Water Vapour Climate Change Initiative (WV_cci) - CCI+ Phase 2

System Specification Document (SSD)

Ref: D3.2

Date: 30 May 2023

Issue: 4.0

For: ESA / ECSAT

Ref: CCIWV.REP.013

This Page is Intentionally Blank

Project	**:**	**Water Vapour Climate Change Initiative (WV_cci) - CCI+ Phase 2**
Document Title:		**System Specification Document (SSD)**
Reference	**:**	**D3.2**
Issued	**:**	**30 May 2023**
Issue	**:**	**4.0**
Client	**:**	**ESA / ECSAT**
Authors	**:**	Olaf Danne (Brockmann Consult), Michaela Hegglin and Hao Ye (University of Reading)
Copyright	**:**	Water_Vapour_cci Consortium and ESA

Document Change Log

Issue/ Revision	Date	Comment
0.1	In Progress	Initial issue
0.9	3 June 2019	First version for internal review
1.0	7 June 2019	First submitted version
1.1	31 August 2019	Revision based on v1.0 RIDs
2.0	31 July 2020	Scheduled document update
2.1	1 October 2020	Revision based on v2.0 RIDs
3.0	19 August 2021	Scheduled document update
4.0	30 May 2023	Scheduled document update

TABLE OF CONTENTS

INDEX OF TABLES

INDEX OF FIGURES

This Page is Intentionally Blank

1. INTRODUCTION

1.1 Purpose

The ESA Climate Change Initiative (ESA CCI) stresses the importance of providing a higher scientific benefit from data acquired by European sensors, especially in the context of the IPCC reports. This implies producing consistent time series of accurate Essential Climate Variables (ECV) products [1] that can be used by climate science and climate service users. Long-term observations and the international links with other agencies currently generating ECV data are key to this activity.

The Water Vapour ECV includes Total Column of Water Vapour (TCWV) and Vertically Resolved profiles of tropospheric and lower stratospheric Water Vapour (VRWV) as primary variables. For TCWV retrievals, the project includes algorithms for the processing of MERIS, MODIS Terra, Sentinel-3 OLCI, and CM SAF HOAPS TCWV data. This combination is applied mainly for the generation of merged products over land (CDR-1), but also with global coverage, i.e., combining the retrievals over land and water (CDR-2) and investigating possible temporal, regional or systematic issues in order to improve the algorithms applied for the various sensors. For VRWV, the project includes the descriptions of two merging algorithms. The first one describes the merging between the satellite limb sounders for SAGE II, HALOE, UARS-MLS, POAM III, SAGE III, SMR, SCIAMACHY, MIPAS, Aura MLS, ACE-FTS, ACE-MAESTRO, and SAGE III/ISS for a stratospheric zonal mean climate data record (CDR-3). Here, corrections for handling spatio-temporal sampling differences and biases will be a focus. The second one refers to the three-dimensional prototype version of the upper tropospheric/lower stratospheric (UTLS) climate data record (CDR-4), for which the input data consist of observations from the satellite limb sounders MIPAS, Aura-MLS, and a combined retrieval product (IMS) from the IASI/MHS/AMSU satellite instruments. Here, the focus is primarily placed on how to best combine observations from nadir and limb sounders across the tropopause region despite their strongly differing viewing geometry and large measurements uncertainties in this region of the atmosphere.

For all the input datasets, the data must be acquired, algorithms and processing workflows must be defined and implemented, and the data processing must be performed.

1.2 Scope

This document describes the system and its elements which were developed within the ESA WV_cci during the first 3-year phase (‘Phase 1’) of the project. In addition, updates which have been applied during the first year of the second 3-year period (‘Phase 2’) of the project are also described.

This SSD exists in the context of other WV_cci documents: The WV_cci PSD [2] describes the content and format of the WV products to be generated, the DARD [3] in Phase 1 describes the satellite and ancillary data required for the generation of the data products, and the ATBD [4] defines the algorithms which are and will be used for WV processing. This SSD describes the design of the system that generates the WV products, including the integration of the processors that implement the algorithms defined in the ATBD, the handling of the input datasets, and the control of the production workflow for the WV_cci processing chains. The main concepts of the elements used by WV_cci and the WV_cci-specific configuration and extension are also in the scope of this document.

This document covers the MERIS, MODIS, OLCI, and CM SAF HOAPS TCWV workflows for the TCWV generation (CDR-1 and CDR-2), as well as the limb sounders (SAGE II, HALOE, UARS-MLS, POAM III, SAGE III, SMR, SCIAMACHY, MIPAS, Aura MLS, ACE-FTS, ACE-MAESTRO, and SAGE III/ISS) and the IMS dataset (as based on the different nadir-viewing instruments IASI/MHS/AMSU) workflows for the generation of the VRWV datasets (CDR-3 and CDR-4).

2. SYSTEM OVERVIEW

This section provides the overview on the TCWV and VRWV processing systems, as sketched out in Figure 2-1. This setup has been established in Phase 1 and is basically still valid in Phase 2. Updates and modifications will be outlined in more detail in the following sections.

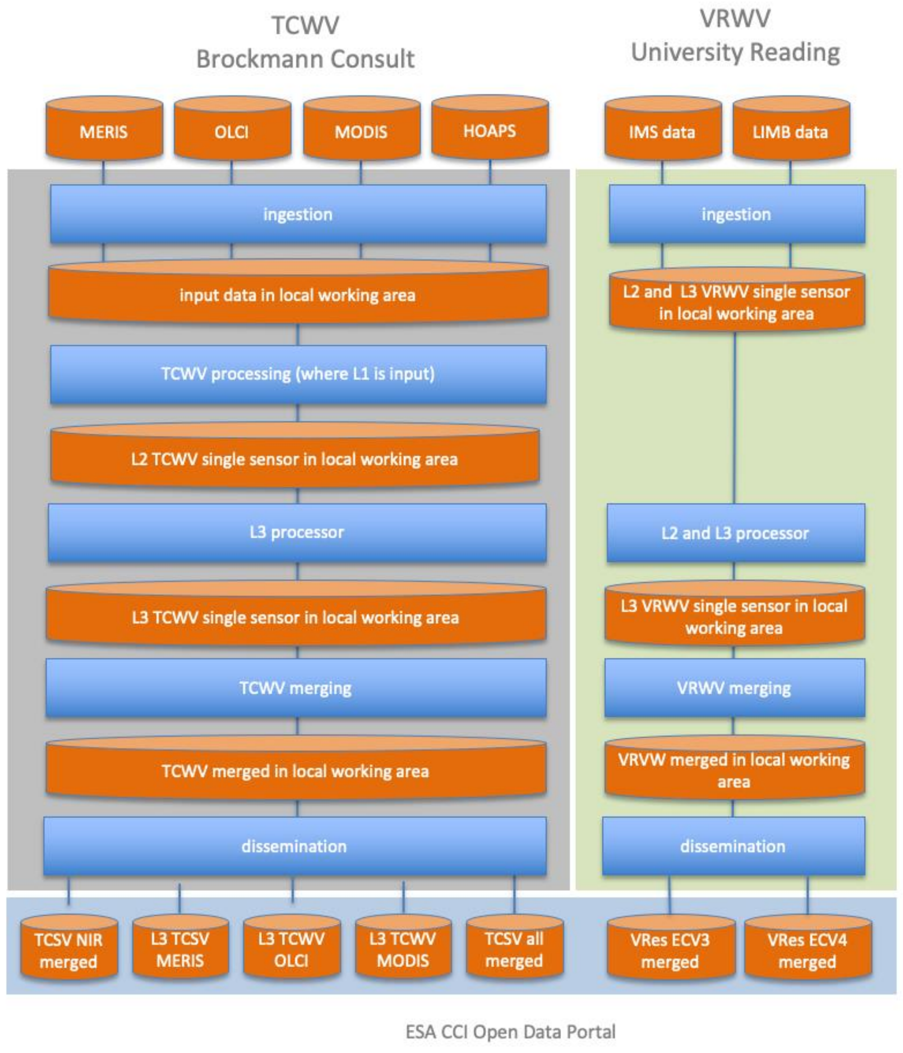

TCWV
Brockmann Consult
VRWV
University Reading
MERIS
OLCI
MODIS
HOAPS
IMS data
LIMB data
ingestion
ingestion
input data in local working area
L2 and L3 VRWV single sensor in local working area
TCWV processing (where L1 is input)
L2 TCWV single sensor in local working area
L3 processor
L2 and L3 processor
L3 TCWV single sensor in local working area
L3 VRWV single sensor in local working area
TCWV merging
VRWV merging
TCWV merged in local working area
VRVW merged in local working area
dissemination
dissemination
TCSV NIR merged
L3 TCSV MERIS
L3 TCWV OLCI
L3 TCWV MODIS
TCSV all merged
VRes ECV3 merged
VRes ECV4 merged
ESA CCI Open Data Portal

Figure 2-1: Symmetric system definition of the TCWV and VRWV processing systems. Updated from [5].

2.1 WV_cci System Context

The WV_cci system interacts with various other entities as illustrated in Figure 2-2. In principle, this system context applies for both WV_cci subsystems for TCWV and VRWV generation (Figure 2-1). Also, this setup is quite similar to other CCIs such as Fire_cci or Landcover_cci, as described in [6] and [7].

The entities of the WV_cci system context are:

- The WV_cci team develops data processors and produces all WV datasets. On the other hand, parts of the team are also responsible for validation, so there is a back-and-forth interaction within the team containing feedback and potentially requests for specific production requests (bug fixing, algorithm improvements). Validated WV datasets will then be provided externally (see below). The processing system for TCWV and VRWV will not be externally distributed, however, the underlying software is free and open source following ESA CCI policy.
- Data providers of the required satellite input data (ESA, NASA, CSA).
- The CCI Open Data Portal which has been fed with validated WV datasets during Phase 1. Extended datasets will follow in Phase 2. In return, the portal serves as platform for users to access these datasets.
- The users: they can access the WV datasets from the CCI Open Data Portal. On request, users may also receive datasets directly from the WV_cci team for validation or other purposes.
- Other ECV projects: they also may receive validated WV datasets on request. In return, they might serve as a resource for ancillary datasets which could be of use to improve the WV_cci system.
- Shared infrastructure used by the WV_cci system. Examples are the JASMIN and the Calvalus clusters being used for TCWV generation, and the RACC and JASMIN clusters for the VRWV generation.

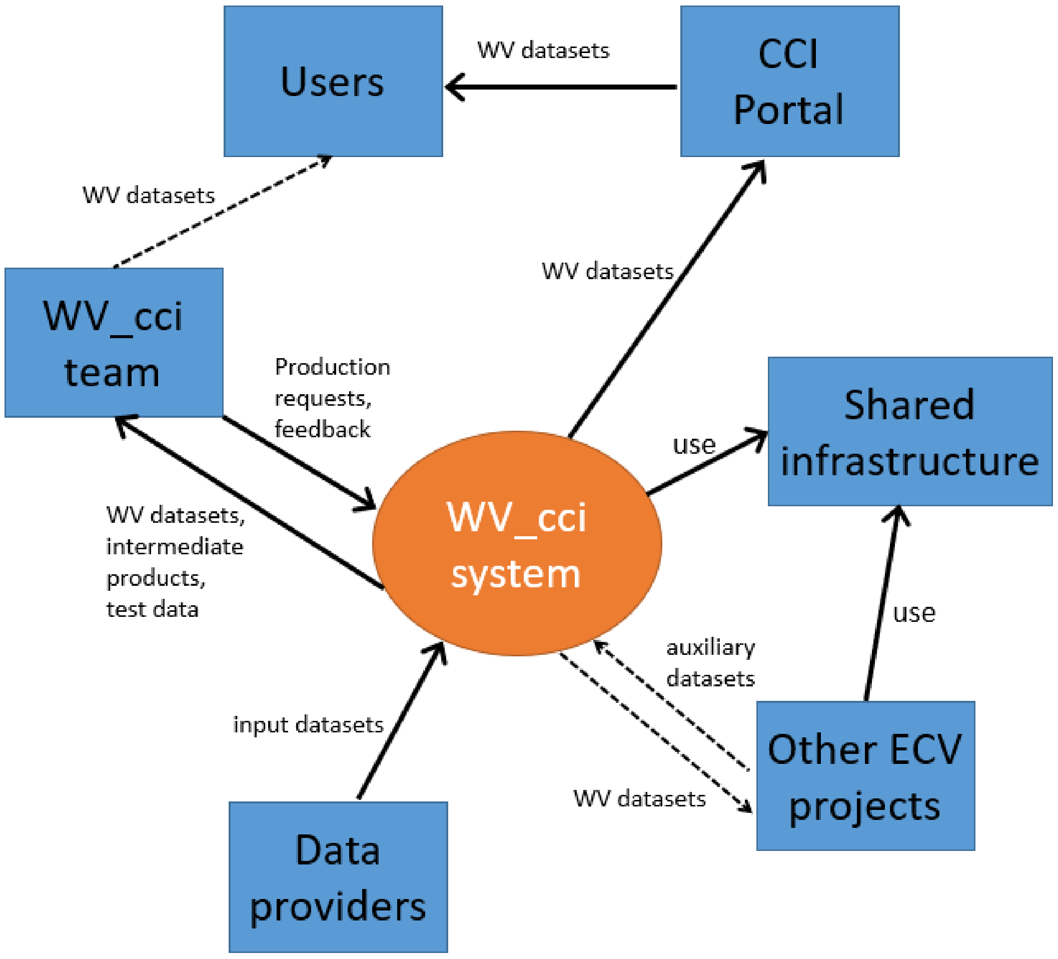


Figure 2-2: System context of the WV_cci system with team, data providers, and other entities.

2.2 Main Function and Processing Chain

Regarding TCWV, one of the main goals of Phase 1 of WV_cci was the merging of the TCWV data of the available sensors to fill all data gaps in global products as far as possible, and to identify regional, temporal, or systematic differences and discontinuities when bringing the data together. This resulted in an improvement of the underlying algorithms, the elimination of existing problems, and finally the production of a high quality global TCWV time series from 2002 to 2017.

For Phase 2, the main objectives are the improvement of the quality of the Phase 1 data records (understand and remove shifts in the long term TCWV time series, retrievals over coastal and inland waters), the spatial and temporal extension of the data records (including regional, high-resolution datasets for selected areas), and finally the update and extension of input data (MERIS 4th reprocessing, Sentinel 3B OLCI, MODIS Aqua, HOAPS version 5).

Regarding VRWV, a main goal for CDR-3 in Phase 1 was the merging of a well characterised and homogeneous long-term zonal mean VRWV CDR in the stratosphere from data obtained from available limb satellite sounders (SAGE II, HALOE, UARS-MLS, POAM III, SAGE III, SMR, SCIAMACHY, MIPAS, Aura MLS, ACE-FTS, ACE-MAESTRO, and SAGE III/ISS) covering the time period 1985–2019. A key aspect in the construction of such a CDR is the requirement for a homogenisation procedure to account for systematic measurement biases, which can vary in time and are often altitude- and latitude-dependent. For VRWV CDR-4, the main goal of WV_cci was to merge the VRWV data from both limb-sounders and nadir instruments into a harmonised prototype VRWV data record from 2010 to 2014. This study improved our knowledge of the consistency between VRWV products from limb-sounders and nadir instruments and provided insight in the underlying merging algorithms between these two kinds of satellite products.

For Phase 2, the primary objectives for CDR-3 are to improve the quality of the Phase 1 data records in terms of self-consistency and homogeneity, and the extension forward in time to 2023. In addition, the data record will potentially also be extended backwards in time to 1978, depending on the assessment of the historical satellite data records that this project envisages to rescue from old data archives. The main objectives for CDR-4 are to extend the latest data record to a longer time scale (2007-present) with the current merging algorithm developed in Phase 1, and to investigate and test alternative merging techniques with new and more reference profiles.

2.3 System Requirements

System requirements are usually derived from use cases, user requirements, and other inputs based on a first decomposition of the system into elements. In the CCI projects, the focus is mainly on the products to be generated, rather than on the system, which is only the means to produce the product but not a deliverable itself. Therefore, the main requirement for the system is to generate the product according to the PSD. The relationship among the different requirements is shown in Figure 2-3.

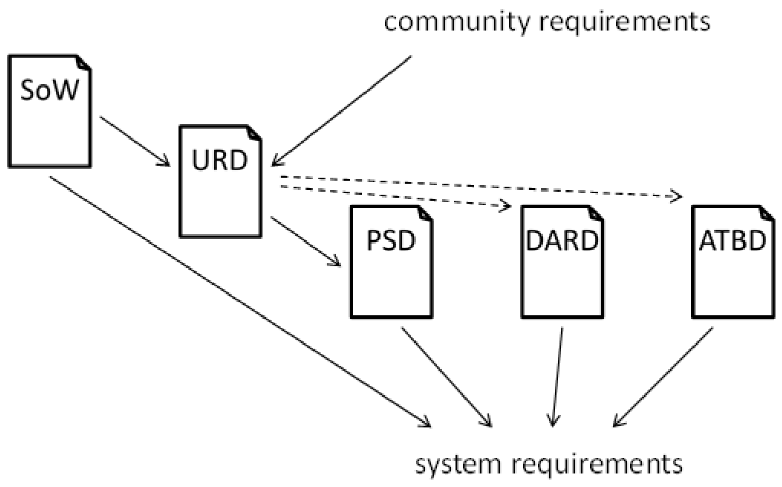


Figure 2-3: Requirements derivation logic as usual in CCI projects with system requirements as specified in project documentation. Taken from [6].

The Statement of Work (SoW) [8] also lists various requirements, a few of them are within system scope. The WV products shall further be compliant with the NetCDF-CF metadata conventions [9] and the CCI Data Standards [10], [11], which leads to another set of system requirements.

All the system requirements related to Phase 1 of the project are explicitly listed and discussed in more detail in the SRD [12]. They are still valid for Phase 2, updates and modifications will be applied as outlined in the Technical Proposal for Phase 2 [13].

2.4 WV_cci System Architecture

2.4.1 TCWV processing system

From the system requirement [TR-30] in the WV_cci SoW [8] and the corresponding cardinal requirements in the project technical proposal [5] it follows that the TCWV processing system shall ingest and handle L1b input data from the MERIS, MODIS and OLCI instruments, complemented by L3 input data from CM SAF HOAPS. All this can easily be fulfilled by the Calvalus system at BC (described in more detail below), except for the MODIS input data sets (Terra: MOD021KM, Aqua: MYD021KM). For these, Calvalus would be able to handle that data, but not to ingest the data with reasonable effort due to limited bandwidth and the fact that NASA does not send the data on physical devices. This is exactly the same sort of problem as in the CCI Fire project (also with BC responsible for data processing) where MERIS and MODIS MOD09 products had to be used as inputs. As a consequence, some alternative high-performance computer clusters were evaluated in CCI Fire for the production using the MODIS inputs (see [6] for more details). Their considered high-performance cluster solutions were the cluster at IT4Innovations [14], the CEDA JASMIN system [15], and Google Earth Engine [16]. From this evaluation, their final decision was to use Calvalus for MERIS and to go with the JASMIN system for MODIS. For WV_cci, the same criteria had to be considered: Although Google Earth Engine boasts a very high performance for free (as WV_cci is a scientific project), they do not have the MOD02/MYD02 datasets available, so the same data transfer problems would apply as for Calvalus. Although IT4Innovations offered overall good conditions (download speed as well as sufficient storage and performance), the fact that BC WV_cci team members as well as other BC staff members involved in CCI projects have experiences with JASMIN and know that it is reliable and simple to use (although not free of charge). Therefore, the decision for WV_cci was made to use JASMIN for MODIS TCWV L2 processing, and Calvalus for everything else, including OLCI L2 processing, MODIS L3 processing and the generation of merged products from all sensors. This requires the transfer of the MODIS TCWV L2 products to Calvalus, but this can be done with reasonable time effort, as the TCWV products are much smaller than the corresponding L1b input products. The only real drawback of the JASMIN system is that the storage space is not free and is limited to 2TB for WV_cci. This means that the processing needs to be done in cycles (e.g. in subsets of a couple of months), which is acceptable.

This approach to share the processing tasks between Calvalus and JASMIN clusters has proven itself during Phase 1 and is being continued in Phase 2.

2.4.1.1 Calvalus system architecture

The WV_cci system is to a large extent based on Calvalus and the underlying infrastructure, with certain elements specifically configured and extended for WV_cci (Figure 2-4). In addition to the software stack with many functions of the WV_cci system provided by Calvalus, Hadoop or other frameworks, there is a shared cluster infrastructure that runs the corresponding services.

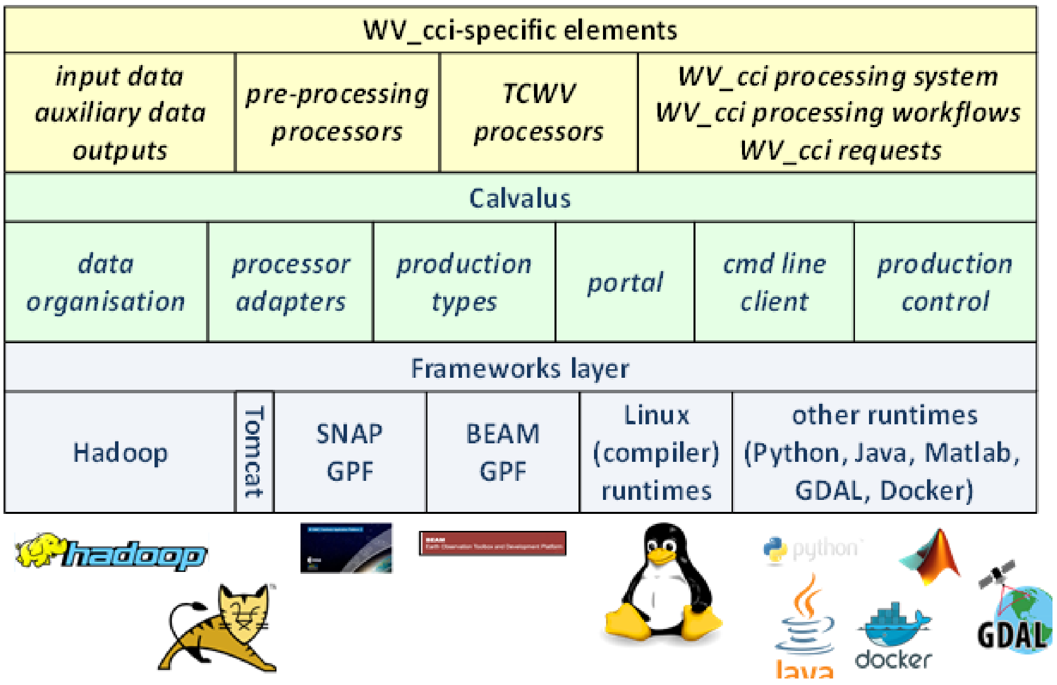


Figure 2-4: WV_cci system architecture layers with specific elements, Calvalus, and Frameworks layer.

The main elements that are specific to WV_cci are:

- Access to the shared input data (MERIS and OLCI RR), space for ancillary data and WV_cci project intermediates and outputs.
- Processors or WV_cci-specific processor configurations of existing processors for preprocessing.
- The TCWV processors and their wrappers for integration into Calvalus.
- The WV_cci processing system instance on a virtual machine for the control of all WV_cci workflows, with rules and request templates for preprocessing, TCWV processing, and formatting and all the respective dependencies.

The main shared elements and functions used from Calvalus and underlying frameworks are:

- The Calvalus data organisation for earth observation (EO) data, ancillary data, and project-specific data, and the corresponding functions of the Hadoop Distributed File System (HDFS) for data management.
- The Calvalus production control tool PMonitor for the control of processing chains and bulk production, in combination with the Calvalus production types for massive-parallel processing, and Hadoop HDFS and YARN for fair scheduling, load

balancing, and robust failure handling for different layers of production management.

- The Calvalus processor adapters to wrap the WV_cci processors, in particular SNAP for the preprocessing (e.g. cloud screening), and the corresponding automated deployment and runtime provisioning for processor integration.
- The Calvalus framework for MapReduce, which employs the Hadoop MapReduce framework, and provides the basis for generic MapReduce algorithms. This is used for the final formatting step.

Section 3.1.2 describes in more detail how these elements and functions are used.

2.4.1.2 JASMIN system architecture

JASMIN is a globally unique data analysis facility. It provides storage and compute facilities to enable data-intensive environmental science. Centred more around storage and data analysis than a “traditional” supercomputer, JASMIN provides flexibility for a wide range of data-intensive analysis workflows. Tens of petabytes of storage are combined with several types of compute resource: managed interactive and batch compute for building and executing large workflows (which is done in WV_cci), and a “community cloud” offering projects and communities a set of service components with which to build and manage their own computing needs. Along with these services, high-performance access to massive data resources, including the curated archives of the Centre for Environmental Data Analysis (CEDA) [17], is provided.

JASMIN is designed, integrated, and operated by the Science & Technology Facilities Council (STFC) [18] on behalf of the Natural Environment Research Council (NERC) [19]. It is architected jointly between STFC’s Scientific Computing Department and CEDA within RAL Space [20]. Since JASMIN came into being in early 2012, it has grown significantly and continuously in scale and complexity, as described in great detail in [21].

The WV_cci project uses a dedicated group workspace within JASMIN. Group workspaces are portions of storage allocated for particular projects to manage themselves, enabling collaborating scientists to share network accessible disks. Users can pull data from external sites to a common cache, process and analyse their data, and where allowed, exploit data available from other group workspaces and from the CEDA archive. Besides WV_cci, other CCI projects such as SST and Ocean Colour CCI project have acquired dedicated group workspaces for disseminating their CDR.

Group Workspaces are usually exploited in conjunction with project specific computing resources, configured and deployed as virtual machines in the JASMIN infrastructure. Such machines can generally mount their own GWS and the CEDA archive. It is important to understand that the project workspaces are not the same as the CEDA archive. Data in a group workspace can be earmarked for ingestion into the CEDA archive, but this is a process that should be discussed directly with CEDA, it is not automatic in any way.

Data within group workspaces are under the responsibility of the designated group workspace manager and are not backed up by CEDA. The manager of the WV_cci workspace is a member of the Development Team.

By deciding to migrate the WV_cci system to the JASMIN/CEDA infrastructure the project has gained the immediate benefit that the required large MODIS MOD021KM/ MYD021KM input datasets are directly available from the NEODC database and accessible from the project group workspace. This circumvents any possible issues with downloading MODIS data from external resources, and moreover, these MODIS input data will not occupy any storage of the project workspace.

The concept of group workspaces as well as of further JASMIN services and service groups is described in detail in [22].

Over the past years, the JASMIN online help has grown significantly along with the system. The starting point is the JASMIN help documentation home page [23], which links to a variety of further articles around the JASMIN system and how to use it.

2.4.2 VRWV processing system

According to the system requirement [TR-30] in the SoW [8] and the Cardinal Requirement [CR-2] in the project technical proposal [5], two VRWV products (CDR-3 and CDR-4) should be produced. To this end, the stratospheric VRWV product (CDR-3) should be merged from multiple limb-sounder instruments as harmonised ensemble or as a merged product, and be based on the SPARC Data Initiative L3 data products from SAGE II, HALOE, UARS-MLS, POAM III, SAGE III, SMR, SCIAMACHY, MIPAS, Aura MLS, ACE-FTS, ACE-MAESTRO, and SAGE III/ISS. The volume of these L3 datasets is small and in principle could be easily ingested and handled by a personal computing system such as a Desktop computer or laptop, although there are advantages of using a University supported system as the RACC, i.e. Reading

Academic Computing Cluster (described below), which offers scheduled and monitored backups of work and storage directories.

For the VRWV CDR-4, the processing system should ingest and handle L2 input data from high-quality nadir and limb sounding retrievals, here including Aura MLS, MIPAS, and IMS instruments. The Aura MLS data can be downloaded from the NASA JPL website directly, and the MIPAS data is available from the WAVAS_SAHAR distribution. For these two input datasets, the UoR RACC system would be able to handle both L2 and L3 data processing with relatively small volumes of input and intermediate data. However, due to the large volume of IMS input data (~ 6 TB) for the final, longer timeseries, the data transfer and storage poses challenges for the processing on the UoR RACC system. Therefore, the high-performance CEDA JASMIN system is chosen for IMS VRWV L2 and L3 processing. The advantage of the JASMIN system is that it has direct access to the IMS data, which are stored on CEDA, thus avoiding transfer and storage problems. This only requires the transfer of auxiliary Derived Meteorological Products (DMP) data for the IMS form the RACC to the JASMIN system and also the transfer of L3 data back to the RACC, which is now feasible given the much smaller volume of the pre-processed data. The generation of the final merged product CDR-4 is done on the RACC system at UoR in a final step.

The RACC system, is deployed at the University of Reading (UoR) and provides essential data processing environment and resources to UoR staffs and students. Most of the cluster resources are available for free for UoR members and students for limited usage and unfunded research, but part of the cluster is dedicated to funded research projects allowing for privileged access to and usage of processing nodes. The RACC provides interactive research computing and batch job submissions after logging into the system through the login nodes. RACC is free for small storage needs within the user’s home directory and also chargeable for large storage requirement with Research Data Volumes. The whole RACC system architecture is described on the UoR website [28].

The JASMIN system is introduced in Section 2.4.1.2. For the processing of CDR-4, as the L2 IMS data is available from the CEDA archive, the user assigned space is sufficient for storage of the DMP, intermediate, and L3 data, thus is free of charge to use. The WV_cci group workspace is not required for IMS processing and storage as the final merging processing is performed on the UoR system.

3. TECHNICAL METHODS AND CONCEPTS

3.1 TCWV Processing System

This section prepares for the detailed descriptions of the parts of the actual processing chain. It consists of an introduction to HDFS (Hadoop Distributed File System) and the hardware and software infrastructure of Calvalus.

The Calvalus infrastructure has been described in a similar manner in various technical and project related documents. The description of the general concepts (not related to WV_cci) in this chapter is mainly taken from [6] and still has a summary character to keep it in the scope of this document. A good starting point for many more technical details and to get a view from the user's perspective would be the Calvalus Software User Manual [29].

3.1.1 HDFS

HDFS, the Apache Hadoop distributed file system, is a distributed and highly scalable file system. A typical cluster running HDFS has a single master plus multiple datanodes. Each datanode stores blocks of data and serves them over the network. This file system is based upon the Google File System (GFS), introduced by Google employees in 2003.

HDFS is designed to store large files, typically in the range of gigabytes to terabytes across multiple machines. This data is replicated across multiple nodes to ensure data availability. Data nodes communicate to rebalance data, to move copies around, and to keep the replication of data.

The main advantage of HDFS is that it allows for data-local processing (see Figure 3-1). This means that the scheduler distributes processing jobs to nodes with an awareness of the data location. For example: if node A contains data X and node B contains data Y, the job tracker schedules node B to perform processing tasks on Y, and node A scheduled to perform processing tasks on X. This considerably reduces the amount of traffic that goes over the network.

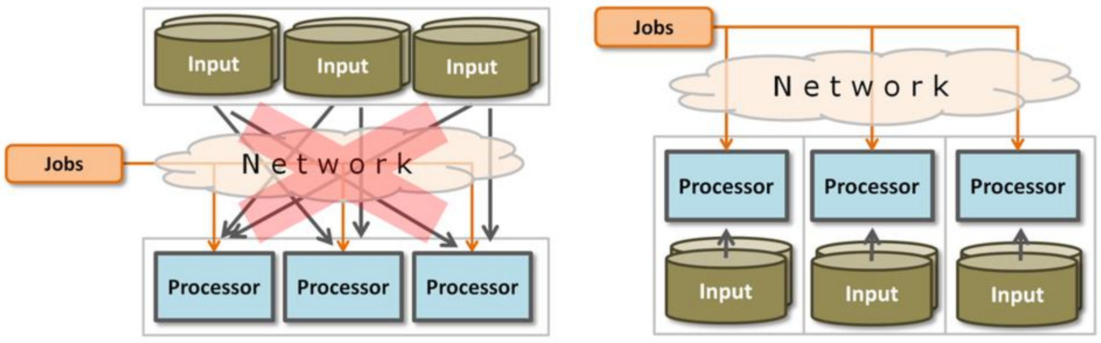


Figure 3-1: HDFS concepts of 'archive centric' and 'data local'.

3.1.2 Calvalus

Calvalus is proprietary software developed by BC since 2010. It employs Apache Hadoop and adapts it to the processing of Earth Observation data. It provides the possibility to perform full mission EO data processing, data aggregation, validation, and value-adding with many frequently updated algorithms and data processors on standard hardware scalable for the amount of data of Sentinels 1, 2 and 3.

3.1.2.1 Software bundles

A software bundle is a Calvalus concept that allows system operators to deploy any runnable software to the system. The processors used in WV_cci, which are provided by project partners, are integrated into software bundles, and as such integrated into the system.

3.1.2.2 Hardware infrastructure

The infrastructure used for WV_cci processing is a Calvalus/Hadoop cluster consisting of computing hardware, storage, input/output elements, and network. This cluster is shared with other projects that each provide parts of the hardware and get in turn a corresponding share of the resources of the cluster. The current cluster has 119 nodes and a storage capacity of about 2.3 PB.

Figure 3-2 shows the layout of the Calvalus/Hadoop cluster with master node, computing and storage nodes, the feeder as input/output element, and an optional test server for services and development. The computing and storage nodes are simple computers with local disks. These disks together are the distributed storage of the cluster managed by the Hadoop Distributed File System.

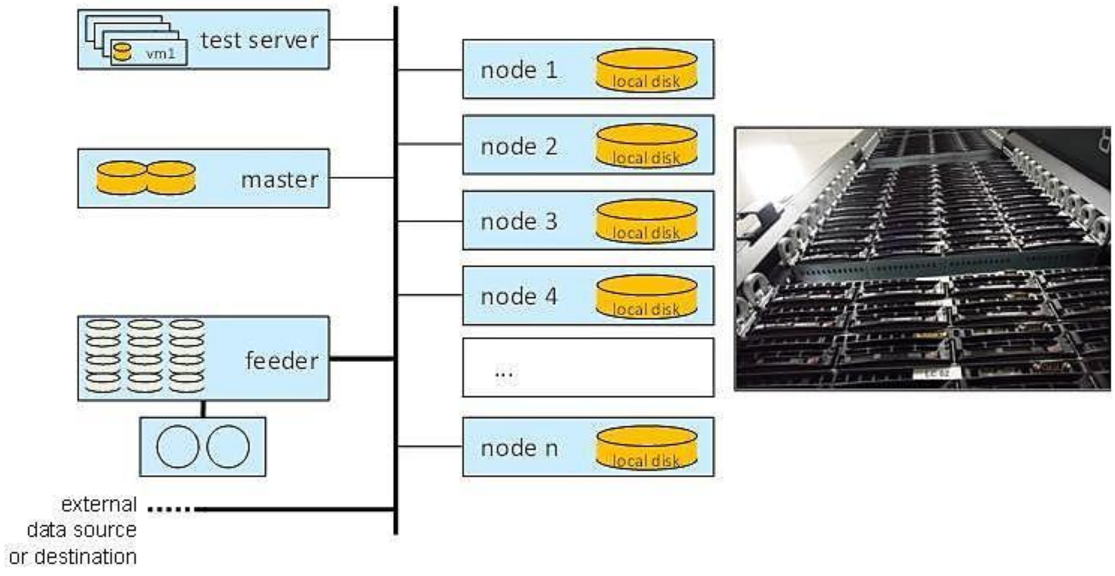


Figure 3-2: Calvalus architecture.

The elements of this infrastructure are described in detail in Table 3-1.

Table 3-1: Calvalus infrastructure elements

Infrastructure elements	Type	Number, size
Computing nodes	Rack-mounted standard computers, 1U, 1 quadcore CPU + hyperthreading, 16-128 GB main memory, 4 hard disks (8–16 TB), 1 Gbit network connection	119 nodes in the cluster
Storage	As listed above the nodes have local disks. They are altogether managed by the Hadoop Distributed File System HDFS. In addition there are offline backups of input datasets at BC.	Overall storage capacity of the cluster is 3 PB. The cluster is used with a replication of 2 for inputs and 1 for outputs which reduces this capacity accordingly. Sharing of input data among projects mitigates the need for storage for all data.
I/O elements	“feeder”: (empty) disk array for 24 hot-pluggable hard disks, 4 Units,	2 feeders for concurrent reading or writing of 48 disks.

Infrastructure elements	**Type**	**Number, size**
	quadcore CPU, 8 or 16 GB main memory, 10 Gbit network connection. FTP server (ftp://ftp.brockmann-consult.de/), used for data exchange with team.	The FTP server is shared with other projects of BC. It is a virtual machine (VM) on a redundant VM server.
Network infrastructure	There is a 10 Gbit backbone linking 4 switches and the 2 feeder nodes for data I/O. Each switch provides 24 1-Gbit ports that connect a set of computing nodes. The cluster Is connected with 10 Gbit to the BC internal network. The FTP server is connected to the BC internal network as well as to the internet. A 3-layers firewall protects the internal production network from the internet. The FTP server is in the outermost layer accessible from the internet. The Calvalus cluster is in the innermost layer.	4 switches 10 Gbit/1Gbit 1 Gbit connection of cluster nodes 10 Gbit connection between switches and to feeders
Operating system	Ubuntu 18.04 LTS	

This approach of Hadoop as middleware and with computing nodes that are at the same time storage nodes has been selected for WV_cci because the knowledge of the location of the respective data allows for data-local processing with minimal use of the network, which makes the approach suitable for massive parallel processing of the large data volumes of preprocessing. The approach is scalable to several petabytes which is an advantage regarding the considerable data volume in later stages of WV_cci.

3.1.2.3 Software infrastructure

The software system for WV_cci processing deployed on the infrastructure above is shown in Figure 3-3. The different layers of the software system start from the operating system up to the individual processors.

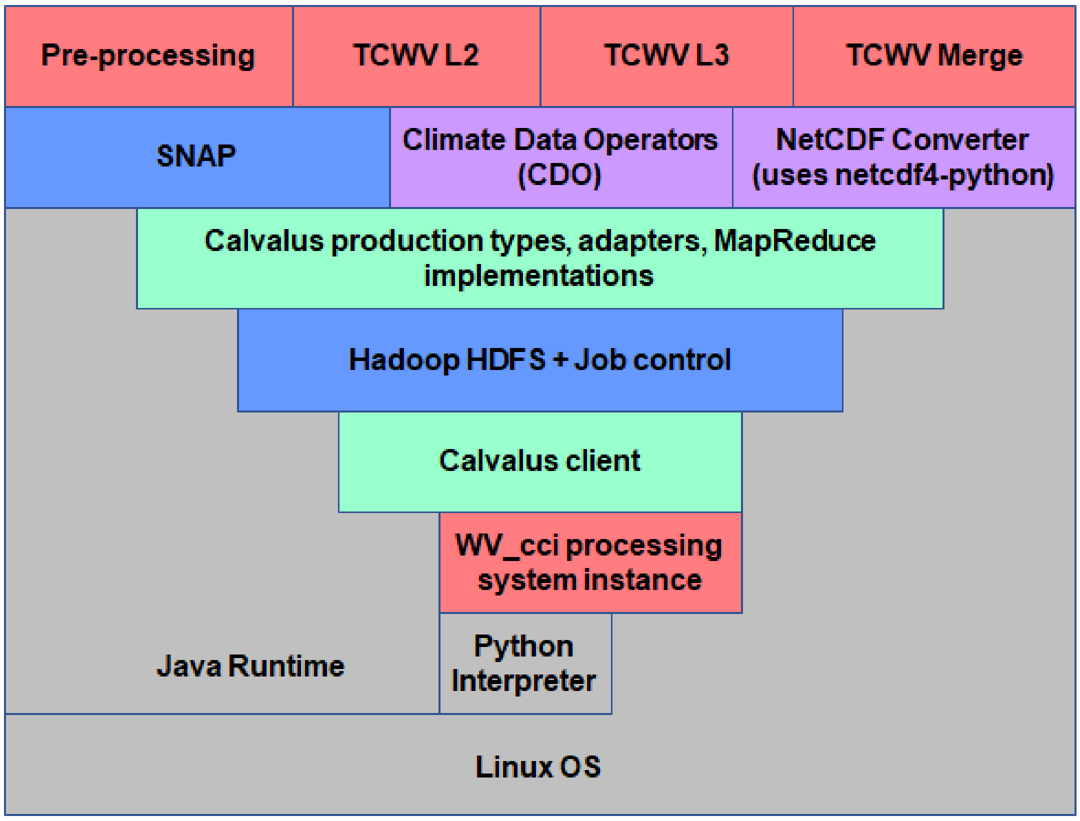

Pre-processing
TCWV L2
TCWV L3
TCWV Merge
SNAP
Climate Data Operators (CDO)
NetCDF Converter (uses netcdf4-python)
Calvalus production types, adapters, MapReduce implementations
Hadoop HDFS + Job control
Calvalus client
WV_cci processing system instance
Java Runtime
Python Interpreter
Linux OS

Figure 3-3: Software architecture.

The layers are:

- Basic software (shown in grey) comprises the Linux operating system, the Java runtime, and the Python interpreter.
- WV_cci processing instance (shown in red) from which the processing flow for a given task is invoked. Explained in more detail in the next subsection.
- Calvalus (shown in green) provides several layers, with a client layer that uses Hadoop and a layer plugged into the Hadoop framework with production types and adapters, both together implementing the Earth Observation functions not available

in bare Hadoop. This part also contains the formatting tool MapReduce implementation.

- Apache Hadoop with its distributed file system and its job scheduling functions and cluster tools as well as ESA SNAP with its graph processing framework are layers used by Calvalus for cluster processing. They are shown in blue.
- Third-party data processing tools (shown in pink) which are wrapped into Unix executables for parallel execution on Calvalus:
  a. Climate Data Operators (CDO): Used for the ingestion of ERA Interim ancillary data, i.e. spatial/temporal interpolation to create ERA time slices matching the satellite input datasets.
  b. a Python-based NetCDF conversion tool for the generation of the final CF- and CCI-compliant NetCDF4 products.
- Specific elements developed for WV_cci (shown in red in the top row) are
  a. the preprocessing, including the pixel classification, band subsetting and ERA Interim ingestion.
  b. the TCWV L2 generation for MERIS and OLCI.
  c. the TCWV L3 generation for MERIS, OLCI, and MODIS.
  d. the TCWV merge, applied for all products including CM SAF HOAPS.

Like the hardware, the software can also be shared with other projects making use of the Calvalus environment. However, it is not always the same version of a software item that is used by different projects. Therefore, the versioning approach has a key role in the software deployment and runtime use for WV_cci in the Calvalus environment:

- Several versions of software items can be deployed in this environment at the same time. The actual requests generated by the processing system instance determine which combinations of versions are used. This is the case for processors (upmost layer) but also for SNAP and Calvalus and the processing system, optionally also for Java and Python. Only the operating system and the Hadoop version being used is one at a time.
- The software items are versioned, and the versions are managed in version control systems. For software items developed by Brockmann Consult the version control system is Git. The software repositories are hosted on GitHub, a cloud service. Some of the repositories are public, e.g., the one for SNAP which is open source.
- Other software items are version-controlled externally. An example is Apache Hadoop maintained as open source by Apache Foundation.

Table 3-2 lists the software items with their role in the WV_cci processing system and its configuration control.

Table 3-2: Software items for WV_cci processing

Software item	Purpose, use	Configuration control
Java runtime environment	Basic software used for Hadoop, Calvalus, ESA SNAP, some processors.	Oracle SDK, Version 8 (1.8.0_121). Java is available from https://www.oracle.com/technetwork/java/index.html
Python interpreter	Basic software used for pmonitor.	Default: version 2.7.6, available from www.python.org. (higher Python versions can be used if needed for specific tasks)
Apache Hadoop	Cluster software with data management HDFS, job scheduling, command line tools and Web operating interface, several APIs: client side for job submission and monitoring, server side for plug-in of map and reduce tasks.	Version 3.2.1, versions maintained by Apache, available from hadoop.apache.org2 (open source)
Calvalus	Earth Observation application layer on top of Hadoop with HDFS archive directory structure with archiving rules, client tool for request submission, software deployment, data ingestion processor adapters for various programming languages (among them for BEAM/SNAP operators processor deployment as versioned software bundles). User portal for on-demand processing. Processing system instances for controlled bulk production.	Version 2.22, version control on GitHub in repository of BC

Software item	Purpose, use	Configuration control
ESA SNAP	Framework for processor development and execution, concepts for product, reader, writer, operator, operator chaining, tile cache and many more, basic software for TCWV preprocessing (subsetting, reprojection), Level 3 aggregator.	Version 9.0.0 SNAP is provided by ESA, maintained and further developed by BC, version control in public repository of BC (open source).
CDO	Climate Data Operators: Used for the ingestion of ERA Interim ancillary data, i.e. spatial/temporal interpolation to create ERA time slices matching the satellite input datasets. It is one of the preprocessing steps before the TCWV L2 generation.	Version 1.9.5, maintained by Max-Planck-Institute for Meteorology, Hamburg, Germany. Available from http://mpimet.mpg.de/cdo
SCRIP product writer	This SNAP product writer implementation is a modification to the NetCDF 4 CF writer in order to generate ERA Interim intermediate products in the CDO-specific SCRIP NetCDF format. This allows a proper extraction of geolocation information from the ERA Interim original products using CDO. It is used within the ERA Interim ingestion preprocessing step.	SNAP writer plugin, provided within the snap-wvcci-2.x software bundle (see below).
SNAP Idepix	SNAP processor for pixel classification (i.e. cloud detection). The pixel classification is one of the preprocessing steps before the TCWV L2 generation.	SNAP processor plugin, maintained and further developed by BC, version control in public repository of BC (open source).
TCWV L2 Processor	SNAP processor for TCWV L2 generation, implementing the TCWV retrievals for MERIS, MODIS and OLCI.	SNAP processor plugin, provided within the snap-wvcci-2.x software bundle (see below).

Software item	Purpose, use	Configuration control
TCWV L3 aggregator	Spatial and temporal aggregation of TCWV values from L2 processing, reprojection onto global WGS-84 lat/lon grid, and NetCDF-formatting of the result.	Maintained as part of Calvalus. Functionality is fully provided by Calvalus software.
TCWV L3 merge	Merge of TCWV L3 products from the different sensors to obtain products with global TCWV coverage in ideal case.	SNAP processor plugin, provided within the snap-wvcci-2.x software bundle (see below).
NetCDF conversion tool	Used to convert SNAP-generated NetCDF 4 files into final format which fully follows CCI and NetCDF CF product conventions as well as the specifications in the PSD.	Python script wrapped into Unix executable, both provided within the snap-wvcci-2.x software bundle (see below).
snap-wvcci-2.x	Software bundle containing the SCRIP writer plugin, the TCWV L2, L3 and merge processors, the NetCDF conversion tool, TCWV lookup tables, and other utility components (wrapper scripts etc.).	Versioned; maintained and further developed by BC, version control in public repository of BC (open source). (Lookup tables are not held in public repository due to their large size.)
wvcci-inst	Processing system instance with workflow control and processing progress and status.	Automated daily backup, version control in private repository of BC.

3.1.2.4 The WV_cci processing instance

To perform bulk processing jobs, Calvalus offers as options the on-demand processsing with the Calvalus portal, but also the use of a processing system instance installed on the master node. This is both explained in much detail in the Calvalus SUM [29]. For WV_cci, the latter option with an instance wvcci_inst is used. In summary, the concept is illustrated in Figure 3-4 for the example of MERIS TCWV L2 processing.

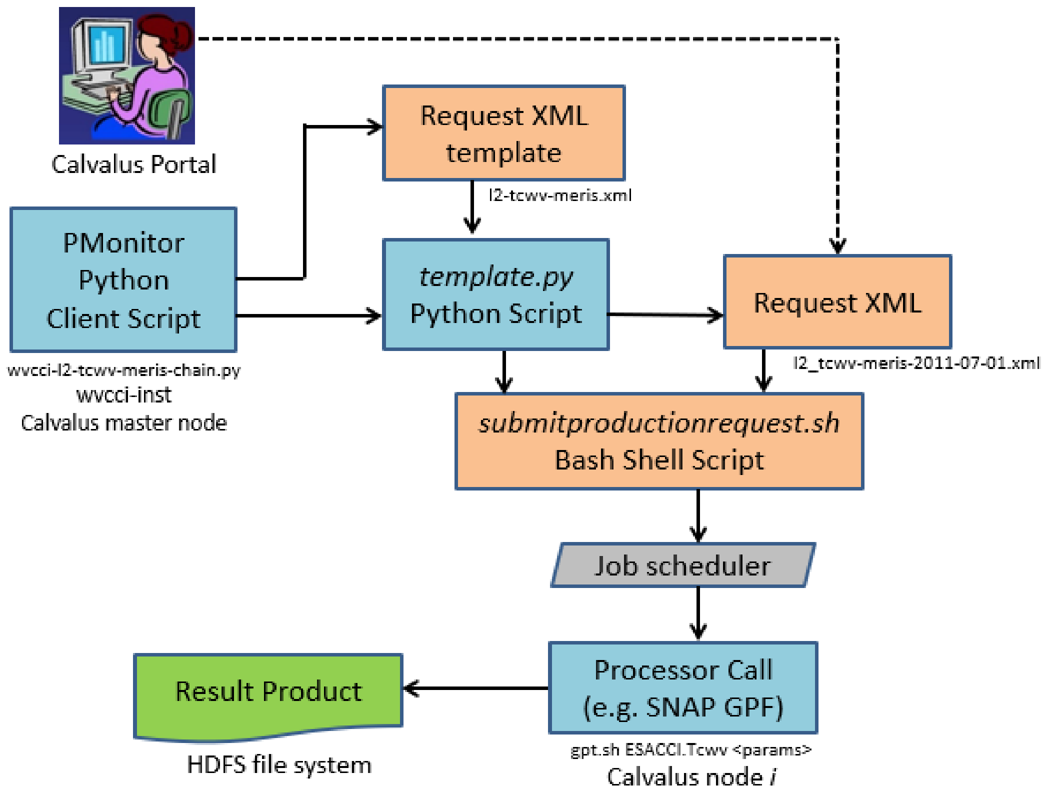

Calvalus Portal
Request XML template
l2-tcwv-meris.xml
PMonitor Python Client Script
wvcci-l2-tcwv-meris-chain.py
wvcci-inst
Calvalus master node
template.py Python Script
Request XML
l2_tcwv-meris-2011-07-01.xml
submitproductionrequest.sh Bash Shell Script
Job scheduler
Processor Call (e.g. SNAP GPF)
gpt.sh ESACCI.Tcwv <params>
Calvalus node i
Result Product
HDFS file system

Figure 3-4: Calvalus options for bulk processing. The processing instance *wvcci-inst* is used for WV_cci. Names of scripts for MERIS TCWV L2 processing are given as examples.

The processing task (e.g., process TCWV L2 from MERIS L1b for two given months) is initiated from a PMonitor Python client script. In this script, a PMonitor instance together with necessary processing parameters (such as year, month, data input/output paths, etc.) are defined. PMonitor is a Python-based workflow engine which handles tasks and their dependencies and executes them on a thread pool. Most importantly, it provides:

- a thread pool with a task queue of mature tasks with inputs available
- a backlog of tasks with inputs not yet available
- a status file with the overall job status (jobs running, in backlog, finished or failed)
- individual log files per job
- a report file that records all completed calls and the paths of output product (set) names
- a set of commands executed in previous runs listed in the initial report.

The PMonitor concept is illustrated in Figure 3-5.

From the client script, another Python script template.py is invoked, which builds up the explicit processing request by 'filling' a task-related request template (here, for the task 'MERIS TCWV L2 processing') with the actual processing parameters. Then, the request is fed into a Bash shell script submitproductionrequest.sh, which passes the request with the dedicated jobs into the Calvalus system. The jobs are then fed into the specified request queue and distributed by the job scheduler onto the available processing nodes. Usually, a single job is related to a single processor call (here, the SNAP TCWV L2 processor) which generated, after successful processing, a single result product (here, a TCWV L2 product) which is available on the HDFS file system.

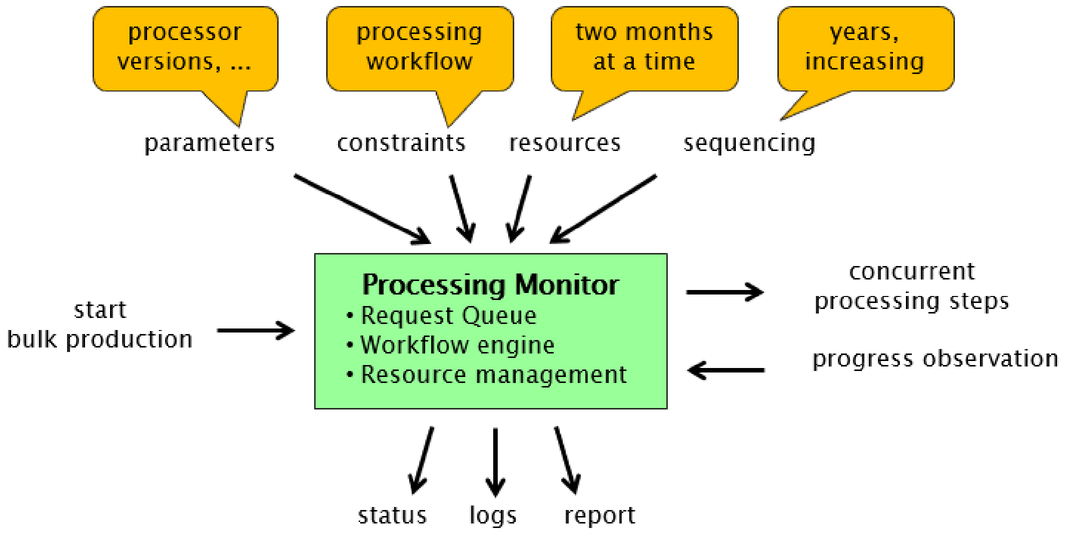


Figure 3-5: Concept of the PMonitor workflow engine. Taken from [29].

The concept outlined above has been used for the whole production on Calvalus during WV_cci Phase 1. During the first year of Phase 2, a modified, more efficient production setup has been established. This setup does not only increase temporal performance (improved job scheduling), but also simplifies the job configuration and the processing maintenance. The setup has been successfully applied to produce TCWV from MERIS 4th reprocessing input data and thus shall be used for all further production during Phase 2. The main differences of this setup compared to the one used in Phase 1 are:

- the initial component to start a workflow is no longer a Python client script, but a request file in YAML format.

- the job configuration is no longer held in a (relatively long) XML template, but in a simpler and shorter (thus less error prone) configuration in JSON format.
- the initialisation and startup of the PMonitor instance, previously part of the client script, has been moved into a 'workflow' Python script.
- the functionalities of 'submitproductionrequest.sh' are now provided by a sequence of Calvalus core Python scripts, mainly 'hstep.py'. They do not need to be changed for different processing workflows.

This modified production setup is illustrated in Figure 3-6 for a workflow <xxx>. For illustration, explicit file names are given for the workflow xxx='tcwv-l3-monthly', the generation of TCWV Level 3 monthly products from TCWV L3 daily inputs. Listings of these files are given in the Appendix.

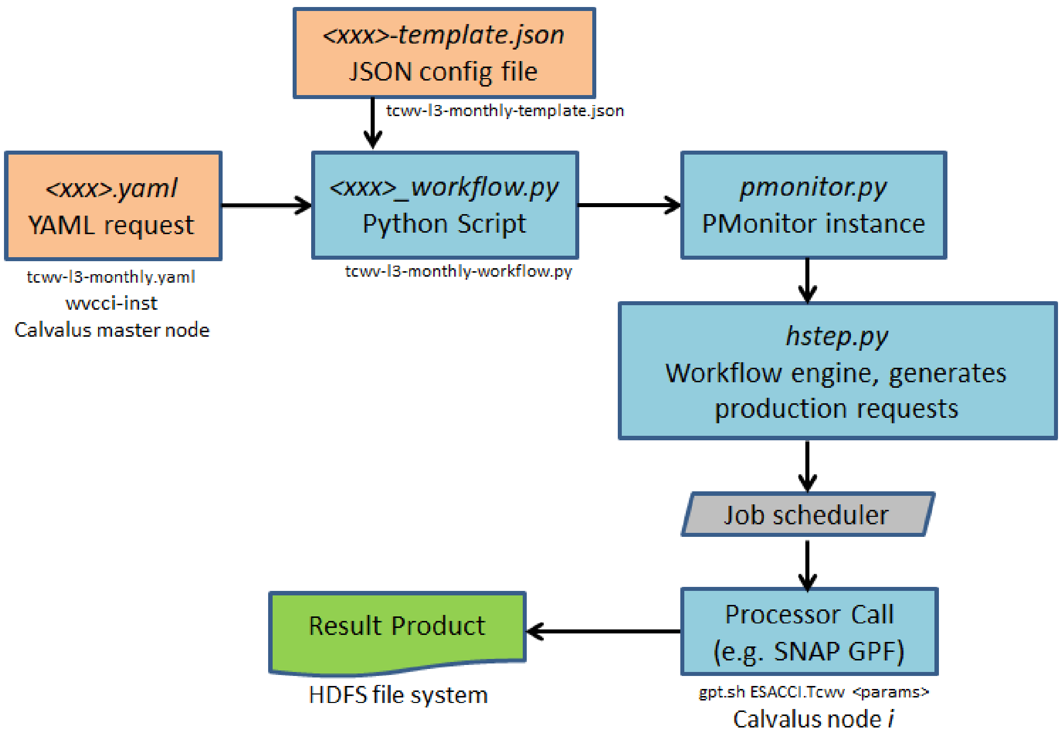


Figure 3-6: Modified Calvalus production setup. Names of scripts for generation of TCWV L3 monthly products are given as examples. See text for more details.

3.1.3 JASMIN

3.1.3.1 Hardware environment

The initial technical architecture was selected to provide a flexible, high-performance storage and data analysis environment, supporting batch computing, hosted processing, and a cloud environment. The increasing need for scientific workflows to “bring the compute to the data” (similar to the Hadoop ‘data local’ approach implemented on Calvalus) drove the development of an infrastructure to support analysis of archive data alongside datasets brought into or generated by projects in their own collaborative workspaces. The most important components deployed in the first phase (2011/12) were:

- Low latency core network
- High-performance disk storage system supporting parallel write
- Access to expandable tape storage for near-line storage
- A batch scheduler
- Block storage for storing virtual machine images

In several phases over the following years (2013-2022), the system has been further evolved, as described in detail in [21]. Table 3-3 summarizes the current status of the main components of the JASMIN hardware environment.

Table 3-3: Main components of JASMIN hardware environment. Taken from [21].

Hardware component	Numbers or specification
Disk storage	• 10 PB SOF + 0.5 PB SSD • 2 PB HPOS • 5 PB PFS • 300TB SSD • 4-500TB Flash
Compute	• 92 x compute nodes with 512 GB RAM, dual AMD Epyc processor, 48-core • Total 92 x 48 = 4416 cores, mostly for deployment in LOTUS cluster.
Network	• 40Gbit/s core network • 25Gbit/sec NIC upgrade for hypervisors in managed cluster

Hardware component	Numbers or specification
Tape storage	• NLDS design & development project underway at CEDA in collaboration with University of Reading. • 23 PB media, 4 drives, 2 data frames, chamber licences & associated costs. • 2 data servers

3.1.3.2 Batch processing

The batch computing environment of JASMIN is the LOTUS processing cluster which is part of JASMIN. LOTUS is not, in itself, a High-Performance Computing (HPC) facility, but provides the batch and parallel processing component of the JASMIN data-intensive scientific analysis environment. LOTUS is a cluster of physical machines, running the SLURM workload manager [24], enabling efficient scheduling of larger data analysis tasks across nodes in the cluster as a single unit -see Figure 1. Each node in the cluster is connected by 10Gbit/s Ethernet to JASMIN's high-performance 40Gbit/s core network.

Many more details around batch processing with LOTUS are given in [23].

3.1.3.3 Job submission and parallel processing environment

As said previously, JASMIN offers to submit commands to the LOTUS cluster, which are scheduled and executed by that engine. See [25] for details. The commands which are submitted may be any executable which is able to run within the prepared environment. Hence, to run a specific task (such as in WV_cci the preprocessing and TCWV L2 retrieval for MODIS), it is necessary that the executable is prepared in the user’s home directory or in the group workspace of the project, and it is ensured that all shared libraries are available there. If these prerequisites are fulfilled, the execution command may be submitted to the LOTUS cluster.

In WV_cci Phase 2, the SLURM job scheduler will be used for parallel processing. In September 2020, SLURM replaced the former LSF scheduler which was used in Phase 1. This change required modifications of the various scripts used in the bulk processing setup (Figure 3-7), following the guideline given in [26]. Explicit examples are given in the Appendix. However, the workflow basically remains the same, and the PMonitor workflow engine is still used for the job control on JASMIN and serves as entry point for the job submission into the LOTUS cluster.

Figure 3-7 illustrates this bulk processing setup. As example, the processing of MODIS TCWV L2 for a given time period is considered. From the client script, a Bash shell script is invoked, which defines (job ID) and registers the jobs to process. The distinct process related to a single job (here, the TCWV Java processor to compute one MODIS scene) is defined in another Bash script. Job definition, job registration and process definition are passed into a third Bash script which in return feeds the job into the LOTUS cluster. Here it is delegated by the SLURM scheduler to an available LOTUS node for processing. After successful processing, a single result product (here, a MODIS TCWV L2 product) is available in the WV_cci group workspace.

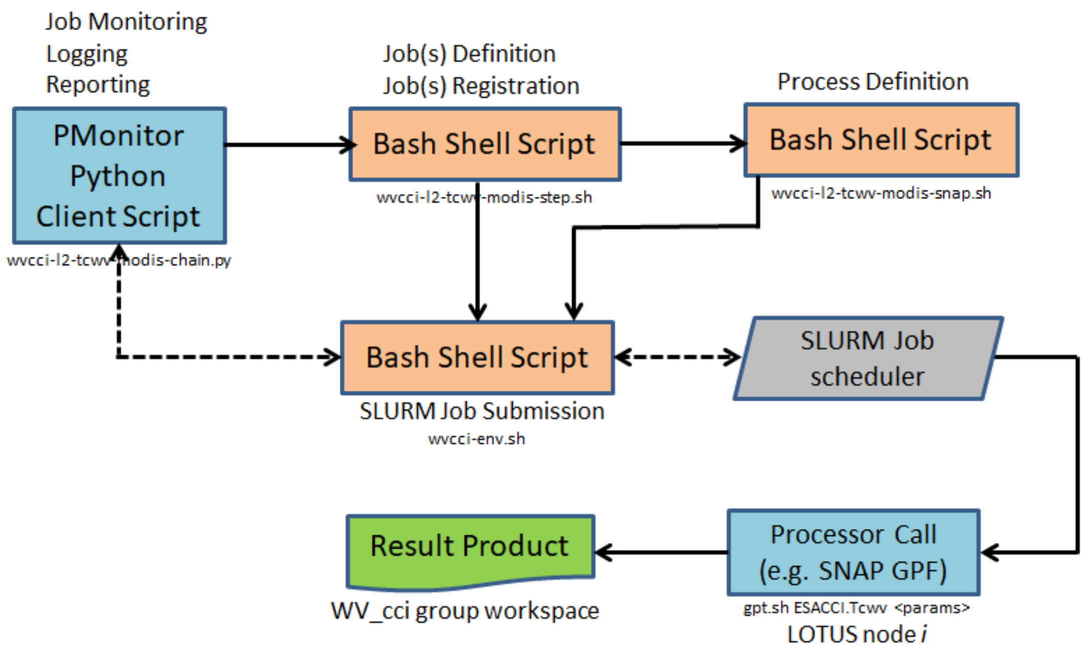


Figure 3-7: Scheme for bulk processing setup on JASMIN, now using the SLURM job scheduler. Names of scripts for MODIS TCWV L2 processing are given as examples.

3.2 VRWV Processing System

The VRWV processing system relies on the Reading Academic Computing Cluster (RACC) system and the JASMIN system. This section introduces the hardware and software infrastructure of the RACC. The detailed description of the JASMIN system can be found in Section 3.1.3.

3.2.1 RACC hardware environment

The RACC provides chargeable Research Data Storage volumes for large storage requirements and also a 150 TB scratch space on request at UoR. Typically, UoR users have home directories for small storage needs (up to 2 TB). The RACC supports both interactive research computing and batch job submission. The interactive computing relies on the login nodes on the RACC and the sessions are limited to 4 CPU and 100 GB of memory per user. The batch computing environment on RACC are managed using SLURM workload manager. The RACC provides 24 hours default time limit for batch jobs, with up to 30 days maximum limit. The cluster consists of around 70 computing nodes with core counts varying between 8-cores up to 24-cores per node. The RACC provides enough compute resources to handle the processing and computing for the VRWV CDR-4 product. As mentioned in Section 2.4.2, the details on the RACC system can be found on the UoR website [28].

3.2.2 RACC processing environment

As said previously, the RACC provides interactive research computing and batch job submission to the cluster. The batch jobs are run according to the command script submitted to RACC. All the programs submitted are prepared in the user's home directory and executable in the RACC cluster. Figure 3-8 shows the schematic layout of the RACC system and the computer nodes provide around 1000 CPU cores for batch jobs.

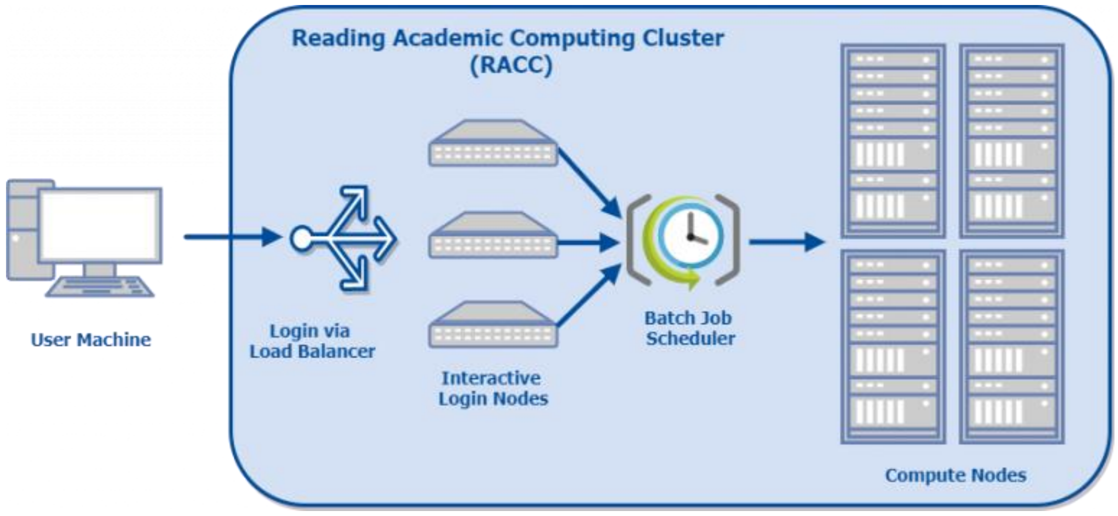


Figure 3-8: Schematic layout of the RACC, taken from [28].

3.2.3 RACC software environment

The software used for the ingestion, processing, testing, and outputting of the climate data record CDR-3 consists of a commercial distribution of the Interactive Data Language by Exelis Visual Information Solutions (IDL version 8.3.0). IDL is vectorised, numerical, and interactive, and is commonly used for interactive processing of large amounts of data. The syntax includes many constructs from Fortran and some from C. For CDR-4, the software of choice is Python version 3.9, with the RACC offering a run environment packaged with the operating system. The Python-based NetCDF conversion tool netcdf4-python for the generation of the final CF- and CCI-compliant NetCDF4 products. For both VRWV CDRs, the software mainly used for testing and visual inspection of the data also includes the NASA/GISS version 4.5 distribution of Panoply, which uses several third-party, open-source Java libraries.

4. WORKFLOWS

This section describes in detail all processing workflows, the input and intermediate data being used, the techniques employed, and the final outputs being created.

4.1 TCWV Processing System

As mentioned previously, the TCWV processing is BC's responsibility and consists of the generation of TCWV products at level 2 and level 3, where the final L3 products will represent the WV_cci CDR-1 and CDR-2 datasets.

4.1.1 TCWV L2 processing

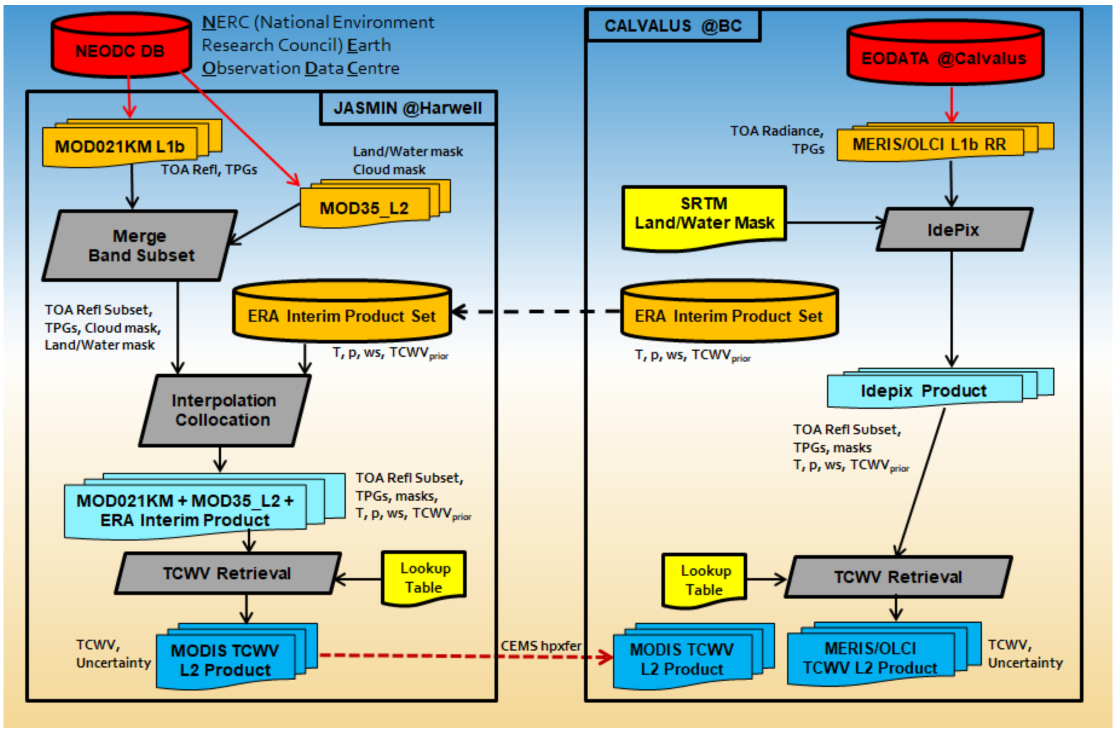


Figure 4-1: The TCWV L2 processing chains for MODIS, MERIS and OLCI. See text for details.

Figure 4-1 shows a schematic overview of the TCWV L2 processing chains. They are mainly separated into the generation of MODIS TCWV L2 products at JASMIN, and the generation of MERIS and OLCI TCWV L2 products at Calvalus. This setup has been established in WV_cci Phase 1 and is basically still applied in Phase 2. The various components of the flows as well as (minor) modifications made for Phase 2 are described in detail in the following sections.

4.1.1.1 MERIS

4.1.1.1.1 Data acquisition

The data inputs used for the MERIS TCWV production are:

- MERIS L1b 4th reprocessing, Reduced Resolution TOA radiance products [31],
- Land/water mask shapefiles derived from the SRTM (Shuttle Radar Topography Mission, [32], [33]) dataset,

ERA Interim [34] ancillary variables are no longer required for the TCWV retrieval algorithm, they are now included in the MERIS L1b 4th reprocessing inputs (see below).

MERIS L1b Reduced Resolution

The MERIS L1b Reduced Resolution 4th reprocessing full archine (06/2002–04/2012) is available at Calvalus and can directly be accessed for processing.

SRTM Land/water mask

For the pixel classification preprocessing step, a reliable land/water mask is needed because the pixel classification scheme is different for land and water.

The SRTM land/water mask is provided as shapefiles with 50-m resolution. If not yet done in an earlier processing, these shapefiles are automatically downloaded by SNAP during the classification for a given L1b scene. Only shapefiles which intersect the region of the scene are downloaded. The downloaded files are stored in the SNAP application directory '.snap' in the user home directory of the Calvalus node1 where the processing job is performed, following the 'data local' principle for job scheduling as illustrated earlier in Section 3.1.1.

The SRTM land/water mask is not available for latitudes below 60°S. In this case, or if the SRTM files cannot be downloaded for any reason, the L1b land/water flag is used as fallback.

Ancillary variables

The TCWV retrieval algorithm requires various ancillary variables as mandatory input:

[1] *On the Calvalus processing nodes operating under Linux, the user is 'yarn', so the SNAP application directory is '/home/yarn/.snap'*

- temperature at 2-m height
- mean sea level pressure
- TCWV initial guess ('prior' value)
- horizontal wind components (for TCWV retrieval over ocean).

All these variables are now available from the tie point grids of the MERIS L1b 4th reprocessing inputs. An extraction from ERA reanalysis data from ECMWF as done in Phase 1 is no longer necessary.

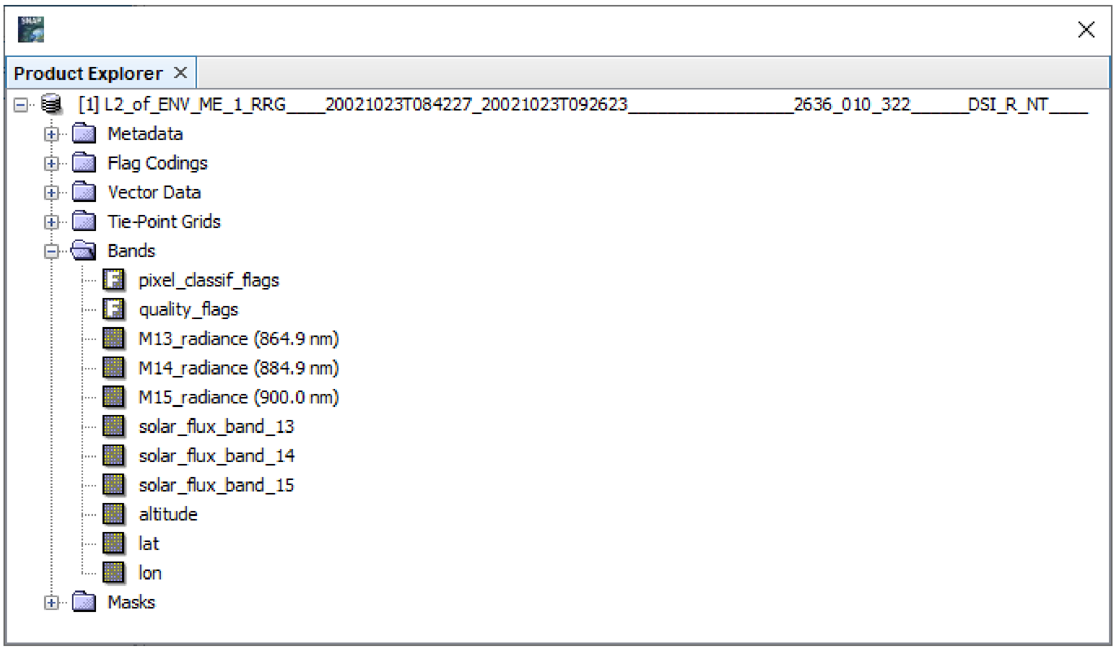
4.1.1.1.2 Preprocessing

The MERIS preprocessing consists of the sub-tasks:

- Idepix: pixel classification based on neural net approaches separately over land and water. Required for cloud pixel identification and their exclusion from TCWV retrieval;
- radiance-to reflectance conversion, required for TCWV retrieval input;
- band subsetting and merging: The TCWV retrieval algorithm requires as input TOA reflectances in MERIS bands 13, 14, 15 (864, 884 and 900nm), the Idepix pixel classification flag, and the ancillary variables listed above

These preprocessing steps are illustrated in the upper right part of Figure 4-1. The radiance-to reflectance conversion is done by an internal SNAP GPF processor. It is called during the Idepix pixel classification. In return, Idepix, which is also a GPF processor developed in Java, is provided as SNAP plugin. It is listed in Appendix 3: Processing scripts (section 2). As said, the integration of ERA Interim ancillary data is no longer needed. The subsetting and merging is also done with a SNAP GPF processor which is provided with the snap-wvcci-2.x software bundle (see Table 3-2).

The final outcome of the preprocessing steps for a given MERIS L1b input is labelled in Figure 4-1 as 'Idepix Product'. This product in return serves as input for the TCWV L2 retrieval. The content of the product is shown in Figure 4-2.



```
Product Explorer
[1] L2_of_ENV_ME_1_RRG____20021023T084227_20021023T092623________________2636_010_322______DSI_R_NT____
    Metadata
    Flag Codings
    Vector Data
    Tie-Point Grids
    Bands
        pixel_classif_flags
        quality_flags
        M13_radiance (864.9 nm)
        M14_radiance (884.9 nm)
        M15_radiance (900.0 nm)
        solar_flux_band_13
        solar_flux_band_14
        solar_flux_band_15
        altitude
        lat
        lon
    Masks
```

Figure 4-2: SNAP Product Explorer: MERIS IdePix pixel classification product after merge with reflectance bands.

4.1.1.1.3 TCWV retrieval

The TCWV retrieval step is illustrated in the lower right part of Figure 4-1. The retrieval is done with a SNAP GPF processor which is provided with the snap-wvcci-1.x software bundle (see Table 3-2). This processor uses specific lookup tables for MERIS required by the implemented optimal estimation algorithm. They are also included in the software bundle. If not yet done in an earlier processing, these lookup tables are automatically installed into the SNAP application directory '.snap' in the user home directory of the Calvalus node (same principle as for the SRTM shapefiles as described above).

The final outcome of the TCWV retrieval step is labelled in Figure 4-1 as 'MERIS/OLCI TCWV L2 Product'. The content of the product is shown in Figure 4-3.

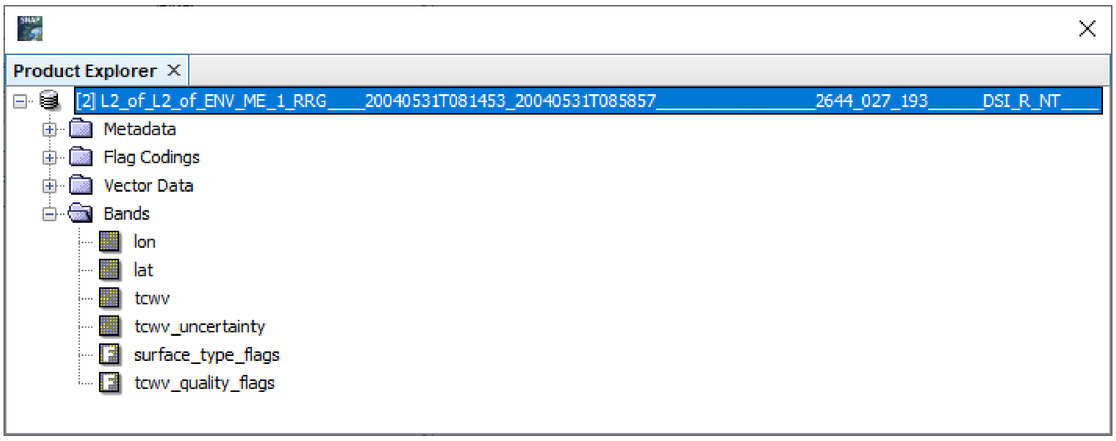

Product Explorer
[2] L2_of_L2_of_ENV_ME_1_RRG____20040531T081453_20040531T085857________________2644_027_193________DSI_R_NT____
- Metadata
- Flag Codings
- Vector Data
- Bands
  - lon
  - lat
  - tcwv
  - tcwv_uncertainty
  - surface_type_flags
  - tcwv_quality_flags

Figure 4-3: SNAP Product Explorer: Contents of a MERIS TCWV L2 product.

The MERIS TCWV L2 products are not part of the TCWV CDR-1 and CDR-2 datasets. However, sets of these products can be made available to users on specific demand.

4.1.1.2 MODIS

4.1.1.2.1 Data acquisition

The data inputs used for the MODIS TCWV production are:

- MODIS MOD021KM (Terra) and MYD021KM (Aqua) Calibrated Radiances 5-Min L1B Swath 1km products [36], [37];
- MODIS MOD35_L2 Cloud Mask and Spectral Test Results 5-Min L2 Swath 250m and 1km products [38];
- ERA Interim ancillary variables required for TCWV retrieval algorithm

MODIS MOD021KM/ MYD021KM

Starting with year 2000/2002, the MODIS MOD021KM and MYD021KM archives have been completely uploaded by CEDA into their archive to the NEODC database [39].. After setting symbolic links to the NEODC file paths, the datasets can directly be accessed from the project group workspace on JASMIN for processing.

MODIS MOD35_L2

Starting with year 2000, the MODIS MOD35_L2 archive has also been completely uploaded by CEDA into their archive to the NEODC database. Again, after setting symbolic links to the NEODC file paths, the datasets can also directly be accessed from JASMIN for processing. The MODIS TCWV processing chain extracts the cloud mask as well as land/water mask from MOD35_L2.

ERA Interim ancillary variables

As for MERIS and OLCI, the TCWV retrieval algorithm requires various ancillary variables as mandatory input:

- temperature at 2-m height
- mean sea level pressure
- TCWV initial guess ('prior' value)
- horizontal wind components (for TCWV retrieval over ocean).

In opposite to MERIS and OLCI, these ancillary variables are not available from the MODIS L1b inputs. Therefore, ERA Interim reanalysis data from ECMWF are still used for WV_cci Phase 2. These datasets are available from the ECMWF public datasets archive [35] after registration and installation of an ECMWF key. The single products are given as NetCDF files containing time slices for each specified variable every 6 hours for one month on a global WGS-84 lat/lon grid with 0.75° horizontal resolution. The download for a given month is done from the command line with a simple Python script listed in Appendix 3: Processing scripts (section 1). Currently, all files from 2002 to 2018 have been downloaded and transferred from Calvalus to JASMIN where they were made available as static ancillary input datasets in the WV_cci group workspace.

4.1.1.2.2 Preprocessing

In principle, the preprocessing steps for MODIS are the same as for MERIS described above. There are the following differences:

- The pixel classification step uses a binary cloud mask from the MOD35L2 product for cloud identification. For the rare case that this product is not available for the given MOD021KM/ MYD021KM L1b input, a neural net approach similar to MERIS is applied for cloud detection.
- Instead of the three MERIS bands 13–15, the five MODIS bands
    a. EV_1KM_RefSB_17 (905nm)
    b. EV_1KM_RefSB_18 (936nm)
    c. EV_1KM_RefSB_19 (940nm)

d. EV_250_Aggr1km_RefSB_2 (859nm)

e. EV_500_Aggr1km_RefSB_5 (1240nm)

are required as input for the TCWV retrieval and passed into the 'Idepix + ERA Interim Product'.

4.1.1.2.3 TCWV retrieval

The TCWV retrieval step is illustrated in the lower left part of Figure 4-1. It is equivalent to MERIS. The processor uses specific lookup tables for MODIS required by the implemented optimal estimation algorithm. They are also included in the software bundle.

The MODIS TCWV L2 products are also not part of the TCWV CDR-1 and CDR-2 datasets. However, sets of these products can be made available to users on specific demand.

4.1.1.2.4 Data transfer JASMIN → Calvalus

An essential preparatory step for the TCWV L3 processing and the merge of products from the different sensors is the transfer of MODIS TCVW L2 products from the JASMIN cluster to Calvalus where the MERIS and OLCI products are generated, the OLCI products will be generated, and the CM SAF HOAPS TCWV provided by DWD are ingested2 This data transfer is done via rsync command, using the hpxfer high speed data transfer service provided by CEDA. The transfer is initiated from a virtual machine in the Calvalus environment. In a second step the data are moved into the HDFS file system to make them accessible for processing.

4.1.1.3 OLCI

As outlined in the following subsections, the TCWV retrieval procedure from OLCI looks very similar to the one for MERIS described above.

4.1.1.3.1 Data acquisition

The data inputs used for the OLCI TCWV production are:

[2] *For the generation of the final TCWV datasets, DWD provided data from CM SAF HOAPS obtained from the SSMI and SSMIS instruments flying onboard platforms DMSP 5D-3/F16-18.*

- OLCI L1b Reduced Resolution TOA radiance products [41] from Sentinel-3A and (new in Phase 2) from Sentinel-3B
- Land/water mask shapefiles derived from the SRTM (Shuttle Radar Topography Mission, [32], [33]) dataset, as for MERIS

OLCI L1b Reduced Resolution

The full archive of OLCI Level 1b reduced resolution products (Earth Observation Mode, OL_1_ERR), from 04/2016 (S3A) and 12/2018 (S3B) to present, is available at Brockmann Consult. As for the MERIS data, these products can directly be accessed from the Calvalus cluster for processing. The S3A dataset has already been used for WV CCI Phase 1 and is also actually used by other projects.

Ancillary variables

The same ancillary variables as described for MERIS in section 4.1.1.1.1, are also needed for OLCI. All these variables are now also taken from the tie point grids of the OLCI L1b inputs and no longer from ERA reanalysis data.

4.1.1.3.2 Preprocessing

The OLCI preprocessing consists of the sub-tasks:

- Idepix: as for MERIS, a pixel classification based on neural net approaches separately over land and water. Required for cloud pixel identification and their exclusion from TCWV retrieval,
- radiance-to reflectance conversion, as for MERIS, required for TCWV retrieval input,
- band subsetting and merging: The TCWV retrieval algorithm requires as input TOA reflectances in OLCI bands 18, 19, 20, 21 (884, 899, 939 and 1015nm), the Idepix pixel classification flag, and the ancillary variables listed in section 4.1.1.1.1.

These preprocessing steps are equivalent to MERIS as described in section 4.1.1.1.1. The final outcome of all preprocessing steps for a given OLCI L1b input is labelled in Figure 4-1 as 'Idepix Product'. This product in return serves again as input for the TCWV L2 retrieval.

4.1.1.3.3 TCWV retrieval

The TCWV retrieval step is again equivalent to MERIS. The processor uses specific lookup tables for OLCI required by the implemented optimal estimation algorithm. They are also included in the software bundle.

The OLCI TCWV L2 products are also not part of the TCWV CDR-1 and CDR-2 datasets. However, sets of these products can be made available to users on specific demand.

4.1.2 TCWV L3 processing

Figure 4-4 shows a schematic overview of the TCWV L3 processing chains, consisting of the generation of daily and monthly products. The TCWV L3 processing is completely done at Calvalus. The various components of the flows are described in more detail in the following sections.

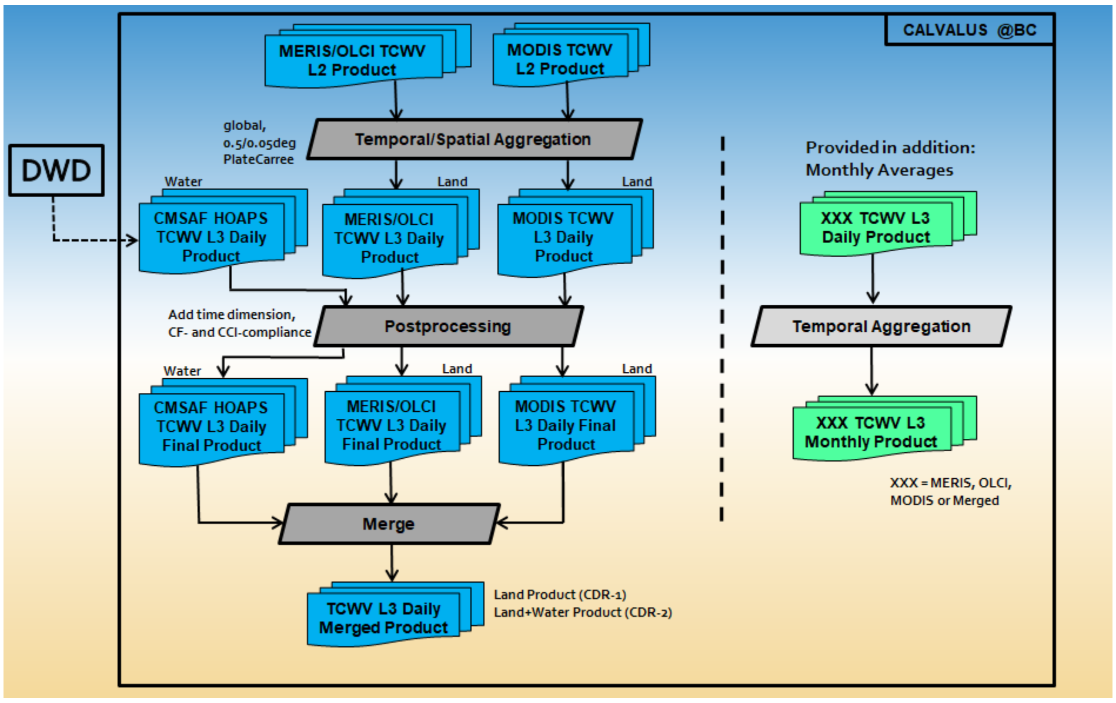

CALVALUS @BC
MERIS/OLCI TCWV L2 Product
MODIS TCWV L2 Product
global, 0.5/0.05deg PlateCarree
Temporal/Spatial Aggregation
DWD
Water
Land
Land
CMSAF HOAPS TCWV L3 Daily Product
MERIS/OLCI TCWV L3 Daily Product
MODIS TCWV L3 Daily Product
Add time dimension, CF- and CCI-compliance
Postprocessing
Water
Land
Land
CMSAF HOAPS TCWV L3 Daily Final Product
MERIS/OLCI TCWV L3 Daily Final Product
MODIS TCWV L3 Daily Final Product
Merge
TCWV L3 Daily Merged Product
Land Product (CDR-1)
Land+Water Product (CDR-2)
Provided in addition: Monthly Averages
XXX TCWV L3 Daily Product
Temporal Aggregation
XXX TCWV L3 Monthly Product
XXX = MERIS, OLCI, MODIS or Merged

Figure 4-4: TCWV L3 processing chains. See text for details.

4.1.2.1 Daily products

The TCWV L3 daily retrieval steps are illustrated in the left part of Figure 4-4.

4.1.2.1.1 Temporal and spatial aggregation

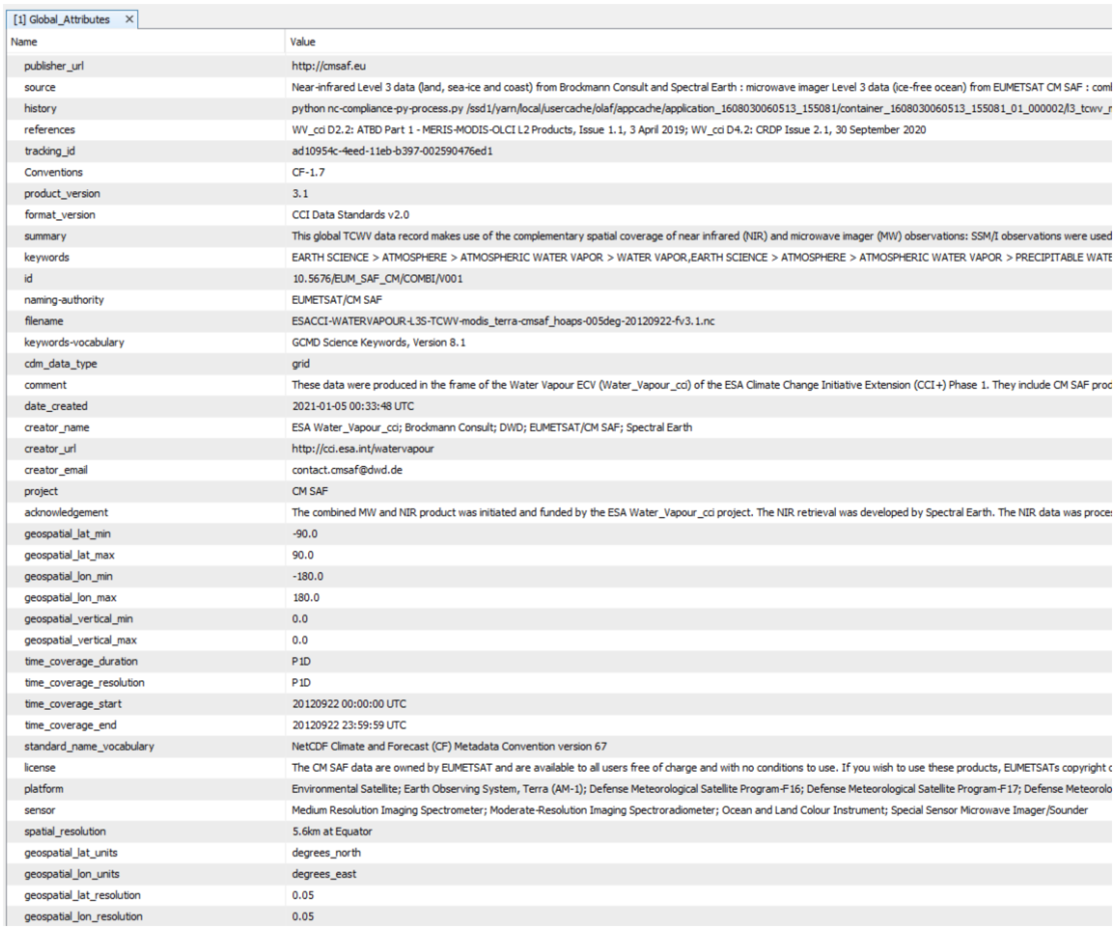
All TCWV L2 products obtained for a particular day from a given NIR instrument (MERIS, MODIS or OLCI) as described in the previous section are taken as input for a temporal and spatial level 3 aggregation. The time aggregation window is the given day, the spatial aggregation is done globally with a reprojection onto a global Plate Carrée grid for two resolutions 0.5 and 0.05 degrees. This results in one (MODIS) or two (MERIS/MODIS or OLCI/MODIS) TCWV L3 products per day from NIR instruments.

As a complementary product at this stage, there is a CM SAF HOAPS TCWV L3 product for the given day, obtained over ice-free ocean from SSM/I or SSMIS microwave instruments. These TCWV products are generated and provided by DWD and are available on the same projection (global Plate Carrée) and with the same resolution (0.5 and 0.05 degrees) as the products from MERIS, MODIS or OLCI, so they can directly be ingested for further processing.

4.1.2.1.2 NetCDF product formatting

The next step is a post-processing procedure where all TCWV L3 products are made compliant to CF- and CCI Data Standards. This includes adding a TIME dimension and variable to the NetCDF products, use of proper variable names, and providing all expected global and variable attributes as well as a comprehensive list of global attributes.

The NetCDF global attributes for a daily TCWV L3 product are shown in Figure 4-5.



[1] Global_Attributes

Name	Value
publisher_url	http://cmsaf.eu
source	Near-infrared Level 3 data (land, sea-ice and coast) from Brockmann Consult and Spectral Earth : microwave imager Level 3 data (ice-free ocean) from EUMETSAT CM SAF : com
history	python nc-compliance-py-process.py /ssd1/yarn/local/usercache/olaf/appcache/application_1608030060513_155081/container_1608030060513_155081_01_000002/l3_tcwv_
references	WV_cci D2.2: ATBD Part 1 - MERIS-MODIS-OLCI L2 Products, Issue 1.1, 3 April 2019; WV_cci D4.2: CRDP Issue 2.1, 30 September 2020
tracking_id	ad10954c-4eed-11eb-b397-002590476ed1
Conventions	CF-1.7
product_version	3.1
format_version	CCI Data Standards v2.0
summary	This global TCWV data record makes use of the complementary spatial coverage of near infrared (NIR) and microwave imager (MW) observations: SSM/I observations were used
keywords	EARTH SCIENCE > ATMOSPHERE > ATMOSPHERIC WATER VAPOR > WATER VAPOR,EARTH SCIENCE > ATMOSPHERE > ATMOSPHERIC WATER VAPOR > PRECIPITABLE WAT
id	10.5676/EUM_SAF_CM/COMBI/V001
naming-authority	EUMETSAT/CM SAF
filename	ESACCI-WATERVAPOUR-L3S-TCWV-modis_terra-cmsaf_hoaps-005deg-20120922-fv3.1.nc
keywords-vocabulary	GCMD Science Keywords, Version 8.1
cdm_data_type	grid
comment	These data were produced in the frame of the Water Vapour ECV (Water_Vapour_cci) of the ESA Climate Change Initiative Extension (CCI+) Phase 1. They include CM SAF pro
date_created	2021-01-05 00:33:48 UTC
creator_name	ESA Water_Vapour_cci; Brockmann Consult; DWD; EUMETSAT/CM SAF; Spectral Earth
creator_url	http://cci.esa.int/watervapour
creator_email	contact.cmsaf@dwd.de
project	CM SAF
acknowledgement	The combined MW and NIR product was initiated and funded by the ESA Water_Vapour_cci project. The NIR retrieval was developed by Spectral Earth. The NIR data was proce
geospatial_lat_min	-90.0
geospatial_lat_max	90.0
geospatial_lon_min	-180.0
geospatial_lon_max	180.0
geospatial_vertical_min	0.0
geospatial_vertical_max	0.0
time_coverage_duration	P1D
time_coverage_resolution	P1D
time_coverage_start	20120922 00:00:00 UTC
time_coverage_end	20120922 23:59:59 UTC
standard_name_vocabulary	NetCDF Climate and Forecast (CF) Metadata Convention version 67
license	The CM SAF data are owned by EUMETSAT and are available to all users free of charge and with no conditions to use. If you wish to use these products, EUMETSATs copyright
platform	Environmental Satellite; Earth Observing System, Terra (AM-1); Defense Meteorological Satellite Program-F16; Defense Meteorological Satellite Program-F17; Defense Meteorolo
sensor	Medium Resolution Imaging Spectrometer; Moderate-Resolution Imaging Spectroradiometer; Ocean and Land Colour Instrument; Special Sensor Microwave Imager/Sounder
spatial_resolution	5.6km at Equator
geospatial_lat_units	degrees_north
geospatial_lon_units	degrees_east
geospatial_lat_resolution	0.05
geospatial_lon_resolution	0.05

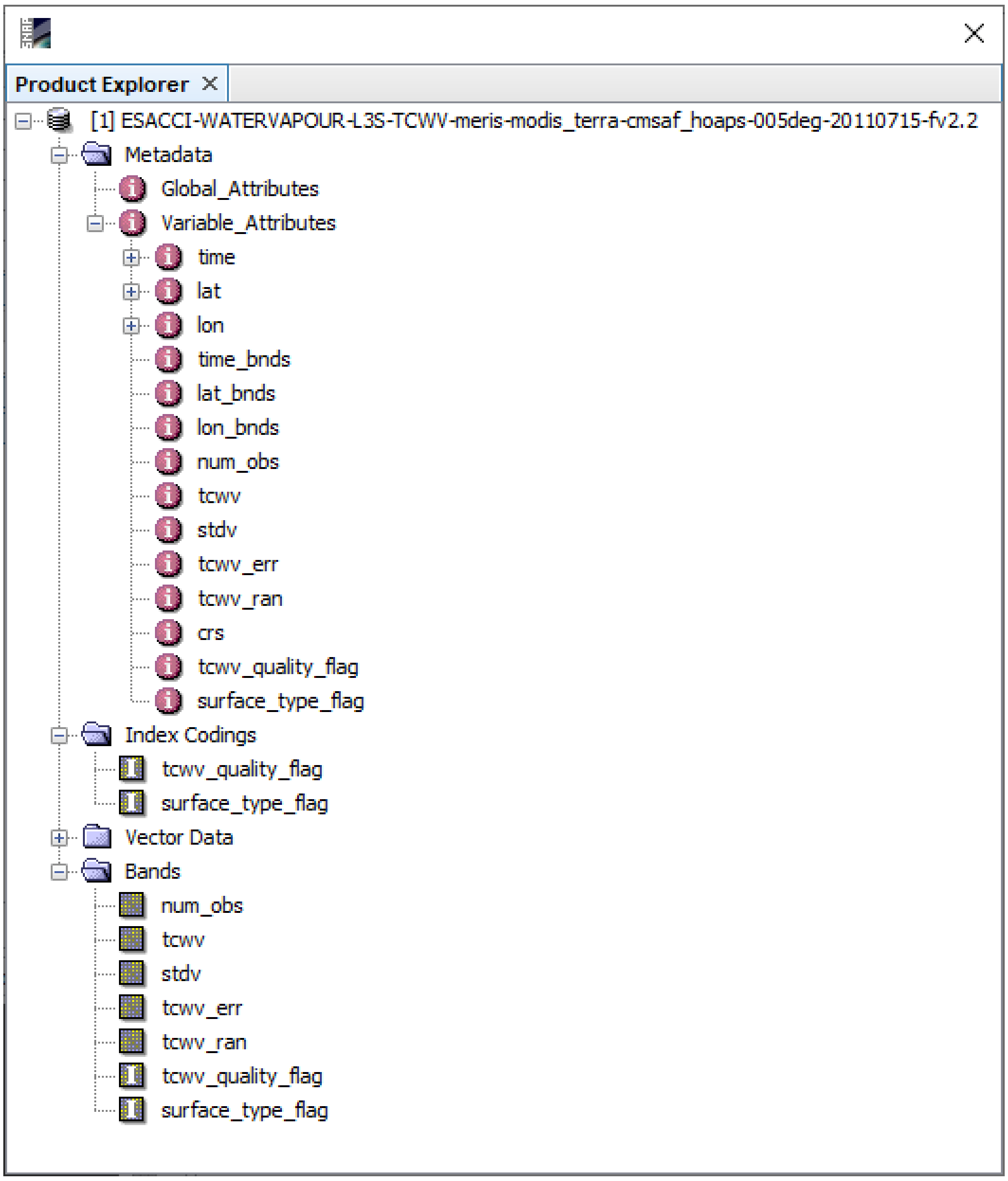
Figure 4-5: SNAP Product Explorer: NetCDF global attributes of a TCWV L3 daily final product.

4.1.2.1.3 Merging of L3 products

The final processing step for the daily L3 products is a merge of the products per day from all products (MERIS/OLCI, MODIS, CM SAF HOAPS) to obtain a global product with a minimum of data gaps. The following merging rules are applied:

- if TCWV from CM SAF HOAPS is available, set merged TCWV to this value (over ocean excl. coastal zones and sea ice)
- if TCWV from CM SAF HOAPS is not available (land, coastal zone or sea ice), set merged TCWV to value from NIR instrument(s). If that value is not available either, set TCWV to NaN.
- if TCWV from CM SAF HOAPS is not available because of coverage, set TCWV to NaN.
- the merged number of observations is the value from CM SAF HOAPS if available, otherwise the value from NIR instrument(s)

The content of a daily TCWV L3 product after this final step is shown in Figure 4-6.



```
Product Explorer
[1] ESACCI-WATERVAPOUR-L3S-TCWV-meris-modis_terra-cmsaf_hoaps-005deg-20110715-fv2.2
    Metadata
        Global_Attributes
        Variable_Attributes
            time
            lat
            lon
            time_bnds
            lat_bnds
            lon_bnds
            num_obs
            tcwv
            stdv
            tcwv_err
            tcwv_ran
            crs
            tcwv_quality_flag
            surface_type_flag
    Index Codings
        tcwv_quality_flag
        surface_type_flag
    Vector Data
    Bands
        num_obs
        tcwv
        stdv
        tcwv_err
        tcwv_ran
        tcwv_quality_flag
        surface_type_flag
```

Figure 4-6: SNAP Product Explorer: Example for a TCWV L3 daily final product generated during WV_cci Phase 1.

The product geolocation information is given through the 'crs' variable and, following CCI standards, also through 1D variables 'lat' and 'lon'. As 2D raster variables we have:

- tcwv (in mm)
- stdv (in mm)
- tcwv_err (in mm)

- tcwv_ran (in mm)
- num_obs (number of TCWV L2 samples to compute L3 value).

The meaning of the statistical quantities and error terms (stdv, tcwv_err, tcwv_ran) is explained in more detail in the PUG [42]

The TCWV global coverage for a single day is illustrated in Figure 4-7. Further examples are shown in the SVR [43].

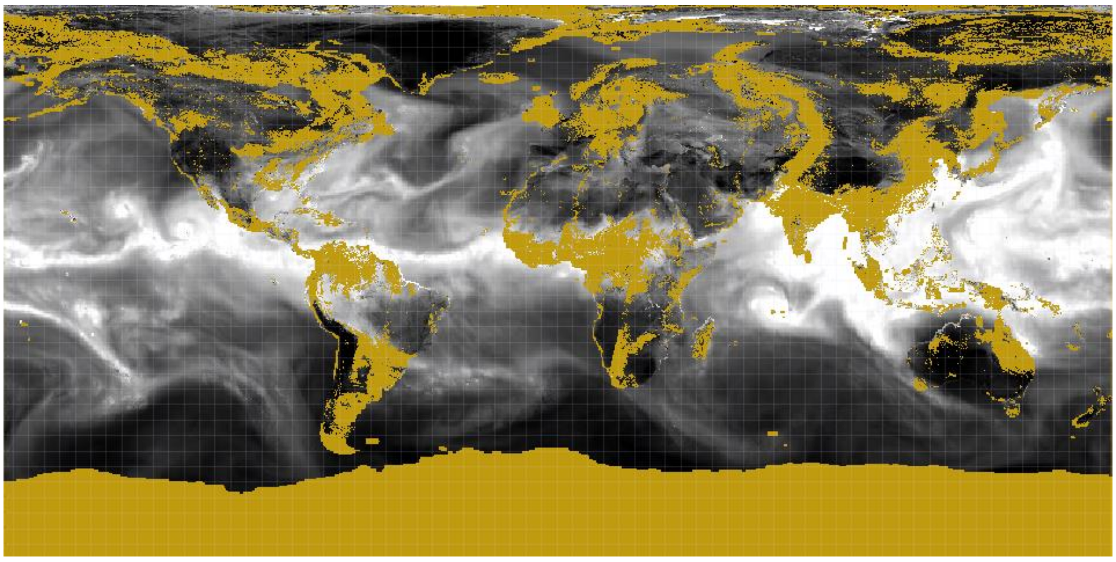


Figure 4-7: OLCI/MODIS/HOAPS TCWV L3 daily merge for July 15th, 2016 (greyscale, 0–70 kg m^{-2}) generated during WV_cci Phase 1. Yellow indicates no data.

4.1.2.2 Monthly products

The TCWV L3 monthly retrieval steps are illustrated in the right part of Figure 4-4. These TCWV L3 monthly products are retrieved from another temporal aggregation of all daily L3 products for a given calendar month. The aggregation rules are explained in the PUG [42]. The product content (variables, attributes) is the same as for the daily products, except that no quality flag is provided with the monthly products. As discussed in more detail in the SVR [43], the monthly aggregation leads to a significantly better global TCWV coverage. An example is shown in Figure 4-8.

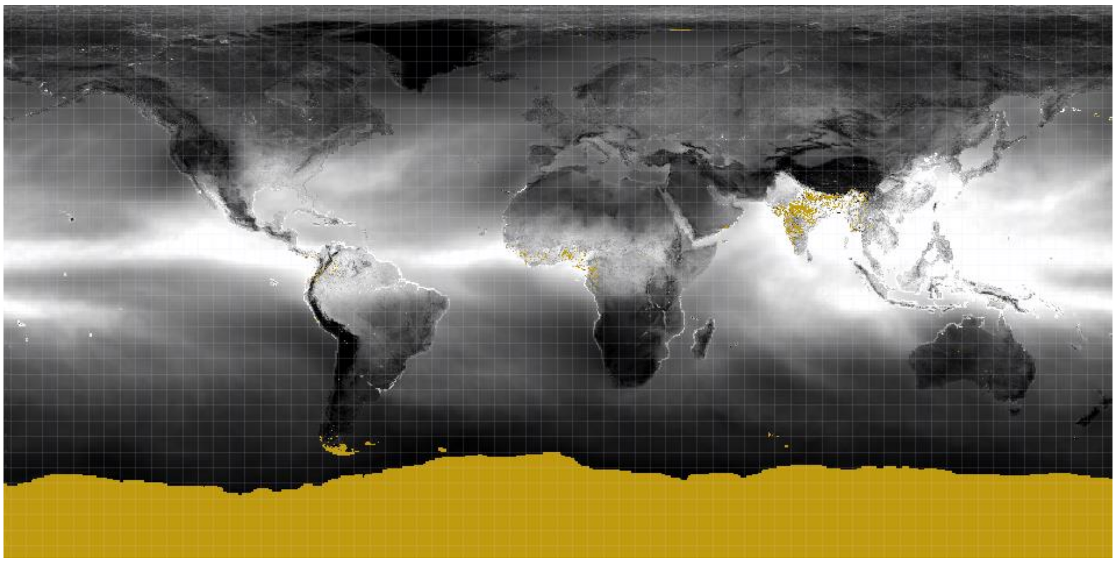


Figure 4-8: OLCI/MODIS/HOAPS TCWV L3 monthly merge for July 2016 (greyscale, 0–70 kg m^{-2}) generated during WV_cci Phase 1. Yellow indicates no data.

4.1.3 TCWV L3 high-resolution processing

For specific case studies during Phase 2, regional TCWV high-resolution (0.01°) products will be generated for sub-periods and for selected areas.

At compilation time of this SSD v4, four ROIs in three areas are favoured (Figure 4-9). Sensors and time intervals still need to be defined in more detail. A first test setup for high resolution processing (0.01deg) has been implemented on Calvalus. Basically, the existing L3 processing chain can be re-used with just a few modifications (mainly the ingestion of the regional subsetting).

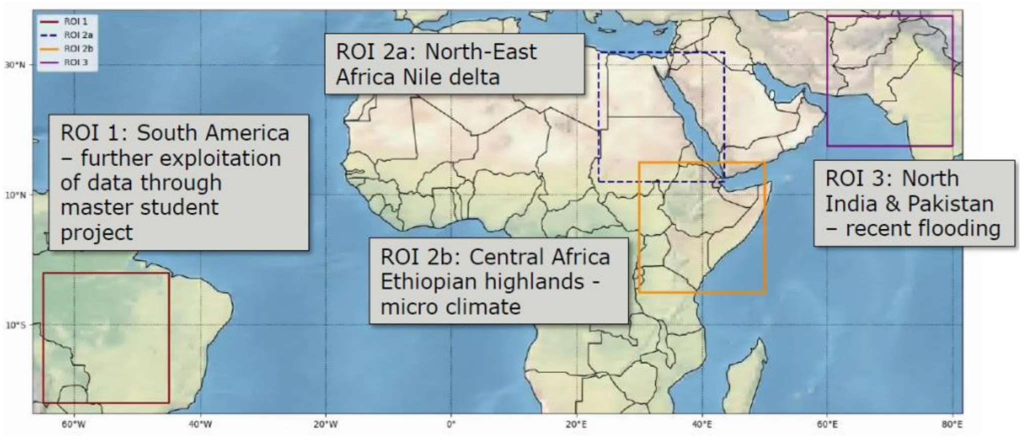

ROI 1
ROI 2a
ROI 2b
ROI 3
ROI 2a: North-East Africa Nile delta
ROI 1: South America – further exploitation of data through master student project
ROI 3: North India & Pakistan – recent flooding
ROI 2b: Central Africa Ethiopian highlands - micro climate
30°N
10°N
10°S
60°W
40°W
20°W
0°
20°E
40°E
60°E
80°E

Figure 4-9: Regions of interest considered for TCWV high-resolution processing in WV_cci Phase 2.

For illustration, Figure 4-10 shows a TCWV high-resolution daily product (MERIS, 20040531) for a 30x30° area over North Africa. Two MERIS swaths from this day partially intersect with this area.

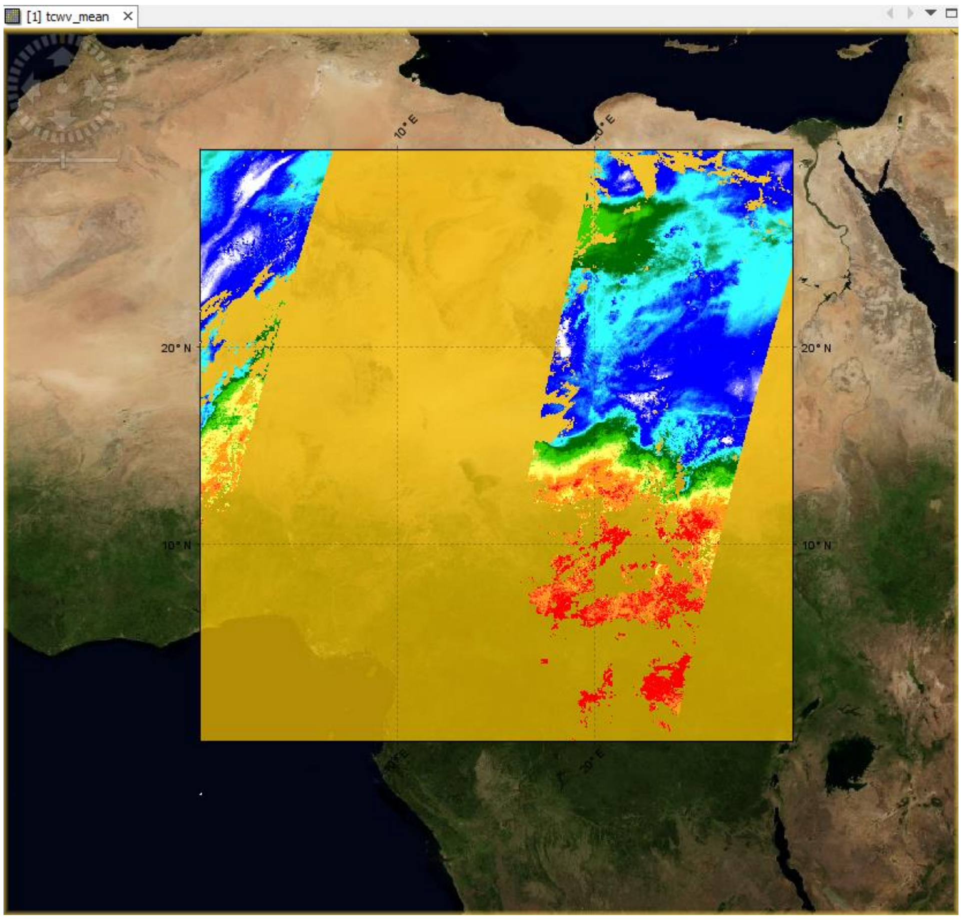


Figure 4-10: TCWV high-resolution daily product (MERIS, 20040531) for a 30x30° area over North Africa.

4.1.4 TCWV production from OLCI/SLSTR Synergy data

In the frame of a feasibility study, TCWV products will be generated from OLCI/SLSTR Synergy input data. This task is foreseen for year 2 of Phase 2 and will be described in more detail (temporal/spatial coverage, specification of required input data, processing setup) in a later version of this SSD.

4.2 VRWV Processing System

The VRWV processing is the responsibility of UoR and consists of two systems for the generation of two different VRWV products at level 3, the stratospheric zonal monthly mean and three-dimensional prototype monthly mean data WV_cci CDR-3 (see Section 4.2.1) and CDR-4 (see Section 4.2.2), respectively. CDR-3 and CDR-4 rely on a range of processed L2 and L3 input datasets from data providers as specified in the DARD [3]. See Figure 4-11 for details.

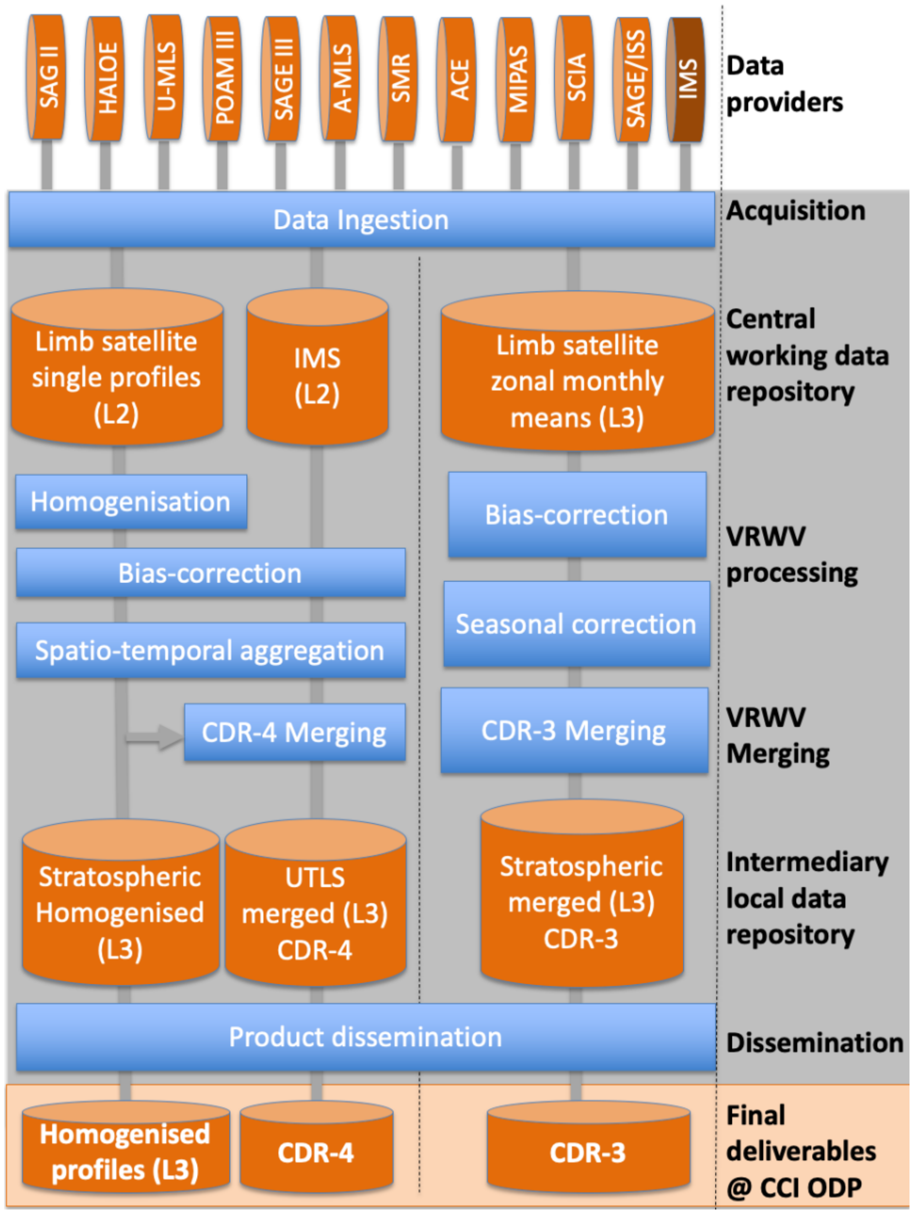


Figure 4-11: System definition of the VRWV processing systems for CDR-3 and CDR-4. Updated from [5].

4.2.1 VRWV CDR-3 processing

Figure 4-11 provides the overview of the processing chains for the generation of the monthly CDR-3 product. The CDR-3 processing starts with the ingestion of L3 SPARC Data Initiative climatologies as input. These include the data from the limb sounders SAGE II, HALOE, UARS-MLS, POAM III, SAGE III, SMR, SCIAMACHY, MIPAS, Aura

MLS, ACE-FTS, ACE-MAESTRO, and SAGE III/ISS. For the bias-correction, specified dynamics simulations from the Canadian Middle Atmosphere Model, a chemistry-climate model participating in the IGAC/SPARC Chemistry-Climate Model Initiative, are used. The VRWV CDR-3 processing is done on the RACC, but could likewise be performed on a powerful notebook. The various components of the flows are described in more detail in the following sections.

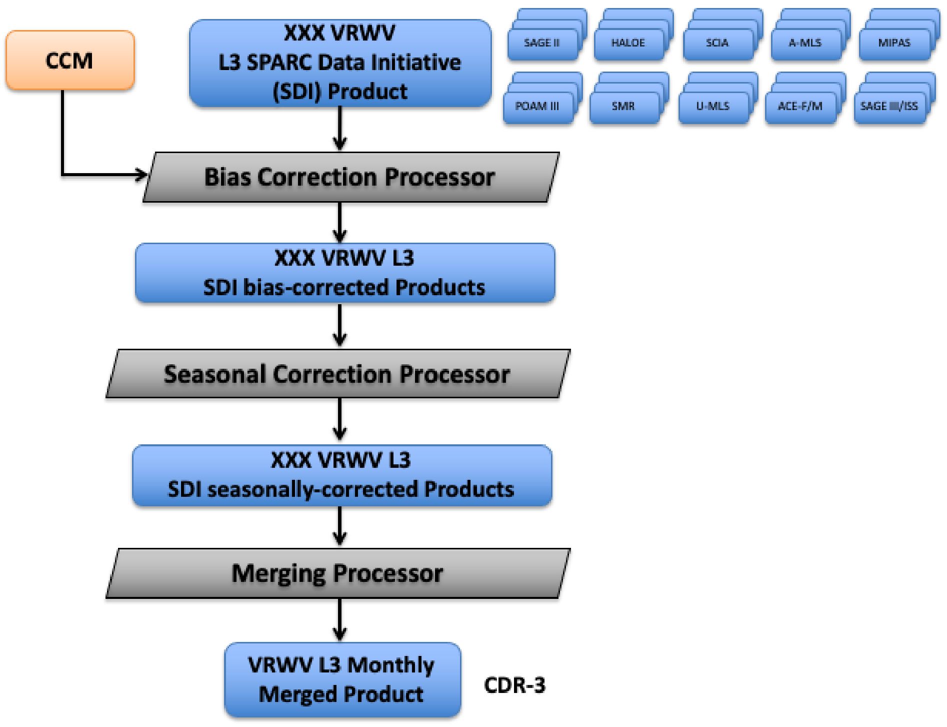


Figure 4-12 VRWV CDR-3 processing chain. SCIA stands for SCIAMACHY, U/A-MLS for UARS-/Aura-MLS, ACE-F/M for ACE-FTS/ACE-MAESTRO, and SAGE III/ISS for SAGE III on Meteor-3M and ISS, respectively. CCM stands for Chemistry–Climate Model. See text for further details.

4.2.1.1 Data acquisition

The input data used for VRWV CDR-3 production include:

- SPARC Data Initiative L3 zonal monthly mean products.
- Stratospheric water vapour fields from chemistry–climate model specified dynamics simulations.
- Stratospheric water vapour data from earlier satellite instruments

The SPARC Data Initiative L3 products

The SPARC Data Initiative L3 zonal monthly mean products [44] are produced by the data providers from L2 observations (national agencies and institutions) and are being made available via the Zenodo data archive [45]. Data after 2019 are available via the data providers in the same SPARC Data Initiative format. The different instruments' zonal monthly mean data products are available over different time periods as specified in [44]. All products have been submitted by the data providers to UoR (the WV_cci) and now available as static ancillary input datasets in the WV_cci workspace on RACC (top level depicted in Figure 4-12).

Chemistry-climate model fields

The chemistry–climate model water vapour fields from specified dynamics simulations from different models can be accessed via the CEDA data archive, as part of the Chemistry-Climate Model Initiative's phase 2 data collection. All products have been transferred once from CEDA and are now available as static ancillary input datasets in the WV_cci workspace on the RACC (top level depicted in Figure 4-12).

Stratospheric water vapour data from earlier satellite instruments

The stratospheric water vapour retrievals from NIMBUS-7 LIMS, SAMS, and UARS iSAMS are downloaded from the respective data archives from the data proviers and saved on the personal laptop for an initial assessment and intercomparison of data quality. These data can potentially extend the CDR-3 further into the past.

4.2.1.2 Bias correction processor

This bias correction processor uses the water vapour fields of a chemistry-climate model as a transfer function to infer biases from the L3 satellite input data and adjust them to a reference derived from a pre-selected set of instruments. The bias correction step is applied to all SPARC Data Initiative L3 input data. The detailed description of the algorithm is available in the ATBD [4] and [46]. The bias-corrected L3 input data are stored as intermediate products in the WV_cci workspace on the RACC.

4.2.1.3 Seasonal correction processor

This seasonal correction processor is applied to a subset of the L3 satellite input datasets only, and is generally necessary only at some of the latitude-pressure grid points. The detailed description of the algorithm is available in the ATBD [4] and [46]. The seasonally corrected L3 input data are stored as intermediate products in the WV_cci workspace on the RACC.

4.2.1.4 Merging of L3 products

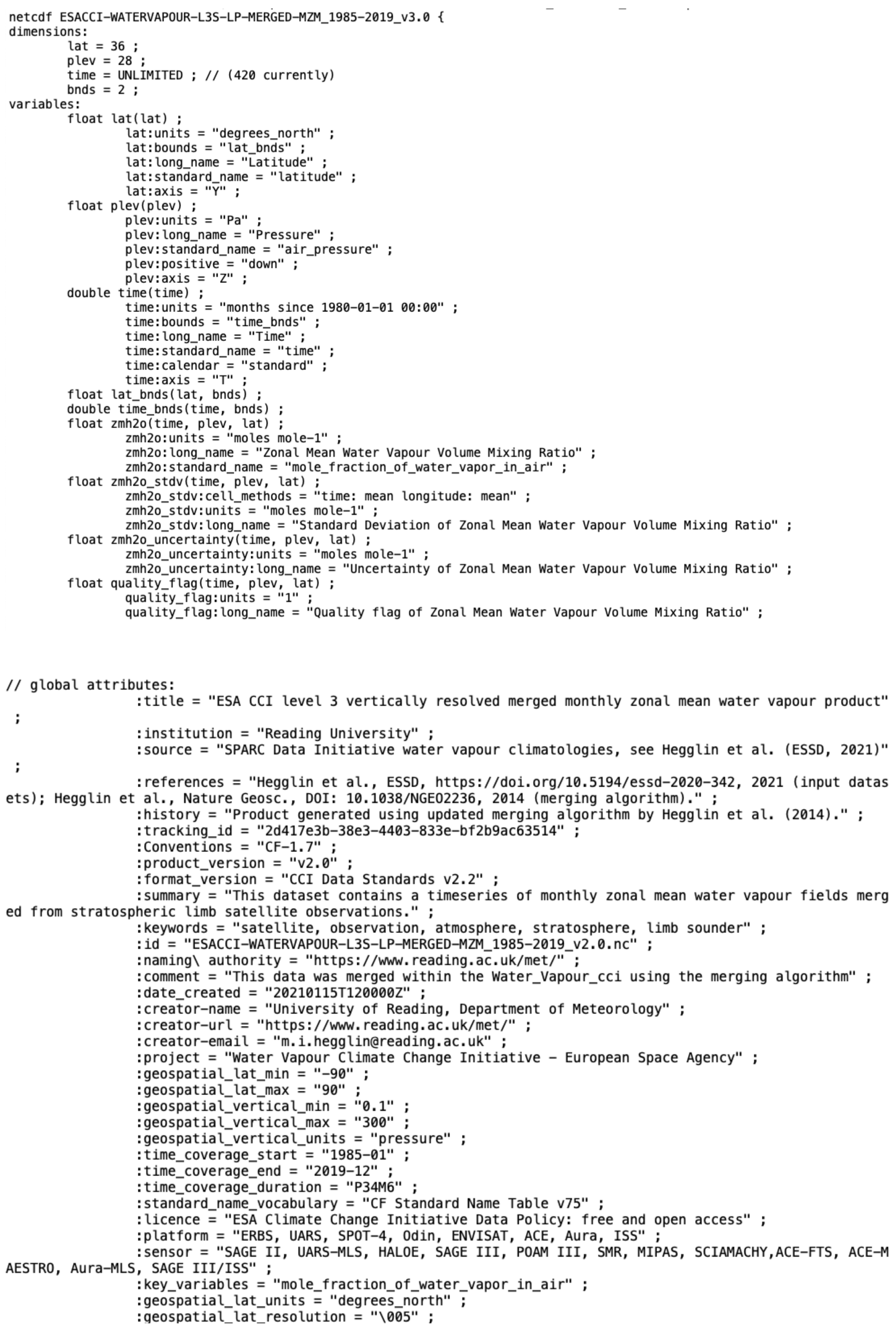
The final processing step for the montly zonal mean L3 CDR-3 is to merge the zonal monthly mean bias- (and later seasonally) corrected SPARC Data Initiative L3 products from all available instruments to obtain a long-term product with a minimum of data gaps and an improved overall bias. The following rules are applied in the merging process:

- If VRWV values from a given SPARC Data Initiative input file at a given latitude and altitude show a bias greater than a specified threshold, based on the biases of other datasets with respect to the model at this grid point, the value is dismissed and not included in the merging.
- If VRWV from the SPARC Data Initiative input files is not available (due to temporal and spatial sampling gaps), set VRWV to NaN.

The content of a montly zonal mean VRWV L3 product after this final step is shown in Figure 4-13.

4.2.1.5 NetCDF product formatting

As can be seen from Figure 4-12, the next step is a post-processing procedure where all VRWV CDR-3 L3 products are made compliant to CF- and CCI Data Standards. This includes adding a TIME dimension and variable to the NetCDF products, use of proper variable names, and providing all expected variable attributes as well as a comprehensive list of global attributes. The content of a merged monthly VRWV L3 end product after this final step is shown in Figure 4-13. The product geolocation information is given through the 1D variables 'lat' and vertical levels are given through 1D variable 'pressure levels'. The two-dimensional VRWV profile is provided in the variable ‘h2ovmr’ in the unit of parts per million per volume (ppmv). The statistical quantities and uncertainty terms are provided as zmh2o_stdv and zmh2o_uncertainty, respectively.



```
netcdf ESACCI-WATERVAPOUR-L3S-LP-MERGED-MZM_1985-2019_v3.0 {
dimensions:
        lat = 36 ;
        plev = 28 ;
        time = UNLIMITED ; // (420 currently)
        bnds = 2 ;
variables:
        float lat(lat) ;
                lat:units = "degrees_north" ;
                lat:bounds = "lat_bnds" ;
                lat:long_name = "Latitude" ;
                lat:standard_name = "latitude" ;
                lat:axis = "Y" ;
        float plev(plev) ;
                plev:units = "Pa" ;
                plev:long_name = "Pressure" ;
                plev:standard_name = "air_pressure" ;
                plev:positive = "down" ;
                plev:axis = "Z" ;
        double time(time) ;
                time:units = "months since 1980-01-01 00:00" ;
                time:bounds = "time_bnds" ;
                time:long_name = "Time" ;
                time:standard_name = "time" ;
                time:calendar = "standard" ;
                time:axis = "T" ;
        float lat_bnds(lat, bnds) ;
        double time_bnds(time, bnds) ;
        float zmh2o(time, plev, lat) ;
                zmh2o:units = "moles mole-1" ;
                zmh2o:long_name = "Zonal Mean Water Vapour Volume Mixing Ratio" ;
                zmh2o:standard_name = "mole_fraction_of_water_vapor_in_air" ;
        float zmh2o_stdv(time, plev, lat) ;
                zmh2o_stdv:cell_methods = "time: mean longitude: mean" ;
                zmh2o_stdv:units = "moles mole-1" ;
                zmh2o_stdv:long_name = "Standard Deviation of Zonal Mean Water Vapour Volume Mixing Ratio" ;
        float zmh2o_uncertainty(time, plev, lat) ;
                zmh2o_uncertainty:units = "moles mole-1" ;
                zmh2o_uncertainty:long_name = "Uncertainty of Zonal Mean Water Vapour Volume Mixing Ratio" ;
        float quality_flag(time, plev, lat) ;
                quality_flag:units = "1" ;
                quality_flag:long_name = "Quality flag of Zonal Mean Water Vapour Volume Mixing Ratio" ;



// global attributes:
                :title = "ESA CCI level 3 vertically resolved merged monthly zonal mean water vapour product"
 ;
                :institution = "Reading University" ;
                :source = "SPARC Data Initiative water vapour climatologies, see Hegglin et al. (ESSD, 2021)"
 ;
                :references = "Hegglin et al., ESSD, https://doi.org/10.5194/essd-2020-342, 2021 (input datas
ets); Hegglin et al., Nature Geosc., DOI: 10.1038/NGEO2236, 2014 (merging algorithm)." ;
                :history = "Product generated using updated merging algorithm by Hegglin et al. (2014)." ;
                :tracking_id = "2d417e3b-38e3-4403-833e-bf2b9ac63514" ;
                :Conventions = "CF-1.7" ;
                :product_version = "v2.0" ;
                :format_version = "CCI Data Standards v2.2" ;
                :summary = "This dataset contains a timeseries of monthly zonal mean water vapour fields merg
ed from stratospheric limb satellite observations." ;
                :keywords = "satellite, observation, atmosphere, stratosphere, limb sounder" ;
                :id = "ESACCI-WATERVAPOUR-L3S-LP-MERGED-MZM_1985-2019_v2.0.nc" ;
                :naming\ authority = "https://www.reading.ac.uk/met/" ;
                :comment = "This data was merged within the Water_Vapour_cci using the merging algorithm" ;
                :date_created = "20210115T120000Z" ;
                :creator-name = "University of Reading, Department of Meteorology" ;
                :creator-url = "https://www.reading.ac.uk/met/" ;
                :creator-email = "m.i.hegglin@reading.ac.uk" ;
                :project = "Water Vapour Climate Change Initiative - European Space Agency" ;
                :geospatial_lat_min = "-90" ;
                :geospatial_lat_max = "90" ;
                :geospatial_vertical_min = "0.1" ;
                :geospatial_vertical_max = "300" ;
                :geospatial_vertical_units = "pressure" ;
                :time_coverage_start = "1985-01" ;
                :time_coverage_end = "2019-12" ;
                :time_coverage_duration = "P34M6" ;
                :standard_name_vocabulary = "CF Standard Name Table v75" ;
                :licence = "ESA Climate Change Initiative Data Policy: free and open access" ;
                :platform = "ERBS, UARS, SPOT-4, Odin, ENVISAT, ACE, Aura, ISS" ;
                :sensor = "SAGE II, UARS-MLS, HALOE, SAGE III, POAM III, SMR, MIPAS, SCIAMACHY,ACE-FTS, ACE-M
AESTRO, Aura-MLS, SAGE III/ISS" ;
                :key_variables = "mole_fraction_of_water_vapor_in_air" ;
                :geospatial_lat_units = "degrees_north" ;
                :geospatial_lat_resolution = "\005" ;
```

Figure 4-13: Panoply product explorer: merged VRWV monthly zonal mean L3 final product (prototype v0).

4.2.2 VRWV CDR-4 processing

Figure 4-14 shows a schematic overview of the VRWV CDR-4 processing chain, consisting of the processing and generation of the monthly mean CDR-4 product. The input data used in the CDR-4 processing include L2 data from Aura-MLS, MIPAS, and IMS and the ancillary variables in the Derived Meteorological Products (DMPs) from NASA JPL. For the bias correction processing, the observations of WV profiles from balloon-borne hygrometer are used. In Phase 2, more WV profiles from other data sources will be tested as the bias correction reference, i.e. profiles from ACE-FTS and chemistry-climate models. The VRWV CDR-4 processing is completely done on the RACC. The various components of the flows are described in more detail in the following sections.

4.2.2.1 Data acquisition

The input data used for VRWV CDR-4 production include:

- Aura-MLS L2 v5.1 VRWV products [47]
- WAVAS_SAHAR dataset
- IMS L2 VRWV Product
- Balloon-borne hygrometer VRWV profiles [37, 38]
- VRWV profiles from chemistry-climate model simulations
- DMP ancillary variables

Aura-MLS L2 VRWV

The Aura-MLS L2 VRWV products are available directly from the data website and stored at RACC for the full archive from 08/2004 to present. This dataset has been used widely by a variety of projects.

WAVAS_SAHAR dataset

The WAVAS_SAHAR database offers harmonised datasets of WV profiles for multiple instruments with specific vertical ranges on both, fixed pressure and fixed altitude grids. The geolocations of the WV profiles are the same as the original retrieved WV profiles for each instrument. The detailed description of the dataset can be found in the DARD [4]. The data used from this dataset include MIPAS, ACE-FTS, and ACE-MAESTRO. The data has been stored at RACC for processing.

IMS L2 VRWV product

The IMS L2 VRWV product from 07/2007 to present is available at CEDA and also stored on the RACC. The product can directly be accessed from the RACC for processing.

Balloon-borne Hygrometer VRWV profiles

The balloon-borne hygrometer WV profiles are available on the RACC and can be directly accessed for processing.

VRWV profiles from chemistry-climate model simulations

The VRWV profiles from chemistry-climate model with specified dynamics from different models can be accessed via the CEDA data archive. Currently, the simulations from CMAM (Canadian Middle Atmosphere Model) are used to test the feasibility of applying the same approach as successfully employed for data merging of CDR-3 into the production of CDR-4.

DMP ancillary variables

The derived meteorological products are provided by NASA JPL for each instrument with multiple ancillary variables. The following ancillary variables are used as input:

- Geopotential height for each profile
- Tropopause height for each profile

4.2.2.2 Bias correction processor

The bias correction step is applied to all L2 input data. Same as in Phase 1, the current bias correction processor using *in situ* VRWV profiles from FPH/CFH observations will continue to assess the biases of the L2 satellite data, which then are used to reduce the bias in the L2 data for the extended time periods from 2007 to 2023. The detailed description of the algorithm is available in the ATBD [4]. After the bias correction step, the original L2 data will be replaced with the bias-corrected L2 data. Note that the bias correction will not be applied to the L2 VRWV data in the stratosphere, defined as above 100 hPa in this study. The corrected VRWV L2 products are not part of VRWV CDR-4 dataset and will not be made available to users. In Phase 2, new bias correction processor based on other possible reference datasets (chemistry-climate model simulations) will be updated after careful assessment.

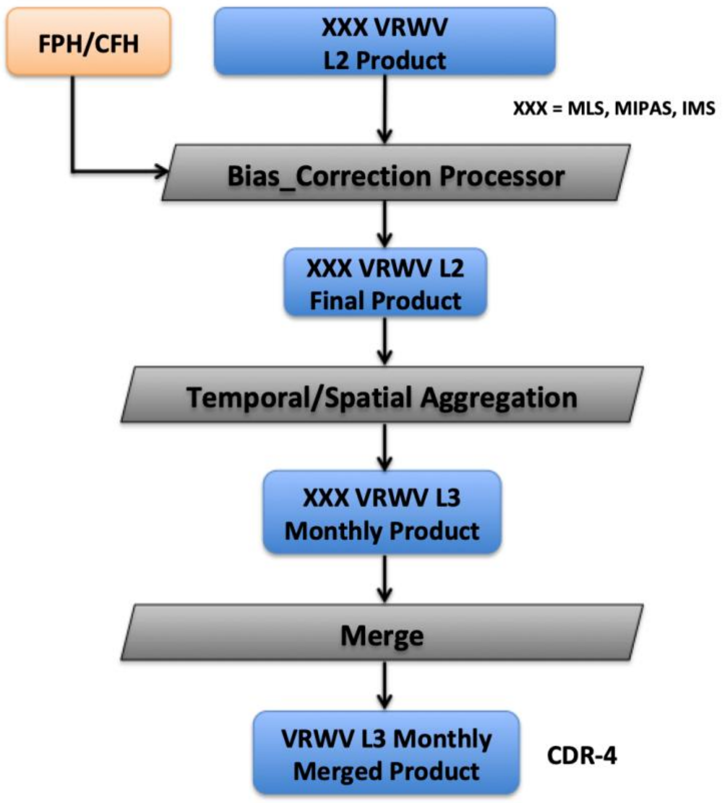

FPH/CFH
XXX VRWV L2 Product
XXX = MLS, MIPAS, IMS
Bias_Correction Processor
XXX VRWV L2 Final Product
Temporal/Spatial Aggregation
XXX VRWV L3 Monthly Product
Merge
VRWV L3 Monthly Merged Product
CDR-4

Figure 4-14: VRWV CDR-4 processing chains. See text for details.

4.2.2.3 Temporal and spatial aggregation

The corrected VRWV L2 products obtained from the previous step for each instrument (MLS, MIPAS, or IMS) are taken as input for a temporal and spatial level 3 aggregation. The aggregation is performed for each month in 2007–2023 with a horizontal resolution of 5 degrees by 5 degrees in latititude and longitude. In the vertical dimension, the L2 VRWV profiles are interpolated to pressure levels as specificed in the PSD [2]. This results in three VRWV L3 products for all available instruments (MLS, MIPAS, and IMS).

4.2.2.4 Merging of L3 products

The last processing step for the monthly L3 products is to merge all the VRWV L3 products obtained from Section 4.2.2.2 into a global monthly VRWV prototype product from 2007 to 2023. The following merging rules are applied:

- At and above 100 hPa (lower stratosphere), only use original VRWV L3 products from MLS and MIPAS before bias correction.

- Between 100 hPa and 300 hPa, include all bias corrected VRWV L3 products into the merged product.
- Below 300 hPa (troposphere), only use original VRWV L3 products from IMS before bias correction.

The product geolocation information is given through 1D variables 'lat' and 'lon' and vertical levels are given through 1D variable 'pressure levels'. The three-dimensional VRWV profile is provided with the variable ‘vmrh2o’ in the unit of ppmv. The statistical quantities and error terms (vmrh2o_stdv and vmrh2o_uncertainty) are also included in the products.

4.2.2.5 NetCDF product formatting

As for CDR-3, the next step is a post-processing procedure where all VRWV CDR-4 L3 products are made compliant to CF- and CCI Data Standards. This includes adding a TIME dimension and variable to the NetCDF products, use of proper variable names, and providing all expected variable attributes as well as a comprehensive list of global attributes.

5. REQUIREMENTS TRACEABILITY

This section traces input technical requirements (TR-xx) listed in the WV_cci Phase 2 Proposal [13] to sections within this document (§). For completeness, all 39 requirements are listed in the following table. However, only the TRs which are at least partially related to the system that generates the WV products (thus a system requirement (SR), marked in blue) are relevant in the context of this SSD document.

Requirement	Description in brief	Reference
TR-1	GCOS requirements	not a SR
TR-2	GCOS requirements, users	not a SR
TR-3	Publication of results	not a SR
TR-4	Ensure that work is complementary to other water vapour ECV development.	not a SR
TR-5	Generation of TCWV products over land	§4.1
TR-6	Merge of global TCWV products (CDR-2)	§4.1.2
TR-7	Upper tropospheric and stratospheric zonal mean water vapour product (CDR-3)	§4.2.1
TR-8	3D UTLS-focussed product (CDR-4)	§4.2.2
TR-9	Consistent stratospheric products	not a SR
TR-10	Requirements from scientists	not a SR
TR-11	Application in climate science	not a SR
TR-12	Quantitative requirements	not a SR
TR-13	Requirements for uncertainties	not a SR
TR-14	Intercomparison of results	not a SR

Requirement	Description in brief	Reference
TR-15	Provide uncertainties in products	§4.1.2
TR-16	Algorithm improvements, use updated and extended input data, prototype dataset from S3 Synergy data	§4.1.1
TR-17	Development and delivery of a harmonised multi-mission ECV data set together with other species like O3, aerosol, and GHGs	not a SR
TR-18	Liaise with the SPARC initiatives and Ozone_cci and Aerosol_cci projects for intercomparison	not a SR
TR-19	Assessment on data quality from earlier satellite instruments to potentially extend the CDR-3 further into the past	§4.2.1
TR-20	Devolop and improve the harmonisation/merging approach for CDR-3	§4.2.1
TR-21	New/alternative merging techniques to improve the prototype CDR-4	§4.2.2
TR-22	Extend CDR-4 by adapting the algorithm to ingest MetOp-B and C: IASI, MHS, AMSU	§4.2.2
TR-23	Comprehensive dataset for validation	not a SR
TR-24	Validation shall quantify accuracy	not a SR
TR-25	Inhomogeneities in data records	not a SR
TR-26	GCOS stability requirement	not a SR
TR-27	Clear sky bias	not a SR
TR-28	Comparison with ECVs from external organisations	not a SR
TR-29	Comparison with climate models	not a SR

Requirement	**Description in brief**	**Reference**
TR-30	Provide automated high-performance processing chain	§3, §4
TR-31	Implement a continuous cycle of improvements	§4
TR-32	Deliver four new ECV data records	§4
TR-33	Use latest available reprocessings of input data, do not exclude sensor overlaps	§4.1.1
TR-34	Liaise with users internal and external to CCI project and programme.	not a SR
TR-35	Include at least three user case studies.	not a SR
TR-36	Investigate consistency of variability and trends.	not a SR
TR-37	Interaction with other relevant science bodies.	not a SR

APPENDIX 1: REFERENCES

[1]: World Meteorological Organization: Essential Climate Variables. © 2019 World Meteorological Organization (WMO).

[2]: ESA CCI Water Vapour: ESA CCI Water Vapour: Product Specification Document. M. Schröder, M. Hegglin, H. Ye, O. Danne, J. Fischer, A. Laeng, R. Siddans, C. Sioris, G. Stiller, and K. Walker. Issue 3.2, 27 July 2021.

[3]: ESA CCI Water Vapour: Data Access Requirement Document. M. Schröder, M. Hegglin, U. Falk, H. Brogniez, J. Fischer, D. Hubert, A. Laeng, R. Siddans, C. Sioris, G. Stiller, T.Trent, K. Walker, O.Danne, and H.Ye. Issue 3.2, 27 July 2021.

[4]: ESA CCI Water Vapour: Algorithm Theoretical Basis Document. Part 1: MERIS-MODIS-OLCI L2 Products. J. Fischer and R. Preusker, Issue 2.1, 21 January 2021.

[5] ESA CCI Water Vapour: Technical Proposal. M. Hegglin and the WV ECV Consortium. Ref: KPT90658, 27 October 2017.

[6]: ESA CCI Fire: System Specification Document. T. Storm, M. Böttcher, and G. Kirches, Version 1.4, 30 September 2017.

[7]: ESA CCI Land Cover (Phase 1): System Specification Document. M. Böttcher, O. Danne, S. Bontemps, and P. Defourny, Version 0.8, 27 November 2012.

[8]: ESA Climate Change Initiative Extension (CCI+) Phase 1 – New Essential Climate Variables – Statement of Work. Annex A: Water Vapour ECV (Water_Vapour_cci). Ref. ESA-CCI-PRGM-EOPS-SW-17-0032, Issue 1, Rev. 3, 22 August 2017.

[9]: Eaton, B., Gregory, J., Drach, B., Taylor, K., Hankin, S., Blower, J., Caron, J., Signell, R., Bentley, P., Rappa, G., Höck, H., Pamment, A., Juckes, M., and M. Raspaud, 2017: NetCDF Climate and Forecast (CF) Metadata Conventions. Version 1.7. http://cfconventions.org/

[10]: ESA Climate Office: CCI Data Standards. Reference CCI-PRGM-EOPS-TN-13-0009, Issue 2.0, date of issue 17/09/2018.

[11]: ESA Climate Office: CCI Data Standards. Reference CCI-PRGM-EOPS-TN-13-0009, Issue 2.1, 2 August 2019.

[12]: ESA CCI Water Vapour: System Requirements Document. O. Danne, M. Hegglin and H. Ye, Issue 3.0, 19 August 2021.

[13] ESA CCI Water Vapour: Detailed Proposal Water Vapour_cci Phase 2. M. Hegglin and the WV ECV Consortium. CCN 3 to ESA CONTRACT 4000123554/18/I-NB - Water Vapour_cci. Ref: ESA-EOP-SC-AMT-2021-22, 1 February 2022.

[14]: IT4Innovations National Supercomputing Center. VSB – Technical University of Ostrava, Czech Republic. https://www.it4i.cz/?lang=en

[15]: JASMIN - Petascale storage and cloud computing for big data challenges in environmental science. http://www.jasmin.ac.uk/

[16]: Google Earth Engine - A planetary-scale platform for Earth science data & analysis. https://earthengine.google.com/

[17]: CEDA - Centre for Environmental Data Analysis. https://www.ceda.ac.uk/

[18]: STFC - Science and Technology Facilities Council. https://www.scd.stfc.ac.uk/

[19]: NERC - Natural Environment Research Council. https://www.ukri.org/councils/nerc/

[20]: RAL Space Home Page. https://www.ralspace.stfc.ac.uk/

[21]: JASMIN - Evolution. https://jasmin.ac.uk/about/evolution/

[22]: JASMIN - Introduction to Group Workspaces. https://help.jasmin.ac.uk/article/199-introduction-to-group-workspaces

[23]: JASMIN - Help Documentation. https://help.jasmin.ac.uk/

[24]: SLURM Scheduler Overview. https://help.jasmin.ac.uk/article/4880-batch-scheduler-slurm-overview

[25]: LOTUS Overview. https://help.jasmin.ac.uk/article/5004-lotus-overview

[26]: SLURM Quick Reference. https://help.jasmin.ac.uk/article/4891-lsf-to-slurm-quick-reference

[27]: IBM Platform LSF documentation.
https://www.ibm.com/support/knowledgecenter/en/SSETD4/product_welcome_platform_lsf.htm

[28]: RACC introduction. https://research.reading.ac.uk/act/knowledgebase/racc-introduction/

[29]: Boettcher, M., Zuehlke, M., and H. Permana; Cal/Val and User Services – Calvalus. Software User Manual. Brockmann Consult, Version 1.7, 19 December 2017.

[30]: Batch Computing on LOTUS. CEDA help documentation,
https://help.ceda.ac.uk/category/107-batch-computing-on-lotus

[31]: MERIS product handbook. European Space Agency, Issue 2.1, 24 October 2006.
http://envisat.esa.int/pub/ESA_DOC/ENVISAT/MERIS/meris.ProductHandbook.2_1.pdf

[32]: Farr, T. G., et al., The Shuttle Radar Topography Mission, Rev. Geophys., 45, RG2004, doi:10.1029/2005RG000183. (2007).

[33]: Shuttle Radar Topography Mission web site. https://www2.jpl.nasa.gov/srtm/index.html

[34]: Berrisford, P, Dee, D.P., Poli, P, Brugge, R, Fielding, M, Fuentes, M, Kållberg, P.W., Kobayashi, S, Uppala, S, and A. Simmons, 2011: The ERA-Interim archive Version 2.0. ERA Report Series, Document Number 1, ECMWF, Reading, UK.

[35]: ECMWF Public Datasets Archive web access.
https://confluence.ecmwf.int/display/WEBAPI/Access+ECMWF+Public+Datasets

[36]: MODIS Characterization Support Team, 2009: MODIS Level 1B Product User's Guide. NASA, Goddard Space Flight Center.
https://ccplot.org/pub/resources/Aqua/MODIS%20Level%201B%20Product%20User%20Guide.pdf

[37]: MODIS Science Data Support Team (SDST), 2015: MODIS/Terra Calibrated Radiances 5-Min L1B Swath 1km Product (MOD021KM). NASA MODIS Adaptive Processing System, Goddard Space Flight Center. Dataset DOI:
http://dx.doi.org/10.5067/MODIS/MOD021KM.006

[38]: Ackerman S. A., and R. Frey, 2015: MODIS Atmosphere L2 Cloud Mask Product (35_L2). NASA MODIS Adaptive Processing System, Goddard Space Flight Center. Dataset DOI: http://dx.doi.org/10.5067/MODIS/MOD35_L2.006.

[39]: CEDA Archive: NEODC database. http://data.ceda.ac.uk/neodc

[40]: CEDA Archive: MODIS temporal coverage in NEODC database. http://data.ceda.ac.uk/neodc/modis/metadata/temporal_coverage/

[41]: Sentinel-3 OLCI Technical Guide: Level-1 Algorithms and Products. https://sentinels.copernicus.eu/web/sentinel/technical-guides/sentinel-3-olci/level-1-algorithms-products

[42]: ESA CCI Water Vapour: Product User Guide. O. Danne, M. Hegglin, H. Ye, M. Schröder, R. Preusker, J. Fischer, C. Brockmann, Issue 2.0, 11 August 2021.

[43]: ESA CCI Water Vapour: System Verification Report. O.Danne, M. Hegglin and H. Ye, Issue 3.0, 19 August 2021.

[44] Hegglin, M. I., S. Tegtmeier, and the SPARC Data Initiative Team, SPARC Data Initiative: Overview, *ESSD*, submitted.

[45] Hegglin, M. I., S. Tegtmeier, J. Anderson, A. E. Bourassa, S. Brohede, D. Degenstein, L. Froidevaux, B. Funke, J. Gille, Y. Kasai, E. Kyrölä, J. Lumpe, D. Murtagh, J. L. Neu, K. Pérot, E. Remsberg, A. Rozanov, M. Toohey, T. von Clarmann, K. A. Walker, H.-J. Wang, R. Damadeo, R. Fuller, G. Lingenfelser, C. Roth, N. J. Ryan, C. Sioris, L. Smith, and Weigel, K. (2020). SPARC Data Initiative monthly zonal mean composition measurements from stratospheric limb sounders (1978-2018) (Version p01) [Data set]. Earth System Science Data (ESSD). Zenodo. http://doi.org/10.5281/zenodo.4265393.

[46] Hegglin, M. I., D. Plummer, J. Scinocca, T. G. Shepherd, J. Anderson, L. Froidevaux, B. Funke, D. Hurst, A. Rozanov, J. Urban, T. v. Clarmann, K. A. Walker, R. Wang, S. Tegtmeier, and K. Weigel, Variation of stratospheric water vapour trends with altitude from merged satellite data, Nature Geoscience, doi: 10.1038/NGEO2236, 2014.

[47]: Aura MLS Level 2 and 3 data quality and description document. NASA, Jet Propulsion Laboratory. Version4.2x-4.0. https://mls.jpl.nasa.gov/data/v4-2_data_quality_document.pdf

[48] Hurst, D. F., S. J. Oltmans, H. Vömel, K. H. Rosenlof, S. M. Davis, E. A. Ray, E. G. Hall, and A. F. Jordan (2011), Stratospheric water vapor trends over Boulder, Colorado: Analysis of the 30 year Boulder record, J. Geophys. Res., 116, D02306, doi:10.1029/2010JD015065.

APPENDIX 2: GLOSSARY

Term	Definition
ACE-FTS	Atmospheric Chemistry Experiment Fourier Transform Spectrometer
ACE-MAESTRO	Atmospheric Chemistry Experiment Measurement of Aerosol Extinction in the Stratosphere and Troposphere Retrieved by Occultation
AMSU	Advanced Microwave Sounding Unit
ATBD	Algorithm Theoretical Basis Document
AURA-MLS	Aura Microwave Limb Sounder
BC	Brockmann Consult
Calvalus	Cal/Val and User Services
CCI	Climate Change Initiative
CCM	Chemistry–Climate Model
CDO	Climate Data Operators
CDR	Climate Data Record
CEDA	Centre for Environmental Data Analysis
CEMS	Climate and Environmental Monitoring from Space
CF	Climate and Forecast
CMAM	Canadian Middle Atmosphere Model
CM SAF	Satellite Application Facility on Climate Monitoring
CPU	Central Processing Unit
CSA	Canadian Space Agency
DARD	Data Access Requirement Document
DMP	Derived Meteorological Products
DMSP	Defense Meteorological Satellite Program
DWD	Deutscher Wetterdienst (German Weather Service)
ECMWF	European Centre for Medium Range Weather Forecast
ECSAT	European Centre for Space Applications and Telecommunications
ECV	Essential Climate Variable
EO	Earth Observation
ERA	European Re-Analysis
ESA	European Space Agency

Term	**Definition**
FTP	File Transfer Protocol
GB	GigaByte
GCOS	Global Climate Observing System
GFS	Google File System
GISS	Goddard Institute for Space Studies
GFS	Google File System (GFS),
GPF	Graph Processing Framework
GWS	Group Workspace
HALOE	HALogen Occultation Experiment
HDFS	Hadoop Distributed File System
HOAPS	Hamburg Ocean Atmosphere Parameters and Fluxes
HPOS	High-Performance Object Storage
IASI	Infrared Atmospheric Sounding Interferometer
IGAC	International Global Atmospheric Chemistry
IMS	Infrared Microwave Sounder
IPCC	Intergovernmental Panel on Climate Change
JASMIN	Joint Analysis System Meeting Infrastructure
JPL	Jet Propulsion Laboratory
JSON	JavaScript Object Notation
KO	Kick-off
LIMS	Limb Infrared Monitor of the Stratosphere
LSF	Load Sharing Facility
LTS	Long Term Support
MERIS	Medium Resolution Imaging Spectrometer
MetOp	Meteorological Operational satellite
MHS	Microwave Humidity Sounder
MIPAS	Michelson Interferometer for Passive Atmospheric Sounding
MODIS	Moderate Resolution Imaging Spectroradiometer
NASA	National Aeronautics and Space Administration
NCAS	National Centre for Atmospheric Science
NEODC	NERC Earth Observation Data Centre
NERC	National Environment Research Council
NetCDF	Network Common Data Form
NIC	Network Interface Controller
NIR	Near Infrared

Term	**Definition**
NLDS	Near-Line Data Store
ODP	Open Data Portal
OLCI	Ocean and Land Colour Instrument
PB	PetaByte
PFS	Physical File System
POAM	Polar Ozone and Aerosol Measurement
PSD	Product Specification Document
PUG	Product User Guide
RACC	Reading Academic Computing Cluster
RAL	Rutherford Appleton Laboratory
ROI	Region of Interest
RR	Reduced Resolution
SAGE	Stratospheric Aerosol and Gas Experiment
SAGE III/ISS	Stratospheric Aerosol and Gas Experiment III on the International Space Station
SAMS	Stratospheric and Mesospheric Sounder
SCIAMACHY	SCanning Imaging Absorption spectroMeter for Atmospheric CartograpHY
SCRIP	Spherical Coordinate Remapping and Interpolation Package
SDK	Software development Kit
SE	Spectral Earth
SLSTR	Sea and Land Surface Temperature Radiometer
SLURM	Simple Linux Utility for Resource Management
SMR	Submillimeter wave Radiometer
SNAP	Sentinel Application Platform
SOF	Scale-out File System
SPARC	Stratosphere-Troposphere Processes and their Role in Climate
SoW	Statement of Work
SR	System Requirement
SRD	System Requirements Document
SRTM	Shuttle Radar Topography Mission
SSD (1)	Solid State Drive
SSD (2)	System Specification Document
SSM/I	Special Sensor Microwave/Imager

Term	Definition
SSMIS	Special Sensor Microwave Imager Sounder
STFC	Science and Technology Facilities Council
TB	TeraByte
TCWV	Total Column of Water Vapour
TOA	Top of Atmosphere
TR	Technical Requirement
UARS-MLS	Upper Atmosphere Research Satellite Microwave Limb Sounder
UARS iSAMS	Upper Atmosphere Research Satellite Improved Stratospheric and Mesospheric Sounder
UoR	University of Reading
URD	User Requirements Document
UTLS	Upper Tropospheric / Lower Stratospheric
VRWV	Vertically Resolved Water Vapour
VM	Virtual Machine
WAVAS_SAHAR	WAVAS SATELLITE COMPONENT IN HARMONIZED FORMAT
WGS	World Geodetic System
WV	Water Vapour
YAML	YAML [= Yet Another Markup Language] Ain't Markup Language (recursive acronym)
YARN	Yet Another Resource Negotiator

APPENDIX 3: PROCESSING SCRIPTS

1. ERA Interim download Python script:

retrieve-era-interim.py

```
#!/usr/bin/env python

#

# (C) Copyright 2012-2013 ECMWF.

#


from ecmwfapi import ECMWFDataServer

import datetime

import calendar


# To run this example, you need an API key

# available from https://api.ecmwf.int/v1/key/


if len(sys.argv) != 3:

    print ('Usage:  python get_erainterim_for_tcwv.py <nc_infile> 
<year> <month>')

    sys.exit(-1)


year = int(sys.argv[1])

month = int(sys.argv[2])


server = ECMWFDataServer()


monthrange = calendar.monthrange(year, month)

start = datetime.date(year, month, 1)

stop = datetime.date(year, month, monthrange[1])

request = {

    'dataset' : "interim",

    'step'    : "0",

    'number'  : "all",

    'levtype' : "sfc",

    'date'    : start.isoformat()+"/to/"+stop.isoformat(),

    'time'    : "00/06/12/18",

    'origin'  : "all",
    'type'    : "an",
    'param'   : "167.128/151.128/137.128/165.128/166.128",
    'area'    : "90/-180/-90/179.25",
    'grid'    : "0.75/0.75",
    'format'  : 'netcdf',
    'target'  : "era-interim-t2m-mslp-tcwv-u10-v10-{0:04d}-
{1:02d}.nc".format(year,month)
    }
print str(request)
server.retrieve(request)
```

2. Bash executable for interpolation of ERA Interim data:

wvcci-l2-erainterim-modis-bash.sh

```
#!/bin/bash
set -x
set -e
ulimit -a

CDO_HOME=cdo
export BEAM_HOME=beam
export LD_LIBRARY_PATH=$CDO_HOME:$LD_LIBRARY_PATH
auxroot=.

# MER_RR__1PRACR20080215_152228_000026032066_00054_31170_0000.N1
merisfile=$1
merisname=$(basename $merisfile)
merisstem=${merisname%.N1}

year=${merisname:14:4}
month=${merisname:18:2}
day=${merisname:20:2}
hour=${merisname:23:2}
minute=${merisname:25:2}
second=${merisname:27:2}
date_in_seconds=$(date +%s -u -d "$year-$month-$day
$hour:$minute:$second")
let day_before_in_seconds="date_in_seconds - 86400"
let day_after_in_seconds="date_in_seconds + 86400"
```

```
date_before=`date +%Y-%m-%d -u -d @$day_before_in_seconds`

date_after=`date +%Y-%m-%d -u -d @$day_after_in_seconds`


# extract geo information in SCRIP format for CDOs


echo "$(date +%Y-%m-%dT%H:%M:%S -u) extracting geo information of
$merisname ..."

scripfile=${merisstem}-scrip.nc

if [ ! -e $scripfile ]; then

  if ! $BEAM_HOME/bin/gpt.sh Write -PformatName=SCRIP -
Pfile=$scripfile $merisfile > gpt.out 2>&1

  then

    cat gpt.out

    exit 1

  fi

fi


# distinguish beginning of month, end of month, or in between and
create time stacks

if [ "$day" = "01" -a "$year$month" != "197901" ]

then

  echo "$(date +%Y-%m-%dT%H:%M:%S -u) merge era stack with previous
month ..."

  eradaybefore=$auxroot/era-interim-t2m-mslp-tcwv-u10-
v10/${date_before:0:4}/era-interim-t2m-mslp-tcwv-u10-v10-
${date_before:0:4}-${date_before:5:2}.nc

  erathisday=$auxroot/era-interim-t2m-mslp-tcwv-u10-v10/$year/era-
interim-t2m-mslp-tcwv-u10-v10-$year-$month.nc

  eratimestack=era-interim-t2m-mslp-tcwv-u10-v10-${date_before:0:4}-
${date_before:5:2}-$year-$month.nc

  $CDO_HOME/cdo -b 32 mergetime $eradaybefore $erathisday
$eratimestack

elif [ "$day" = "31" -a "$year$month" != "201509" ]

then

  echo "$(date +%Y-%m-%dT%H:%M:%S -u) merge era stack with next month
..."

  erathisday=$auxroot/era-interim-t2m-mslp-tcwv-u10-v10/$year/era-
interim-t2m-mslp-tcwv-u10-v10-$year-$month.nc

  eradayafter=$auxroot/era-interim-t2m-mslp-tcwv-u10-
v10/${date_after:0:4}/era-interim-t2m-mslp-tcwv-u10-v10-
${date_after:0:4}-${date_after:5:2}.nc
```

```
  eratimestack=era-interim-tcwv-u10-v10-$year-$month-${date_after:0:4}-${date_after:5:2}.nc
  $CDO_HOME/cdo -b 32 mergetime $erathisday $eradayafter $eratimestack
else
  eratimestack=$auxroot/era-interim-t2m-mslp-tcwv-u10-v10/$year/era-interim-t2m-mslp-tcwv-u10-v10-$year-$month.nc
fi

# interpolate temporally

echo "$(date +%Y-%m-%dT%H:%M:%S -u) interpolate temporally ..."
eratimeslice=era-interim-t2m-mslp-tcwv-u10-v10-$year$month$day$hour$minute.nc
$CDO_HOME/cdo inttime,$year-$month-$day,$hour:$minute:01 $eratimestack $eratimeslice

# interpolate spatially

echo "$(date +%Y-%m-%dT%H:%M:%S -u) interpolate spatially ..."
#erameris=${year:2:2}$month$day$hour$minute-era-interim.nc
erameris=${merisstem}_era-interim.nc
$CDO_HOME/cdo -L -f nc4c remapbil,$scripfile $eratimeslice $erameris

# list results

mkdir -p $day
mv $erameris $day

echo "CALVALUS_OUTPUT_PRODUCT $day/$erameris"
```

3. PMonitor workflow script on Calvalus:

tcwv-l3-monthly-workflow.py

```
import glob
import os
import datetime
import calendar
```

```
from datetime import date
from calendar import monthrange, isleap
from pmonitor import PMonitor

class TcwvL3Monthly(PMonitor):

    def __init__(self, parameters):

        print('tcwv-l3-monthly-workflow.py: init...')

        self._l3_day_root = parameters['l3DayRoot']
        self._l3_month_root = parameters['l3MonthRoot']
        self._version = parameters['version']
        self._sensor = parameters['sensor']



        self._start = parameters['start']
        self._stop = parameters['stop']
        self.start_of_request =
datetime.datetime.strptime(self._start, '%Y-%m-%d')  # e.g. 2019-01-
01
        self.stop_of_request = datetime.datetime.strptime(self._stop,
'%Y-%m-%d')  # e.g. 2020-08-01

        inputs = []
        self._l3_resolutions = [ '005', '05' ]
        cursor = self.start_of_request
        while cursor < self.stop_of_request:
            lastdayofmonth = calendar.monthrange(cursor.year,
cursor.month)[1]
            year_month_day = cursor.strftime('%Y-%m-%d')
            year = year_month_day[0:4]
            month = year_month_day[5:7]
            for res in self._l3_resolutions:
                inputs.append(self._l3_day_root + '/'  + res + '/' +
year + '/' + month)
            cursor += datetime.timedelta(days=lastdayofmonth)
        PMonitor.__init__(self,
                          inputs,
                          request=parameters['requestName'],
                          hosts=[('localhost', 6)],
                          types=[('tcwv-l3-monthly', 60)],
                          logdir='log',
                          script='hstep.py',
                          polling='hjobs.py',
                          period=30,
                          simulation='simulation' in parameters and 
parameters['simulation'])

  def create_workflow(self):

        # monthly loop
        cursor = self.start_of_request
        while cursor < self.stop_of_request:
            lastdayofmonth = calendar.monthrange(cursor.year, 
cursor.month)[1]
            year_month_day = cursor.strftime('%Y-%m-%d')
            cursor_end_month = cursor  + 
datetime.timedelta(days=lastdayofmonth-1)
            print ('month start: ' + year_month_day)
            print ('month end: ' + cursor_end_month.strftime('%Y-%m-
%d'))
            year_month_day_stop = cursor_end_month.strftime('%Y-%m-
%d')

            year = year_month_day[0:4]
            month = year_month_day[5:7]

            for res in self._l3_resolutions:
                num_rows = 3600 if res == '005' else 360

                inputs = []
                inputs.append(self._l3_day_root + '/'  + res + '/' + 
year + '/' + month)
                l3_month_dir = self._l3_month_root + '/'  + res + '/' 
+ year + '/' + month
                params = ['start', year_month_day,
                          'stop', year_month_day_stop,
                          'sensor', self._sensor,
                          'year', year,
                          'month', month,
                          'version', self._version,
                          'res', res,
                          'numRowsInL3', str(num_rows)
                          ]

                self.execute('tcwv-l3-monthly',
                             inputs,
                             [l3_month_dir],
                              parameters = params
                            )

            cursor += datetime.timedelta(days=lastdayofmonth)


    def run(self):
        self.wait_for_completion()
```

4. YAML request for TCWV L3 monthly processing on Calvalus:

tcwv_l3_monthly.yaml

```
requestName : tcwv_l3_monthly
productionType : tcwv-l3-monthly
version : "4.0"
simulation : False
#sensor : 'meris'
sensor : 'meris-cmsaf_hoaps'
start : '2002-07-01'
stop : '2012-04-01'
l3DayRoot : "/calvalus/projects/wvcci/tcwv/meris/l3-daily-final-nc"
l3MonthRoot : "/calvalus/projects/wvcci/tcwv/meris/l3-monthly"
```

5. JSON config file for TCWV L3 monthly processing on Calvalus:

tcwv-l3-monthly-template.json

```
{
    "productionType"    : "l3-agg",
    "productionName"    : "WVCCI ${sensor} L3 monthly ${res}deg
${year}-${month}",

    "inputPath"         :
"${input}/${dd}/.*${yyyy}.*${MM}.*${dd}.*fv.*.nc",
    "dateRanges"        : "[${start}:${stop}]",

    "aggregationParameters" :
"<parameters><planetaryGrid>org.esa.snap.binning.support.PlateCarreeG
rid</planetaryGrid>  <numRows>${numRowsInL3}</numRows>
<compositingType>BINNING</compositingType>
<superSampling>3</superSampling>  <aggregators>
<aggregator><type>AVG</type><varName>tcwv</varName></aggregator>
<aggregator><type>AVG</type><varName>num_obs</varName><outputSums>tru
e</outputSums></aggregator>
<aggregator><type>AVG</type><varName>stdv</varName></aggregator>
<aggregator><type>AVG</type><varName>tcwv_err</varName></aggregator>
<aggregator><type>AVG</type><varName>tcwv_ran</varName><targetName>tc
wv_ran</targetName><weightCoeff>1.0</weightCoeff><outputCounts>true</
outputCounts><outputSums>false</outputSums></aggregator>
<aggregator><type>AVG</type><varName>tcwv_ran</varName><targetName>tc
wv_ran_sums</targetName><weightCoeff>1.0</weightCoeff><outputCounts>f
alse</outputCounts><outputSums>true</outputSums></aggregator>
<aggregator><type>MAJORITY_CLASS</type><varName>surface_type_flag</va
rName><classes>0,1,2,3,4,5,6,7</classes></aggregator>
<aggregator><type>MIN_MAX</type><varName>surface_type_flag</varName><
/aggregator>
<aggregator><type>MAJORITY_CLASS</type><varName>tcwv_quality_flag</va
rName><classes>0,1,2</classes></aggregator>
<aggregator><type>MIN_MAX</type><varName>tcwv_quality_flag</varName><
/aggregator>  </aggregators></parameters>",


    "outputDir"         : "${output}",
    "outputVersion"     : "${version}",
    "outputFormat"      : "NetCDF4",
    "calvalus.output.prefix" : "l3_tcwv_${sensor}_${res}deg",

    "maxReducers"       : "1",
    "timeout"           : "7200",
    "failurePercent"    : "0",
    "attempts"         : "2",
    "processingMemory" : "8192",
    "aggregationMemory": "8192",
    "snap.tileCache"   : "1792",
    "queue"            : "wvcci",

    "calvalus"         : "calvalus-2.22",
    "snap"             : "snap-wvcci-2.0-SNAPSHOT",
    "processorBundles" : "snap-wvcci-2.0-SNAPSHOT"

}
```

6. Status file for bulk processing on Calvalus after successful execution:

wvcci-l2-tcwv-meris.status

```
219 created, 0 running, 0 backlog, 219 processed, 0 failed
```

7. Report file entries for TCWV L3 monthly processing on Calvalus after successful execution:

tcwv_l3_monthly.report

```
hstep.py tcwv-l3-monthly start 2004-05-01 stop 2004-05-31 sensor
meris-cmsaf_hoaps year 2004 month 05 version 4.0 res 05 numRowsInL3
360 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-daily-final-
nc/05/2004/05 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-
monthly/05/2004/05

hstep.py tcwv-l3-monthly start 2004-06-01 stop 2004-06-30 sensor
meris-cmsaf_hoaps year 2004 month 06 version 4.0 res 05 numRowsInL3
360 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-daily-final-
nc/05/2004/06 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-
monthly/05/2004/06

hstep.py tcwv-l3-monthly start 2004-05-01 stop 2004-05-31 sensor
meris-cmsaf_hoaps year 2004 month 05 version 4.0 res 005 numRowsInL3
3600 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-daily-final-
nc/005/2004/05 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-
monthly/005/2004/05

hstep.py tcwv-l3-monthly start 2004-06-01 stop 2004-06-30 sensor
meris-cmsaf_hoaps year 2004 month 06 version 4.0 res 005 numRowsInL3
3600 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-daily-final-
nc/005/2004/06 /calvalus/projects/wvcci/tcwv/meris-cmsaf_hoaps/l3-
monthly/005/2004/06
```

8. Log file example for TCWV L3 monthly processing on Calvalus:

tcwv-l3-monthly-0003.out

```
2023-03-27 01:06:38,726 INFO com.bc.calvalus: reading request
from /home/cvop/wvcci-inst/requests/tcwv-l3-monthly-2002-08-01-
meris-cmsaf_hoaps-2002-08-4.0-005-3600.json with type l3-agg and
20 parameters

2023-03-27 01:06:38,759 INFO com.bc.calvalus: reading .calvalus
configuration with 15 parameters

2023-03-27 01:06:38,760 INFO com.bc.calvalus: reading production
type definition from etc/l3-agg-cht-type.json with 42 parameters
and 38 rules

684 [main] WARN org.apache.hadoop.util.NativeCodeLoader - Unable
to load native-hadoop library for your platform... using builtin-
java classes where applicable

1336 [main] WARN
org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory - The
short-circuit local reads feature cannot be used because
libhadoop cannot be loaded.

2023-03-27 01:06:39,788 INFO com.bc.calvalus: no bundle or no
processor requested

2023-03-27 01:06:39,788 INFO com.bc.calvalus: reading processor
descriptor from bundle with 0 parameters

2023-03-27 01:06:40,067 INFO com.bc.calvalus: clearing output dir
/calvalus/projects/wvcci/tcwv/meris/l3-monthly/005/2002/08

1752 [main] INFO
org.apache.hadoop.yarn.client.ConfiguredRMFailoverProxyProvider -
Failing over to rm2

2017 [main] INFO org.apache.hadoop.mapreduce.JobResourceUploader
- Disabling Erasure Coding for path: /tmp/hadoop-
yarn/staging/olaf/.staging/job_1671707876855_517134

2028 [main] WARN org.apache.hadoop.mapreduce.JobResourceUploader
- No job jar file set.  User classes may not be found. See Job or
Job#setJar(String).

2023-03-27 01:06:41,559 INFO com.bc.calvalus: query for [2002-08-
01:2002-08-31] done

2023-03-27 01:06:41,561 INFO com.bc.calvalus: file system query
done in [ms]: 686

2023-03-27 01:06:41,561 INFO com.bc.calvalus: Total files to
process : 31

3983 [main] INFO org.apache.hadoop.mapreduce.JobSubmitter -
number of splits:31

4159 [main] INFO org.apache.hadoop.mapreduce.JobSubmitter -
Submitting tokens for job: job_1671707876855_517134

4161 [main] INFO org.apache.hadoop.mapreduce.JobSubmitter -
Executing with tokens: []

4301 [main] INFO org.apache.hadoop.mapred.YARNRunner - Job jar is
not present. Not adding any jar to the list of resources.

5181 [main] INFO org.apache.hadoop.conf.Configuration - resource-
types.xml not found
```

```
5181 [main] INFO
org.apache.hadoop.yarn.util.resource.ResourceUtils - Unable to
find 'resource-types.xml'.

5232 [main] INFO
org.apache.hadoop.yarn.client.api.impl.YarnClientImpl - Submitted
application application_1671707876855_517134

5263 [main] INFO org.apache.hadoop.mapreduce.Job - The url to
track the job:
http://master01:8088/proxy/application_1671707876855_517134/

2023-03-27 01:06:43,706 INFO com.bc.calvalus: Production
successfully ordered with ID job_1671707876855_517134

job_1671707876855_517134
```

9. Jobs definition and registration script on JASMIN:

wvcci-l2-tcwv-modis-step-slurm.sh

```
#!/bin/bash


. ${WVCCI_INST}/bin/wvcci-env-slurm.sh   # this script shall now be
used for everything!


echo "entered NEW wvcci-l2-tcwv-modis-step-slurm..."
l1bPath=$1
l1bFile=$2
cloudMaskPath=$3
sensor=$4
year=$5
month=$6
day=$7
hhmm=$8
wvcciRootDir=$9


task="wvcci-l2-tcwv-modis"
jobname="${task}-${year}-${month}-${day}-${hhmm}"
command0="./bin/${task}-bash-slurm.sh"
command="${command0} ${l1bPath} ${l1bFile} ${cloudMaskPath} ${sensor}
${year} ${month} ${day} ${wvcciRootDir}"


echo "jobname: $jobname"
echo "command0: $command0"
echo "command: $command"

echo "`date -u +%Y%m%d-%H%M%S` submitting job '${jobname}' for task
${task}"

if [ -z ${jobs} ]; then
    timelim=120
    memlim=16000
    echo "submit_job ${jobname} ${command} ${timelim} ${memlim}"
    submit_job ${jobname} "${command}" ${timelim} ${memlim}
fi
```

10. TCWV L2 part in process definition script on JASMIN:

# wvcci-l2-tcwv-modis-bash-slurm.sh

```
#!/bin/bash

## TCWV
if [ -f $l1bEraInterimMerge ]; then

#auxdataPath=/gws/nopw/j04/esacci_wv/software/dot_snap/auxdata/wvcci
    auxdataPath=/home/users/odanne/.snap/auxdata/wvcci
    tcwv=$tcwvDir/${modisstem}_tcwv.nc

    # Add MOD35_L2 product as cloud product.
    # We want to write TCWV with NetCDF4_WVCCI in order NOT to write
lat,lon!
    echo "START gpt Tcwv - wallclock time is: `date`"
    if [ -f $cloudMaskPath ]; then
      echo "${SNAP_HOME}/bin/gpt ESACCI.Tcwv -q 1 -e -
SsourceProduct=$l1bEraInterimMerge -Smod35Product=$cloudMaskPath -
PauxdataPath=$auxdataPath -Psensor=$sensor -PprocessOcean=true -f
NetCDF4-WVCCI -t $tcwv"
      #echo "${SNAP_HOME}/bin/gpt ESACCI.Tcwv -q 4 -e -
SsourceProduct=$l1bEraInterimMerge -Smod35Product=$cloudMaskPath -
PauxdataPath=$auxdataPath -Psensor=$sensor -PprocessOcean=true -f
NetCDF4-WVCCI -t $tcwv"
      #echo "${SNAP_HOME}/bin/gpt ESACCI.Tcwv -e -
SsourceProduct=$l1bEraInterimMerge -Smod35Product=$cloudMaskPath -
PauxdataPath=$auxdataPath -Psensor=$sensor -PprocessOcean=true -f
NetCDF4-WVCCI -t $tcwv"
```

```
      ${SNAP_HOME}/bin/gpt ESACCI.Tcwv -q 1 -e -
SsourceProduct=$l1bEraInterimMerge -Smod35Product=$cloudMaskPath -
PauxdataPath=$auxdataPath -Psensor=$sensor -PprocessOcean=true -f
NetCDF4-WVCCI -t $tcwv

      #${SNAP_HOME}/bin/gpt ESACCI.Tcwv -q 4 -e -
SsourceProduct=$l1bEraInterimMerge -Smod35Product=$cloudMaskPath -
PauxdataPath=$auxdataPath -Psensor=$sensor -PprocessOcean=true -f
NetCDF4-WVCCI -t $tcwv

      #${SNAP_HOME}/bin/gpt ESACCI.Tcwv -e -
SsourceProduct=$l1bEraInterimMerge -Smod35Product=$cloudMaskPath -
PauxdataPath=$auxdataPath -Psensor=$sensor -PprocessOcean=true -f
NetCDF4-WVCCI -t $tcwv

    fi

    echo "END gpt Tcwv - wallclock time is: `date`"

    status=$?

    echo "Status: $status"

fi


# cleanup: remove l1b-erainterim merge product (TCWV input)

echo "START cleanup - wallclock time is: `date`"

sleep 10

if [ -f $l1bEraInterimMerge ]; then

  # Merge products are large. As examples, just keep L1bEraInterim of
Jan 15th and Jul 15th of each year:

  if [ "$day" != "15" ] || ([ "$month" != "01" ] && [ "$month" !=
"07" ]); then

    echo "DELETING L1bEraInterim merge product : $l1bEraInterimMerge"

    rm -f $l1bEraInterimMerge

  fi

fi


status=$?

if [ $status = 0 ] && [ -e "$tcwv" ]

then

    echo "TCWV product created."

    echo "Status: $status"

else

    echo "TCWV product NOT successfully created (corrupt or MODIS L1b
is not a Day product)."

    echo "Status: $status"

    if [ -e "$tcwv" ]
    then
      echo "rm -f $tcwv"
      rm -f $tcwv   # delete unwanted file
    fi
fi
echo "END cleanup - wallclock time is: `date`"

echo "FINISHED job wvcci-tcwv-l2-modis - wallclock time is: `date`"
```

11. SLURM job submission script on JASMIN:

# wvcci-env-slurm.sh

```
#!/bin/bash

# WVCCI function definitions
# usage ${mms_home}/bin/${WVCCI_ENV_NAME}  (in xxx-start.sh and xxx-
run.sh)

# project and user settings
# -------------------------
export PROJECT=wvcci

# Java and Python runtime definitions
# -----------------------------------

# ensure that processes exit
set -e

if [ -z "${WORKING_DIR}" ]; then
    WORKING_DIR=`pwd -P`
fi

export PM_LOG_DIR=${WORKING_DIR}/log

if [ "$SCHEDULER" == "LSF" ]; then

    submit_job() {
        jobname=$1
        command=$2
        timelim=$3  # job time limit in minutes, see
https://help.jasmin.ac.uk/article/113-submit-jobs, default: 60

        memlim=$4   # job memory limit in MB, see
https://help.jasmin.ac.uk/article/113-submit-jobs, default: 4000


        if [ ! -z "$timelim" ];

        then

          timelim="-W ${timelim}"

        fi

        if [ ! -z "$memlim" ];

        then

          memlim="-R rusage[mem=${memlim}] -M ${memlim}"

        fi


        bsubmit="bsub -q short-serial -We 2:00 -W 4:00 ${memlim} -P
${PROJECT} -cwd ${WVCCI_INST} -oo ${WVCCI_LOG}/${jobname}.out -eo
${WVCCI_LOG}/${jobname}.err -J ${jobname} ${WVCCI_INST}/${command}
${@:3:${mynumargs}}"


        rm -f ${PM_LOG_DIR}/${jobname}.out

        rm -f ${PM_LOG_DIR}/${jobname}.err


        # line contains the console output of the bsub command

        line=`${bsubmit}`


        if echo ${line} | grep -qF 'is submitted'

        then

            # extract the job_id from the bsub message, concatenate
'_' and jobname to form an identifier

            # and dump to std_out to be fetched by pmonitor

            job_id=`echo ${line} | awk '{ print
substr($2,2,length($2)-2) }'`

            echo "${job_id}_${jobname}"

        else

            echo "`date -u +%Y%m%d-%H%M%S` - submit of ${jobname}
failed: ${line}"

            exit 1

        fi

    }
```

```
elif [ "$SCHEDULER" == "SLURM" ]; then

    submit_job() {
            jobname=$1
            command=$2
            timelim=$3  # job time limit in minutes, see
https://help.jasmin.ac.uk/article/113-submit-jobs, default: 60
            memlim=$4   # job memory limit in MB, see
https://help.jasmin.ac.uk/article/113-submit-jobs, default: 4000

            if [ ! -z "$timelim" ];
            then
              timelim="-W ${timelim}"
            fi
            if [ ! -z "$memlim" ];
            then
              memlim="--mem=${memlim}"
            fi

            command="${WVCCI_INST}/${command} ${@:3:${mynumargs}}"
            bsubmit="sbatch -p short-serial --time-min=01:00:00 --
time=02:00:00 ${memlim} --chdir=${WVCCI_INST} -o
${WVCCI_LOG}/${jobname}.out -e ${WVCCI_LOG}/${jobname}.err --open-
mode=append --job-name=${jobname} ${command}"
            echo "bsubmit: ${bsubmit}"

            rm -f ${PM_LOG_DIR}/${jobname}.out
            rm -f ${PM_LOG_DIR}/${jobname}.err

            # line contains the console output of the bsub command
            line=`${bsubmit}`

            if echo ${line} | grep -qF 'Submitted batch job'
            then
                # extract the job_id from the bsub message,
concatenate '_' and jobname to form an identifier
                # and dump to std_out to be fetched by pmonitor
                job_id=`echo ${line} | awk '{ print
substr($4,0,length($4)) }'`
                echo "${job_id}_${jobname}"
            else
```

```
                echo "`date -u +%Y%m%d-%H%M%S` - submit of ${jobname}
failed: ${line}"
                exit 1
            fi
        }

else
    echo "Invalid scheduler"
    exit 1
fi
```

End of Document