and 35 meters for horizontal and vertical limits respectively. For CAT-I approaches, HAL and VAL are 40 and [35 down to 10][1] meters respectively.
   - **A flag to activate the generation of the file *sigma_flt.dat*.** This flag is a binary parameter which can assume either 0 or 1 value. When set to 1, the prototype will generate an additional output file during the xPLs computation that contains σ_{flt} values for each epoch and satellite.

   RP Configuration File content can be seen in detail in[RD.3].

2. Reference Conditions File: this file, common to the SPP, allows the user to define the reference conditions parameters for the simulation. For further information on Reference Conditions file, please refer to [RD.6].

5.3.4. OUTPUTS

The overall Receiver Prototype outputs consist in a set of data files and figures of merit generated by the *SoL Solution Computation*, the *User Analysis* tools and the Visualization modules. Proper functions will be implemented in order to process the binary files generated within the User Level Analysis Module and to provide the intended graphs and maps to the user.

A comprehensive explanation of the RP outputs can be found in [RD.14].

5.4. PROSBAS PROTOTYPE ENVIRONMENT, INSTALLATION AND EXECUTION

In this section information concerning PROSBAS Prototype environment, installation, execution and folder structure is provided.

5.4.1. PROSBAS PROTOTYPE ENVIRONMENT

PROSBAS Prototype environment, execution time and disk usage will be described in [RD.6].

5.4.2. PROSBAS PROTOTYPE INSTALLATION

PROSBAS Prototype installation is described in [RD.6].

[1] For Category I precision approach, a VAL greater tan 10m for a specific system design may only be user if a system-specific analysis has been completed.

5.4.3. PROSBAS PROTOTYPE EXECUTION

A complete guide for PROSBAS Prototype execution will be provided in PROSBAS document "Operating Manuals for Prototype and Tools" [RD.6] that will be delivered for MTR.

5.4.4. DIRECTORIES STRUCTURE

A detailed description of the directories structure provided with PROSBAS Prototype can be found in [RD.6].

6. TRACEABILITY

6.1. TRACE MODULES TO REQUIREMENTS

The following requirements are general to all PROSBAS System Prototype:

Table 6-1: PROSBAS System Prototype General Requirements.

Req.	Req. Title	Module
REQ-PROSBAS-1-01-1A	Configuration	All
REQ-PROSBAS-1-02-1A	Reproducibility	All
REQ-PROSBAS-1-03-1A	Implementation of SBAS ICD	All
REQ-PROSBAS-1-04-1A	SBAS ICD Flexibility	All
REQ-PROSBAS-1-05-1A	Reference Conditions Flexibility	All
REQ-PROSBAS-1-06-1A	Configurability of key parameters characterisation	All
REQ-PROSBAS-1-07-1A	Nominal and degraded scenarios	All
REQ-PROSBAS-1-08-1A	Expandability up to four constellations	All
REQ-PROSBAS-1-09-1A	L1 mode (deleted requirement)	All
REQ-PROSBAS-1-10-1A	L5 mode	All
REQ-PROSBAS-1-11-1A	Hosting Platform	All

The design shown includes the following executables:

At Service Provider Level:

- Message Sequence Generator (MSG).
- Message Loss Simulator (MLS).
- Message Sequence Analyzer (MSA).
- NOF Generator (NOFG).
- SBAS Message Emulator (SME).

The following table shows the trace from requirements into executable modules.

Table 6-2: Trace of requirements to modules at Service Provider Level.

Req.	Req. Title	Module
REQ-PROSBAS-2-01-1A	Interface with Receiver Prototype	All
REQ-PROSBAS-3-01-1A	SBAS Message Sequence Generation	MSG
REQ-PROSBAS-3-02-1A	Enhanced ICD configuration	MSG
REQ-PROSBAS-3-03-1A	Configurable Number of Simulation Epochs	MSG
REQ-PROSBAS-3-04-1A	Message types and subtypes in SBAS Message Sequence	MSG
REQ-PROSBAS-3-05-1A	Message types update interval	MSG
REQ-PROSBAS-3-06-1A	Issue of Data and Extended Issue of Data in SBAS Message Sequence	MSG
REQ-PROSBAS-3-07-1A	IODE out of Satellite Monitoring Model	MSG
REQ-PROSBAS-3-08-1A	PRN Mask	MSG

Req.	Req. Title	Module
REQ-PROSBAS-3-09-1A	Alert Generation Model	MSG
REQ-PROSBAS-4-01-1A	Generation of Message Sequence with User Message Losses	MLS
REQ-PROSBAS-4-02-1A	User Message Losses in Alerts	MLS
REQ-PROSBAS-5-01-1A	Analysis of SIS Key Performance Indicators	MSA
REQ-PROSBAS-5-02-1A	Static Bandwidth	MSA
REQ-PROSBAS-5-03-1A	Analysis of Message Sequences	MSA
REQ-PROSBAS-5-04-1A	Update Intervals	MSA
REQ-PROSBAS-5-05-1A	Schedulability	MSA
REQ-PROSBAS-5-06-1A	Message Timeouts	MSA
REQ-PROSBAS-5-07-1A	Global Satellite Monitoring	MSA
REQ-PROSBAS-5-08-1A	Issue of Data Ephemeris evolution	MSA
REQ-PROSBAS-5-09-1A	Misleading Information	MSA
REQ-PROSBAS-6-01-1A	L1/L5 NOF generation	NOFG
REQ-PROSBAS-6-02-1A	Operational Modes	NOFG
REQ-PROSBAS-6-03-1A	Operational Mode 1	NOFG
REQ-PROSBAS-6-04-1A	Operational Mode 2	NOFG
REQ-PROSBAS-6-05-1A	NOFG configuration parameters	NOFG
REQ-PROSBAS-7-01-1A	SBAS Message generation	SME

At User Reciever Level:

- Navigation RINEX Reader Module
- SBAS Decoder Module
- User Analysis Module
- Visualization Module

The following table shows the trace from requirements into executable modules.

Table 6-3: Trace of requirements to modules at Receiver Prototype level.

Req.	Req. Title	Module
REQ-PROSBAS-8-01-1A	Interface with Service Provider Prototype	All
REQ-PROSBAS-9-01-1A	Navigation Files	NavRinexReader
REQ-PROSBAS-9-02-1A	SBAS Message Files	SBAS Decoder Module
REQ-PROSBAS-10-01-1A	Configuration File	All
REQ-PROSBAS-10-02-1A	Reference Condition File	All
REQ-PROSBAS-11-01-1A	Udre.dat File	SBAS Decoder Module
REQ-PROSBAS-11-02-1A	Give_XXXXX.dat Files	SBAS Decoder Module

Req.	Req. Title	Module
REQ-PROSBAS-11-03-1A	KPI Output Files	User Analysis Module
REQ-PROSBAS-11-04-1A	KPI Figures of Merit	Visualization Module
REQ-PROSBAS-12-01-1A	SBAS L1/L5 ICD implementation	SBAS Decoder Module
REQ-PROSBAS-12-02-1A	Protection Levels	User Analysis Module
REQ-PROSBAS-12-03-1A	Position Errors	User Analysis Module
REQ-PROSBAS-12-04-1A	Availability	User Analysis Module
REQ-PROSBAS-12-05-1A	Integrity	User Analysis Module
REQ-PROSBAS-12-06-1A	Continuity	User Analysis Module
REQ-PROSBAS-12-07-1A	Statistics	User Analysis Module

PROSBAS project is funded by the European Commission.

7. ANNEX A: NOFG MODULE DETAILED DESCRIPTION

In this section, the way the L1/L5 NOF is generated with NOFG module will be explained.

As stated, two Operational Modes can be selected by the user.

In Operational Mode 1, L1 MTs content will be simulated according to the reference conditions and the ICD. Then, the L1 MTs content is transformed to L1/L5 MTs content according to the ICD. Note that only FCs, UDREs are degraded from L1 to L1/L5 (and also Iono in the case of degradation to L5 in case of running in L5-only backup). Finally, the L1/L5 NOF is built from the L1/L5 MTs content.

In Operational Mode 2, real L1 MTs data files are used as input. Those files are obtained by GMV from different sources outside PROSBAS Prototype. For example, from EGNOS outputs or ***magicSBAS*** L1 SBAS decoded messages. It is important to note that the L1 MTs content data files are generated externally with GMV's tools that will not be delivered. Then, as before, the L1 MTs content is transformed into L1/L5 MTs content according to the ICD. Note that, again, only FCs and UDREs are degraded from L1 to L1/L5 (and also Iono in the case of degradation to L5). Finally, the L1/L5 NOF is built from the L1/L5 MTs content.

Section 7.1 is devoted to explain how L1/L5 NOF is generated.

- In subsection 7.1.1, the concept of Gauss-Markov process that will be used to simulate L1 MTs content in Operational Mode 1 for certain parameters will be briefly reviewed.
- In subsection 7.1.2, it will be explained how to generate the different simulated L1 MTs content files in Operational Mode 1.
- In subsection 7.1.3, it will be explained where the different input L1 MTs content files are obtained in Operational Mode 2.
- In subsection 7.1.4, it will be explained how to transform L1 MTs content into L1/L5 MTs content.
- In section 7.1.5, it will be explained how to build the L1/L5 NOF from the different L1/L5 MTs content.

Finally, in Section 7.2 the detailed design of NOFG Module will be explained.

Conceptually, NOFG Module functionality is represented in the following Figure:

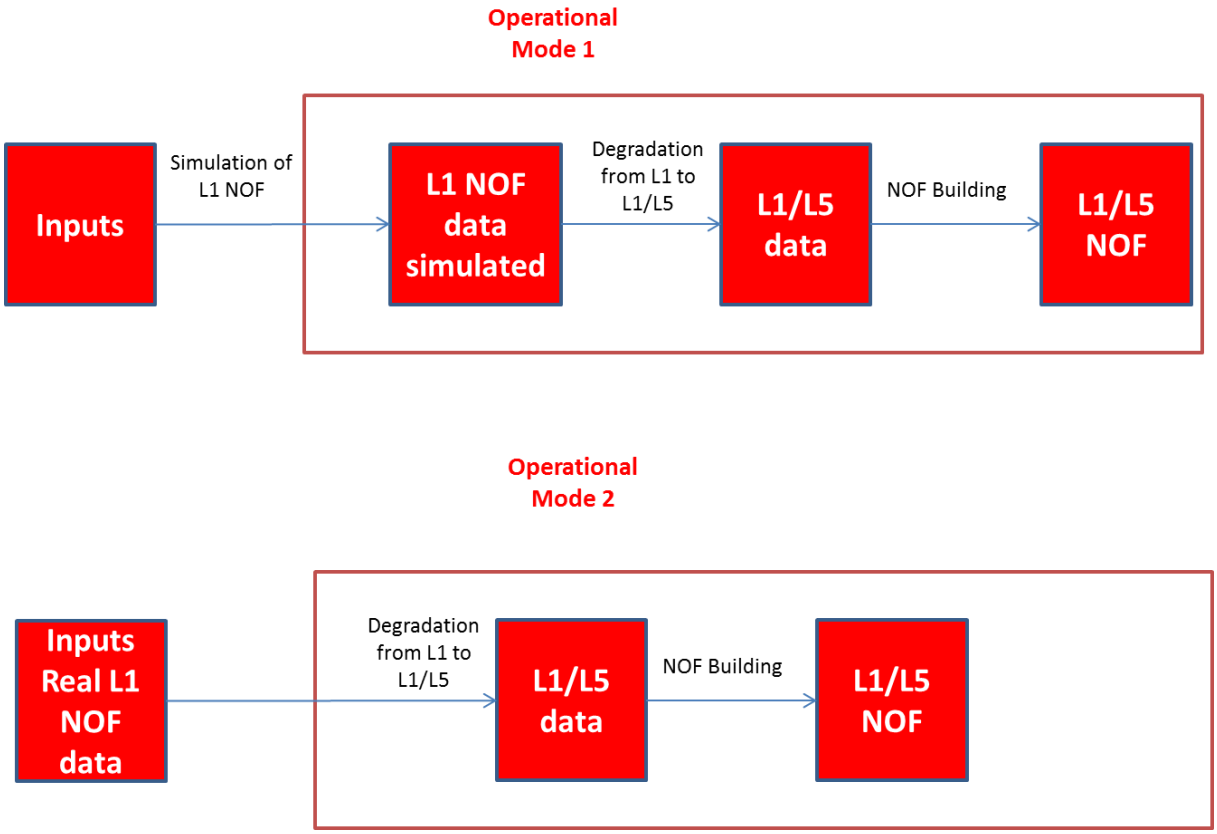


Figure 7-1: Conceptual scheme of NOFG functionality

Please, note that all the information regarding the Ionosphere corrections provided in this section is only applicable to the case when the SPP is run in L5-only backup mode. If the SPP is run in L1/L5, then the ionosphere-related data files and sub-modules are not considered.

7.1. L1/L5 NOF GENERATION

This section is devoted to explain how L1/L5 content is generated .

7.1.1. GAUSS-MARKOV PROCESS REVIEW

In this section, we will review the Gauss-Markov process that will be used in Operational Mode 1 to simulate the evolution in time of several parameters, as it will be explained.

A Gauss-Markov process is a discrete dynamical system defined by:

$$x_{n+1} = k\, x_n + \alpha' + \sigma'\, N(0,1), \qquad n \in \mathbb{N},$$

where:

$k = e^{-\Delta t/\tau}$, Δt is the time step and τ is the autocorrelation time;

$\alpha' = (1-k)\cdot\alpha$ and α is the mean value;

$\sigma' = \sqrt{1-k^2}\cdot\sigma$ and σ is the standard deviation;

$N(0,1)$ is a standard normal distribution shot.

The user can configure the following parameters:

- the autocorrelation time τ;
- the mean value α;
- the standard deviation σ;
- the initial value of the process x_1.

Then, for all the parameters in Operational Mode 1 whose evolution in time will be simulated with a Gauss-Markov process, the user can select the autocorrelation time, the mean value and the standard deviation in the corresponding SPP configuration file.

In particular, the UDRE associated to each satellite will undergo such a process, with different parameters depending on the number of stations in view from the satellite.

Likewise, the ionospheric errors associated to each IGP will be simulated by means of a Gauss-Markov process. In this case, the parameters will also be variable, depending on the number of IPPs —as computed from the configured stations— found in a specific region around the IGP.

7.1.2. L1 GENERATION OPERATIONAL MODE 1

In this section, we will explain how to build the different L1 MTs with simulated pseudo-realistic values according to the configuration.

The L1 MTs content generated in this manner will be used to fill the message sequence generated with MSG module.

According to the Enhanced ICD structure, the following files should be present in order to generate the data for L1 and L5, so that the message sequence with L1/L5 NOF content can be filled:

- **ref_cond_cfg.dat**: this file includes the initial epoch of the simulation together with its duration; moreover, the satellite mask is specified and the operational mode is set. The IGP zone and the frequency are configured here. Finally, the rate at which satellite corrections are sent is set here as well.

- **deg_params_cfg.dat**: this file includes the length of the time interval and the slope used to estimate the Delta_FC addend in the degradation of the UDRE from L1 to L5. In addition, it includes the parameters for the degradation of the fast corrections in time as well as the table to encode the UDRE indicators in L5. Please, let us note that DeltaT_IP is obtained from the update interval of the integrity configured in ref_cond_cfg.dat.

- **gen_L1_NOF_DFRE_cfg.dat**: this file includes the values of the aforementioned Gauss-Markov process used to propagate the following data corresponding to L1 in time: fast corrections, UDREs, GIVEs, clock (slow) corrections and orbit corrections. Besides, the initial data of some other parameters that do not undergo a Gauss-Markov process are established in this file; namely, the initial values of the IODP of the satellite mask, the degradation parameters, the time offset parameters, the IODI of the IGP mask and the covariance matrix are set in this file.

- **SatPos.txt**: this file includes the positions of all the satellites throughout the simulation.

- **IGP_world_ecac.cfg**: this file includes the location of all the configurable IGPs.

- **stations.cfg**: this file includes the positions of the ground stations.

For a detailed description of configuration files and intermediate files generated, please refer to refer to [RD.3].

7.1.3. L1 INPUTS IN OPERATIONAL MODE 2

In this section, we will explain where the different L1 data files used as input in Operational Mode 2 are obtained.

Those real L1 content files are generated externally by GMV for all the tests that will be executed both in validation and in experimentation. GMV's tools that will not be delivered with PROSBAS Prototype, might be used for this purpose. These real input L1 data files are obtained from decoded SBAS L1 Messages from different sources.

In Operational Mode 2, the L1 data content is not simulated with configuration values or Gauss-Markov processes. In this case, real L1 SBAS data will be used.

It is worth noticing here two important differences with respect to Operational Mode 1.

First, in Operational Mode 1, it was necessary to compute the number of stations in view in order to simulate the UDREs with a Gauss-Markov process depending on such number. In the case of Operational Mode 2, the number of stations in view is not used.

Finally, it is also important to note that a module equivalent to the one in charge of L1 data generation in Operational Mode 1 is not present in Operational Mode 2 since, in Operational Mode 1, this module is devoted to setting values with configuration parameters and generating data with a Gauss-Markov process.

Let us focus on the extraction of L1 inputs needed for Operational Mode 2 from L1 SBAS Messages produced externally in the case of UDRE ICD.

According to Enhanced ICD Message Structure, the following data files should be used as inputs with real L1 values:

- ***SV_Mask.dat*** (to build MT1): containing the satellite mask. It is important to point out that the satellite mask configured in *ref_cond_cfg.dat* should be coherent with the one present in the L1 SBAS Messages used as input in Operational Mode 2.

- ***MT2_5.dat*** and ***MT24_fast.dat*** (optional, to build MT2-5): containing the fast corrections. Note that the FCs will be coherent with the alerts generated by MSG module.

- ***MT6.dat*** (optional, to build MT2-5 and MT6): containing the UDREs. Note that the UDREs will be coherent with the alerts generated by MSG module.

- ***MT10.dat*** (to build MT7+10): containing the system latency and FCs degradation factors indicators.

- ***MT12.dat*** (to build MT12): containing SBAS Network time / UTC offset parameters.

- ***MT18.dat*** (to build MT18): containing the IGP mask.
- ***MT25_VelCode0.dat*** and ***MT24_slow.dat*** (optional, to build MT25): containing the slow corrections with velocity code 0.
- ***MT26.dat*** (to build MT26): containing the GIVDs and GIVEIs.
- ***MT28.dat*** (to build MT28): containing the covariance matrices.

For information on the format of those files, please refer to [RD.3].

Moreover, the same configuration files listed in section 7.1.2 are required in this operational mode. It is noteworthy, however, that the file *gen_L1_NOF_DFRE_cfg.dat* is used just to activate the debug outputs in Operational Mode 2.

7.1.4. DEGRADATION FROM L1 TO L1/L5

In this section we will explain how to degrade the L1 NOF in order to build the L1/L5 NOF. Let us note that this process is independent of the Operational Mode; in both Operational Modes, a similar L1 NOF content is obtained.

Let us note that a large part of the L1 NOF information does not have to be degraded to obtain the L1/L5 NOF.

The only data that should be considered for degradation from L1 to L1/L5 are the following:

- ionospheric corrections (in L5 backup mode instead of L1/L5);
- fast corrections;
- UDREs.

The degradation from L1 to L1/L5 of ionospheric corrections, FCs and UDREs are respectively performed by the following modules: NOFG Iono Degradator (that is only active in L5-only backup mode), NOFG FC Degradator and NOFG UDRE Degradator.

After this process, the NOF Builder module translates all the available data to build the L1/L5 NOF.

7.1.4.1. Ionospheric Degradation

The ionospheric corrections will have to be degraded considering a frequency-dependent factor: $(f_1 / f_5)^2$. Note that this is only done in the case of running in single L5 frequency instead of L1/L5 since in the case of L1/L5 no ionosphere information is broadcast by the SPP.

That is, the GIVDs and GIVEs in L1 for each IGP as a function of the epoch are available. Then, NOFG Iono Degradator module transforms each GIVD and GIVE with the frequency dependent factor according to the following expressions:

$$GIVD\_L5 = GIVD\_L1 * (f_1 / f_5)^2$$

$$GIVE\_L5 = GIVE\_L1 * (f_1 / f_5)^2$$

As a result, one obtains the ionospheric corrections in L5.

7.1.4.2. FCs degradation

We will assume that:

FC_L5 = FC_L1

Note that no ISC is simulated because it is not a critical parameter for ICD analysis.
Nevertheless, let us note that MSG module has the capability of activating the broadcast of MT30.

7.1.4.3. UDREs/DFREs degradation

NOFG UDRE Degradation Module performs the computation of L1/L5 UDRE/DFRE as it will be explained below. NOFG Module accepts as input the file *deg_params_cfg.dat* with configuration information to degrade the UDREs to L5 as e.g. the DeltaT_FC interval.

Let us assume that we have the L1 UDREs for all the epochs to be simulated and SVs. Then, for each of the epochs and SVs, the computation of L5 UDREs will be performed according to the following expression:

UDRE_L5 = UDRE_L1 + Delta_FC/1.62 + UDRE_Degradation/1.62

where

- UDRE_L1 term is the information the UDRE in L1,
- Delta_FC term contains the effect of the degradation of the Fast Corrections,
- UDRE_Degradation term contains e.g. the UDRE prediction error (including UDRE Border Effect).

UDRE_L5 is the final result to be used by NOFG Builder module.

Note that UDRE_L1 term is obtained directly from SBAS message and consequently it is scaled to a confidence level of (10^{-3}) according to the standards.
Delta_FC and UDRE_Degradation parameters are terms to be added to the UDRE internal computation (10^{-7} confidence level), therefore they should be scaled assuming gaussianity (1.62 factor to pass from 10^{-3} to 10^{-7}).

Please, note that Delta_FC term is needed because UDREs must not be computed with the internally computed fast correction but with the broadcast one. Adding Delta_FC makes UDRE_L5 equivalent to UDRE of type 6 computation in the current CPF design. This systematic term will be the difference between the broadcast fast corrections and the estimated fast corrections. Furthermore, the UDRE of type 2 in the current CPF design, which ignores the addend Delta_FC, is also stored.

Let us first review the concept of Delta_FC with the help of the following Figure from [RD.9].

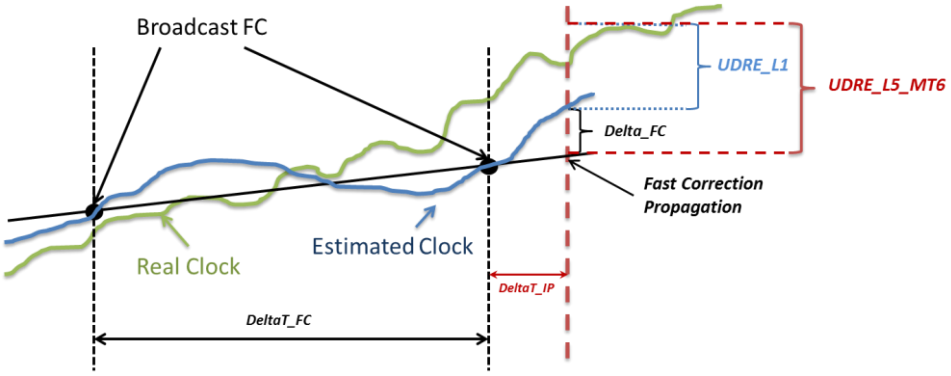


Figure 7-2: Delta_FC

Delta_FC is defined as:

$$Delta_{FC} = \left| Clock_{est} - \frac{PRC_{current} - PRC_{previous}}{DeltaT_{FC}} \left(t - t_{of}\right) \right|$$

where:

- $PRC_{current}$ is the most recent fast correction.
- $PRC_{previous}$ is a previous fast correction.
- *DeltaT_FC* is the update interval between fast corrections.
- t_{of} is the time of applicability of the $PRC_{current}$

Then, Delta_FC is computed as the difference between the real FC and the two previous FCs sent, taking into account the DeltaT_FC parameter present in *deg_params_cfg.dat* and the message sequence generated by MSG module.

In practice, the Delta_FC term is computed in the following way in Operational Mode 2. In this case, Delta_FC is computed by using the offset δB and the drift δB_dot, that is, with a clock model. First, let us remember that the clock is reconstructed by adding the fast and slow corrections. Then, the offset and the drift are computed with a least squares method over the reconstructed clock using a specific (configurable) estimation arc and taking into account that the clock model can be described by the following equation:

Clock (t) = δB + δB_dot x t

Then, Delta_FC would represent the difference of the sum of the fast correction and slow correction with the clock estimated with the model.

As for Operational Mode 1, a linearly increasing function in terms of the time elapsed since the last clock corrections were broadcast is assumed to model the Delta_FC addend properly. The slope of this function has a configurable value which is specified in the configuration file *deg_params_cfg.dat*.

Let us explain the UDRE_Degradation term that takes into account the UDRE Border effect studied in [RD.9].

The following figure from [RD.9] can be checked in order to clarify the UDRE border effect.

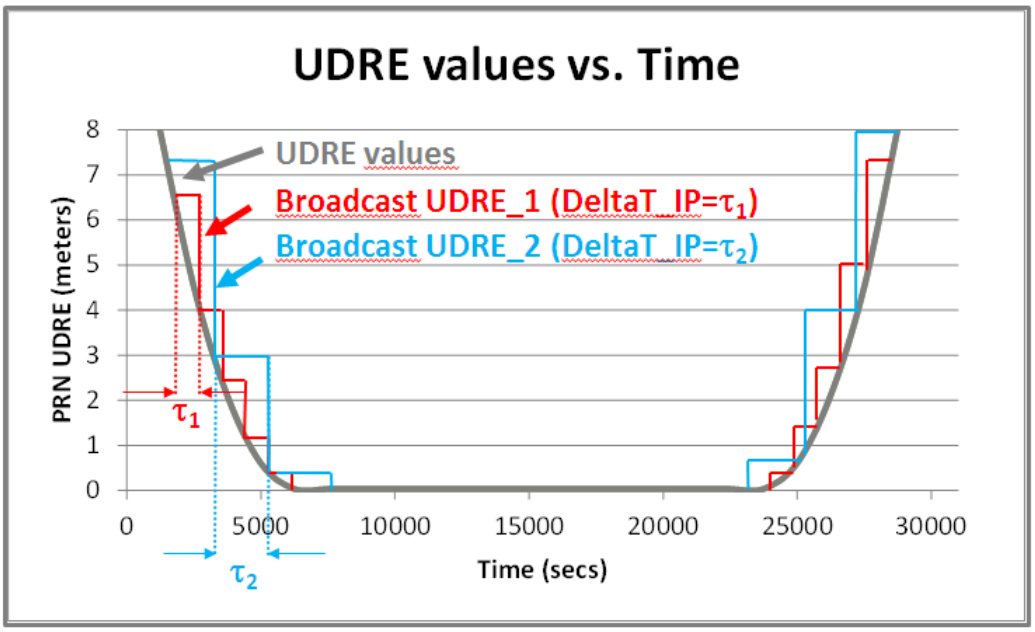


Figure 7-3: UDRE Border effect. Grey line: UDRE values; Red line: UDRE values Broadcast for DeltaT_IP=τ_1; Blue line: UDRE values Broadcast for DeltaT_IP=τ_2.

The first case is trivial at SPP level since UDRE_Degradation is zero. The second case is explained with the following expression.

The UDRE_Degradation term can be obtained from the file UDREs.dat for each epoch and each SV as

$$UDRE\_Degradation = | UDRE\_L1\_t - MAX\_(t, t+UDRE\_timeout)\ UDRE\_L1 |$$

That is, for each epoch *t* and each SV, UDRE_Degradation term is obtained as the absolute value of the difference between UDRE_L1 at epoch *t* and the maximum value of UDRE_L1 in the time interval (t, t + 2 DeltaT_IP), where DeltaT_IP is also a configuration parameter present in the file *deg_params_cfg.dat*.

Note that this is equivalent to assume that we have an ideal model to estimate the UDRE degradation, since information on the future (not available in a real time CPF) is being used to estimate the UDRE degradation.

7.1.5. BUILDING THE L1/L5 NOF

In this section, we will explain how to build the L1/L5 NOF once the L1 NOF has been generated.

The task of organizing all the generated information for L1/L5 in an L1/L5 structure that will be used by SME module to fill the Message Sequence with L1/L5 NOF content is performed by NOFG Builder module.

NOFG Builder Module writes the information described previously in the NOF structure according to the Enhanced ICD. This NOF structure will try to facilitate the filling of the message sequence with NOF content performed by SME Module.

7.2. NOFG DETAILED DESIGN

In this section, the NOFG design architecture will be explained, differentiating between both Operational Modes cases. This will give a clearer explanation of the behavior of NOFG module. Nevertheless, it is worth pointing out that, internally, during the development of the prototype, the two Operational Modes cases have been managed in a compact way.

First, let us briefly summarize the behavior of NOFG Module without entering in details on the Operational Mode.

A NOFG Driver sub-module reads the configuration contained in Reference Conditions files. In particular, the Operational Mode determines the modules to be executed and the expected inputs. Then, in Operational Mode 1, an L1 Generation module generates files with L1 NOF content, simulated with Gauss-Markov processes. In Operational Mode 2, those files are obtained externally. After that, some sub-modules degrade certain L1 information to L1/L5. Finally, the NOFG Builder module builds the L1/L5 NOF.

7.2.1. NOFG MODULE IN OPERATIONAL MODE 1

In the following figure, the NOFG module design for the Operational Mode 1 is shown.

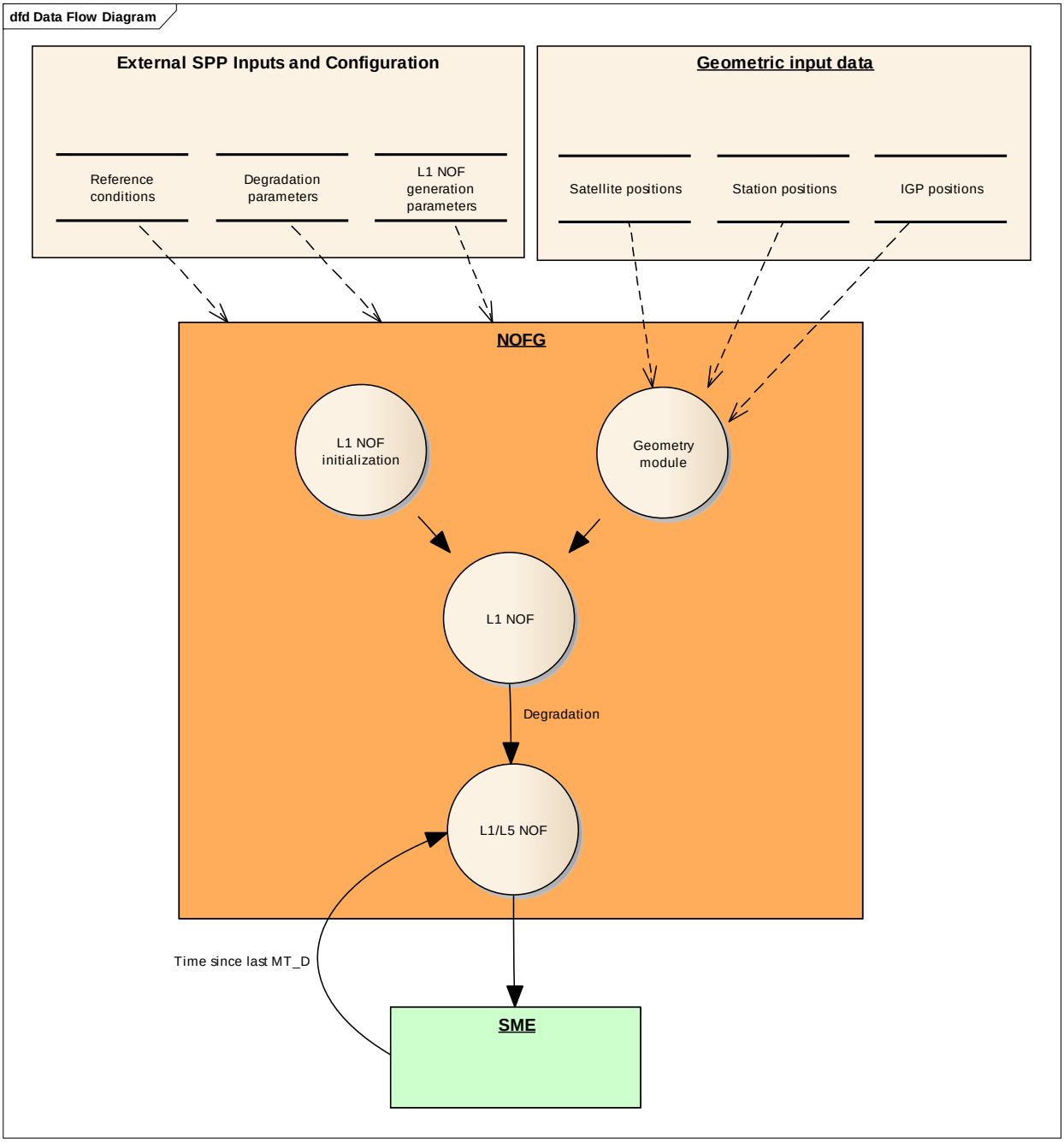


Figure 7-4: NOFG architecture in Operational Mode 1

The following sub-modules can be identified:

- L1 NOF initialization
- Geometry module
- L1 NOF module
- L1/L5 NOF module

The following external inputs are needed by NOFG Module:

- Reference conditions: *ref_cond_cfg.dat*
- Degradation parameters: *deg_params_cfg.dat*

- L1 NOF generation parameters: *gen_L1_NOF_DFRE_cfg.dat*
- Satellite positions: *SatPos.txt*
- Station positions: *stations_cfg.dat*
- IGP positions: *IGP_world_ecac.cfg*

Some intermediate outputs might also be generated:

- IODE file: *iode*
- UDRE in L1 file: *udre_l1*
- UDRE in L5 file: *udre_l5*
- GIVE in L5 file: *give_l5*

Let us briefly summarize the NOFG functioning in the case Operational Mode 1 from a high-level perspective.

NOFG Driver sub-module reads the configuration file *ref_cond_cfg.dat*, in particular the Operational Mode in order to manage the different sub-modules that should be executed and the different expected inputs.

Then, the Geometry sub-module receives RINEX navigation data as inputs and the stations position in order to compute the number of stations visible for each satellite and each epoch (this information will be used to generate the UDREs with a Gauss-Markov process depending on the number of stations in view).

In turn, the L1 NOF initialization sub-module initializes the data of the L1 NOF from the configuration parameters read in the file *gen_L1_NOF_DFRE_cfg.dat*. Namely, the following information is used to build the simulated L1 NOF:

- parameters that are simulated as constant in time (e.g. the fast corrections degradation factors);
- Gauss-Markov parameters to simulate a time variation (e.g. the FCs).

For more information on how the L1 NOF is generated in Operational Mode 1, please refer to Section 7.1. L1 NOF sub-module is mainly in charge of generating the Gauss-Markov processes and writing the L1 NOF data.

At this point, L1/L5 NOF sub-module is executed in order to generate the NOF for L1 and L5 (taking into account Delta_FC and UDRE Border effect; see Section 7.1 for more information). Note that a degradation process has been carried out, which includes the estimation of the term Delta_FC with the aid of SME module's feedback regarding the last emission of a message of type D, as shown in Figure 7-4. This sub-module and this process retrieve the configuration for the degradation from L1 to L1/L5 from the file *deg_params.dat*.

Finally, the L1/L5 NOF is handed over to SME module.

7.2.2. NOFG MODULE IN OPERATIONAL MODE 2

In the following figure, NOFG module design in the case of Operational Mode 2 is shown.

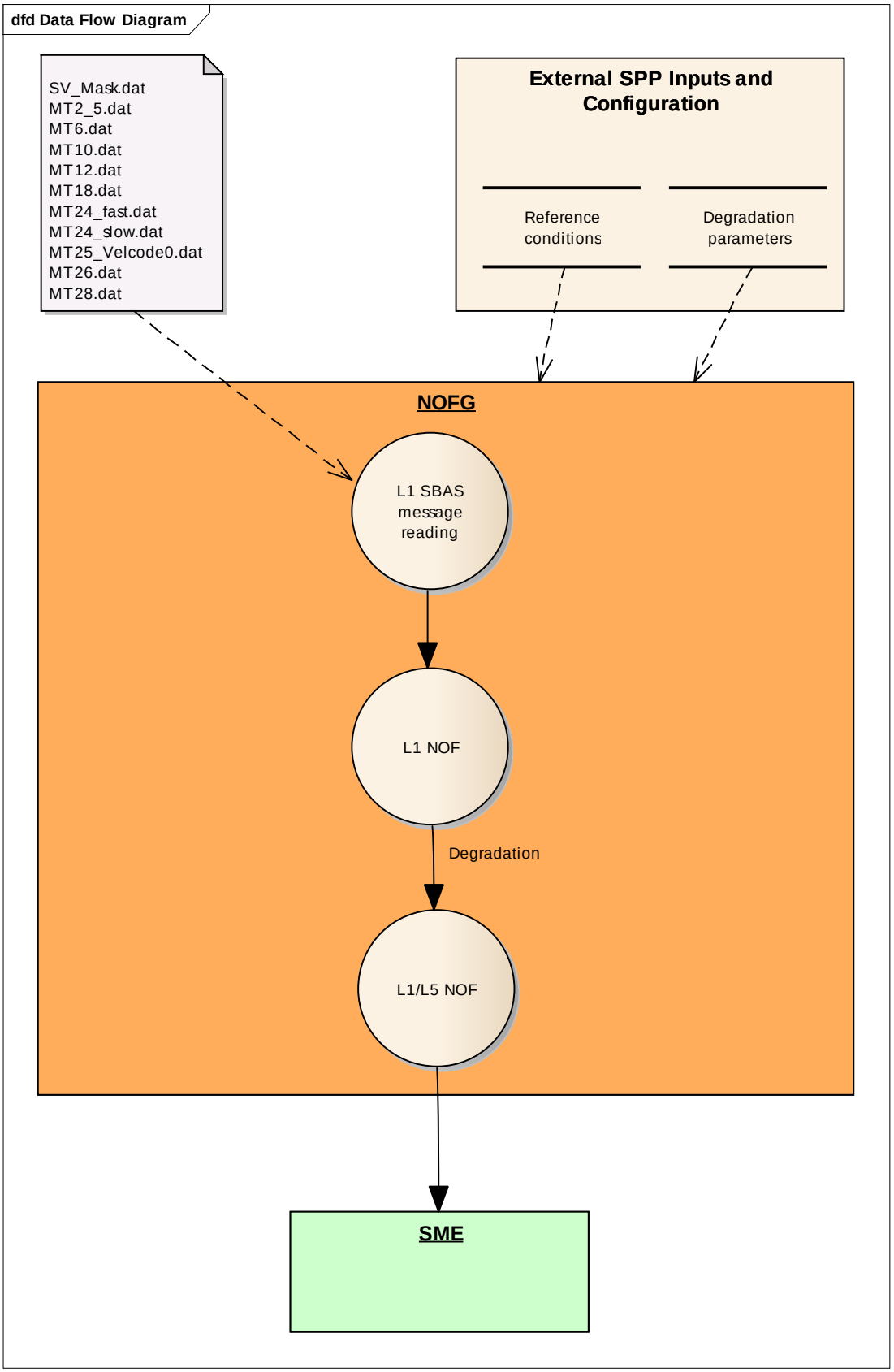


Figure 7-5: NOFG architecture in Operational Mode 2

The following sub-modules can be identified:

- L1 SBAS message reading
- L1 NOF module
- L1/L5 NOF module

The following external inputs are needed by NOFG Module:

- Reference conditions: *ref_cond_cfg.dat*
- Degradation parameters: *deg_params_cfg.dat*
- L1 SBAS messages: *SV_Mask.dat, MT2_5.dat, MT6.dat, MT10.dat, MT12.dat, MT18.dat, MT24_fast.dat, MT24_slow.dat, MT25_Velcode0.dat, MT26.dat* and *MT28.dat*

Some intermediate outputs might also be generated:

- IODE file: *iode*
- UDRE in L1 file: *udre_l1*
- UDRE in L5 file: *udre_l5*
- GIVE in L5 file: *give_l5*

Let us briefly summarize the NOFG functioning in the case of Operational Mode 2 from a high-level perspective.

NOFG module reads the configuration file *ref_cond_cfg.dat*, in particular the Operational Mode, in order to manage the different sub-modules that should be executed and the different expected inputs.

Several L1 data files have been externally prepared. The different necessary files are listed in Figure 7-5 as well as in [RD.6].

Henceforth, the process is similar to that of Operational Mode 1. L1 NOF sub-module is mainly in charge of generating the L1 NOF data.

At this point, L1/L5 NOF sub-module is executed in order to generate the NOF for L1/L5 (taking into account Delta_FC and UDRE Border effect; see Section 7.1 for more information). Note that a degradation process has been carried out; in this case, the term Delta_FC is computed by estimating the clock error as explained in Section 7.1. This sub-module and this process retrieve the configuration for the degradation from L1 to L1/L5 from the file *deg_params.dat*.

Finally, the L1/L5 NOF is passed to SME module.

8. ANNEX B: ENHANCED SBAS L1/L5 ICD REVIEW

Enhanced ICD model can be considered as a hybrid ICD model that takes the advantages of both UDRE Alternative ICD and DFRE ICD while solving certain of their drawbacks.

The different message types of Enhanced ICD are the ones of DFRE ICD but adding the use of the integrity messages of UDRE Alternative ICD model MT6_1 and MT6_2 in order to perform a quick refresh of DFREs on asynchronous events when more than 7 DFREs are changed. The purpose of using in Enhanced ICD MT6_1 and MT6_2 from UDRE Alternative ICD together with all the MTs from DFRE ICD in the context of Enhanced ICD is to solve integrity issues related with the fact of more than 7 DFREIs changing at once.

In addition, other UDRE Alternative ICD messages are considered in Enhanced ICD as MT12, or in case of L5-only back-up mode, MT18 and MT26.

The purpose of this section is not to extensively describe the different Message Types considered in Enhanced ICD. The purpose of this section is only to summarize the different MTs considered in Enhanced ICD in order to know which Message Types have been prototyped in PROSBAS software.

In the following table there are shown the different message types of Enhanced ICD that have been implemented in PROSBAS Prototype.

Table 8-1: Enhanced ICD Message Types Summary

Message Type Name	Description	Frequency
MT_B	PRN Mask Message	L1/L5
MT_C	Alert Message containing DFRECIs of up to 91 SVs and up to 7 DFREIs	L1/L5
MT_6_1 & MT_6_2	Integrity Messages containing up to 51 DFREIs in MT_6_1 and up to 40 DFREIs in MT_6_2 (please note that Enhanced ICD is limited to a maximum of 91 SVs in mask). MT_6_1 and MT_6_2 are refreshed with a large Update Interval but are broadcast with a high priority when asynchronous events implying a simultaneous change of more than 7 DFREs take place.	L1/L5
MT_D	Satellite Correction Message	L1/L5
MT_E	SBAS Satellite Orbit Message	L1/L5
MT_F	Degradation Parameters Message	L1/L5
MT_G	SBAS Almanac Message	L1/L5
MT_12	UTC Timing Data Message	L1/L5
MT_18	IGP Mask Message	L5 back-up mode
MT_26	Ionosphere Corrections Message	L5 back-up mode
MT_30	Inter-signal Values Message	L5 back-up mode

Please, let us note that three options concerning integrity can be configured by the user.

First, the user can configure the Update Interval of MT_C to 6 seconds, the one of MT_6_1/2 to be very large and the use of MT_6_1/2 in asynchronous events deactivated. In this case, Enhanced ICD converges to DFRE ICD.

Second, the user can configure the Update Interval of MT_6_1/2 to 6 seconds and the Update Interval of MT_C very large. Then, in this case, the integrity is broadcast through MT_6_1/2 while MT_C is only used in alerts.

Finally, the user can configure the Update Interval of MT_C to 6 seconds, the Update Interval of MT_6_1/2 to 300 seconds (in order that receivers process this message) and activate the use of MT_6_1/2 in asynchronous events. In this case, the integrity is broadcast through MT_C but when more than 7 DFREIs are being changed during the time-out of MT_C, in this case, MT_C is substituted by MT_6_1 (immediately followed by MT_6_2 if more than 51 SVs have been configured in mask).

For a complete description of Enhanced ICD, please refer to the last version of [RD.1].

For a detailed description and analysis of Enhanced ICD, please refer to [RD.13].

9. ANNEX C: MESSAGE ANALYSIS SUPPORT TOOLS (MAST) OVERVIEW

As stated before, the SPP will be based on SUGAST Prototype Message Analysis and Support Tools (MAST). In this subsection we will briefly review MAST tool. For further information regarding MAST, please refer to [RD.4].

Let us first review the MAST Missions, High Level Functions and Architecture.

MAST Missions can be summarized as follow:

- To support the definition of SBAS L1 and L1/L5 ICD as regards the dynamic aspects of the message sequence.
- To determine the bandwidth performance of different SBAS L1 and L1/L5 ICD models under different configurations.
- To test different message scheduling algorithms.
- To test different configurations of PRN mask selection algorithms.
- To anticipate the performance achieved at user level upon message losses.

MAST provides its missions by implementing the next high-level functions:

- Generation of SBAS message sequences with capability to configure:
    - The SBAS L1 and L1/L5 ICD models and constraints among message types.
    - The message scheduling algorithms.
    - The PRN mask selection algorithms.
    - SBAS service parameters as needed.
    - Generation of alerts.
- Generation of reference data in line with message sequence for performance analysis.
- Simulation of user message losses.
- Analysis of achieved performance.

Nevertheless, it is worth noticing that not all MAST functionalities will be implemented/used in PROSBAS SPP, for example, the message scheduling algorithm will be fixed (not configurable), or the dynamic mask functionality will not be used.

MAST has been decomposed in three architectural components in line with the following data flow model.

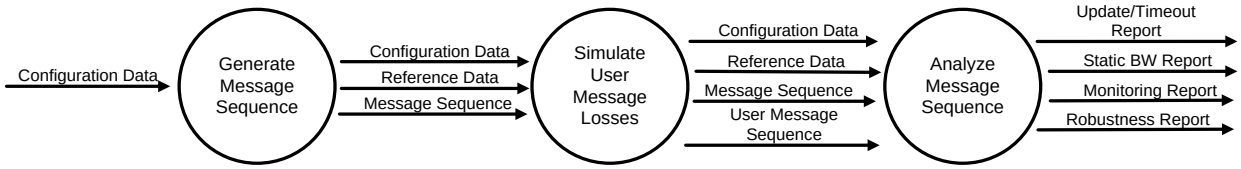


Figure 9-1: MAST data flow model

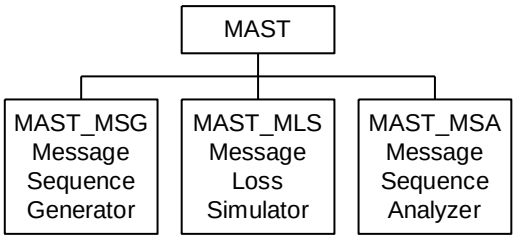


Figure 9-2: MAST functional decomposition

- MAST_MSG (MAST Message Sequence Generator): This module reads the configuration and generates a message sequence and reference data accordingly.
- MAST_MLS (MAST Message Loss Simulator): Depending on the configuration, this module generates a new message sequence omitting some messages assumed to have been lost by the user. This module is optional.
- MAST_MSA (MAST Message Sequence Analyzer): With the output either from MAST_MSG or MAST_MLS if message loss is to be analyzed, this executable performs the needed analyses to support the experimentation KPIs.

END OF DOCUMENT