SOLAR ORBITER

RPW Operation Centre

ROC Software Development Plan

ROC-GEN-SYS-PLN-00015-LES
Iss.02, Rev.02

Prepared by:	Function:	Signature:	Date
Xavier Bonnin	RPW Ground Segment Deputy Project Manager		27/09/2016
Verified by:	**Function:**	**Signature:**	**Date**
Yvonne de Conchy	RPW Ground Segment Project Manager	Yvonne de Conchy 2017.01.04 10:18:04 +01'00'	27/09/2016
Approved by:	**Function:**	**Signature:**	**Date**
Milan Maksimovic	RPW PI		Dd/mm/yyyy
For application:	**Function:**	**Signature:**	**Date**
Name	Team Member #4		Dd/mm/yyyy

CLASSIFICATION **PUBLIC** ☒ **RESTRICTED** ☐

l'Observatoire de Paris — LESIA
Laboratoire d'Études Spatiales et d'Instrumentation en Astrophysique

CNRS-Observatoire de PARIS
Section de MEUDON – LESIA
5, place Jules Janssen
92195 Meudon Cedex – France

Change Record

Issue	Rev.	Date	Authors	Modifications
00	00	23/12/2014	Xavier Bonnin	First draft
00	01	17/02/2015	Xavier Bonnin	Second draft
00	02	25/06/2015	Xavier Bonnin	Third draft
01	00	30/06/2015	Xavier Bonnin	First release
01	01	15/10/2015	Xavier Bonnin	Update the software responsibilities and deliveries
02	01	22/07/2016	Xavier Bonnin	Major updates: Re-organize sections to be consistent with ECSS-E-ST-40C(6March2009).
02	02	27/09/2016	Xavier Bonnin	Complete missing sections

Acronym List

Acronym	Definition
AIT	Assembly, Integration Tests
AIV	Assembly, Integration Validations
API	Application Programming Interface
CDF	Common Data Format
CIRD	Concept and Implementation Requirements Document
CNES	Centre National d'Etudes Spatiales
CoI	Co-Investigator
CP	Cruise Phase
DAS	DPU Application Software
DBS	DPU Boot Software
DPS	Data Processing System
DPU	Data Processing Unit
ECSS	European Cooperation for Space Standardization
EDDS	EGOS Data Dissemination System
E-FECS	Enhanced-Flight Events Communications Skeleton
EM	Engineering Model

ESA	European Space Agency
ESAC	European Space Astronomy Centre
ESOC	European Space Operation Centre
Faust	Flight Operation Request Editor
FCT	Flight Control Team
Figaro	Flight Operation Procedure Editor
FM	Flight Model
FOP	Flight Operation Plan
GIGL	Groupe Informatique Générale du LESIA
GSE	Ground Support Equipment
GUI	Graphical User Interface
HF	High Frequency
HFR	High Frequency Receiver
IAP	Institute of Atmospheric Physics
ICD	Interface Control Document
IDB	Instrument Database
IDL	Interactive Data Language
IOR	Instrument Operation Request
IRF	Institutet för rymdfysik
IT	Instrument Team
I/O	Input/Output
JSON	JavaScript Object Notation
LESIA	Laboratoire d'Etudes Spatiales et d'Instrumentation en Astrophysique
LF	Low Frequency
LFR	Low Frequency Receiver
LL	Low Latency
LPC2E	Laboratoire de Physique et Chimie de l'Environnement et de l'Espace
LPP	Laboratoire de Physique des Plasmas
MADAWG	Modeling And Data Analysis Working Group
MCS	Monitoring and Control System
MDOR	Memory Direct Operation Request
MEB	Main Electronic Box
MIB	Mission Information Base
MOC	Mission Operation Centre
MUsIC	MCS User Interfaces

NECP	Near Earth Commissioning Phase
NP	Nominal Phase
OPera	Operation Planning Interface
ORM	Object Relational Mapping
OS	Operating System
PDOR	Payload Direct Operation Request
PFM	Preliminary Flight Model
PI	Principal Investigator
PMP	Project Management Plan
POPPy	Plugin-Oriented Pipeline for Python
PA	Product Assurance
QA	Quality Assurance
RCS	RPW Calibration Software
RDBMS	Relational Database Management System
REG	ROC Engineering Guidelines
RGS	RPW Ground Segment
ROC	RPW Operation Centre
RODS	ROC Operation and Data System
ROT	RPW Operation Toolkit
RPL	RPW Packet parsing Library
RPW	Radio and Plasma Waves
RSS	ROC Software System
SAP	Science Activity Plan
SBM	Selected Burst Mode
SCM	Search-Coil Magnetometer
SDD	Software Design Document
SDDS	Solar Orbiter EDDS
SDP	Software Development Plan
SGSE	Software Ground Support Equipment
SISSI	SBM Interactive Selection Software Interface
SOC	Science Operation Centre
SolO	Solar Orbiter
SOOP	Solar Orbiter Operation Plan
SOV	System Operations Validation
SRDB	Spacecraft Reference Database

SSD	Software Specification Document
SSH	Secure SHell
SSMM	State Solid Mass Memory
SSS	Software System Specification
SUM	Software User Manual
SVT	System Validation Tests
S/C	Spacecraft
S/W	Software
TBC	To Be Confirmed
TBD	To Be Defined
TBW	To Be Written
TC	Telecommand
TDS	Time Domain Sampler
THR	Thermal Noise and High Frequency Receivers
TM	Telemetry
TMC	TM Corridor
TNR	Thermal Noise Receiver
TV	TM/TC Viewer
URD	User Requirement Document
XML	eXtended Markup Language

Table of Contents

List of figures

List of tables

1 GENERAL

1.1 Scope of the Document

This document presents the software development plan (SDP) of the RPW Operation Centre (ROC).

According to [RD26], the purpose of the SDP is "*to describe the established management and development approach for the software items to be defined by a software supplier to set up a software project in accordance with the customer requirements*".

The present document is a tailored version of the SDP as defined above, but covering all of the software units of the ROC software system (RSS).

This SDP addresses more specifically the needs to be foreseen in terms of:

- RSS development, validation and maintenance organisation and responsibilities
- RSS data production, validation and distribution organisation and responsibilities
- RSS related management and collaboration tools
- RSS related development approach, environment and integrated logistic support
- RSS related identified risks

These needs must meet the requirements initially listed in the ROC Concept and Implementation Requirements Document (CIRD) [AD1] in terms of: technical developments, software tools, data format, and functional facilities to support the ROC science and operational activities. Nevertheless the content of the SDP is expected to follow the evolution of the RSS specification and design.

The SDP shall be always consistent with the approach described in the ROC Project Management Plan (PMP) [AD3].

1.2 Applicable Documents

This document responds to the requirements of the documents listed in the following table:

Mark	Reference/Iss/Rev	Title of the document	Authors	Date
AD1	ROC-GEN-SYS-PLN-00002-LES/01/03	ROC Concept and Implementation Requirements Document (CIRD)	Yvonne de Conchy	22/12/2016
AD2	ECSS-S-ST-00-01C(1October2012), Third issue	ECSS System – Glossary of terms	ECSS consortium	01/10/2012
AD3	ROC-GEN-MGT-PLN-00026-LES/1/3	ROC Project Management Plan	Yvonne de Conchy Xavier Bonnin	20/12/2016
AD4	ROC-GEN-MGT-QAD-00033-LES/0/3	ROC Software Assurance /Product Assurance Plan	Stephane Papais	14/11/2016

1.3 Reference Documents

This document is based on the documents listed in the following table:

Mark	Reference/Iss/Rev	Title of the document	Authors	Date

RD1	SOLO-RPWSY-IF-55-CNES_0401(=EID-B).pdf/04/01	Experiment Interface Document Part B (EID-B) for RPW	RPW Team	21/12/2012
RD2	SOL.EST.RCD.0050/03/00	Experiment Interface Document Part A (EID-A)	Erik de Witt	03/08/2012
RD3	RPW-GEN-DAT-SPC-00006-LES/01/00	RPW Data Products	Xavier Bonnin	23/12/2016
RD4	RPW-TST-GSE-NTT-00017-LES/02/01	Data format and metadata definition for the ROC-SGSE	Xavier Bonnin	14/10/2016
RD5	SOL-SGS-ICD-0005/1/1	Solar Orbiter Low-Latency Data: Concept and Implementation	Anik de Groof	Sept., 2014
RD6	SOL-SGS-TN-0009/2/2	Metadata Definition for Solar Orbiter Science Data	Anik De Groof	15/01/2015
RD7	SOL-SGS-ICD-0002_DPICD/0/2	Data Producer to archive ICD	Luis Sanchez	05/09/2014
RD8	SOL-SGS-ICD-0003/0/2	Solar Orbiter Instrument Operation Request Interface Control Document (IOR ICD)	Christopher Watson	19/08/2014
RD9	SOL-SGS-0004/0/3	Solar Orbiter Interface Control Document for Low Latency Data CDF files	Andrew Walsh	Feb., 2015
RD10	SOL-SGS-TN-0006/1/0	SOC Engineering Guidelines for External Users	Richard Carr	04/03/2015
RD11	SOL-ESC-IF-05011/1/0	Solar Orbiter Data Delivery Interface Control Document	Luca Michienzi	10/09/2013
RD12	SOL-SGS-TN-0009/1/0	Metadata Definition for Solar Orbiter Science Data	Anik De Groof	15/01/2015
RD13	SOL-ESC-PL-10001/1/0	Solar Orbiter FOP Preparation Plan	B. Sousa	05/09/2013
RD14	SOL-ESC-TN-12000/1/0	Solar Orbiter Mission Planning Concept	Ignacio Tanco	27/09/2013
RD15	SOL-SGS-ICD-0006/0/1	Solar Orbiter Enhanced-Flight Events Communications Skeletons Interface Control Document	Christopher Watson	05/07/2015
RD16	ROC-GEN-SYS-NTT-00008-LES/1/3	ROC Engineering Guidelines (REG)	Xavier Bonnin	09/11/2016
RD17	ROC-GEN-SYS-NTT-00019-LES/1/1	ROC Engineering Guidelines for External Users (REGU)	Xavier Bonnin	18/11/2015
RD18	TN_GSE-017/01/02	RPW MEB SGSE Science Data Format	Loïc Guegen	23/06/2015
RD19	ROC-TST-GSE-SUM-00024-LES/1/1	ROC-SGSE User Manual	M.Duarte	16/12/2016
RD20	SOL-SGS-TS-0006/0/0	Solar Orbiter Instrument Teams – SOC Test Specification	Nana Bach Christopher J. Watson	16/12/2015
RD21	ROC-TST-GSE-ICD-00023-LES/2/2	ROC-SGSE Calibration Software ICD	Manuel Duarte	10/05/2016
RD22	http://www.redmine.org/, Version 3.2.3	Redmine	Redmine developer team	05/06/2016
RD23	https://www.atlassian.com/software/jira	JIRA software	Atlassian team	N/A
RD24	http://nvie.com/posts/a-successful-git-branching-model/	A successful Git branching model	Vincent Driessen	05/01/2010
RD25	TMC-ICD SOL-SGS-ICD-0007 v0_4	Solar Orbiter Telemetry Corridor ICD	Christopher J.	25/01/2016

			Watson	
RD26	ECSS-E-ST-40C(6March2009)/3	Space engineering: Software	ECSS team	06/03/2016
RD27	SOL-ESC-RP-05500 - Issue 3r1 20121012 - Solar Orbiter CReMA Issue 3 Rev 1	Solar Orbiter: Consolidated Report on Mission Analysis	José Manuel Sanchez Perez	2012-10-12
RD28	SOLO-RPWSY-PT-1235-CNES/01/00	RPW Instrument Calibration Plan	RPW teams	2014-11-12
RD29	2A- SOL-ESC-HO-05014/1/1	Instrument Command Workshop, ESOC : Commanding Interface and Testing	I.Tanco	05/09/2016
RD30	LL-pipelines at SOC schedule.pptx	LL-Pipelines@SOC Proposed schedule	Chris Watson	06/07/2015
RD31	ROC-TST-GSE-SUM-00035-LES/1/1	Plugin-Oriented Pipeline for Python framework User Manual	Manuel Duarte	26/06/2016
RD32	https://www.djangoproject.com/	Django	Django software foundation	2016
RD33	https://docs.python.org/3/library/venv.html	Creation of virtualenv	Python software foundation	2016
RD34	https://docs.python.org/3/install/	Installing Python modules (Legacy Version)	Python software foundation	2016
RD35	ROC-TST-GSE-SUM-00035-LES/1/0	POPPy framework user manual	M.Duarte	24/06/2016

1.4 About this document

1.4.1 Access policy

The present document is accessible without any restriction.

Any modification of this document shall be verified by the RPW Ground Segment Deputy Project Manager and approved by the RPW Ground Segment Project Manager before publication.

1.4.2 Terminology

All terms used in this document, and which are not listed in the table 1 shall follow the definition in [AD2].

Name	Definition
Collaboration tools	Tools or interfaces that can help teams to collaborate on a document, file, or S/W development, test and maintenance. (e.g., SVN, JIRA, etc.)
Data products	Generic term for data produced by S/W or humans. They can be files, entries in a database, images on a screen, a given data set, etc.
Data set	A fully identified set of data (e.g., LFR level 1 data set). It shall be associated with a given support (e.g., CDF file format).
Device	Hardware, such as servers or disks.
External team/person/user	Team/person/user that does not belong to the ROC team at LESIA.
File	Any type of file including database dump files, source codes, documents, etc.
(on-)ground tests	All of the tests performed on the RPW system or sub-systems on-ground,

	especially before the launch. (e.g., performances/calibrations, SBM algorithm validations, etc.)
In-flight data	Data produced in space after the launch of the Solar Orbiter probe.
Integration	Integration of a given S/W into a given ROC pipeline.
Input keyword	Keyword provided as an input argument in a S/W. It could be a boolean flag (e.g., '--help') or not (e.g., '-f filepath').
Mission	Refers to the in-flight phase of the project (i.e., after the launch of the Solar Orbiter probe).
(Software) module	See "Software unit" definition
On-ground data	Data produced on-ground, especially before the launch of the Solar Orbiter probe.
Pipeline	Set of S/W, files or databases that allows the processing and the production of the different levels of required data.
(POPPy) Plugin	A software unit designed to be used in the POPPy framework
Production pipeline	Stable release of a pipeline used for the production of fully validated and ready to be distributed RPW data.
ROC team	People working in the ROC based at the LESIA (Paris Observatory, Meudon).
Root directory	The lower level directory in a given arborescence tree.
RPW Calibration Software	The RPW Calibration Software (RCS) regroup the software allowing the ROC to produce RPW calibrated science data.
RPW ground segment team	People working on the RPW ground segment activities, including the ROC team, the RPW PI and the RPW sub-system Lead-CoIs.
Software life-cycle	The Software life-cycle regroups all of the step to be passed by a S/W from its conception to its possible re-use. It namely concerns: the concept, specification and design steps, the development, test and validation processes, the delivery, application, maintenance and upgrades, then the sustainability and possible re-cycle.
Software unit	A software equipment of a system or sub-system
Stable release	Release of Software that has been tested and validated. A stable release is ready to be deployed in the ROC servers.
Software (root) directory	The main directory of a given S/W. It corresponds to the root directory of the Software.

Table 1. Terminology.

2 ROC SOFTWARE PRODUCTS OVERVIEW

2.1 ROC Software System (RSS)

The ROC Software System (RSS) definition gathers all of the engineering systems required to reach the ROC functional capabilities defined in the CIRD. It is composed of two systems:

- The ROC Ground Equipment Support (ROC GSE), which regroups software equipment relative to the instrument tests performed on-ground before and after the launch.
- The ROC Operations And Data System (ROADS), which concerns software equipment relative to the instrument monitoring, commanding and data processing capabilities.

Figure 1 shows the RSS product tree. The sub-systems of the ROC GSE and ROADS are presented in the next sections. The specification requirements of the RSS are given in the "ROC Software System Specification" (RSSS) document and the RSS design in the "ROC Software System Design Document" (RSSDD).

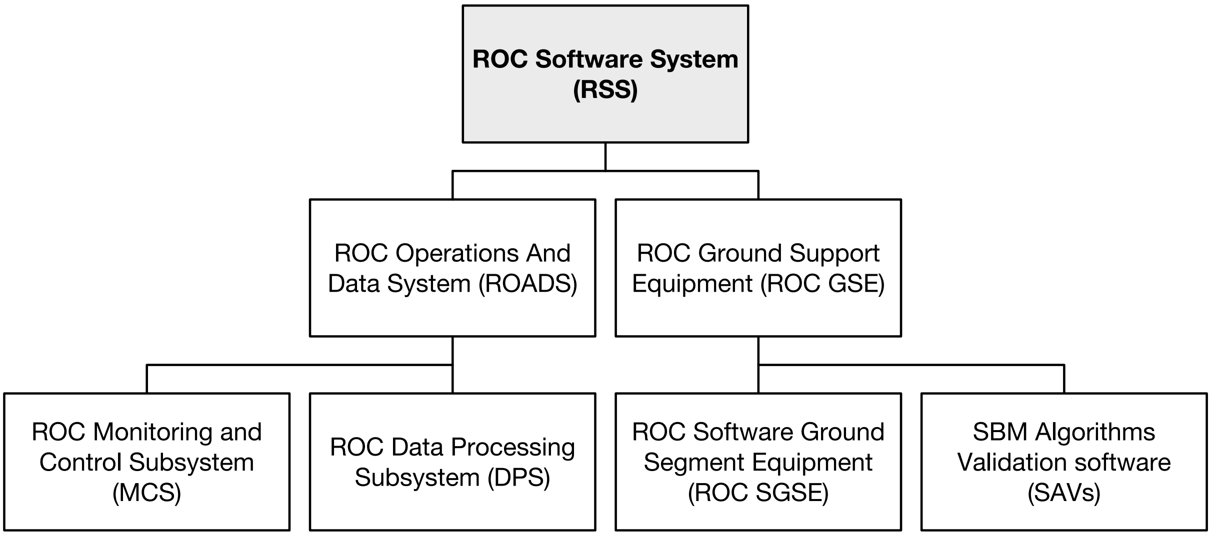

Figure 1. ROC Software System product tree.

2.1.1 ROC Operations And Data System (ROADS)

Figure 2 presents the product tree of the ROC Operations And Data System (ROADS), which is the highest-level software system in support to the RPW ground segment (RGS) activities during the Solar Orbiter mission.

It is composed of two sub-systems:

- The ROC Monitoring and Control Subsystem (MCS), which regroups the software units in charge of the RPW monitoring and commanding.
- The ROC Data Processing Subsystem (DPS), which regroups the software units in charge of the RPW data processing, dissemination and archiving.

The primary instance of the ROADS will be hosted at the LESIA (Meudon). Secondary instances will have to be deployed and run at the MOC site during the RPW related Near Earth Commissioning Phase (NECP) operations.

The software units and databases related to these subsystems are presented in the next sections.

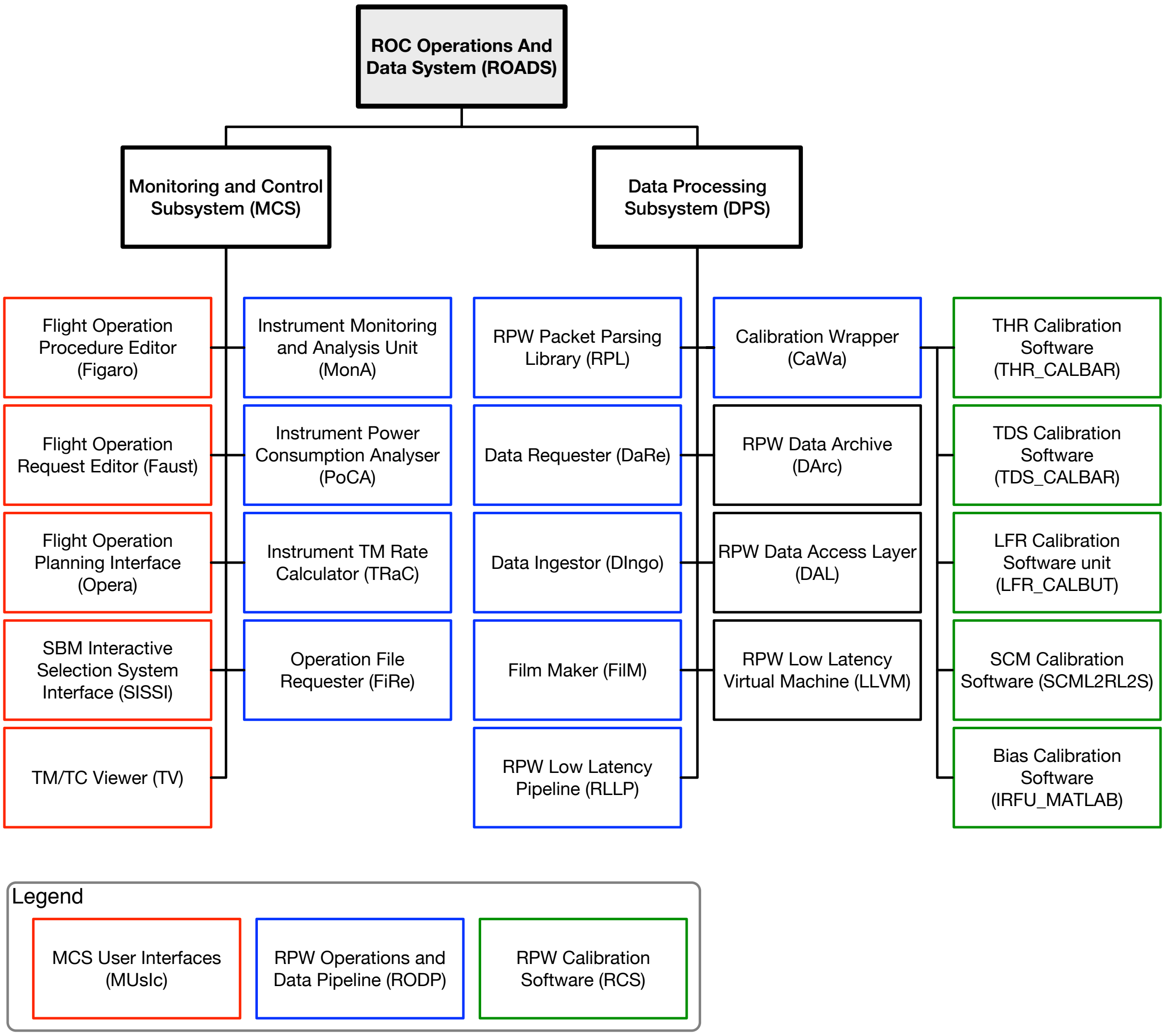

Figure 2. ROC Operations And Data System (ROADS) software products.

2.1.1.1 The ROC Operations and Data Pipeline (RODP) software units

Most of the MCS and DPS software units are gathered into common shared equipment called the "ROC Operations and Data Pipeline" (RODP). The RODP is a plugin-oriented architecture pipeline based on the "POPPy" framework concept (see section 2.3.1). It allows the ROC team to perform all of the main backend tasks in terms of data processing and operation requests, using plugins assigned to a specific task.

This approach avoids possible redundancies of processes and interfaces between the two subsystems. It also provides a centralized and homogenous core of software units / plugins built with "POPPy", a Python 3 framework inherited from the ROC-SGSE development. Finally it optimizes the development, the use and the maintenance of the ROADS software backend.

The RODP contains the following modules relative to the DPS and MCS.

RODP module name	Function	Subsystem
Data Requester (DaRe)	Module in charge of requesting data from the SOC/MOC DDS (TM/TC	DPS

	data), as well as auxiliary and input operation data from the SOC/MOC GFTS.	
RPW Packet parsing Library (RPL)	Module in charge of parsing the RPW TM/TC packets	DPS
File Maker (FilM)	Module in charge of producing the L0, L1 and HK data files from the RPW TM packets analysis	DPS
Data Ingestor (Dingo)	Module in charge of ingesting data into the MDB	DPS
ROC Calibration Wrapper (CaWa)	Module in charge calling the RPW Calibration Software (RCS)	DPS
RPW Low Latency Pipeline (RLLP)	Module in charge to run the Low Latency Data Pipeline (LLDP) for RPW.	DPS
Instrument Monitoring and Analysis Unit (MonA)	Module in charge of perform a first automated analysis of the downlinked RPW data	MCS
Instrument Power consumption Analyser (PoCA)	Module in charge of the analyse of the instrument power consumption	MCS
Instrument TM rate calculator (TraC)	Module in charge of the telemetry (TM) rate calculation	MCS
Operation File Requester (FiRe)	Module in charge of preparing the instrument operation requests	MCS

Table 2. DPS and MCS related RODP modules.

The RODP share interfaces with the ROC Mission Database (MDB), the SOC/MOC Data Dissemination System (DDS), the SOC/MOC Generic File Transfer System (GFTS) and the RPW Calibration Software (RCS), through dedicated plugins.

2.1.1.2 The Monitoring and Control Subsystem User Interfaces (MUsIc)

The MCS User Interfaces (MUsIC) are a set of Graphical User Interfaces (GUIs) dedicated to the preparation of the instrument operations and to the analysis of the instrument data. The primary instance of the MUsIC will be used at the LESIA site only.

- The RPW TM/TC Viewer (TV) is the main GUI to promptly monitor the instrument status, TM/TC history and statistics and HK/science data.
- The RPW Flight Operation Procedure Editor (Figaro) is a Web GUI to edit and save the RPW flight procedures (RFP). The RFP will serve as a primary library to generate the RPW operation requests (IOR, MDOR, PDOR).
- The RPW Flight Operation Request Editor (Faust) is the ROC GUI to prepare and submit the instrument commanding sequences, in accordance with the mission planning constraints and interface specification.
- The RPW Operation Planning Interface (Opera) provides a Web GUI to visualize and plan the instrument operations timeline.
- The SBM Interactive Selection System Interface (SISSI) is a Web GUI that allows ROC team to manage and select the SBM event data to downlink.

The architecture of these GUIs includes interfaces with the RODP and MDB to retrieve/store related data and meta-data.

The user requirements in terms of the MUSiC visual aspect and functionalities are reported in the dedicated "ROC Human-Machine interface user requirements" document.

2.1.1.3 RPW Calibration Software (RCS)

The RPW Calibration Software (RCS) are in charge of producing science calibrated data files in the CDF format during the mission. In practice a specific plugin of the RODP using a dedicated interface launches the RCS execution. The teams in charge have developed a first version of the RCS during the ground calibrations at the system level, in order to produce L2R and L2S calibrated data files.

RCS name	Function	Team in charge	Programming Languages
THR data Calibration Software (THR_CALBAR)	Produce TNR-HFR calibrated science data files in the CDF format	TNR-HFR team (LESIA, Meudon)	IDL
TDS data Calibration Software (TDS_CALBAR)	Produce TDS calibrated science data files in the CDF format	TDS team (IAP, Pragues)	IDL
LFR data Calibration Unit (LFR_CALBUT)	Produce LFR calibrated science data files in the CDF format	LFR team (LPP, Palaiseau)	Python
SCML2RL2S	Produce SCM calibrated science data files in the CDF format	SCM team (LPC2E, Orléans)	IDL
IRFU_MATLAB	Produce Bias calibrated science data files in the CDF format	Bias team (IRF, Uppsala)	Matlab

Table 3. RPW Calibration Software (RCS).

2.1.1.4 RPW Low Latency Virtual Machine (LLVM)

The RPW Low Latency Virtual Machine (LLVM) is dedicated to process the Low Latency (LL) data for RPW. It contains a lighter stateless version of the RODP, which is driven by the RLLP dedicated plugin.

An instance of this pipeline shall be delivered as a virtual appliance to the SOC.

2.1.1.5 ROC Data Access Layer (DAL)

The RPW Data Access Layer (DAL) definition encompasses all of the interface layers to access RPW data produced by the ROC.

2.1.1.6 The RPW Data Archive (DArc)

The ROC Data Archive (DArc), which regroups the ROC functional capabilities in terms of data archiving both at LESIA and external archive centres (i.e., ESA, CDPP).

2.1.2 ROC Ground Support Equipment (GSE)

Figure 3 shows the product tree concerning the ROC Ground Support Equipment (GSE). The ROC GSE application firstly concerns instrument tests performed on-ground before launch, namely: EM2/PFM instrument calibrations and SBM1/SBM2 detection algorithm validation campaigns. Nevertheless some parts of these GSE tools will be re-used during the Solar Orbiter mission to support ROC specific tasks.

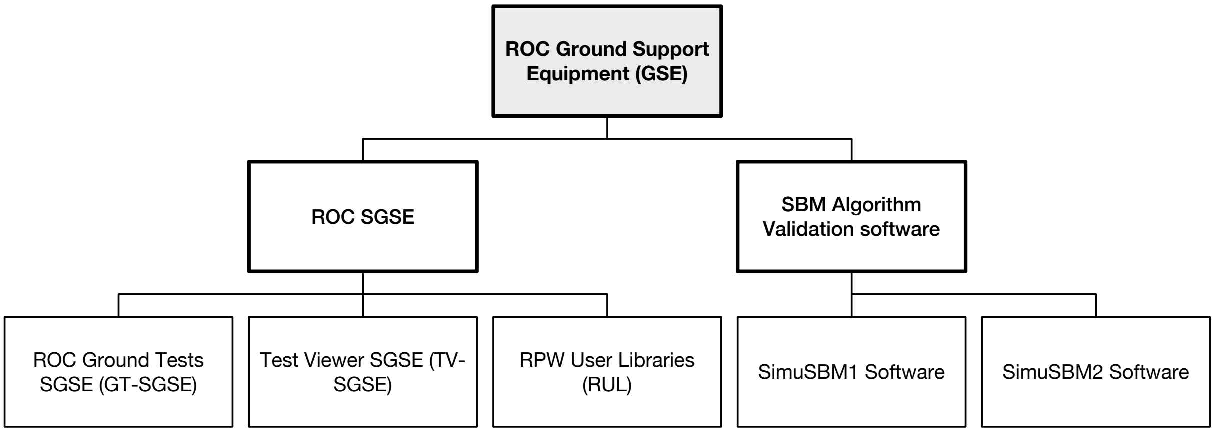

Figure 3. ROC Ground Support Equipment related software products.

2.1.2.1 ROC SGSE

The ROC Software Ground Support Equipment (ROC SGSE) provides SGSE in support to the AIT/AIV and sub-system teams, in order to analyse RPW packet data during the EM2/PFM ground calibration tests at system level. An instance of the ROC-SGSE will also be used to support the validation of the RPW Operation Requests during the mission.

The following units shall operate in the ROC SGSE:

- The ROC Ground Tests SGSE (GT-SGSE), a pipeline to process the data generated during ground tests.
- The Test Viewer SGSE (TV-SGSE), an integrated Graphical User Interface (GUI) to load and visualize data processed by the GT-SGSE.
- The ROC-SGSE Test Database (TDB), which is used to store information and data related to the tests.
- The RPW User Libraries, IDL and Python software libraries to retrieve and handle ROC SGSE data.

The developments and validation tests performed on the ROC SGSE shall benefit as much as possible to the ROADS.

2.1.2.2 RPW SBM1/SBM2 algorithm validation software

The SBM algorithm validation software (SAVs) for RPW is a set of two IDL programs "*SimuSBM1*" and "*SimuSBM2* ", in charge of helping the RPW flight software team to validate respectively the SBM1 and SBM2 detection algorithms of the DAS.

Both programs perform the following tasks:

- Read synthetized input data and produce CDF format files that comply the specification defined in [RD17].
- Perform SBM event detection and report results into XML format files that comply the specification defined in [RD17].

These two programs shall be updated to be able to process and analyse real RPW data on-board Solar Orbiter. This functionality will mainly serve to optimize the SBM event detection rate, by assessing the algorithm input parameters.

The description of the "SimuSBM1" and "SimuSBM2" programs are given in the "SBM1 Detection Algorithm Simulator" and "SBM2 Detection Algorithm Simulator" documents respectively.

Details about responsibilities, products and planning concerning these additional software products are TBD.

2.2 ROC databases

2.2.1 ROC Mission Database (MDB)

The ROC Mission Database (MDB) is the central base to run and manage the ROADS software. Especially, it permits to:

- Save information and metadata related to the data processing, event reports, instrument operation planning and requests
- Keep a history log of the ROADS software activity
- Host a local copy of the RPW Instrument database (IDB)

The MDB will be hosted on a dedicated Postgres RDBMS server, maintained by the LESIA computer service (GIGL). It inherits the developments made on the ROC-SGSE Test Database (TDB).

2.2.2 RPW Instrument Database (IDB)

The RPW Instrument Database (IDB) contains all of the information to analyse TM/TC packets for RPW. It also includes data mapping with the Solar Orbiter Mission Information Database (MiB), using SCOS-2000 format definition.

The IDB is developed and maintained by the RPW flight software team.

The ROC Instrument Database is a local copy of the IDB in the Postgres RDBMS format. The current version of the ROC IDB is built from the PALISADE XML format version of the flight software team. An upgrade is thus needed to use the SCOS-2000 format version as expected.

2.2.3 Solar Orbiter Mission Information base (MIB)

The MOC will distribute a "reference" Mission Information base (MIB) to the IT. This MIB is made up of the latest Spacecraft Reference Database (SRDB), including payload IDBs, used by the Flight Control Team (FCT) for mission operations.

The ROC shall plan to host a copy of this database, and ensure to use an IDB compatible with this MIB version.

2.2.4 ROC-SGSE Test database (TDB)

The ROC-SGSE Test database (TDB) is used by the ROC-SGSE to manage the data production during the ground calibration campaigns.

2.3 ROC auxiliary software products

The auxiliary software products concern all of the software and interfaces developed by the ROC team in parallel to or outside the scope of the RSS.

2.3.1 The Plugin-Oriented Pipeline for Python (POPPy)

The ROC-SGSE and RODP software are both built using a dedicated Python 3 framework named "POPPy" for "Plugin-Oriented Pipeline for Python".

This framework was initially written during the ROC-SGSE development, in order to offer a standard and efficient way to build a data processing pipeline with the Python 3 programming language. The initial framework was then used and extended in order to develop the RODP plugins.

The concept of POPPy is based on a network of plugins. Each plugin is assigned to a specific function (e.g., retrieving data, producing L1 science data files, etc.), performed using associated "tasks", "commands" and "targets" classes. POPPy architecture design relies on the Python 3 package mechanism, which is very powerful in terms of software installation, configuration and execution.

A full description of this framework can be found in the document [RD35].

2.3.2 RPW user software libraries

The ROC team should make available helpful routines that can be used by the RPW consortium and other IT to retrieve and handle the data produced by the ROC-SGSE.

These routines will then constitute the base of software libraries for the RPW data user community, and should be freely accessible through the RPW Web portal during the whole mission.

2.3.3 Software in support to the operation and data processing activities

In addition to the ROADS and ROC GSE software products, the ROC shall ensure that tools are developed and run in order to perform the following specific tasks:

- On-board RPW TM and power budget analysis and optimization
- On-board BIAS current calibrations from sweeping data
- Effective electrical antenna length and direction determination in-flight

A first version of these tools shall be up-and-running at the beginning of the Cruise Phase (CP).

2.3.4 Project management software products

The ROC does not plan to develop any specific software relative to the project management. The centre will only rely on existing commercial software to supervise its activities (see PMP).

2.3.5 Solar Orbiter Inter-instrument software products

The ROC does not currently plan to develop specific software related to inter-instrument collaborations inside Solar Orbiter project.

2.3.6 External collaboration software products

The ROC does not currently plan to develop specific software related to inter-mission science collaborations.

2.3.7 RPW science user community software products

2.3.7.1 MASER project collaboration

TBW

2.3.7.2 HELIO Virtual Observatory (VO) project collaboration

TBW

3 ROC DATA PRODUCTS OVERVIEW

The ROC is in charge of the data processing, dissemination and archiving tasks for RPW. The exhaustive list of instrument data to be produced during the Solar Orbiter mission is reported in the "RPW Data Products" document [RD3].

This document must at least supply a short description of each data product, including the format and the context of creation.

3.1 File naming convention, data format and metadata definition

The file naming data format, metadata and processing level definition for the RPW science data must comply the conventions defined at Solar Orbiter level [RD12].

Conventions for the data produced during on-ground calibration tests must be defined separately in [RD4].

Other rules and procedures concerning the data production at the ROC must be listed into the REG document [RD16].

3.2 Data products by category

The ROC will have to retrieve, process, generate, validate and deliver several categories of data products during the project. The following table gives a summary of these data, with the format, the provider and the main users.

Data product category	Data format	Data provider	Data user
MEB SGSE test log	MEB SGSE test log XML format	RPW MEB GSE team	ROC
ROC-SGSE data (HK and uncalibrated science L1 data)	CDF	ROC	RPW Consortium, AIT-AIV CNES/LESIA teams
Downlinked RPW TM packet data	EDDS XML format	MOC, SOC	ROC
Uplinked RPW TC packet data history	EDDS XML format	MOC, SOC	ROC
RPW TC packet sequences for	IOR XML format	ROC	SOC

instrument nominal operation requests			
RPW TC packet sequences for instrument non-routine operation requests (e.g., contingency, patches)	PDOR/MDOR XML format	ROC	MOC
RPW Flight Operation Procedures	MOIS importer format (Microsoft Excel©)	ROC	MOC
MEB SGSE GEDEON import script	MEB SGSE GEDEON XML format	ROC	ROC
RPW science data	CDF	ROC	RPW Consortium, ESA/CDPP data archives, Science Community
RPW HK parameters	CDF	ROC	RPW Consortium
Solar Orbiter MOC Orbit/attitude data	SPICE kernels	MOC, SOC	ROC
Pre-Processed Orbit/attitude data for RPW	CDF	ROC	RPW Consortium, ESA/CDPP data archives, Science Community
Quick-looks	TBD	ROC	RPW Consortium, ESA/CDPP data archives, Science Community

Table 4. Categories of ROC data products.

4 ROC SOFTWARE MANAGEMENT APPROACH

4.1 Master schedule

The general project master schedule is presented in the PMP.

4.2 Assumptions, dependencies and constraints

As a software supplier the ROC developer team is limited by the following assumptions, dependencies and constraints:

- The Solar Orbiter mission schedule and constraints
- The instrument ground segment design and acceptance reviews
- The SOC/MOC interface test schedule [RD20]

- The SOV/SVT campaign planning [RD29]
- The SOC Low Latency (LL) test and delivery schedule [RD30]
- The RPW instrument calibration plan [RD28]
- The RPW DPU flight software release timeline
- The MEB GSE software release timeline

4.3 Software development related risks

The following table attempts to identify the categories of risk that could potentially become points of potential failures for the ROC software activities.

Category of risk	Cause(s)	Consequence(s)	Severity	Probability	Solution(s) to mitigate the risk
Hardware/Operating System (OS) failure at LESIA	Obsolescence, overvoltage	Loss of instrument operation and monitoring capabilities, science data delivery delay	High	Medium	Plan backup systems to be rapidly deployed. To mitigate the risk at OS level, the ROC uses virtual machines as primary servers for the RSS. (Hardware/OS recovery is the responsibility of the GIGL)
ROC Software System failure at LESIA	Bugs, regression, retro-compatibility not supported	Loss of instrument operation and monitoring capabilities, science data delivery delay	High	TBD	A stable version of the RSS shall be ready to be deployed quickly (use of Git repos + Python installation package capabilities)
Hardware/OS/RSS failure at the MOC site during the RPW related NECP operations	See previous risks	NECP operations related to RPW cannot be performed correctly by the ROC team	Critical	TBD	Plan to have two backup software systems ready to be used at the MOC. This solution might impact the hardware/software facilities to be deployed at the MOC site.
Obsolete software/hardware	Software updates are not retro-compatible, unavailable hardware devices	Risk of software facilities failures, loss of ROC facilities	Low	Low	Use as much as possible stable, portable and time-honoured software technology
Unexpected personnel reduction in the developer team	Transfer, voluntary redundancy, contract end, disease,	Under-sized team, loss of expertise	High	Low	Prompt Non permanent post hiring or internal replacement by the LESIA

	pregnancy, ...				
Lack of experience	Developer not familiar with a software technology	Software quality loss, required specification not reached, software delivery delay	Medium	Medium	The ROC shall offer the possibility to its developer team to get professional training.

Table 5. Categories of risk related to software development.

4.4 Monitoring and controlling mechanisms

Except reviews/key points, no specific monitoring and controlling mechanisms are planned to follow the ROC software development.

Nevertheless, the RPW Ground Segment Project Manager must plan regular progress meetings, as defined in the PMP. Additionally, the ROC software validation process must lead to the edition of a validation report, as documented in the ROC software Validation Plan (SVP).

Design and acceptance reviews shall also be key dates to evaluate the software concept and progression.

4.5 Staffing plan

This section presents the responsibilities and organization of key personnel and institutes involved in the ROC software development.

4.5.1 ROC project organization and responsibilities

The organization, personnel and responsibilities in the ROC project are detailed in the PMP.

4.5.2 ROC software product responsibilities

4.5.2.1 Customer versus supplier roles

The following table resumes the customer versus supplier roles concerning the software and database units required to perform the ROC tasks.

Software System	Customer	Supplier
ROADS	ROC	ROC
ROC GSE	ROC	ROC
LLVM	SOC / ROC	ROC
RCS	ROC	RPW analyser/sensor teams (BIAS, SCM, THR, TDS and LFR), see details in the next sections.
MEB GSE tools and databases	ROC	MEB GSE team
ROC MDB	ROC	ROC
ROC-SGSE TDB	ROC	ROC
RPW IDB	ROC	Flight software team
SolO Mission Information Database	ROC	MOC

Table 6. ROC software customers versus suppliers.

4.5.2.2 Key personnel responsibilities

According to the CIRD, The RPW Ground Segment Project Manager, **Yvonne de Conchy**, shall ensure that the design of the ROC software and data products meet the requirements addressed at both the RPW and Solar Orbiter levels. She approves the ROC engineering documentation.

The RPW Ground Segment Deputy Project Manager, **Xavier Bonnin**, supervises the ROC software and data design definition, development, testing and validation. After the launch, he ensures that the ROC software is always up and running.

The RPW sensor/analyser lead CoIs: **Jan Soucek** (TDS), **Thomas Chust** (LFR), **Milan Maksimovic** (THR), **Andris Vaivads** (BIAS) and **Volodya Krasnosselskikh** (SCM) shall ensure that the software, data and documentation they have to deliver to the ROC are compliant, operational and up-to-date during all of the phases of the mission, starting at the instrument switch-on during the Near Earth Commissioning Phase (NECP). This concerns essentially the RCS and corresponding data products.

4.5.3 RPW Calibration Software (RCS) responsibilities

4.5.3.1 During the Solar Orbiter mission

The following table summarizes the list of RCS to be developed by whom and for each category of data products during the Solar Orbiter mission.

Software name	Entity in charge of the software development, test, validation, maintenance and delivery	Entity in charge of the software execution	Data products category	Software programming language
IRFU-MATLAB	BIAS team (IRF-U)	ROC team (LESIA)	- TDS (LFM) / LFR electric waveforms measured at antenna level (L1➔L2) TBC - BIAS specific data (e.g., sweep data, derived data, etc.)	Matlab
LFR_CALBUT	LFR team (LPP)	ROC team (LESIA)	- All calibrated data measured at SCM level (L1➔L2), except magnetic waveforms TBC - All calibrated data measured at antenna level (L1➔L2), except electrical waveforms TBC	Python
SCML2RL2S	SCM team (LPC2E)	ROC team (LESIA)	- TDS/LFR magnetic waveforms measured at SCM level (L1➔L2) TBC	IDL
TDS_CALBAR	TDS team (IAP)	ROC team (LESIA)	- All calibrated data measured at SCM level (L1➔L2), except magnetic waveforms TBC - All calibrated data measured at antenna level (L1➔L2), except electrical	IDL

			waveforms in LFM mode. TBC	
THR_CALBAR	THR team (LESIA)	ROC team (LESIA)	- All calibrated data measured at antenna level (L1➔L2)	IDL

Table 7. RCS responsibilities during Solar Orbiter mission.

4.5.3.2 During the ground calibrations

The following table summarizes the list of RCS to be developed by whom and for each type of science data products during the ground calibrations.

Software name	Entity in charge of the software development, test, validation, maintenance and delivery	Entity in charge of the software execution	Science data product type	Software programming language
IRFU-MATLAB	BIAS team (IRF-U)	BIAS team (IRF-U)	- TDS (LFM) / LFR electric waveforms measured at PA LF level (L2S) - BIAS specific data (e.g., sweep data, derived data, etc.)	Matlab
LFR_CALBUT	LFR team (LPP)	LFR team (LPP)	- All calibrated data measured at LFR receiver level (L2R) - All calibrated data measured at SCM level (L2S), except magnetic waveforms - All calibrated data measured at PA level (L2S), except electrical waveforms	Python
SCML2RL2S	SCM team (LPC2E)	SCM team (LPC2E)	- TDS/LFR magnetic waveforms measured at SCM level (L2S)	IDL
TDS_CALBAR	TDS team (IAP)	TDS team (IAP)	- All calibrated data measured at LFR receiver level (L2R) - All calibrated data measured at SCM level (L2S), except magnetic waveforms - All calibrated data measured at PA level (L2S), except electrical waveforms in LFM mode.	IDL
THR_CALBAR	THR team (LESIA)	THR team (LESIA)	- All calibrated data measured at THR receiver level (L2R) - All calibrated data measured at PA	IDL

			level (L2S)	

Table 8. RCS responsibilities during ground calibrations.

4.5.4 RPW science data production, validation and archiving responsibilities during the Solar Orbiter mission

4.5.4.1 Science data production

The ROC will have in charge to produce all of the RPW science data during the Solar Orbiter mission. This includes also the calibrated science data using the RCS delivered by the sub-system teams.

4.5.4.2 Science data quality validation responsibilities

Several steps of validation will have to be realized between the RPW packet data retrieval at the LESIA and the delivery of definitive science calibrated data to the ESA archive centre.

Details about the data validation processes have still TBD, but the ROC will have to lead this task and be the final contributor before delivery.

Table 9 gives the person in charge of the calibrated science data validation (i.e. data validation leader) at the LESIA, given by science data products category.

Science data products category	Data validation leader at LESIA	Data validation contributor(s)
LFR spectral products – electric component	M.Maksimovic (TBC)	T.Chust (LPP)
LFR waveform products – electric component	F.Pantellini (TBC)	A.Vaivads (IRF-U), T.Chust (LPP)
LFR spectral products – magnetic component	O.Alexandrova	M.Kretzschmar (LPC2E), T.Chust (LPP)
LFR waveform products – magnetic component	O.Alexandrova	M.Kretzschmar (LPC2E), T.Chust (LPP)
LFR BP products	TBD	T.Chust (LPP)
THR products - electric component	L.Matteini	M.Maksimovic
THR products - magnetic component	L.Matteini	M.Kretzschmar (LPC2E), M.Maksimovic
TDS histogram, stats, MAMP and SM-PDS products	M.Maksimovic (TBC)	J.Soucek (IAP)
TDS waveform products – electric	M.Maksimovic (TBC)	J.Soucek (IAP)

component		
TDS waveform products – magnetic component	M.Maksimovic (TBC)	M.Kretzschmar (LPC2E), J.Soucek (IAP)

Table 9. Science calibrated data validation leader.

4.5.4.3 Science data archiving responsibilities

All of the processed science data files will be stored at LESIA. Preliminary data will have to be distributed within a short time - less than a TBD - to the RPW consortium for analysis. Then, definitive data will be delivered to ESA for archiving within three months after the TM packet data were received by the ROC (TBC).

Data archiving at CDPP (Toulouse) shall also be planned; details have TBD.

4.5.5 RPW science data production, validation and archiving responsibilities during the ground calibrations at system level

4.5.5.1 Science data production responsibilities

During the EM blank calibrations in CNES (Toulouse), the AIT/AIV CNES team will be in charge to ensure the production of L1 uncalibrated science data files, using the local instance of the ROC-SGSE. Resulting CDF format files will have to be then sent to the LESIA using the dedicated exchange file interface (SEF).

During the thermal calibration on PFM in LESIA (Meudon), the ROC team will produce it-self the L1 uncalibrated science data from its ROC-SGSE instance. This instance will be deployed on its production server at LESIA. Moreover it shall have a direct access to the CNES MEB GSE database in order to be able to retrieve test log data produced during calibrations.

The L2R/L2S science calibrated data production will be ensured by the sub-system teams them-selves, from the L1 data files provided by the ROC. Data will have to be available within a short time - less than 72h - to the RPW consortium for analysis.

At the end all of the processed science data files will be archived at the LESIA.

4.5.5.2 Science data quality validation responsibilities

The ROC team does not plan to validate the science data quality during the ground calibrations. The primary validation of these data will have to be performed directly by the sub-system teams involved.

4.5.5.3 Science data archiving responsibilities

All of the relevant data produced during the ground calibration campaigns, including stand-alone, shall be archived at the ROC. The detailed list of data products and associated software and documentation to be archived must be defined in a specific document TBD.

4.5.6 ROC software logistics support responsibilities

4.5.6.1 At LESIA and Paris Observatory

It has been decided that the computer services of the LESIA (GIGL) and of the Observatoire de Paris (DIO), will have in charge to supply facilities to the ROC for the following needs:

- Hosting the ROC hardware equipment, namely: servers and disks

- Ensuring the availability of the network and accesses, which permit data exchange between the ROC equipment and with the external servers (i.e., intranet/internet accesses, NFS-like mounting systems, SFTP/HTTPS servers).
- Offering up-and-running collaboration tools (Web page, Git/SVN, Wiki, mailing lists, etc.)
- Ensuring the maintenance and availability of these device and facilities.
- Providing a prompt helpdesk to the ROC team in case of equipment failures
- Supporting ROC team in operational logistics (e.g., telecon. user assistance).

The following table gives the list of equipment and the computer service in charge of the maintenance/assistance. Details about the concerned ROC device can be found in the section. Obsolete equipment is written in grey.

ROC facilities	Service in charge of the maintenance/assistance
Servers	GIGL
Data disks	GIGL
Communication accesses	GIGL (DIO at Observatory level)
RPW Web portal	GIGL
ROC Wiki pages	GIGL
ROC SVN repository	GIGL
ROC Git repositories	DIO
ROC mailing lists	DIO
ROC JIRA project pages	GIGL
Telecon system	GIGL
Visiocon	GIGL

Table 10. ROC logistics support at LESIA and Paris Observatory.

4.6 Software procurement process

ROC software development, test and application environments essentially rely on free software projects (e.g., Python, Linux.) to work.

For commercial software such as IDLⒸ and MatlabⒸ programming languages, it has been decided with the GIGL that the ROC developer teams can use the licences supplied by the LESIA.

An Operating System (OS) on WindowsⒸ must be also available in order to run a ROC instance of the MEB GSE software suite.

4.7 Supplier management

4.7.1 RCS specific development plan

4.7.1.1 Convention and procedures

In the case of the RCS, the teams must follow the specific convention and procedures defined in the "ROC Engineering Guidelines for external User" (REGU) document [RD17].

4.7.1.2 Monitoring and controlling mechanisms

Monthly engineering telecons must be organized by the ROC team to follow the RCS development, validation and delivery, including the software data products and the interface with the RODP.

4.7.1.3 RCS integration philosophy

Since the RCS are developed with different programming languages, the ROC shall plan to develop a specific interface to call the RCS from the RODP. The description of this interface shall be given in a dedicated Interface Control Document (ICD).

4.7.1.4 RCS integration tests schedule

The table below gives the current schedule of the main integration tests concerning the RCS for the mission. Details will be described into the RVP.

Validation Test	Description and main objectives	Due date
RCS ICD validation test	Test the RCS command line interfaces are compliant with the RCS ICD.	2017-03-31 (TBC)
RCS implementation test	ROC team to test the RCS implementation into the RODP.	2017-06-30 (TBC)
RCS data products validation	Test to validate that data files produced by the RCS are fully compliant with the ROC data format and metadata definition.	2017-10-31 (TBC)
Full validation test	An end-to-end test to check the full process is ready for the mission.	2018-03-31 (TBC)

Table 11. RCS implementation tests schedule.

4.7.1.5 RCS documentation delivery schedule

Concerning the RCS documentation, the teams in charge shall produce a Software Requirement Specification (SRS) document and a Software User Manuel (SUM). First versions of these documents shall be delivered to the ROC according to the following schedule. After, up-to-date versions of the SUM shall be supplied to the ROC with new software release.

Document(s)	Version	Due date
SRS	Preliminary version	2017-03-31 (TBC)
SRS / SUM	First release / Preliminary version	2017-10-31 (TBC)
SUM	First release	2018-03-31 (TBC)

Table 12. RCS documentation.

Note: templates of SRS and SUM will be provided by the ROC.

5 ROC SOFTWARE DEVELOPMENT APPROACH

5.1 Strategy to the software development

5.1.1 General convention and philosophy

5.1.1.1 ROC software development, test and validation philosophy

Except the RCS, all of the software relative the RSS will be developed, tested and validated by the ROC at the LESIA on a dedicated development server. The RCS must be developed and tested by the sub-system teams before delivery. The ROC team will only performed end-to-end integration tests of the RCS. The final validation of the RPW LLVM is performed by the SOC.

5.1.1.2 ROC software execution and maintenance philosophy

All of the software relative to the RSS, including the RCS, shall be hosted and run at the LESIA only. It required the RCS to be delivered to the ROC by teams in charge, following the procedures defined in the REGU document. Nevertheless, the primary instance of the LLVM for RPW will be run at the SOC. And the ROC team shall plan to deploy secondary instance(s) of the ROADS at the MOC site during the RPW related NECP operations.

Note that the RCS teams can produce data from their local software instance, to support calibration validation and software upgrading for instance, but these data shall be not released publicly without the permission of the RPW PI.

5.1.1.3 ROC software development conventions, procedures and rules

The RPW Ground Segment Software Manager shall define general conventions, procedures and rules concerning the ROC software development at LESIA. These conventions shall be listed into the dedicated REG document [RD16].

An extension document “ROC Engineering Guidelines for External Users” (REGU) document [RD17] shall address to the sub-system teams the additional rules specific to the RCS development.

In all cases, the RPW Ground Segment Software Manager shall ensure the homogeneity and the re-usability of the software development and application environment. Moreover it shall ensure the technologies used are as much as possible sustainable over all of the phases of the project.

5.1.2 Software versioning convention

Each software release shall be tagged with a unique version number sequence “X.Y.Z”, where “X” is an integer indicating S/W major changes, “Y” is an integer indicating S/W minor changes and “Z” is an integer indicating a revision (e.g., bug correction, small corrections).

Software preliminary versions (e.g., alpha, beta, etc.) shall have their release number “X” equals to 0.

The first stable release of S/W shall have its major number “X” equals to 1, its minor number “Y” equals to 0 and its revision number “Z” equals to 0 (i.e., “1.0.0”).

The version number shall never contain letters as prefix (e.g., “0.1.2b”).

5.1.3 Software project development configuration

5.1.3.1 Teams collaboration tools

The S/W development inside the ROC project shall rely on the following collaboration tools:

- **Web portal** (http://rpw.lesia.obspm.fr/) that gives a single centralized access to the interfaces, software and data used by team inside the project. Note that confidential content can require a restricted access to the intranet. This portal is managed using the SPIP framework and hosted on the GIGL Web server.
- **Two Wiki pages** "ROC" and 'RPW Ground Segment" in support to the project development activities at the LESIA (restricted access) and RPW teams (public access) levels respectively. The Wiki page run on the Redmine open source project, which is hosted and maintained by the GIGL.
- **JIRA Issue tracker tool** to keep track of development/maintenance activities, and to perform SCRUM sprints. The exhaustive lists of JIRA projects are given in the PMP.
- **Sympa Mailing-lists** to communicate on specific topics inside the project. The exhaustive lists of mailing lists are given in the PMP.
- **Git revision control software** to manage the software development and delivery. The Git server shall at least contain the following repositories:
    - **ROC-SGSE**, stores ROC-SGSE source files
    - **TV-SGSE**, stores TV-SGSE source files
    - **RODP**, stores RODP source files
    - **TV**, stores TV source files
    - **Faust**, stores Faust source files
    - **Figaro**, stores Figaro source files
    - **Opera**, stores Opera source files
    - **SISSI**, stores SISSI source files
    - **LLVM**, stores RPW LLVM source files
    - **SimuSBM1**, stores SimuSBM1 source files
    - **SimuSBM2**, stores SimuSBM2 source files
    - **AdminTools**, stores programs to administrate the ROC systems
    - **TestTools**, stores programs to perform RSS validation tests
    - **DataPool**, stores ROC support data (e.g., CDF skeleton tables)
- **SubVersioN (SVN) revision control software**, was initially used by the ROC developer team. The ROC SVN repository is now only used by the teams to deliver the RCS and the corresponding CDF skeletons.

The way these tools must be used by the ROC and external teams is described in the REG and the REGU respectively.

5.1.3.2 ROC software versioning management file

Information history about the software run at the ROC shall be reported into a dedicated Excel© file:

"*roc_sw_info_IssXX_RevYY.xlsx*",

where "*XX*" and "*YY*" are respectively the issue and revision 2-digits numbers of the document.

This document shall be under the responsibility of the RPW Ground Segment Deputy Project Manager and must be always up-to-date.

5.1.3.3 ROC facilities access management file

Information about the user access privileges to the ROC facilities, servers and disks, shall be reported into a dedicated Excel© file:

"*roc_user_access_IssXX_RevYY.xlsx*",

where "*XX*" and "*YY*" are respectively the issue and revision 2-digits numbers of the document.

This document shall be under the responsibility of the RPW Ground Segment Deputy Project Manager and must be always up-to-date.

5.1.4 ROC engineering validation plan

The way the whole ROC engineering infrastructure will be validated shall be described in the SVP document. It shall include the list of tests to be achieved, their purpose, the procedure to follow to perform and validate them, the person in charge and the logistic required for the tests. A timeline presenting the deadlines of the validation test campaigns, in agreement with the ESA schedule [RD20], shall also be provided.

5.1.5 ROC software product assurance plan

The software product assurance plan of the ROC is described in the "ROC Software Product Assurance Plan" document (SPAP) [AD4].

5.1.6 ROC data product management approach

5.1.6.1 ROC data set listing files

The ROC team shall write and maintain an up-to-date list of data to be produced by the RODP at the LESIA during the whole mission. Each line of this list shall correspond to a given data set. The list of fields to be provided for each data set is given in the section 5.1.6.3.

The same work shall be first done for the data processed by the ROC-SGSE tool during ground calibration tests.

5.1.6.2 ROC data set listing file naming convention

List of RPW data sets produced by the ROC during the whole mission shall be reported into a dedicated Excel© file named:

"*rpw_datasets_IssXX_RevYY.xlsx*",

where "*XX*" and "*YY*" are respectively the issue and revision 2-digits numbers of the document.

List of data sets produced by the ROC SGSE during ground tests shall be reported into a similar Excel© file named:

"*roc-sgse_datasets_IssXX_RevYY.xlsx*",

where "*XX*" and "*YY*" are respectively the issue and revision 2-digits numbers of the document.

Both documents shall be under the responsibility of the RPW Ground Segment Deputy Project Manager and must be always up-to-date.

5.1.6.3 ROC data set listing file description

The following table gives the list of columns to be found in the ROC data set files.

Column name	Description	Priority
Dataset ID	Identifier of the dataset in the ROC system. It shall be unique.	Mandatory
Dataset name	Full name of the dataset	Mandatory
Dataset description	Description of the dataset	Mandatory
Data processing level	Processing level of the dataset	Mandatory
Data file format	File format of the dataset	Mandatory
Data file periodicity	Periodicity of production of the data file for the current dataset (e.g., daily, monthly, per event, etc.)	Mandatory
Data validation leader	Entity (person, institute or team) in charge of the dataset validation	Mandatory
Data validation contributor(s)	Entities (person, institute or team) that contribute to the data validation.	Optional
Data definition schema file	Name of file containing the dataset definition schema (e.g., CDF skeleton, XSD, etc.)	Optional
Parent dataset IDs	ID(s) of the parent dataset(s) required to produce the current dataset.	Optional
Calibration table file(s)	Name of the file(s) containing the table to calibrate the data of the current dataset.	Optional
Software name	Name of the software used to produce the current dataset	Mandatory
Software development/maintenance leader	Entity (person, institute or team) in charge of developing and maintaining the software.	Mandatory
Software development/maintenance contributor(s)	Entities (person, institute or team) that contribute to the software development and maintenance.	Optional
Software application leader	Entity (person, institute or team) which is effectively run the software.	Mandatory

Table 13. ROC dataset listing file columns.

5.2 Software project development life cycle

5.2.1 ROADS development life cycle

The ROADS development life cycle is illustrated on the figure below.

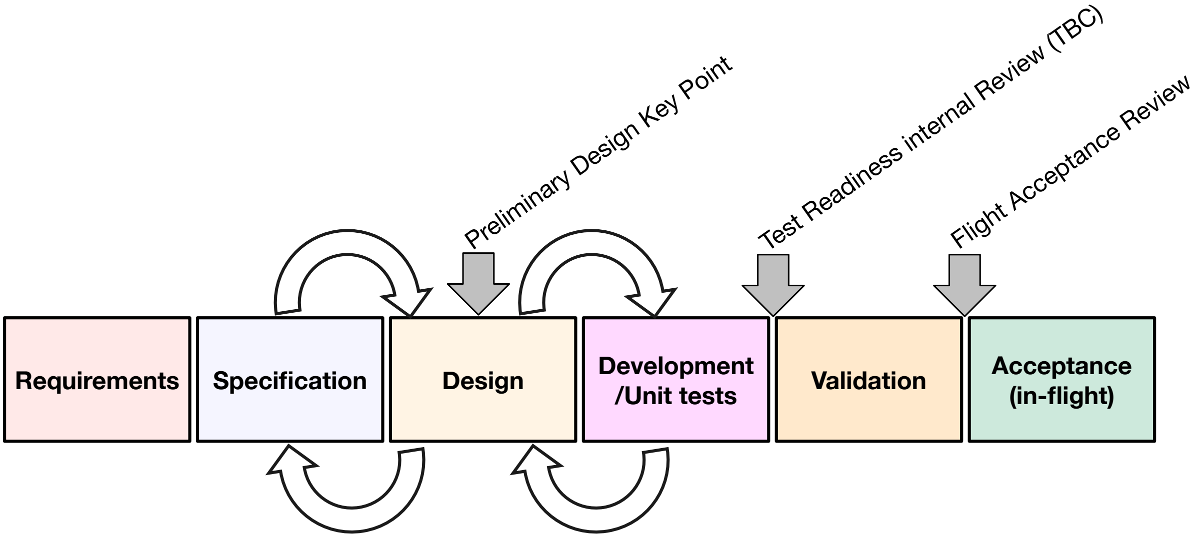

Figure 4. ROADS project development life cycle.

It is closed to a typical waterfall life cycle, but iterations (curved arrows on the figure) must be introduced since the initial specification is susceptible to be regularly updated; essentially due to the evolution concerning the design of the interface with the SOC/MOC. There are iterations between the specification and design in one hand, and the design and the development phases in the other hand thus concern mainly the RODP. Especially dedicated unit tests will have to be implemented to test the main functionalities of the RODP.

The table below gives the list of main technical documents to be expected at the end of each phase. The list of documents to be delivered before each review/key point is given in the SPAP [AD4].

Development phase	Expected technical documents
Requirements	ROC concept and implementation requirements, ROC human-machine interface user requirements
Specification	ROC Software System Specification and related ICD
Design	ROC Software System Design Document
Development / unit tests	
Validation	ROC Software system User Manual, ROC Software System Reference Manual
Acceptance (in-flight)	

Table 14. Expected technical documents during ROADS development life-cycle.

5.3 Software engineering standards and techniques

5.3.1 RODP engineering standards and techniques

The RODP is written using the Python 3 programming language with the virtual environment [RD33] and package mechanisms [RD34].

5.3.2 MUsIc engineering standards and techniques

The MUsIc GUIs will be developed and deployed using Django [RD32], a Web framework for Python.

5.3.3 LLVM engineering standards and techniques

The LLVM engineering standards and techniques can be found in [RD10].

5.4 Software development and software testing environment

5.4.1 Hardware equipment

According to the responsibilities defined in the section 4.5.6, all of the ROC hardware: servers, data disks and interfaces, are maintained and administrated by the GIGL.

5.4.1.1 ROC processing servers

Table below provides the name and function of the ROC servers hosted at LESIA.

Server name	Function	Configuration	Access privileges
roc.obspm.fr	The "production server" is used to run the primary instance of the RODP in prod.	• A x86 Intel architecture VM • Inter(R) Xeon(R) 2.4 GHz • Linux Debian Jessie OS. • 871 GBytes	The administration of this server is done by the GIGL. Only accessible to the ROC developer team through SSH.
roc-dev.obspm.fr	The "development server" is dedicated to develop, test and validate the RODP, before implementation on the "production" server. Teams to test and validate their RCS, before delivery to the ROC, can also use it.	• A x86 Intel architecture VM • Inter(R) Xeon(R) 2.4 GHz • Linux Debian Jessie OS. 871 GBytes	The administration of this server is done by the GIGL. Accessible to the ROC team and RPW consortium people involved in the ground segment activities.
roc-web.obspm.fr	The "web server" is used to host the MUsIC (i.e., Figaro, Faust, Opera, SISSI and TV)	• A x86 Intel architecture VM • Inter(R) Xeon(R) 2.4 GHz • Linux Debian Jessie OS. 871 GBytes	The administration of this server is done by the GIGL. Only accessible to the ROC developer team through SSH.
roc-low.obspm.fr	The "low latency server" is dedicated to prepare the LLVM for the SOC	• Same than the EUI LLVM	The administration of this server is done by the GIGL. Only accessible to the ROC developer team through SSH.
roc-nfsvm	VM used by the LLVM to test the NFS mounting	• Same than the EUI NFS testing VM	
rocpc1.obspm.fr	This is the server used by the ROC to host and run its MEB GSE tools and database instance	• Windows 7	The administration of this server is done by the ROC developer team.
arietis.obspm.fr	The *arietis.obspm.fr* server hosts the RPW Web page at the LESIA.	TBD	The administration of this server is done by the GIGL, with specific accesses for the ROC team in order to edit the RPW Web page through the SPIP[1] framework.
lesia11.obspm.fr	The "data storage" server is	TBD	The administration of this

[1] http://www.spip.net/

	used to reach the ROC data disks, which contain the ROC data products. It currently provides 16 Terabytes of space. This server is only accessible in read/write privilege from the roc-dev.obspm.fr and roc.obspm.fr via a ZFS mounting system. A HTTPS server also runs, giving access of private/public directories.		server is done by the GIGL, with specific accesses for the ROC and sub-system teams.
TBD	The "LESIA database server" will be used to host the ROC MDB and MIB databases.	TBD	The administration of this server is done by the GIGL, with specific accesses for the ROC team to its databases.
TBD	One ore more terminals to monitor the RPW activity by displaying the TV 24h/24h. It will be in the operation room of the ROC	TBD	TBD
TBD	Specific server/terminal to be deployed and run at the MOC during the NECP operations.	TBD	TBD
git.obspm.fr	Used to host the git repositories of the ROC	TBD	The administration of this server is done by the DIO, with specific accesses for the ROC team to create and update the repositories. Some repo. must also be accessible to the RPW teams.

Table 15. ROC operating servers.

Note that the development and production servers are not sized to store a large amount of data volume, and shall be used for program testing and execution only.

5.4.1.2 ROC data storage capability

The ROC data product files are currently stored using two 8 Terabytes data disks of the *lesia11.obspm.fr* server. The total data storage capacity will be extended to 32 Terabytes before the launch.

5.4.2 ROC hardware communication interfaces

The deployment and maintenance of the ROC device interfaces are under the GIGL responsibilities. It includes:

- Internal communication interfaces between the ROC servers
- Internal communication interfaces between the ROC servers and data disks
- Internal communication interfaces between the ROC servers and ROC team computers.
- Communication with the Internet network through the dedicated interfaces.

5.4.3 Software environment

5.4.3.1 ROC server Operating Systems (OS)

The ROADS servers will run on the same **SMP Debian Jessie x86_64** Operating System (OS).

The default command processor of the system will be the **Bourne-Again SHell** (BASH) V4. In consequence, any S/W executed on the ROC servers shall work within this OS environment.

An additional Windows 7 server is also available to use the MEB GSE tools and database instances of the ROC.

5.4.3.2 ROC LLVM OS

The RPW LLVM is a tailored copy of the LLVM developed by the Solar Orbiter EUI team using the CentOs OS.

5.4.3.3 User account management

Each person involved in the RGS activities can have a dedicated user account on the *roc-dev* development server, as well as a dedicated space on the ROC data disk to store its personal data related to the project.

By default the free spaces allowed to users on the ROC server and data disk are respectively limited to **5 Gigabytes** and **20 Gigabytes**. These quotas can be increased on demand.

The complete list of user accounts and access privileges shall be reported on the "ROC device access document".

Access to the ROC development and production servers shall be only possible using the SSH protocol.

5.4.3.4 Technical account management

In addition to the personnel user accounts, the following so-called "technical" accounts shall be created to manage and run the ROC pipelines.

Technical account	Description	Development server	Execution server	Comment
roc_rdop	Technical account for the RPW Data and Operations Pipeline execution	roc-dev.obspm.fr	roc.obspm.fr	Access to this account is restricted to the ROC developer team only
roc_sgse	Technical account for the ROC SGSE execution	roc-dev.obspm.fr	roc.obspm.fr	Access to this account is restricted to the ROC developer team only
roc_rllp	Technical account for the RPW Low Latency Pipeline execution	roc-low.obspm.fr	roc-low.obspm.fr	Access to this account is restricted to the ROC developer team only
rocuser	Technical account used to have a generic read-only access to the Git repositories of the ROC			

Table 16. Technical accounts.

Note: an additional technical account called "rocuser" has been created to provide a generic read-only access to the Git repositories of the ROC.

5.4.3.5 Root account management

The OS administration and maintenance of the servers and disks are ensured by the GIGL. However people needing to perform specific operations at the OS level can ask for a root access. These people shall be clearly identified and belong to the ROC team only. Any request for a root access shall be sent to the RPW Ground Segment Software Manager.

5.4.3.6 ROC data access policy

All of the data produced by the ROC will be stored in dedicated data disks (see section 5.4.1.2). Access to the data disks will be possible from the ROC servers using the ZFS mounting system. Specific access privileges shall be defined for each user. Some directories in the data disks will be also visible from Internet to allow people to retrieve data.

User access privileges shall be reported into the "ROC device access management file".

5.4.3.7 Programming languages

Most of the ROC software is based on the Python 3 programming language, which becomes both a major programming language, and a more and more used language in the astronomy community.

The following table gives the list of main ROC software, the programming language name and the main modules to be used for the development.

Software	Programming Languages	Main external modules/libraries
RODP	Python 3 + Cython	numpy, SQLAlchemy
LLVM	Debian CentOs, Python 3 + Cython	numpy, SQLAlchemy
RPL	Python 3 + Cython	numpy
TV	Python 3 + Django (TBC)	numpy, matplotlib (TBC)
Figaro	Python 3 + Django (TBC)	numpy, matplotlib (TBC)
Opera	Python 3 + Django (TBC)	numpy, matplotlib (TBC)
Faust	Python 3 + Django (TBC)	numpy, matplotlib (TBC)
SISSI	Python 3 + Django (TBC)	numpy, matplotlib (TBC)
GT-SGSE	Python 3 + Cython	numpy, SQLAlchemy
TV SGSE	Python 3 + PyQt5	numpy, matplolib, SQLAlchemy
SimuSBM1 Software	IDL	
SimuSBM2 Software	IDL	
THR_CALBAR	IDL	
TDS_CALBAR	IDL	
SCML2RL2S	IDL	
LFR_CALBUT	Python 3	
IRFU-MATLAB	Matlab 5	

Table 17. ROC software programming languages.

5.4.3.8 Database management systems

The following table presents the databases involved into the ROC activities and the associated management system.

Database	Functions	Management system

ROC-SGSE Test Database (TDB)	Database of the ROC-SGSE	PostgreSQL
ROC Instrument Database (IDB)	ROC instance of the RPW IDB	PostgreSQL
ROC Mission Database (MDB)	Database of the ROADS	PostgreSQL
MEB SGSE Database	MEB GSE database	MySQL
RPW Instrument Database (IDB)	RPW IDB	XML (PALISADE) / PostgreSQL (SCOS-2000)

Table 18. List of databases involved in the ROC activities.

Note: in practice the ROC IDB will be included as a schema in the ROC MDB Postgres database.

5.4.3.9 Common software libraries

The following table presents the list of common libraries that are currently installed on the *roc* and *roc-dev* servers.

Library	Description
CDAWlib	CDAW IDL library
CDF	NASA CDF library
Cython	C for Python module
jsonschema	JSON format handler for Python
matplotlib	Mathematical plotting library for Python
mysqlclient	MySQL-python fork
nose	extended unittest for Python
numpy	Scientific Python library
psycopg2	Python PostgreSQL database adaptater
pyparsing	Python parsing module
qtconsole	Qt-based console for working with Jupyter kernels
SolarSoft	IDL Solar Software (SSW) library
SPICE - ICY	IDL toolkit of the NAIF SPICE library
SPICE - JNISPICE	Java toolkit of the NAIF SPICE library
SQLAlchemy	Database ORM module for Python
django	Python Web framework

Table 19. List of ROC software libraries.

Note: the SolarSoft (SSW) library only works under a csh/tcsh shell environment.

5.4.3.10 Software development and testing support tools

The ROC developer team shall use the virtual environment and the package mechanisms to develop and test the software written in Python 3, namely: RODP, ROC-SGSE and MUsiC GUIs. These mechanisms are flexible, robust and widely used in the developer community.

Unit tests shall also be implemented in the RODP to validate critical functions. The list of unit tests shall be given in the SVP.

Any specific procedure and/or convention related to the development and testing support tools shall be reported into the REG.

5.5 Software documentation plan

5.5.1 General

The software documentation plan presents the management of the ROC engineering documentation. The ROC GSE documentation is not covered in this plan.

5.5.2 Software documentation identification

The software documentation identification shall follow the convention defined in the PMP.

5.5.3 Deliverable items

5.5.3.1 ROC deliverable documentation

The main documentation concerning the ROC software is given in the PMP.

In addition, the ROC developer team shall plan to deliver the following documents:

- ROC Software System User Manual (RSSUM)
- ROC Software System Reference Guide (RSSRG)
- RPW LLVM User Manual (RLUM)

5.5.3.2 RCS deliverable documentation

Consult the section 4.7.1.5 for more details.

5.5.4 Software documentation standards

The software documentation shall follow the standards specified in the PMP.

Any additional procedures and rules concerning the software documentation shall be listed into the REG.

6 ROC SOFTWARE DELIVERY PLAN

This section presents all of the S/W products to be delivered to the ROC and by whom in preparation of the RPW ground segment activities. The following schedule shall always be consistent with the planning defined at project level in the PMP.

6.1 ROC software development planning

The ROC software development planning shall be detailed in the *ROC planning file* as defined in the PMP. The next sections give the main schedule.

6.2 ROC software delivery main schedule

Table below gives the schedule of the main software products to be developed and delivered by the ROC team.

Software deliverable	Main available capabilities	Software validation leader	Developer(s) in charge	Delivery date
RPL V0.1.0	Parsing of the RPW TM/TC packet data encapsulated in the MEB SGSE test log. Compatible with the	Xavier Bonnin (LESIA)	- Thierry Sauzière (AVISTO)	04/07/2014

	RPW IDB V2.2.3			
RPL V0.2.0	Ready for RPW packet parsing during EM2/PFM calibration campaigns. Implemented into the ROC-SGSE using XML (PALISADE) version of the IDB	Xavier Bonnin (LESIA)	- Manuel Duarte (VIVERIS) - Xavier Bonnin (LESIA)	31/12/2015
RPL V1.0.0	Ready for RPW packet parsing during PFM calibration campaigns. Implemented into the ROC-SGSE using XML (PALISADE) version of the IDB	Xavier Bonnin (LESIA)	- Sonny Lion (LESIA)	31/12/2016
TRaC V0.1.0	Demo version of the TraC. Only valid for THR and LFR. Compatible with the V2.2.3 IDB	Xavier Bonnin (LESIA)	Thierry Sauzière (AVISTO)	04/07/2014
ROC-SGSE V0.1.0	Version to be used for the EM2 blank calibration campaigns at CNES.	Xavier Bonnin (LESIA)	- Manuel Duarte (VIVERIS), - Xavier Bonnin (LESIA)	15/03/2016
ROC-SGSE V1.0.0	Version to be used for the PFM thermal calibration campaigns at LESIA.	Xavier Bonnin (LESIA)	- Manuel Duarte (VIVERIS), - Xavier Bonnin (LESIA)	30/10/2016
RPW user libraries V0.1.0	First demo version (CDF parser)	Xavier Bonnin (LESIA)	- Quynh Nhu (LESIA), - Xavier Bonnin (LESIA),	30/04/2015
RPW user libraries V1.0.0	First stable release.	Xavier Bonnin (LESIA)	- Quynh Nhu (LESIA), - Xavier Bonnin (LESIA)	31/12/2017
RPW Low Latency Virtual Machine (LLVM) V0.1.0	First "Hello World" VM configuration and test version.	Richard Carr (ESAC/SOC)	- Xavier Bonnin (LESIA)	01/02/2016
RPW Low Latency Virtual Machine (LLVM) V0.X.0	Second "processing data" version.	Richard Carr (ESAC/SOC)	- Xavier Bonnin (LESIA)	30/06/2016
RPW Low Latency Virtual Machine	Full version. - Fully compliant	Richard Carr (ESAC/SOC)	- Xavier Bonnin (LESIA)	31/08/2016

(LLVM) V1.0.0	and validated.			
ROC Operations and Data Pipeline (RODP) V0.X.0	First not validated version	Xavier Bonnin (LESIA)	- Sonny Lion (LESIA) - Xavier Bonnin (LESIA)	31/12/2017
ROC Operations and Data Pipeline (RODP) V1.0.0	Full validated version, ready for the mission - Full process of the RPW data products – Full ROT capabilities - Full RMU capabilities - SOC/MOC interfaces up and running	Xavier Bonnin (LESIA)	- Quynh Nhu Nguyen (LESIA) - Xavier Bonnin (LESIA) - Sonny Lion (LESIA)	31/03/2018
MCS Users Interfaces (MUsIcs) V0.X.0	Preliminary demo version	Xavier Bonnin (LESIA)	- Quynh Nhu Nguyen (LESIA) - Developer #1	30/06/2017
MCS Users Interfaces (MUsIcs) V0.X.0	First not validated version	Xavier Bonnin (LESIA)	- Quynh Nhu Nguyen (LESIA) - Developer #1	31/12/2017
MCS Users Interfaces (MUsIcs) V1.0.0	Full stable version. Ready for the mission.	Xavier Bonnin (LESIA)	- Quynh Nhu Nguyen (LESIA) - CDD IE	31/03/2018
SimuSBM1 processing V1.0.0	Version ready for SBM1 detection algorithm validation on-ground	Xavier Bonnin (LESIA)	- Xavier Bonnin (LESIA) - Oksana Kruparova (LESIA)	30/03/2015
SimuSBM1 Processing V2.0.0	Second version. Ready to be used with real RPW data produced on-board.	TBD	TBD	30/06/2018 (TBC)
SimuSBM2 Processing V1.0.0	Version ready for SBM2 detection algorithm validation on-ground	Xavier Bonnin (LESIA)	- Xavier Bonnin (LESIA) - Milan Maksimovic (LESIA) - Olga Alexandrova (LESIA)	30/06/2015
SimuSBM2 Processing V2.0.0	Second version. Ready to be used with real RPW data produced on-board.	TBD	TBD	30/06/2018 (TBC)

Table 20. ROC software main releases.

Note that software shall be always released with the required documentation.

6.3 RPW calibration software delivery schedule

The following table provides the planning of the main releases of the RCS to be delivered by the RPW analyser/sensor teams.

Team	Software name	Software function	Version	Delivery date
THR	THR CALibration softwARe SGSE (THR CALBAR-SGSE)	Produce THR calibrated data during ground calibration campaigns	1.0.0	31/10/2016
THR	THR CALibration softwARe (THR CALBAR)	Produces L2 electrical spectral data files for THR during the Solo mission	TBD	31/10/2017 (TBC)
LFR	LFR CALiBration UnitT SGSE (LFR CALBUT-SGSE)	Produces L2s HF electric component and L2r data files for LFR during the on-ground tests	1.0.0	31/10/2016
LFR	LFR CALiBration UnitT (LFR CALBUT)	L2 data files for LFR during the Solo mission	TBD	31/10/2017 (TBC)
TDS	TDS CALibration SoftwARe SGSE (TDS CALBAR-SGSE)	Produces L2s HF electric component and L2s data files for TDS during on-ground tests	1.0.0	31/10/2016
TDS	TDS CALibration SoftwARe (TDS CALBAR)	Produces L2s HF electric component and L2r data files for TDS during SolO mission	TBD	31/10/2017 (TBC)
BIAS	IRFU_MATLAB SGSE	Produces L2s data files involving BIAS during the ground tests	1.0.0	31/10/2016
BIAS	IRFU_MATLAB	Produces L2 data files involving BIAS during the SolO mission	TBD	31/10/2017 (TBC)
SCM	SCML2RL2S SGSE	Produces L2 waveform data files involving SCM during the on-ground tests	1.0.0	31/10/2016
SCM	SCML2RL2S SGSE	Produces L2 waveform data files involving SCM during the SolO mission	TBD	31/10/2017 (TBC)

Table 21. Calibration software main releases.

7 APPENDIX

7.1 Summary list of ROC software products

7.1.1 ROC software products

The following table provides the list of S/W and interfaces that shall be delivered to the ROC.

System	Name	Function	Main language	S/W Provider	S/W Maintainer	S/W user
DPS/MCS	ROC Operations and Data Pipeline (RODP) – core	Pipeline based on POPPy framework and used to retrieve, process data and prepare operations. In	Python 3, Cython,	ROC	ROC	ROC

		practice, it regroups the "backend" modules of the DPS and MCS.				
DPS	RPW Packet parsing Library (RPL)	- Decommutation/decompression of the RPW TM/TC packet data. Implemented as a plugin of the RODP.	Python 3, Cython,	ROC	ROC	ROC
DPS	Data Requester (DaRe)	Software unit in charge of retrieving data from the SOC/MOC DDS interface. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
DPS	File Maker (FilM)	Software unit in charge of producing L0/L1/HK data files. Implemented as a plugin of the RODP.	Python 3, Cython	ROC	ROC	ROC
DPS	Data ingestor (Dingo)	Software unit in charge of ingesting data into the MDB database. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
DPS	Calibration Wrapper (CaWa)	Software unit in charge of running the RCS. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
DPS	RPW Low Latency Pipeline (RLLP)	Software unit in charge of producing RPW LL data files. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
DPS	RPW Calibration S/W for the LFR receiver (LFR_CALBUT)	- Produce CDF format files containing science calibrated data for the LFR receiver. Run by the CaWa module of the RODP.	Python	LPP	LPP	ROC
DPS	RPW Calibration S/W for the TDS receiver (TDS_CALBAR)	- Produce CDF format files containing science calibrated data for the TDS receiver. Run by the CaWa module of the RODP.	IDL	IAP	IAP	ROC
DPS	RPW Calibration S/W for the TNR-HFR receiver (THR_CALBAR)	- Produce CDF format files containing science calibrated data for the TNR-HFR receiver. Run by the CaWa module of the RODP.	IDL	LESIA	LESIA	ROC
DPS	RPW Calibration S/W for the BIAS unit (IRFU-MATLAB)	- Produce CDF format files containing science data derived from the BIAS unit parameters (e.g., plasma density) . Run by the CaWa module of the RODP.	Matlab	IRF	IRF	ROC
DPS	RPW Calibration S/W for the SCM sensors	- Produce CDF format files containing science calibrated data as measured by the SCM sensors. Run by the CaWa	IDL	LPC2E	LPC2E	ROC

	(SCML2RL2S)	module of the RODP.				
DPS	RPW Low Latency Virtual Machine (LLVM)	Virtual Machine that contains a stateless/lighter version of the RODP dedicated to the data processing of the RPW Low Latency data	Python 3, Cython	ROC	ROC	ROC/SOC
DPS	RPW Data Archive (DARC)	Software units of the DPS in charge of the RPW data archiving	TBD	ROC	ROC	ROC
DPS	RPW Data Access Layer (DAL)	Software units of the DPS in charge of the external RPW data access interfaces at LESIA	TBD	ROC	ROC	Science community
MCS	Instrument Monitoring and Analysis Unit (MonA)	Software unit in charge of the RPW automated part of the monitoring and analysis. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
MCS	Instrument Power Consumption Analyser (PoCA)	Software unit in charge of the RPW power consumption analysis. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
MCS	Instrument TM Rate Calculator (TRaC)	Software unit in charge of the RPW TM rate computation. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
MCS	Operation File Requester (FiRe)	Software unit in charge of preparing the instrument operation requests. Implemented as a plugin of the RODP.	Python 3	ROC	ROC	ROC
MCS	RPW TM/TC Viewer (TV)	Main GUI for the instrument monitoring. An interface of the MUsIC	Python 3	ROC	ROC	ROC, RPW flight software teams
MCS	RPW Flight Procedure Editor (Figaro)	Main GUI to create/edit flight procedures for RPW. An interface of the MUsIC				
MCS	RPW Flight Operation Request Editor (Faust)	Main GUI to create/edit RPW Operating Modes and IORs. An interface of the MUsIC.	Python 3	ROC	ROC	ROC
MCS	RPW Operation Planning Interface (OPera)	Main GUI to visualize and plan the RPW observation campaign. An interface of the MUsIC	Python 3, Django framewor k	ROC	ROC	ROC
MCS	SBM Interactive Selection Software Interface (SISSI)	Main GUI to manage SBM event data selection. An interface of the MUsIC.	Python 3, Django framewor k	ROC	ROC	ROC / TBD
ROC GSE	ROC-SGSE	SGSE to process RPW packet data generated during the ground calibration tests.	Python 3, Cython, POPPy framewor k	ROC	ROC	ROC / CNES
ROC GSE	Test Viewer SGSE (TV-	- Display RPW data produced on-ground during	Python 3, POPPy	ROC	ROC	ROC / CNES

	SGSE)	calibration tests at the system level.	framewor			
ROC GSE	SimuSBM1	Simulate SBM1 event detection	IDL	ROC	ROC	ROC
ROC GSE	SimuSBM2	Simulate SBM2 event detection	IDL	ROC	ROC	ROC
ROC GSE	RPW User Libraries (RUL)	Provide routines in Python and IDL languages that allow to retrieve read, plot, etc., RPW data	IDL / Python 3	ROC	ROC	ROC/Community
N/A	POPPy framework	Python framework to build plugin-oriented pipeline	Python 3	ROC	ROC	ROC

Table 22. ROC software products.

7.1.2 ROC Database products

The following table provides the list of databases that shall be used by the ROC.

Name	Description	S/W Provider	S/W Maintainer
RPW Instrument Database (RPW-IDB)	Contains RPW TM/TC packet structures.	RPW Flight Software team (LESIA)	RPW Flight Software team (LESIA)
ROC Instrument Database (ROC-IDB)	RDBMS copy of the RPW IDB. Used by the ROC pipelines to analyze RPW packet data.	ROC	ROC
ROC-SGSE Test Database (ROC-TDB)	Contains information required to run the ROC Ground Tests SGSE (ROC GT-SGSE)	ROC	ROC
ROC Mission Database (ROC-MDB)	Contains information related to the ROC Software system use.	ROC	ROC
SolO mission information Database (MIB)	Local copy of the MIB as sent by the MOC.	MOC	MOC
ROC MEB GSE mission database	Instance of the MEB GSE database for the mission.	MEB GSE team (LESIA)	MEB GSE team (LESIA)

Table 23. ROC database products.

8 List of TBC/TBD/TBWs

TBC/TBD/TBW			
Reference/Page/Location	Description	Type	Status

9 DISTRIBUTION LIST

LISTS See Contents lists in "Baghera Web": Project's informations / Project's actors / RPW_actors.xls and tab with the name of the list or NAMES below		Tech_LESIA
		Tech_MEB
		Tech_RPW
		[Lead-]Cols
		Science-Cols

INTERNAL

LESIA CNRS	x	M. MAKSIMOVIC	LESIA CNRS		
	x	Y. DE CONCHY			
	x	X. BONNIN			
	x	QN NGUYEN			
	x	B. CECCONI			
	x	A. VECCHIO			

EXTERNAL (To modify if necessary)

CNES		C. FIACHETTI	AsI/CSRC		J.BRINEK
		C. LAFFAYE			P.HELLINGER
		R.LLORCA-CEJUDO			D.HERCIK
		E.LOURME			P.TRAVNICEK
		M-O. MARCHE	IAP		J.BASE
		E.GUILHEM			J. CHUM
		J.PANH			I. KOLMASOVA
		B.PONTET			O.SANTOLIK
				x	J. SOUCEK
				x	D. PISA
IRFU		L. BYLANDER	IWF		G.LAKY
		C.CULLY			T.OSWALD
		A.ERIKSSON			H. OTTACHER
	x	E.JOHANSSON			H. RUCKER
	x	A.VAIVADS			M.SAMPL
	x	Y. KHOTYAINTSEV			M. STELLER
LPC2E		P. FERGEAU	LPP	x	T.CHUST
	x	G. JANNET			A. JEANDET
		T.DUDOK de WIT			P.LEROY
	x	M. KRETZSCHMAR			M.MORLOT
		V. KRASNOSSELSKIKH		x	B. KATRA
SSL		S.BALE			