

# Water Vapour Climate Change Initiative (WV\_cci) - CCI+ Phase 1



System Specification Document (SSD)

Ref: D3.2

Date: 1 October 2020

Issue: 2.1

For: ESA / ECSAT

Ref: CCIWV.REP.013



aeronomie.be



UNIVERSITY OF  
**LEICESTER**

UNIVERSITÉ DE  
**VERSAILLES**  
SAINT-QUENTIN-EN-YVELINES



Science & Technology Facilities Council  
Rutherford Appleton Laboratory

Universidade de Vigo

***This Page is Intentionally Blank***

**Project** : **Water Vapour Climate Change Initiative (WV\_cci) - CCI+ Phase 1**

**Document Title:** **System Specification Document (SSD)**

**Reference** : **D3.2**

**Issued** : **1 October 2020**

**Issue** : **2.1**

**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

# TABLE OF CONTENTS

<b>1. INTRODUCTION .....</b>	<b>9</b>
1.1 Purpose.....	9
1.2 Scope.....	10
<b>2. SYSTEM OVERVIEW.....</b>	<b>11</b>
2.1 WV_cci System Context .....	11
2.2 Main Function and Processing Chain .....	13
2.3 System Requirements.....	14
2.4 WV_cci System Architecture.....	15
2.4.1 TCWV processing system (BC).....	15
2.4.1.1 Calvalus system architecture.....	16
2.4.1.2 JASMIN system architecture .....	17
2.4.2 VRWV Processing System (UoR) .....	19
<b>3. TECHNICAL METHODS AND CONCEPTS.....</b>	<b>20</b>
3.1 TCWV Processing System (BC) .....	20
3.1.1 HDF5.....	20
3.1.2 Calvalus .....	21
3.1.2.1 Software bundles .....	21
3.1.2.2 Hardware infrastructure .....	21
3.1.2.3 Software infrastructure.....	24
3.1.2.4 The WV_cci processing instance.....	29
3.1.3 JASMIN .....	31
3.1.3.1 Hardware environment.....	31
3.1.3.2 Job submission and processing environment.....	32
3.2 VRWV Processing System (UoR).....	33
3.2.1 RACC hardware environment .....	34
3.2.2 RACC processing environment.....	34
3.2.3 RACC software environment.....	35
<b>4. WORKFLOWS .....</b>	<b>36</b>
4.1 TCWV Processing System (BC) .....	36
4.1.1 TCWV L2 processing.....	36
4.1.1.1 MERIS.....	37
4.1.1.2 MODIS .....	40
4.1.1.3 OLCI.....	43
4.1.2 TCWV L3 processing.....	44
4.1.2.1 Daily products .....	45
4.1.2.2 Monthly products.....	49
4.2 VRWV Processing System (UoR).....	51
4.2.1 VRWV CDR-3 processing .....	52
4.2.1.1 Data acquisition .....	53
4.2.1.2 Bias Correction Processor.....	53
4.2.1.3 Seasonal Correction Processor.....	53
4.2.1.4 Merging of L3 products .....	54
4.2.1.5 NetCDF product formatting.....	54
4.2.2 VRWV CDR-4 processing .....	55
4.2.2.1 Data acquisition .....	55
4.2.2.2 Bias Correction Processor.....	57
4.2.2.3 Temporal and spatial aggregation .....	57

4.2.2.4 Merging of L3 products .....	58
4.2.2.5 NetCDF product formatting .....	58
<b>5. REQUIREMENTS TRACEABILITY .....</b>	<b>59</b>
<b>APPENDIX 1: REFERENCES .....</b>	<b>63</b>
<b>APPENDIX 2: GLOSSARY .....</b>	<b>66</b>
<b>APPENDIX 3: PROCESSING SCRIPTS.....</b>	<b>68</b>

## INDEX OF TABLES

Table 3-1: Calvalus infrastructure elements .....	22
Table 3-2: Software items for WV_cci processing .....	26

## INDEX OF FIGURES

Figure 2-1: Symmetric system definition of the TCWV and VRWV processing systems. Updated from [5]. .....	11
Figure 2-2: System context of the WV_cci system with team, data providers, and other entities. .	13
Figure 2-3: Requirements derivation logic as usual in CCI projects with system requirements as specified in project documentation. Taken from [6]. .....	14
Figure 2-4: WV_cci system architecture layers with specific elements, Calvalus, and Frameworks layer. ....	16
Figure 3-1: HDFS concepts of 'archive centric' and 'data local'.....	21
Figure 3-2: Calvalus architecture. ....	22
Figure 3-3: Software architecture. ....	24
Figure 3-4: Calvalus options for bulk processing. The processing instance <i>wvcci-inst</i> is used for WV_cci. Names of scripts for MERIS TCWV L2 processing are given as examples. ....	30
Figure 3-5: Concept of the PMonitor workflow engine. Taken from [18]. ....	31
Figure 3-6: Scheme for bulk processing setup on JASMIN. Names of scripts for MODIS TCWV L2 processing are given as examples. ....	33
Figure 3-7: Schematic layout of the RACC, taken from [17].....	34

Figure 4-1: The TCWV L2 processing chains for MODIS, MERIS and OLCI. See text for details.	36
Figure 4-2: SNAP Product Explorer: MERIS IdePix pixel classification product after merge with reflectance bands and ERA Interim anillary variables. ....	39
Figure 4-3: SNAP Product Explorer: Contents of a MERIS TCWV L2 product. ....	40
Figure 4-4: TCWV L3 processing chains. See text for details. ....	45
Figure 4-5: SNAP Product Explorer: NetCDF global attributes of a TCWV L3 daily final product.	46
Figure 4-6: SNAP Product Explorer: TCWV L3 daily final product. ....	48
Figure 4-7: OLCI/MODIS/HOAPS TCWV L3 daily merge for July 15th, 2016 (greyscale, 0-70 kgm-2). Yellow indicates no data. ....	49
Figure 4-8: OLCI/MODIS/HOAPS TCWV L3 monthly merge for July 2016 (greyscale, 0-70 kgm-2). Yellow indicates no data. ....	50
Figure 4-9: System definition of the VRWV processing systems for CDR-3 and CDR-4. Updated from [5]. ....	51
Figure 4-10 VRWV CDR-3 processing chain. SCIA stands for SCIAMACHY, U/A-MLS for UARS-/Aura-MLS, ACE-F/M for ACE-FTS/MAESTRO and CCM for Chemistry-Climate Model, respectively. See text for futher details. ....	52
Figure 4-11: Panoply product explorer: merged VRWV monthly zonal mean L3 final product (prototype v0). ....	55
Figure 4-12: VRWV CDR-4 processing chains. See text for details. ....	57

***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 retrieval, 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, ACE-FTS, ACE-MAESTRO, and a combined retrieval product (IMS) from the IASI/MHS/AMSU satellite instruments. Here, 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.

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

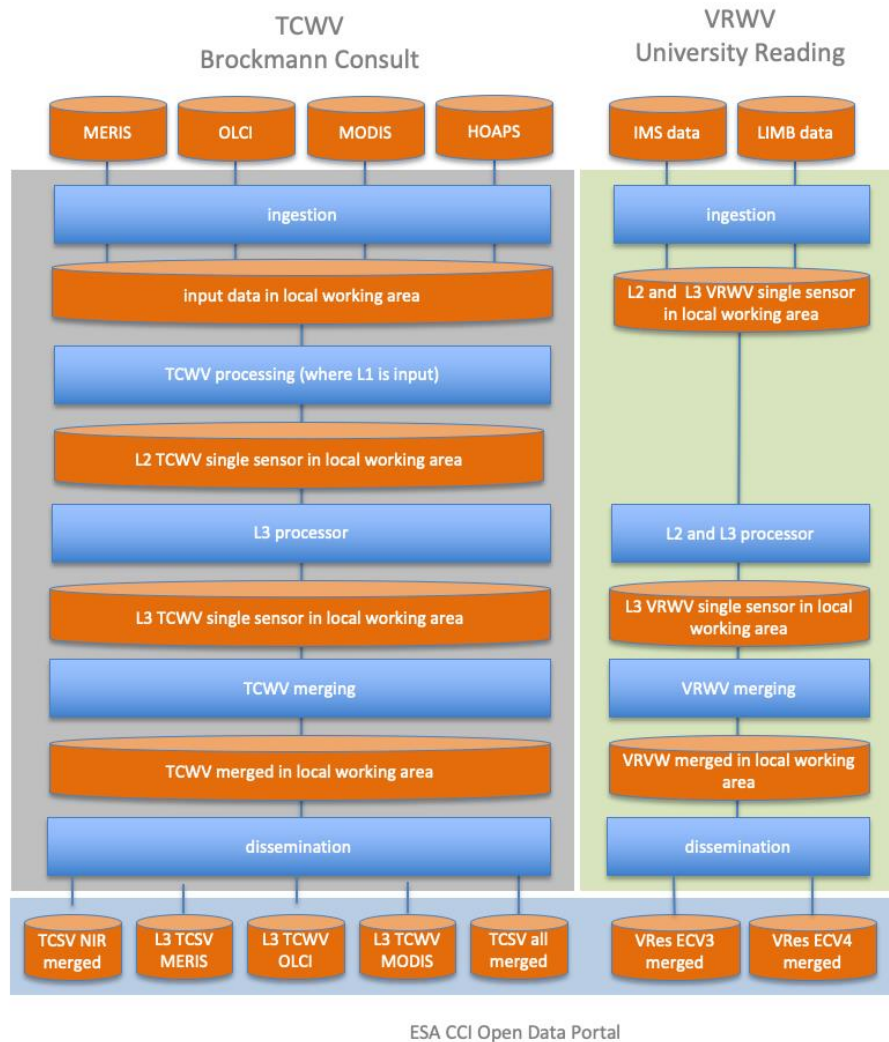
## 1.2 Scope

This document describes the system developed for the ESA WV\_cci project and its elements used in the current phase.

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] 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 currently covers the MERIS, MODIS, OLCI, and CM SAF HOAPS TCWV workflows for the TCWV generation. The current version 2 of this SSD has been extended for OLCI, in line with the introduction of OLCI into the retrieval following the project plan. For VRWV, the merging algorithm description in its first draft form in this version 2 and will be finalised in version 3, in line with the VRWV merging activities within the project plan.

## 2. SYSTEM OVERVIEW



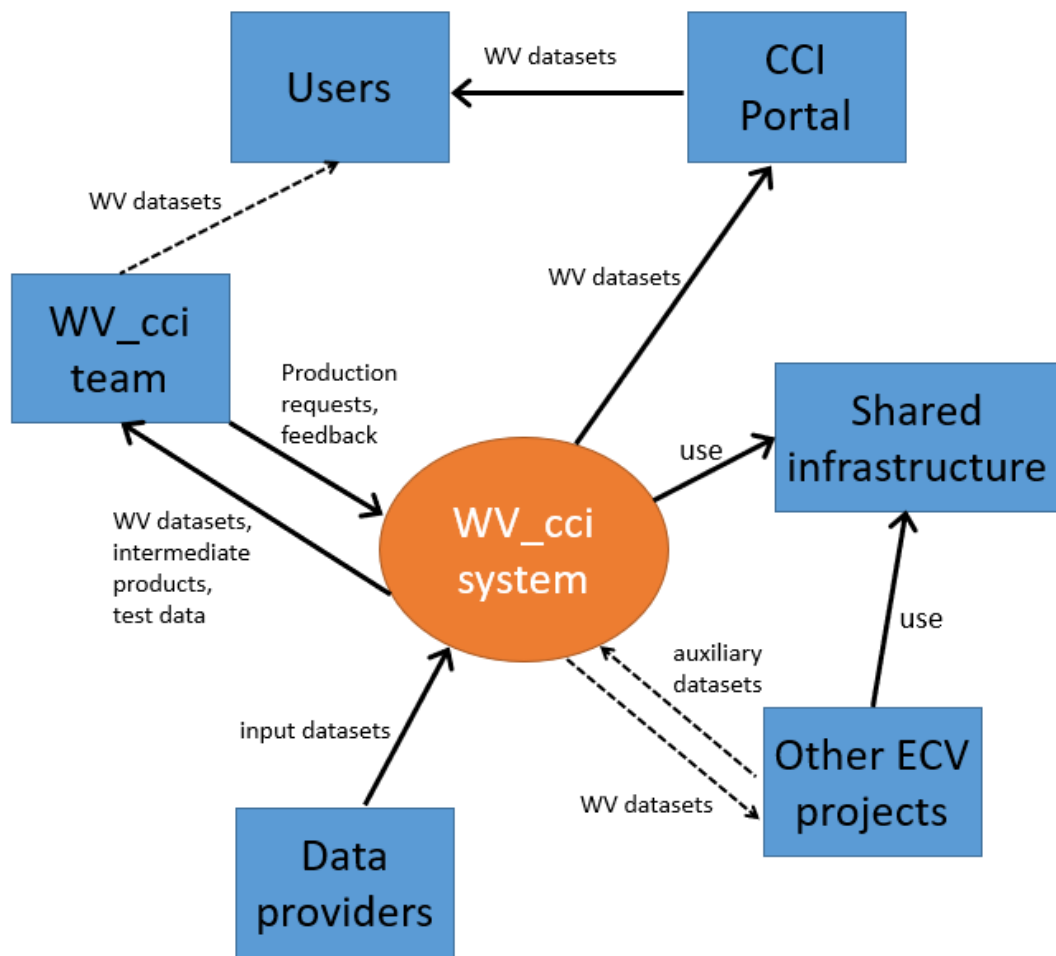
**Figure 2-1: Symmetric system definition of the TCWV and VRWV processing systems. Updated from [5].**

### 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).
- The CCI Open Data Portal which will be fed with validated WV datasets. 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, or the RACC for the VRWV.



**Figure 2-2: System context of the WV\_cci system with team, data providers, and other entities.**

## 2.2 Main Function and Processing Chain

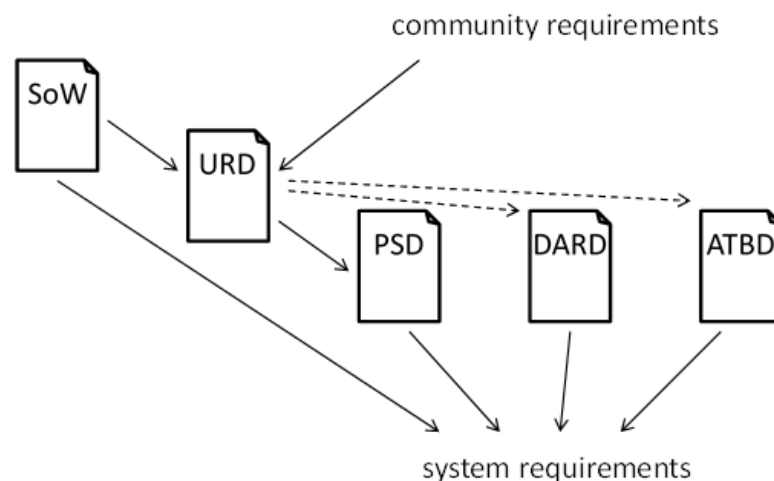
With regard to TCWV, one of the main goals of WV\_cci is the merging of the TCWV data of the available sensors in order to fill all data gaps in global products as far as possible, and to identify regional, temporal, or systematic differences and discontinuities when bringing the data together. This should ideally result 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.

With regard to VRWV, a main goal for CDR-3 is the merging of a well characterized and homogeneous long-term zonal mean VRWV CDR in the stratosphere from data obtained from available limb satellite sounders (SAGE II, HALOE, POAM III, SMR,

SCIAMACHY, MIPAS, ACE-FTS, ACE-MAESTRO, and Aura-MLS) covering the time period 1985–2019. A key aspect in the construction of such a CDR is the requirement for a homogenization 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 is to merge the VRWV data from both limb-sounders and nadir instruments into a harmonised prototype VRWV data records from 2010 to 2014. This study is expected to improve knowledge of the consistency between VRWV products from limb-sounders and nadir instruments and provide insight in the underlying merging algorithm between these two kinds of satellite products.

## 2.3 System Requirements

System requirements are usually derived from use cases, user requirements, and other inputs on the basis of 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 these system requirements are explicitly listed and discussed in more detail in the SRD [12].

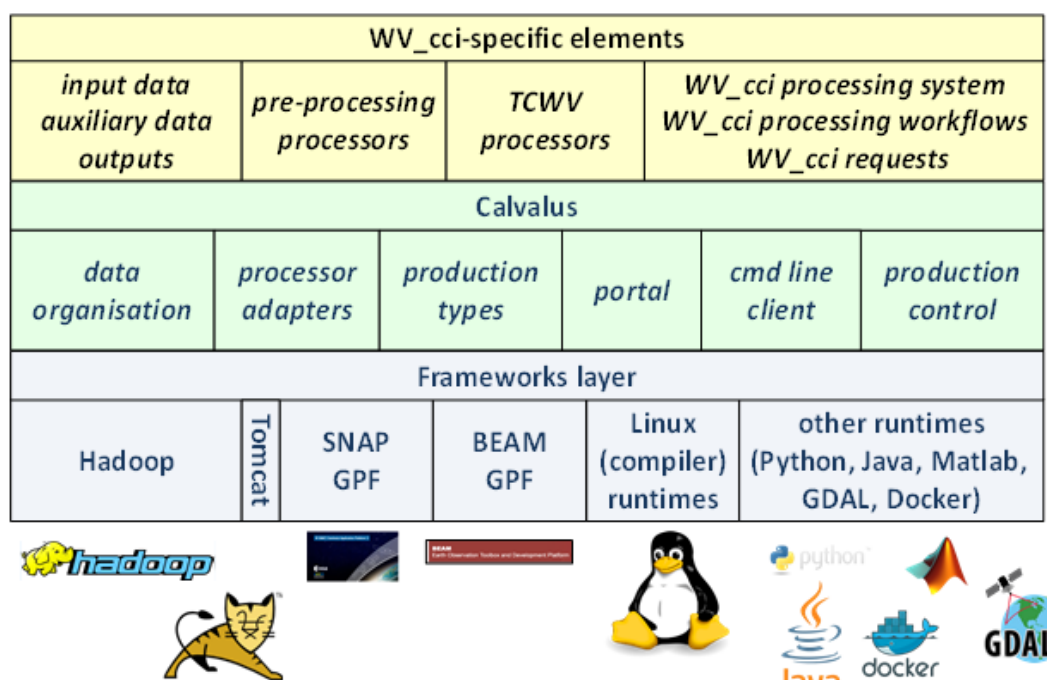
## 2.4 WV\_cci System Architecture

### 2.4.1 TCWV processing system (BC)

From the system requirement [TR-30] in the 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 (MOD021KM, optionally 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 [13], the CEDA JASMIN system [14], and Google Earth Engine [15]. 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. Therefore, the decision for WV\_cci was made to use JASMIN for MODIS TCWV L2 processing, and Calvalus for everything else, including 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.

### 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

CEDA provides the JASMIN and CEMS data processing environments. The JASMIN super-data-cluster is deployed on behalf of NCAS at the STFC Rutherford Appleton Laboratory (RAL) and supports the data analysis requirements of the UK and European climate and earth system modelling community. It consists of multi-Petabyte fast storage co-located with data analysis computing facilities, with satellite installations at Bristol, Leeds and Reading Universities. It shares infrastructure with an equivalent facility in the Earth Observation domain, for Climate and Environmental Monitoring from Space (CEMS).

JASMIN is deployed in the e-Science department at RAL to deliver three main functions: the infrastructure for the data storage and services of CEDA, including the NCAS British Atmospheric Data Centre; an environment for data intensive scientific computation for the climate and earth system science communities; and flexible access to high-volume and complex data for the climate and earth observation communities.

The WV\_cci project uses a dedicated group workspace within CEMS. 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, which facilitates these ECVs sharing a joint mini portal to their respective CDRs.

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.

An "Elastic Tape" system is available, which enables group workspace managers to manage secondary copies of their data and to move less-used data out to near-line tape to enable optimal use of high-performance disk space.

By deciding to migrate the WV\_cci system to the CEDA/CEMS infrastructure the project has gained the immediate benefit that the required large MODIS MOD021KM 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 JASMIN "super-data-cluster" is deployed on behalf of NERC at the STFC Rutherford Appleton Laboratory (RAL). JASMIN supports the data analysis requirements of the UK and European climate and earth system modelling community. It consists of multi-Petabyte fast storage co-located with data analysis computing facilities, with dedicated light paths to various key facilities and institutes within the community.

The whole JASMIN system architecture (services, how to use it, help documentation etc.) is described in-depth on the JASMIN web site, with [14] as entry point to get started. It employs the LSF software for scheduling and running batch processing jobs, which is described in detail [16].

## 2.4.2 VRWV Processing System (UoR)

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, POAM III, SMR, SCIAMACHY, MIPAS, ACE-FTS, ACE-MAESTRO, and Aura-MLS. 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 (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, ACE-FTS, ACE-MAESTRO, and IMS instruments. For the Aura MLS and IMS data, the data can be downloaded from the data websites directly, and for MIPAS, ACE-FTS, and ACE-MAESTRO, the data is available from WAVAS\_SAHAR distribution. At UoR, the RACC system would be able to handle both L2 and L3 data processing for all input data as well as the generation of merged product CDR-4.

The RACC system, i.e. Reading Academic Computing Cluster, 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. 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 [17].

## 3. TECHNICAL METHODS AND CONCEPTS

### 3.1 TCWV Processing System (BC)

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.

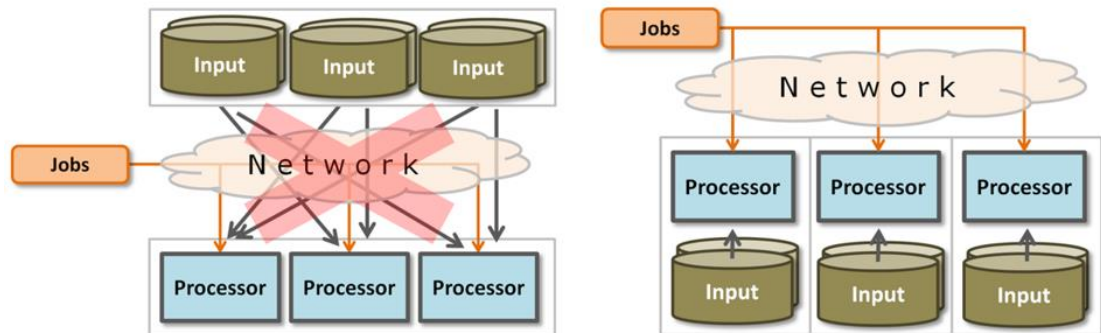
The Calvalus infrastructure has been described in a similar manner in various technical and project related documents. The description of the general concepts (not related to WV\_cci) in this chapter is mainly taken from [6] and still has a summary character to keep it in the scope of this document. A good starting point for many more technical details and also to get a view from the user's perspective would be the Calvalus Software User Manual [18].

#### 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 in order to ensure data availability. Data nodes communicate in order 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

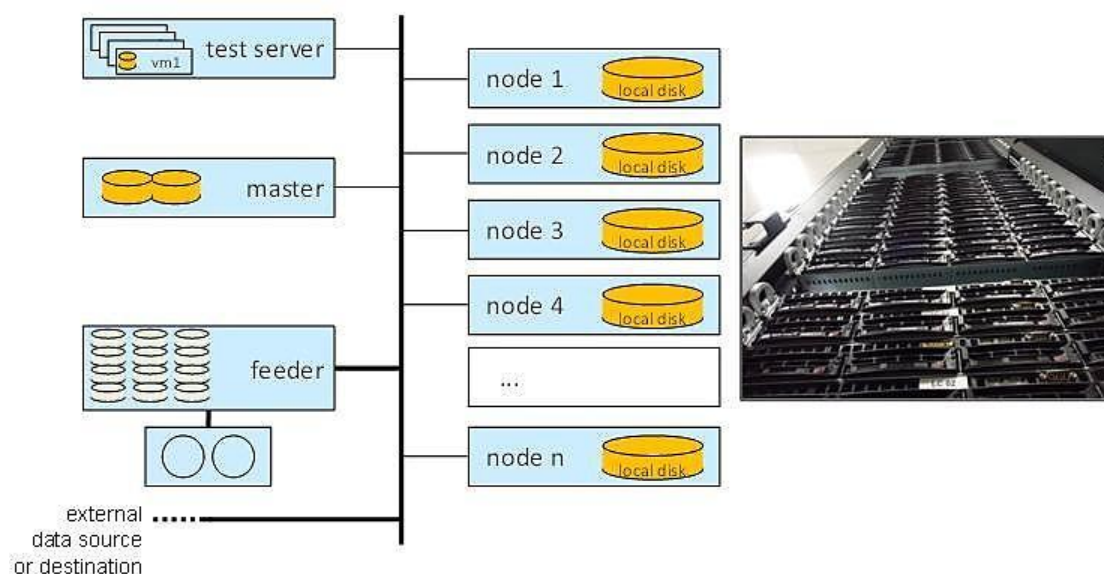
A software bundle is a Calvalus concept that allows system operators to deploy any runnable software to the system. The processors used in WV\_cci, which are provided by project partners, are integrated into software bundles, and as such integrated into the system.

#### 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**

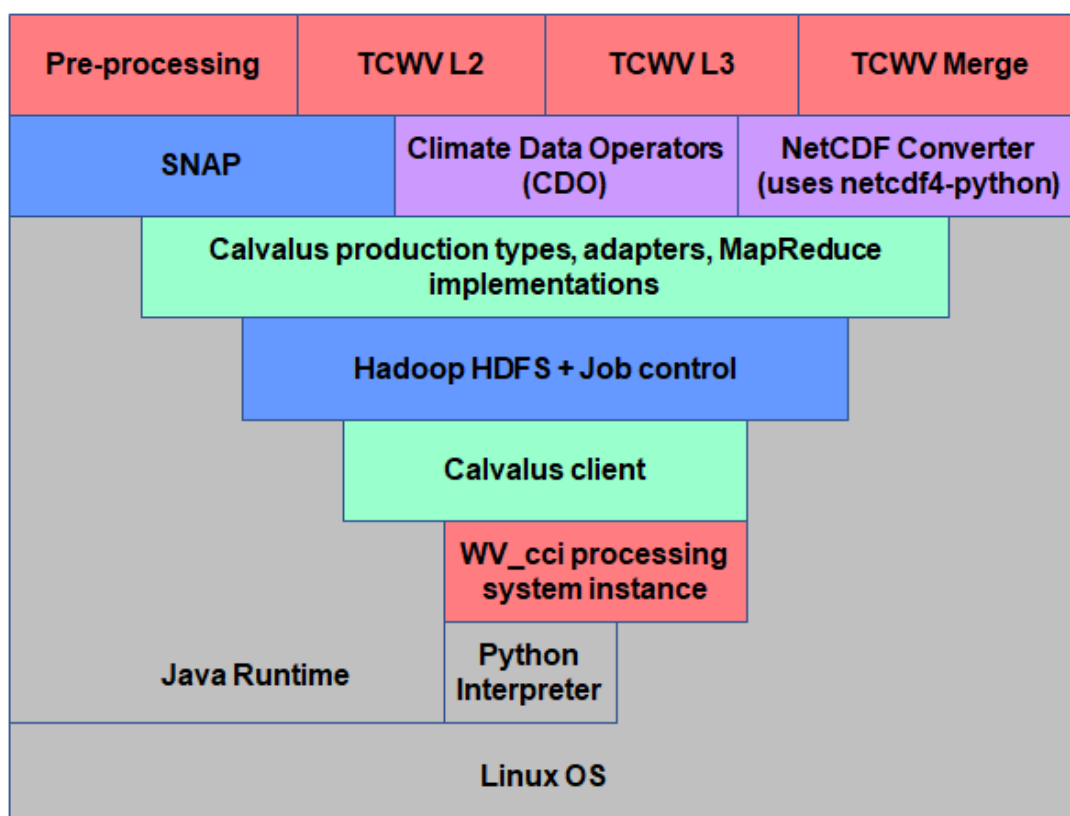
Infrastructure elements	Type	Number, size
Computing nodes	Rack-mounted standard computers, 1U, 1 quadcore CPU + hyperthreading, 16 GB main memory, 4 hard disks (11-16 TB), 1 Gbit network connection  (there are other nodes with 8 GB main memory and one node with two 6-core CPUs and 128 GB main memory in the same cluster)	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.	Overall storage capacity of the cluster is ~2.3 PB. The cluster is used with a replication of 2 for inputs and 1 for outputs which reduces this capacity accordingly.

Infrastructure elements	Type	Number, size
	In addition there are offline backups of input datasets at BC.	Sharing of input data among projects mitigates the need for storage for all data.
I/O elements	<p>“feeder”: (empty) disk array for 24 hot-pluggable hard disks, 4U, quadcore CPU, 8 or 16 GB main memory, 10 Gbit network connection.</p> <p>FTP server (ftp://ftp.brockmann-consult.de/), used for test and validation data exchange with DWD.</p>	<p>2 feeders for concurrent reading or writing of 48 disks.</p> <p>The FTP server is shared with other projects of BC. It is a virtual machine (VM) on a redundant VM server.</p>
Network infrastructure	<p>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.</p> <p>The cluster is connected with 1 Gbit to the BC internal network.</p> <p>The FTP server is connected to the BC internal network as well as to the internet.</p> <p>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.</p>	<p>4 switches 10 Gbit/1Gbit</p> <p>1 Gbit connection of cluster nodes</p> <p>10 Gbit connection between switches and to feeders</p> <p>Internet connection via the backbone of the Geesthacht Innovation and Technology Centre (GITZ).</p>
Operating system	Ubuntu 14.04 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.



**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 actually 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 a single 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 <a href="https://www.oracle.com/tech/network/java/index.html">https://www.oracle.com/tech/network/java/index.html</a>
Python interpreter	Basic software used for pmonitor.	Version 2.7.6, available from <a href="http://www.python.org">www.python.org</a>
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 2.7.3, versions maintained by Apache, available from <a href="http://hadoop.apache.org2">hadoop.apache.org2</a> (open source)

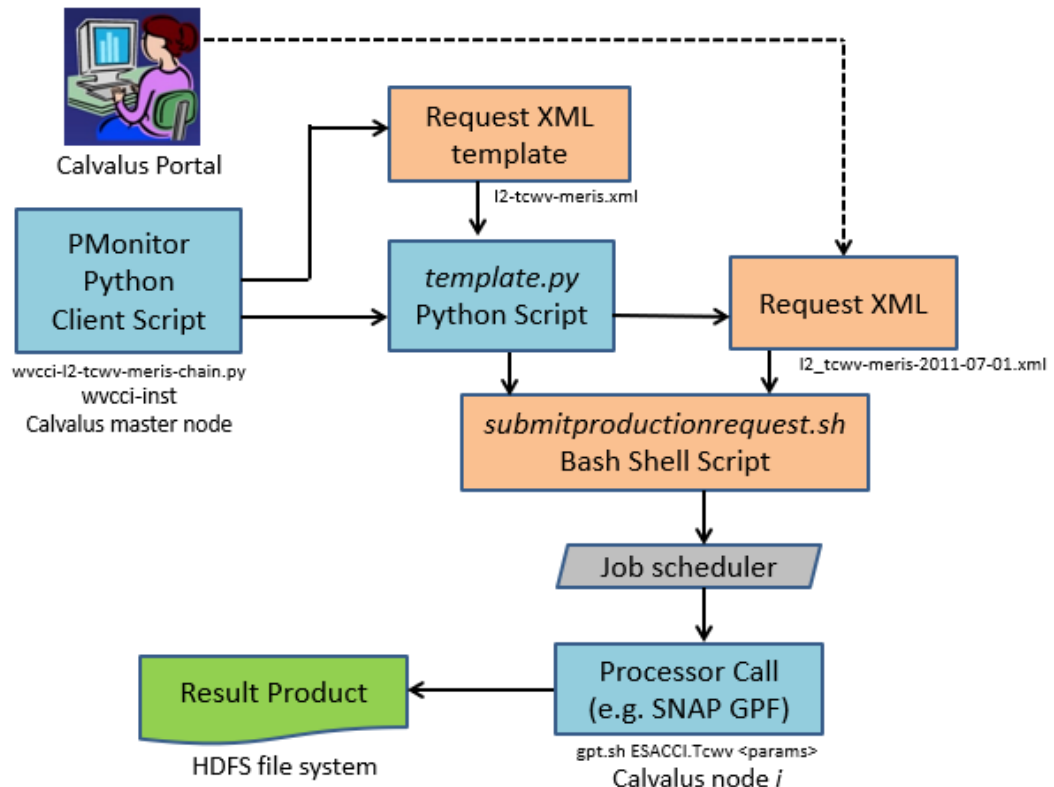
Software item	Purpose, use	Configuration control
Calvalus	<p>Earth Observation application layer on top of Hadoop with HDFS archive directory structure with archiving rules, client tool for request submission, software deployment, data ingestion processor adapters for various programming languages (among them for BEAM/SNAP operators processor deployment as versioned software bundles).</p> <p>User portal for on-demand processing.</p> <p>Processing system instances for controlled bulk production.</p>	Version 2.18, version control on GitHub in repository of BC
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 7.0.0  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 <a href="http://mpimet.mpg.de/cdo">http://mpimet.mpg.de/cdo</a>

Software item	Purpose, use	Configuration control
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-1.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-1.x software bundle (see below).
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-1.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-1.x software bundle (see below).

Software item	Purpose, use	Configuration control
snap-wvcci-1.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 processing 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 [18]. 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.



**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. Both explicit request and corresponding template are given in XML format, examples are listed in Appendix 3: Processing scripts. 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.

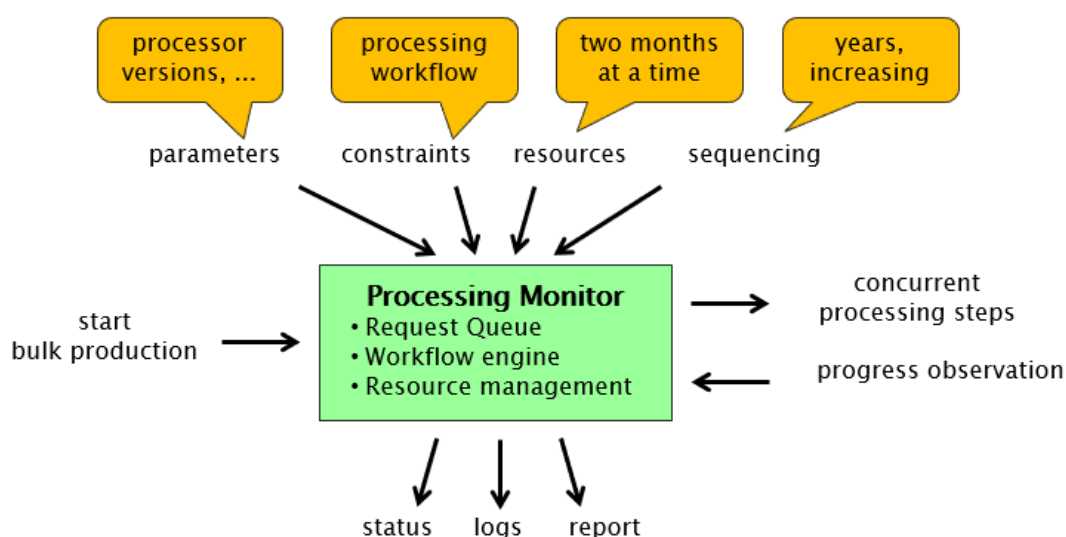


Figure 3-5: Concept of the PMonitor workflow engine. Taken from [18].

### 3.1.3 JASMIN

#### 3.1.3.1 Hardware environment

Together with CEMS, a total of 4.6 PB of fast storage is deployed at RAL, connected via its own low-latency network to the JASMIN and CEMS data compute facilities. Satellite systems at Bristol, Leeds and Reading Universities consist of significant disks (500, 100 and 150 TB, respectively) coupled with additional compute resources. It is envisaged that virtual machines for analysis can be constructed for particular purposes

at satellite sites, then migrated to the central system for processing against the major data store.

JASMIN provides both interactive and batch computing environments, recognising that scientists often need to develop and test workflows interactively before running those workflows efficiently at scale. The batch computing environment is the LOTUS processing cluster which is part of JASMIN. LOTUS is a group of physical machines, running the LSF workload manager, enabling efficient scheduling of larger data analysis tasks across nodes in the cluster as a single unit. Each node in the cluster is connected by 10Gbit/s Ethernet to JASMIN's high-performance 40Gbit/s core network. Nodes within LOTUS run the same stack of software and can access the same high-performance storage as the JASMIN Scientific Analysis servers, ensuring a consistent working environment for all phases of users' workflows. LOTUS currently has around 4000 cores, but is heavily used and implements a fair-share scheduling system between users. LOTUS can be accessed from various JASMIN scientific analysis servers. From these servers, it is possible to estimate and allocate resources for jobs, to submit, monitor and control jobs, via commands which interact with the LSF workload manager.

As mentioned in section 2.4.1.2, the whole JASMIN system including the hardware environment is described in-depth on the JASMIN web site [14].

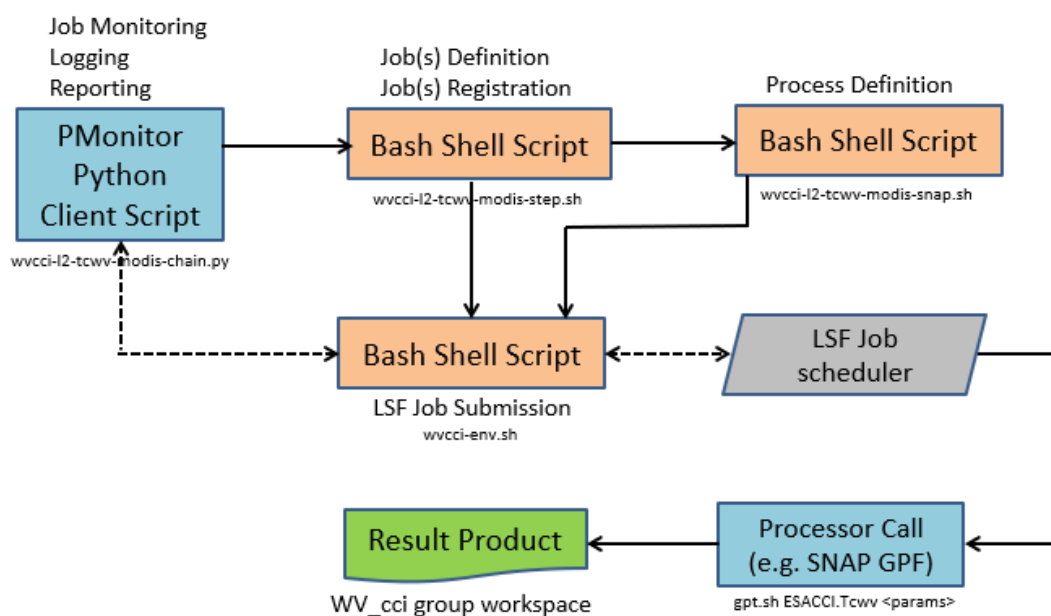
### 3.1.3.2 Job submission and processing environment

As said previously, JASMIN offers to submit commands to the LOTUS cluster, which are scheduled and executed by that engine. See [19] for details. The commands which are submitted may be any executable which is able to run within the prepared environment. Hence, in order 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 short, the LSF parallel processing approach on JASMIN is quite different from Calvalus, as we do not have the 'data local' approach here. On Calvalus, the software (e.g. SNAP) is brought to the processing nodes where the data are stored ('bring the software to the data'), whereas on JASMIN both data and software are stored (or are at least made accessible via symbolic links) in the user's home directory or in the group workspace of the project, from where they can be accessed by all nodes of the LOTUS cluster. However, the PMonitor workflow engine is used for the job control on JASMIN as well and serves as entry point for the job submission into the LOTUS cluster.



Figure 3-6 illustrates the bulk processing setup on JASMIN/LOTUS. As example, the processing of MODIS TCWV L2 for a given time period is considered (equivalent to the MERIS example on Calvalus). 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 LSF 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 3-6: Scheme for bulk processing setup on JASMIN. Names of scripts for MODIS TCWV L2 processing are given as examples.**

### 3.2 VRWV Processing System (UoR)

The VRWV processing system relies on the Reading Academic Computing Cluster (RACC) system. This section introduces the hardware and software infrastructure of the RACC.

### 3.2.1 RACC hardware environment

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). RACC supports both interactive research computing and batch job submission. The interactive computing relies on the login nodes on 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. 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. 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 [17].

### 3.2.2 RACC processing environment

As said previously, 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 3-7 shows the schematic layout of the RACC system and the computer nodes provide around 1000 CPU cores for batch jobs.

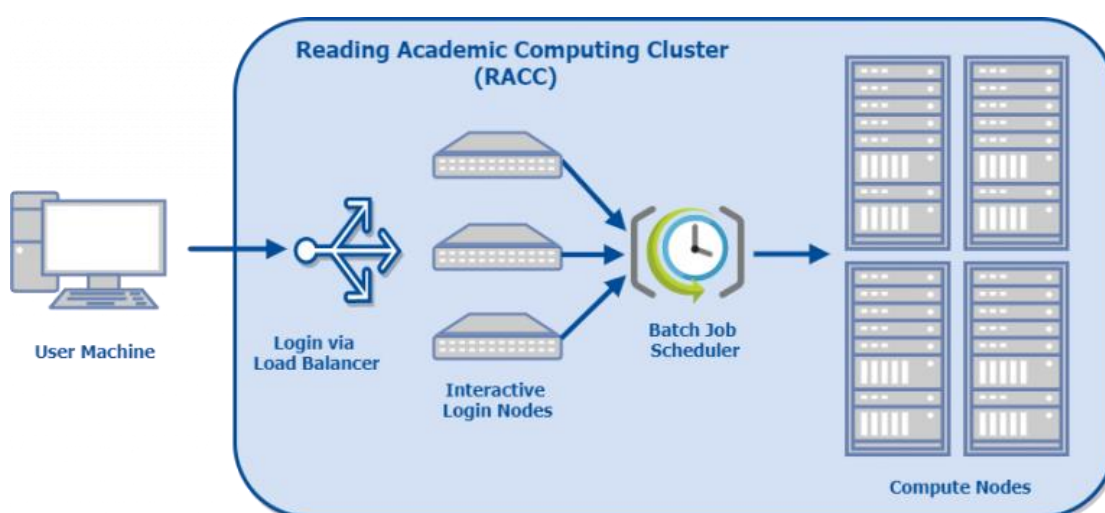


Figure 3-7: Schematic layout of the RACC, taken from [17].

### 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 distribution of the Interactive Data Language by Exelis Visual Information Solutions (IDL version 8.3.0). IDL is vectorized, 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 3.5, with the RACC offering a run environment packaged with the operating system. 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.

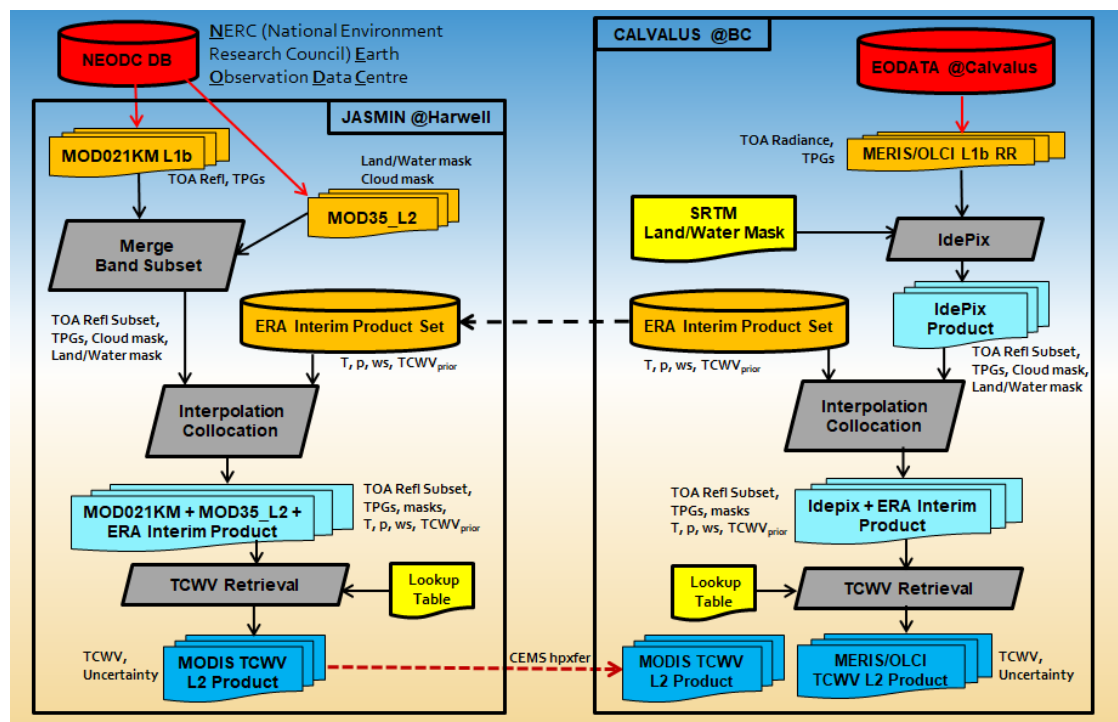
## 4. WORKFLOWS

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 (BC)

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. The various components of the flows 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 Reduced Resolution TOA radiance products [20]
- Land/water mask shapefiles derived from the SRTM (Shuttle Radar Topography Mission, [21], [22]) dataset
- ERA Interim [23] ancillary variables required for TCWV retrieval algorithm.

##### MERIS L1b Reduced Resolution

The MERIS L1b Reduced Resolution 3rd reprocessing full archive (06/2002–04/2012) is available at Calvalus and can directly be accessed for processing<sup>1</sup>. This dataset has already been used by a variety of projects.

##### 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 node<sup>2</sup> where the processing job is performed, following the 'data local' principle for job scheduling as illustrated earlier in section 3.1.1.

---

<sup>1</sup> If possible, it is planned to use data from the 4<sup>th</sup> reprocessing for the generation of the final TCWV datasets. At compilation time of this version 1 of the SSD, the MERIS L1b data from the 4<sup>th</sup> reprocessing are still under review at ESA and not yet publicly available.

<sup>2</sup> On the Calvalus processing nodes operating under Linux, the user is 'yarn', so the SNAP application directory is '/home/yarn/.snap'

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.

#### ERA Interim ancillary variables

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).

For a reliable TCWV retrieval it is usually not sufficient to assume 'reasonable' constant values for these variables. Therefore, ERA Interim reanalysis data from ECMWF are used for WV\_cci. These datasets are available from the ECMWF public datasets archive [24] 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). All required files from 2002 to 2018 have been downloaded and are now available on Calvalus for direct access.

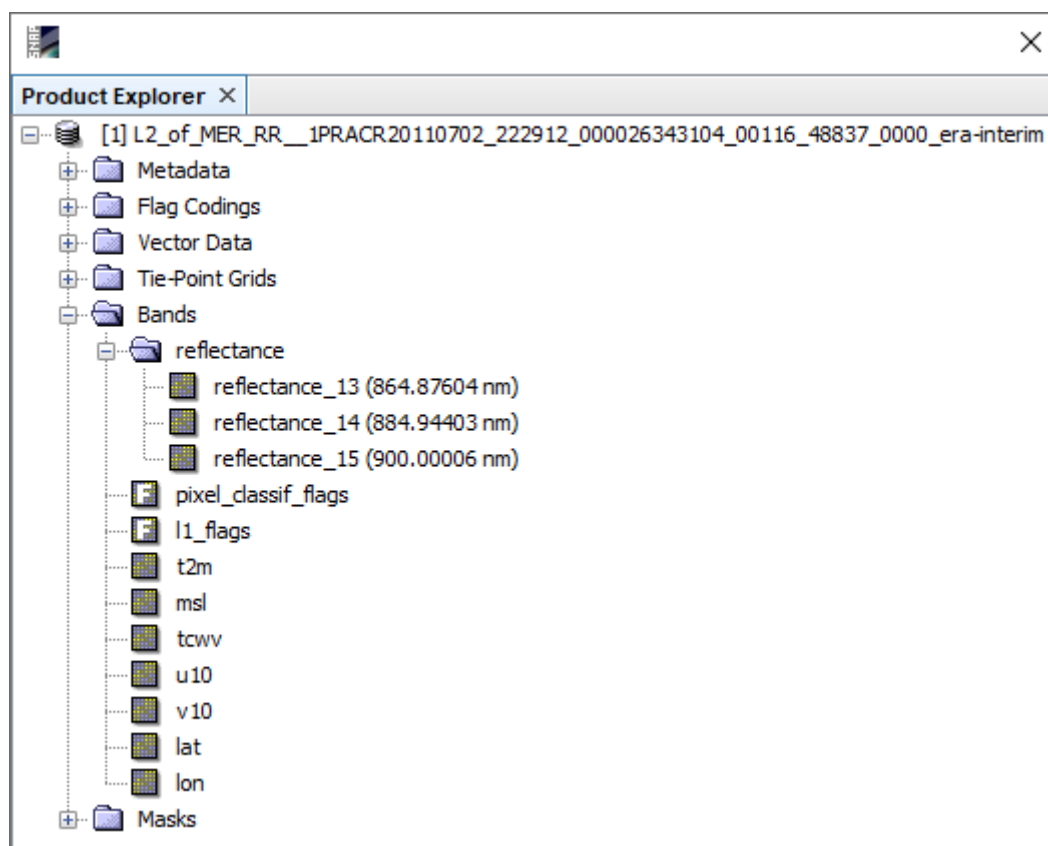
#### 4.1.1.1.2 Preprocessing

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 ERA interim variables listed above;
- ERA Interim interpolation and colocation: For TCWV L2 processing from a given L1b product, the ERA Interim data needs to be mapped in space and time onto the 'master' grid of the L1b product.

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. The ERA Interim interpolation and reprojection is done with appropriate CDO commands invoked from a Unix executable which is available on Calvalus for parallel execution. It is listed in Appendix 3: Processing scripts, section 2. The subsetting and merging is also done with a SNAP GPF processor which is provided with the snap-wvcci-1.x software bundle (see Table 3-2).

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

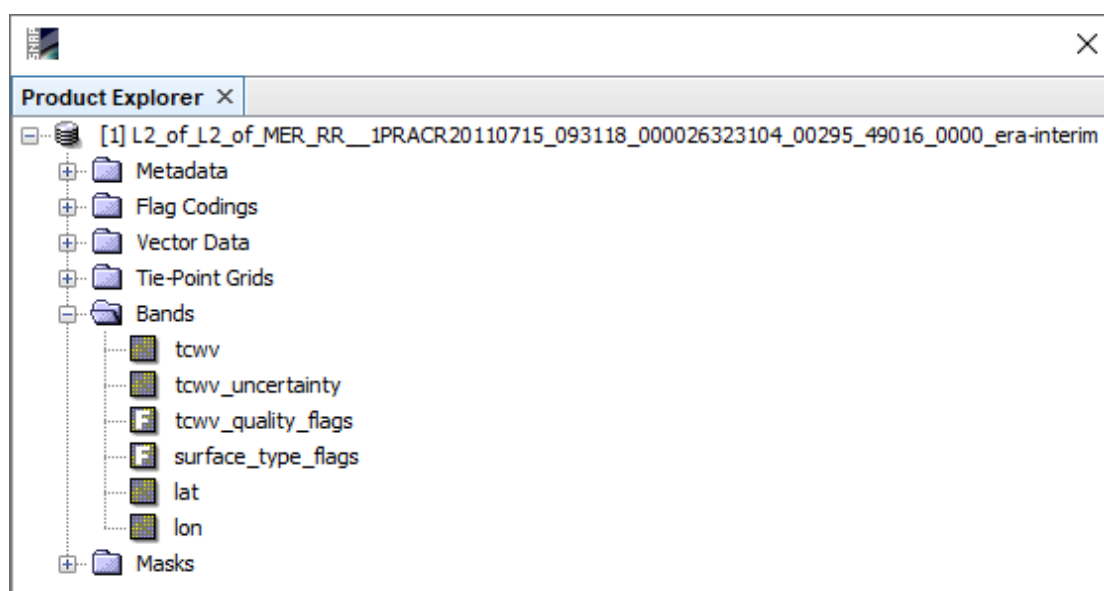


**Figure 4-2: SNAP Product Explorer: MERIS IdePix pixel classification product after merge with reflectance bands and ERA Interim auxiliary variables.**

#### 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.



**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) Calibrated Radiances 5-Min L1B Swath 1km products [25], [26];
- MODIS MOD35\_L2 Cloud Mask and Spectral Test Results 5-Min L2 Swath 250m and 1km products [27];
- ERA Interim ancillary variables required for TCWV retrieval algorithm (as for MERIS).

#### MODIS MOD021KM

Starting with year 2005, the MODIS MOD021KM archive has been completely uploaded by CEDA into their archive to the NEODC database [28]. As can be seen from [29], the number of available products is obviously close to the theoretical value. For by far most of the days, there are 288 products (every 5 minutes) available as expected. 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 2006, the MODIS MOD35\_L2 archive has also been completely uploaded by CEDA into their archive to the NEODC database. The data coverage is obviously as good as for MOD021KM. 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

The ERA Interim datasets are also used in the same way as for MERIS. All products have been transferred once from Calvalus to JASMIN and 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 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 TCWV 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 ingested<sup>3</sup> 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.

---

<sup>3</sup> For the generation of 'Dataset 1' and 'Dataset 2' which are the baseline in this version 2 of the SSD, DWD provided data from CM SAF HOAPS obtained from the SSM/I and SSM/IS instruments flying onboard platforms DMSP 5D-3/F16-18.

#### 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:

- OLCI L1b Reduced Resolution TOA radiance products [30]
- ERA Interim [23] ancillary variables required for TCWV retrieval algorithm.

##### OLCI L1b Reduced Resolution

The full archive of OLCI Level 1b reduced resolution products (Earth Observation Mode, OL\_1\_ERR), from 04/2016 to present, is available at Brockmann Consult. As for the MERIS data, these products can directly be accessed from the Calvalus cluster for processing. This dataset has already been used for the WV CCI test processing cycles and is also actually used by other projects.

##### ERA Interim ancillary variables

The acquisition of ERA Interim ancillary variables works exactly in the same way as described for MERIS in section 4.1.1.1.1.

##### 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 ERA interim variables listed in section 4.1.1.1.1;
- ERA Interim interpolation and colocation: For TCWV L2 processing from a given L1b product, the ERA Interim data needs to be mapped in space and time onto the 'master' grid of the L1b product. Again, same as for MERIS.

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 + ERA Interim Product'. This product in return serves again as input for the TCWV L2 retrieval.

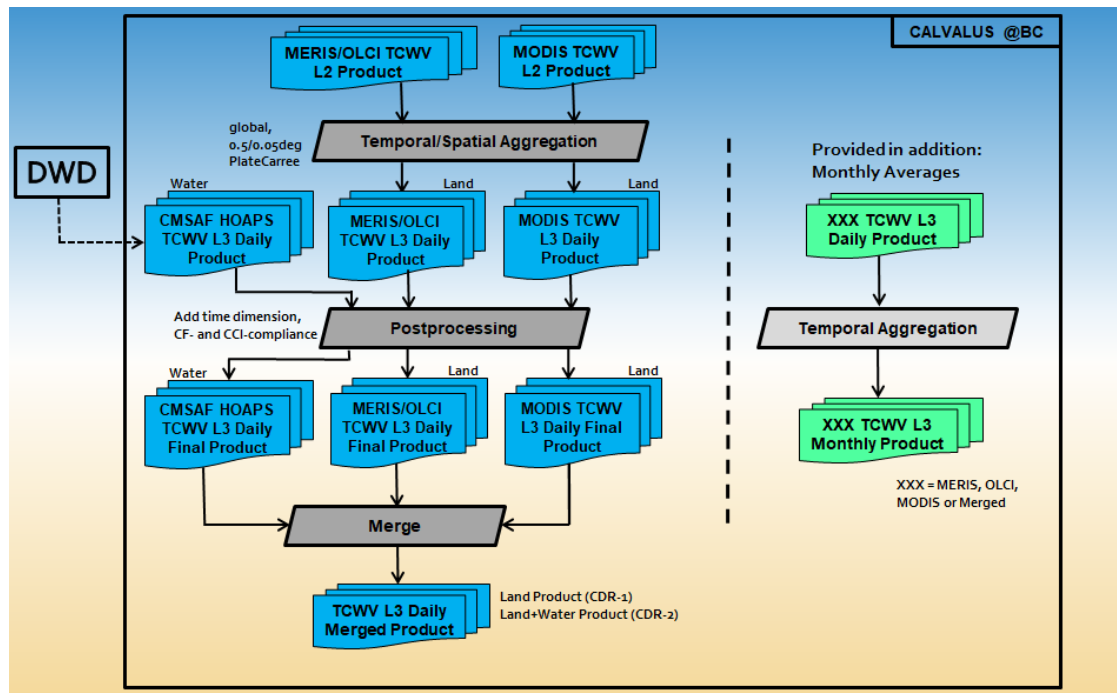
#### 4.1.1.3.3 TCWV retrieval

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.



**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

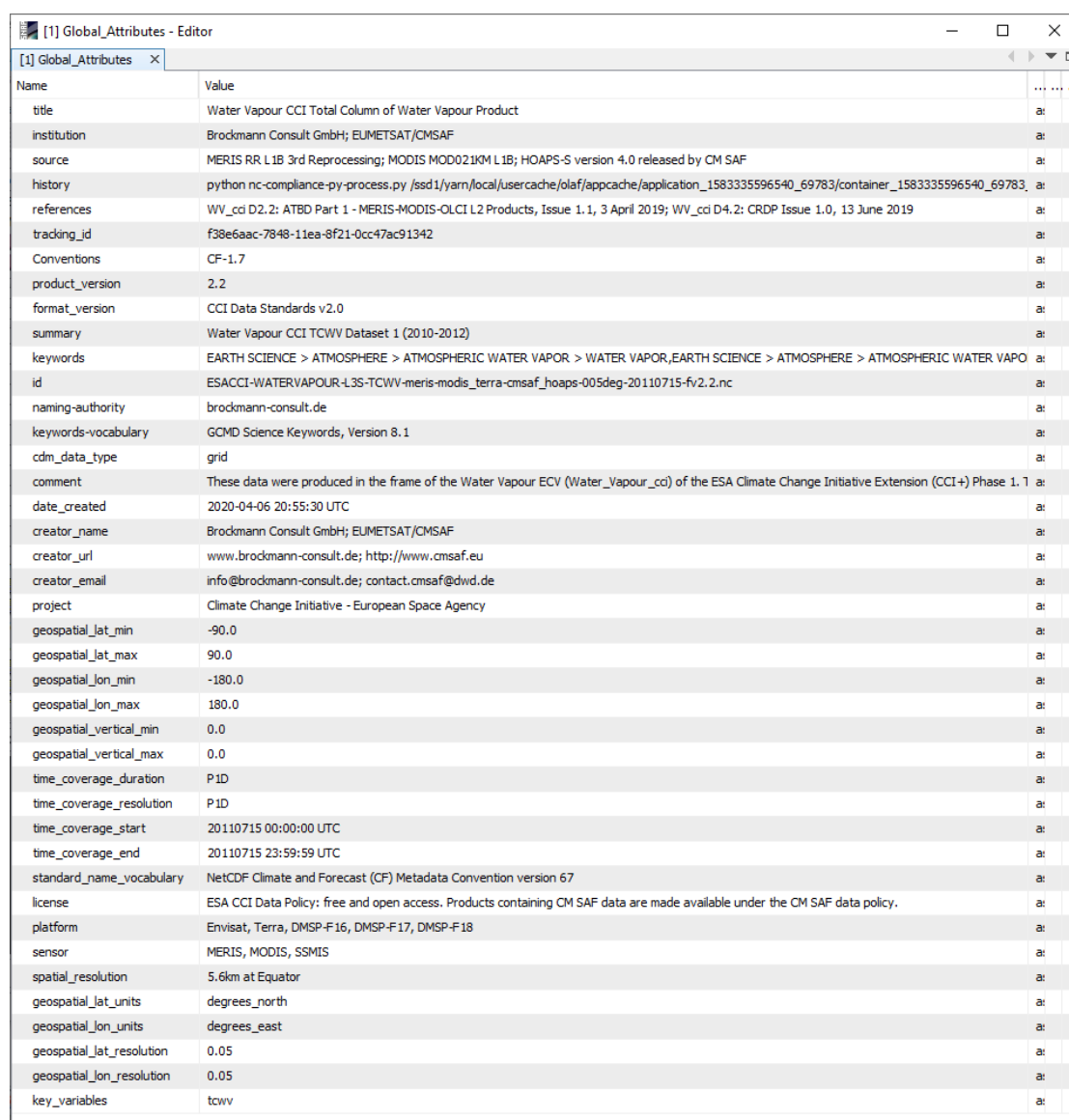
All TCWV L2 products obtained for a particular day from a given NIR instrument (MERIS, MODIS or OLCI) as described in the previous section are taken as input for a temporal and spatial level 3 aggregation. The time aggregation window is the given day, the spatial aggregation is done globally with a reprojection 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.



Name	Value	
title	Water Vapour CCI Total Column of Water Vapour Product	a:
institution	Brockmann Consult GmbH; EUMETSAT/CMSAF	a:
source	MERIS RR L1B 3rd Reprocessing; MODIS MOD021KM L1B; HOAPS-S version 4.0 released by CM SAF	a:
history	python nc-compliance-py-process.py /ssd1/yarn/local/usercache/olaf/appcache/application_1583335596540_69783/container_1583335596540_69783	a:
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 1.0, 13 June 2019	a:
tracking_id	f38e6aac-7848-11ea-8f21-0cc47ac91342	a:
Conventions	CF-1.7	a:
product_version	2.2	a:
format_version	CCI Data Standards v2.0	a:
summary	Water Vapour CCI TCWV Dataset 1 (2010-2012)	a:
keywords	EARTH SCIENCE > ATMOSPHERE > ATMOSPHERIC WATER VAPOR > WATER VAPOR,EARTH SCIENCE > ATMOSPHERE > ATMOSPHERIC WATER VAPO	a:
id	ESACCI-WATERVAPOUR-L3S-TCWV-meris-modis_terra-cmsaf_hoaps-005deg-20110715-fv2.2.nc	a:
naming-authority	brockmann-consult.de	a:
keywords-vocabulary	GCMD Science Keywords, Version 8.1	a:
cdm_data_type	grid	a:
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. 1	a:
date_created	2020-04-06 20:55:30 UTC	a:
creator_name	Brockmann Consult GmbH; EUMETSAT/CMSAF	a:
creator_url	www.brockmann-consult.de; http://www.cmsaf.eu	a:
creator_email	info@brockmann-consult.de; contact.cmsaf@dwd.de	a:
project	Climate Change Initiative - European Space Agency	a:
geospatial_lat_min	-90.0	a:
geospatial_lat_max	90.0	a:
geospatial_lon_min	-180.0	a:
geospatial_lon_max	180.0	a:
geospatial_vertical_min	0.0	a:
geospatial_vertical_max	0.0	a:
time_coverage_duration	P1D	a:
time_coverage_resolution	P1D	a:
time_coverage_start	20110715 00:00:00 UTC	a:
time_coverage_end	20110715 23:59:59 UTC	a:
standard_name_vocabulary	NetCDF Climate and Forecast (CF) Metadata Convention version 67	a:
license	ESA CCI Data Policy: free and open access. Products containing CM SAF data are made available under the CM SAF data policy.	a:
platform	Envisat, Terra, DMSP-F16, DMSP-F17, DMSP-F18	a:
sensor	MERIS, MODIS, SSMIS	a:
spatial_resolution	5.6km at Equator	a:
geospatial_lat_units	degrees_north	a:
geospatial_lon_units	degrees_east	a:
geospatial_lat_resolution	0.05	a:
geospatial_lon_resolution	0.05	a:
key_variables	tcwv	a:

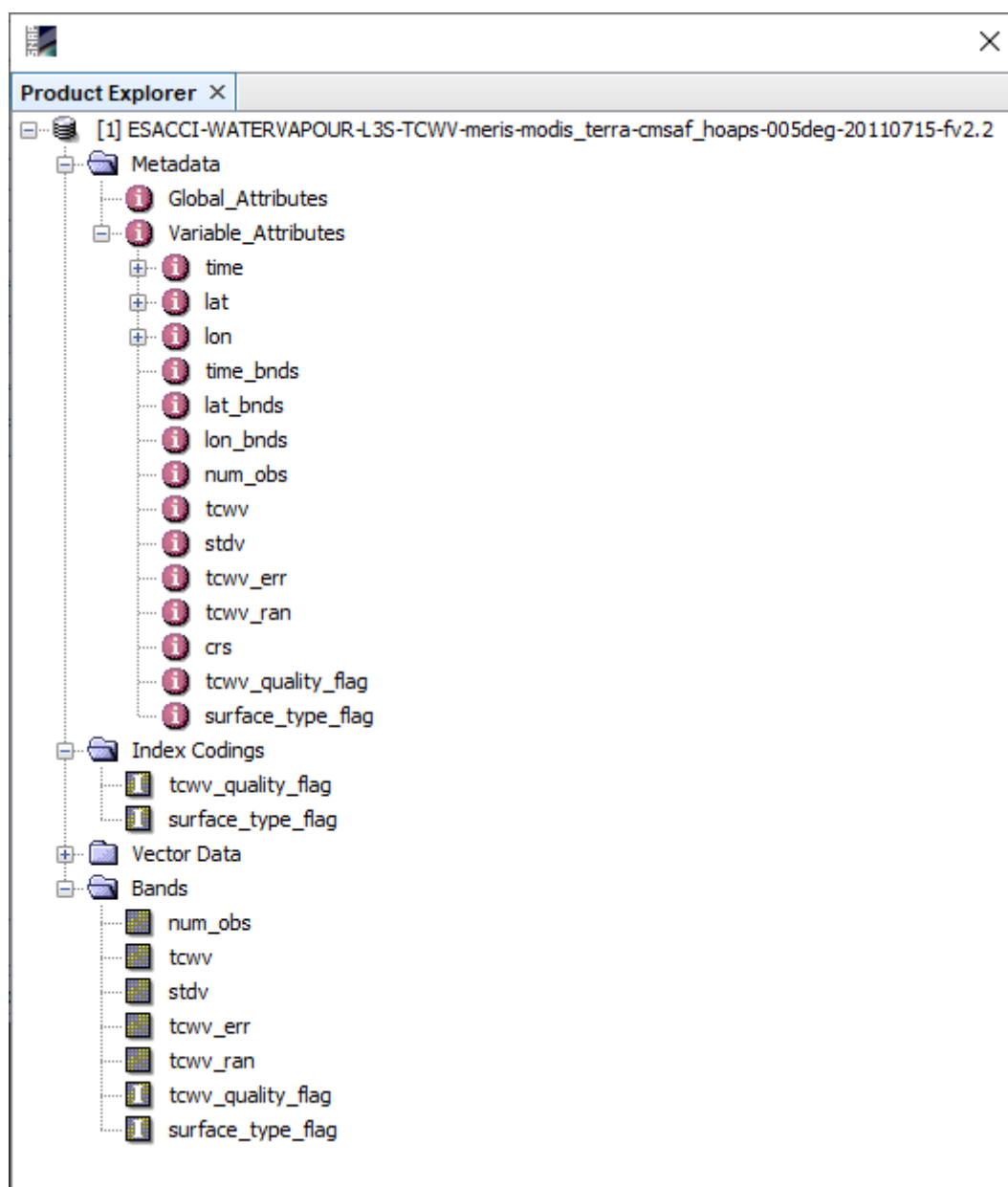
**Figure 4-5: SNAP Product Explorer: NetCDF global attributes of a TCWV L3 daily final product.**

#### 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.



**Figure 4-6: SNAP Product Explorer: TCWV L3 daily final product.**

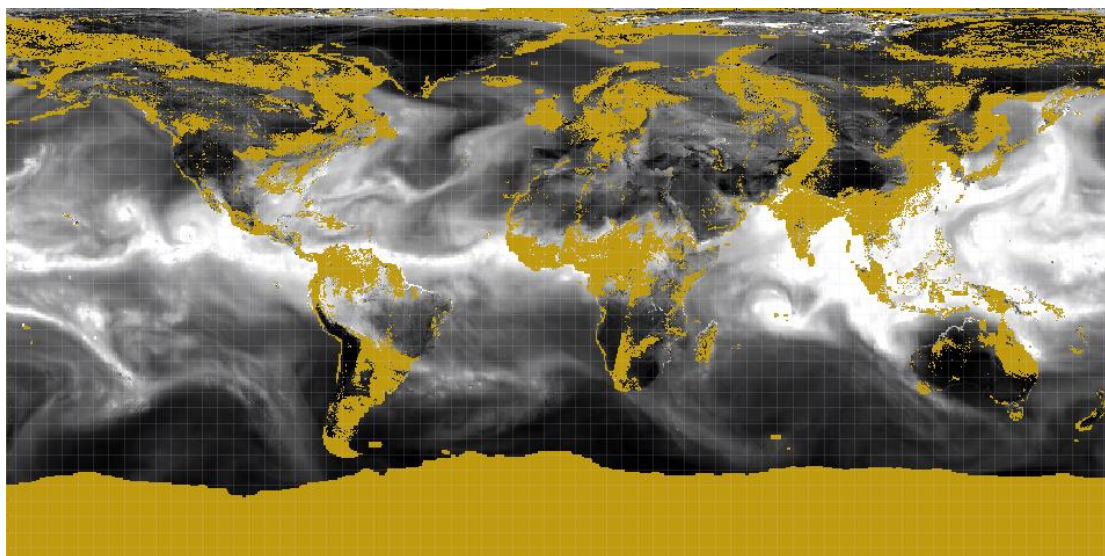
The product geolocation information is given through the 'crs' variable and, following CCI standards, also through 1D variables 'lat' and 'lon'. As 2D raster variables we have:

- 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 [31]

The TCWV global coverage for a single day is illustrated in Figure 4-7. Further examples are shown in the SVR [32].

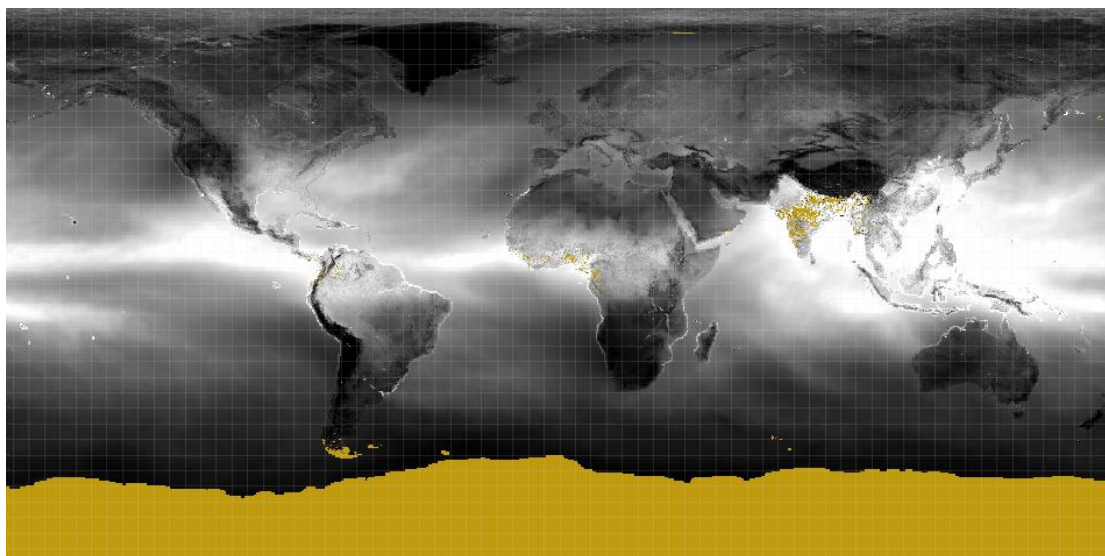


**Figure 4-7: OLCI/MODIS/HOAPS TCWV L3 daily merge for July 15th, 2016 (greyscale, 0-70 kgm-2). Yellow indicates no data.**

#### 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 [31]. 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 [32], 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 kgm<sup>-2</sup>). Yellow indicates no data.**

## 4.2 VRWV Processing System (UoR)

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 4-9 for details.

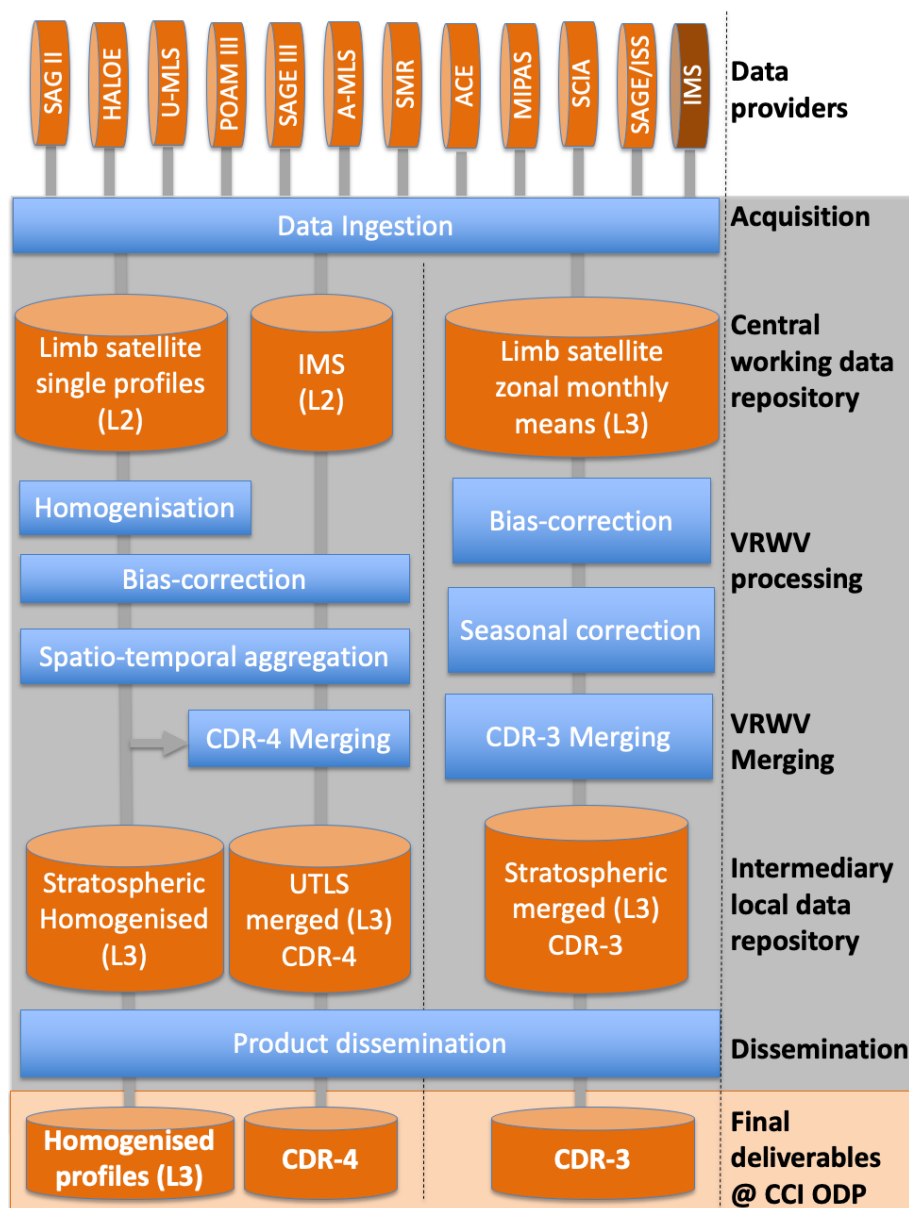
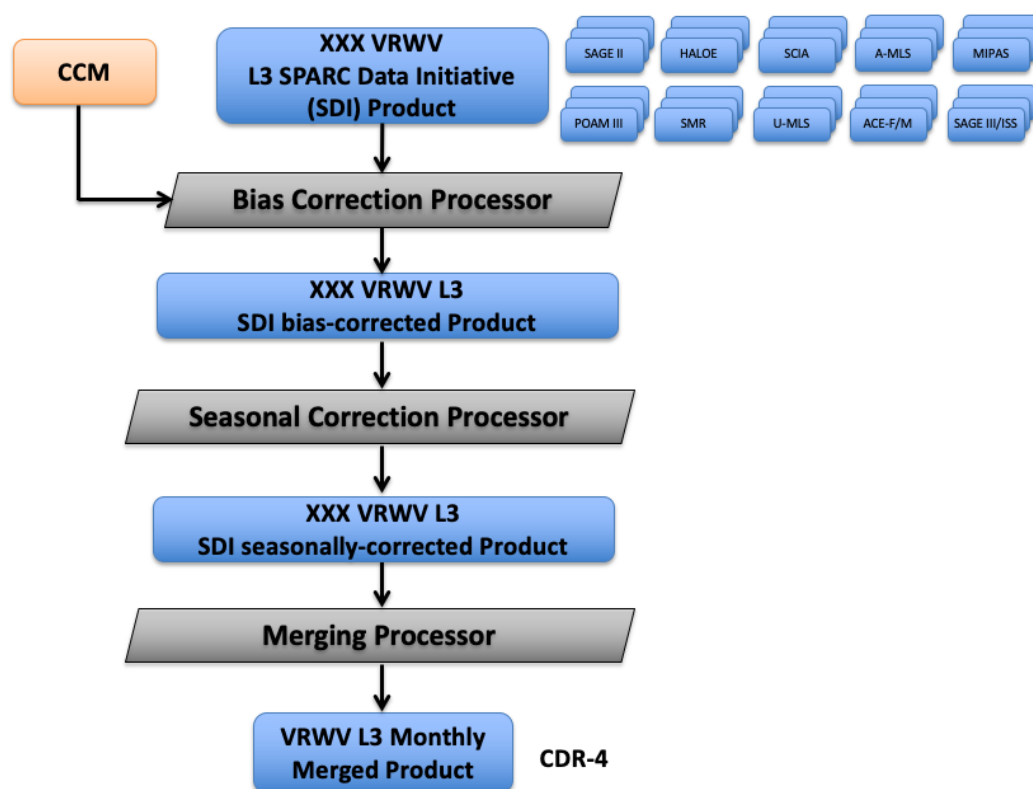


Figure 4-9: System definition of the VRWV processing systems for CDR-3 and CDR-4. Updated from [5].

#### 4.2.1 VRWV CDR-3 processing

Figure 4-9 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 different chemistry–climate models 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 4-10 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.

##### The SPARC Data Initiative L3 products

The SPARC Data Initiative L3 zonal monthly mean products [33] are produced by the data providers from L2 observations (national agencies and institutions) and are being made available via the Pangea data archive. The different instruments' zonal monthly mean data products are available over different time periods as specified here in [33]. 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 4-10).

##### Chemistry-climate model fields

The chemistry-climate model water vapour fields from specified dynamics simulations can be accessed via the CEDA data archive. All products have been transferred once from CEDA and are now available as static ancillary input datasets in the WV\_cci workspace on RACC (top level depicted in Figure 4-10).

#### 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 [34]. 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

[Details of this processor will be added in the final version of the SSD only.]

#### 4.2.1.4 Merging of L3 products

The final processing step for the monthly 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 monthly zonal mean VRWV L3 product after this final step is shown in Figure 4-11.

#### 4.2.1.5 NetCDF product formatting

As can be seen from Figure 4-10, 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. The content of a merged monthly VRWV L3 end product after this final step is shown in Figure 4-11. 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 error terms (h2ovmr\_stdv, vrvw\_err, vrvw\_ran) are to be added into the product in its next version. Note, further adjustments will need to be made to make this product compatible with the CCI formatting standards.

```
netcdf file:/DOCS/CMAM20/MERGING/H2O_1985-2020_v0.nc {
  dimensions:
    lat = 36;
    pressure = 28;
    time = UNLIMITED; // (432 currently)
  variables:
    float h2ovmr(time=432, pressure=28, lat=36);
      :units = "vmr";
      :long_name = "water vapour volume mixing ratio";
      :standard_name = "water vapour";

    float h2ovmr_std(time=432, pressure=28, lat=36);
      :long_name = "water vapour standard deviation vmr";
      :units = "vmr";

    float lat(lat=36);
      :units = "degrees_north";
      :axis = "X";
      :long_name = "latitude";
      :standard_name = "latitude";

    float pressure(pressure=28);
      :units = "hPa";
      :axis = "Z";
      :long_name = "pressure";
      :standard_name = "pressure";

    double time(time=432);
      :units = "year";
      :axis = "T";
      :long_name = "time";
      :standard_name = "time";

  // global attributes:
  :variable_id = "sdi_std";
  :units = "volume mixing ratio";
  :activity_id = "ESA WV_cci";
  :license = "ESA WV_cci. The data producers and data providers make no warranty,
  :history = "v0.0 merged water vapour product";
  :institution = "University of Reading, Reading, UK";
  :contact = "Michaela Hegglin (m.i.hegglin@reading.ac.uk)";
  :creation_date = "2020-01-29T12:00:00Z";
  :source_version = "0.0";
}
```

**Figure 4-11: Panoply product explorer: merged VRWV monthly zonal mean L3 final product (prototype v0).**

## 4.2.2 VRWV CDR-4 processing

Figure 4-12 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, ACE-FTS, ACE-MAESTRO, and IMS and the ancillary variables in the derived meteorological products from NASA JPL. For the bias correction processing, the observations of WV profiles from GRUAN are used. 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 v4.2 VRWV products [35]
- WAVAS\_SAHAR dataset

- IMS L2 VRWV Product
- GRUAN VRWV profiles [36]
- 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 harmonized 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.

#### GRUAN VRWV profiles

The GRUAN WV profiles are available on the RACC and can be directly accessed for processing.

#### DMP ancillary variables

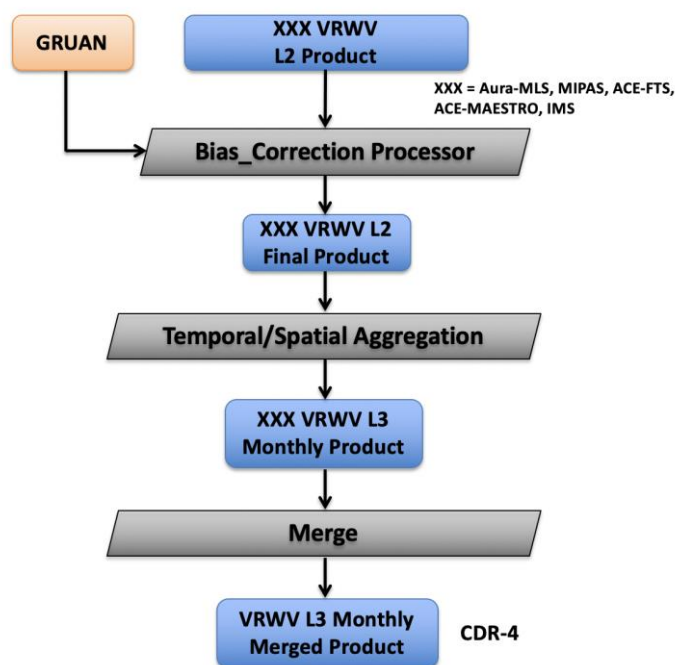
The derived meteorological products are provided by NASA JPL for each instrument with multiple ancillary variables. The following ancillary variables are used as input:

- 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. This processor uses *in situ* VRWV profiles from GRUAN observations to assess the biases of the L2 satellite data, which then are used to reduce the bias in the L2 data. 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.



**Figure 4-12: 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, ACE-FTS, ACE-MAESTRO, or IMS) are taken as input for a temporal and spatial level 3 aggregation. The aggregation is performed for each month in 2010–2014 with a horizontal resolution of 5 degrees by 5 degrees in latitude and longitude (for now, the possibility of a finer resolution grid will be explored for the production of the the final data set). In the vertical dimension, the L2 VRWV profiles are interpolated to pressure levels as specified in the PSD [2]. This results in five VRWV L3 products for all available instruments (MLS, MIPAS, ACE-FTS, ACE-MAESTRO, 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 2010 to 2014. 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 variables 'pressure levels'. The three-dimensional VRWV profile is provided with the variable 'vmrh2o' in the unit of ppmv. The statistical quantities and error terms (stdv, vrwv\_err, vrwv\_ran) is to be added into the products in next version.

#### 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 requirements (WVI-SR-xxxx) of the SRD [13] to sections within this document (§).

Requirement	Subject	Reference
WV-SR-0010	Generate and disseminate WV products	§4
WV-SR-0100	Processing workflow and scenarios	§4
WV-SR-0200	Output products and stability	§4.1.2, §3.1
WV-SR-0300	Data dissemination	§4.1.2
WV-SR-0400	Processing system evolution	§3.1
WV-SR-1100	Production control	§3.1
WV-SR-1110	Failure handling	§3.1
WV-SR-1120	Transfer to operations	§3.1
WV-SR-1130	Automated preprocessing	§4.1.1.1.2, §4.1.1.2.2
WV-SR-1140	Automated WV processing	§3.1
WV-SR-1200	Integration of TCWV processors	§3.1, §4.1
WV-SR-1210	Radiance-to-Reflectance processor	§4.1.1.1.2
WV-SR-1220	Idepix processor	§4.1.1.1.2, §4.1.1.2.2
WV-SR-1230	TCWV preprocessor	§4.1.1.1.2, §4.1.1.2.2
WV-SR-1240	TCWV L2 retrieval processor	§4.1.1.1.3, §4.1.1.2.3

Requirement	Subject	Reference
WV-SR-1250	TCWV L3 processor	§4.1.2
WV-SR-1260	TCWV L3 sensor merge processor	§4.1.2.1.3
WV-SR-1300	Ingestion of input data for TCWV retrieval	§4.1.1.1.1, §4.1.1.2.1
WV-SR-1310	L1 input for TCWV	§4.1.1.1.1, §4.1.1.2.1
WV-SR-1320	L2 input for TCWV	§4.1.1.1.1, §4.1.1.2.1
WV-SR-1330	Anxiliary data input for TCWV	§4.1.1.1.1, §4.1.1.2.1
WV-SR-1340	Ingestion of HOAPS products	§4.1.2.1.1, §4.1.2.1.3
WV-SR-1350	Updates of input data	§4.1.1.1.1, §4.1.1.2.1
WV-SR-1400	Integration of VRWV processors	§3.2, §4.2
WV-SR-1410	VRWV pre-processor (CDR-4 only)	§4.2.2.2
WV-SR-1420	VRWV L2 processor (CDR-4 only)	§4.2.2.2
WV-SR-1430	VRWV L3 processor	§4.2.1.2, §4.2.1.3, §4.2.2.3
WV-SR-1440	VRWV L3 merging processor	§4.2.1.4, §4.2.2.4
WV-SR-1500	Ingestion of input data for VRWV retrivals	§4.2.1, §4.2.2
WV-SR-1510	L2 or L3 input	§4.2.1.1, §4.2.2.1

Requirement	Subject	Reference
WV-SR-1520	Auxiliary data input	§4.2.1.1, §4.2.2.1
WV-SR-1530	Updates of input data	§4.2.1.1, §4.2.2.1
WV-SR-1600	Production performance	§3.1.2, §3.1.3
WV-SR-1610	Processing concurrency	§3.1.2.4, §3.1.3.2
WV-SR-1620	Map reduce for TCWV L3	§3.1.1
WV-SR-2100	Stability and quality of outputs	§4.1.2.1, §4.1.2.2
WV-SR-2110	Processing quality control	§3.1.2.4, §3.1.3.2
WV-SR-2120	Size of generated products	§4.1.2.1, §4.1.2.2
WV-SR-2130	Archive capacity	§3.1.2.2, §3.1.3.1
WV-SR-3100	Product distribution	§4.1.2.1, §4.1.2.2
WV-SR-3110	Data inventory	§3.1.2.2, §3.1.3.1
WV-SR-3120	Structured data storage	§3.1.2.2, §3.1.3.1
WV-SR-3130	On demand data delivery	§3.1.2.2, §3.1.3.1
WV-SR-3140	FTP access	§3.1.2.2, §3.1.3.1

Requirement	Subject	Reference
WV-SR-3150	NetCDF4 format	§4.1.2.1.2
WV-SR-4100	Evolution and improvements	§2.1, §2.2, §3, §4
WV-SR-4110	Expert driven validation	§2.1, §2.2, §3, §4
WV-SR-4120	GCOS requirements	§2.1, §2.2, §3, §4
WV-SR-4130	Intermediate and final processing results	§4
WV-SR-4200	Interaction with users and groups of scientific experts	§2.1, §2.2, §3, §4
WV-SR-4210	Information about products	§2.1, §2.2, §4
WV-SR-4300	Reprocessing support	§2.1, §2.2, §3, §4
WV-SR-4310	Processor versioning	§2.1, §3, §4
WV-SR-4320	Processing granularity	§3, §4
WV-SR-4330	Reprocessing performance	§3.1.2.4, §3.1.3.2
WV-SR-4400	Configuration control	§3.1.2.4, §3.1.3.2
WV-SR-4500	Modularity	§3, §4

## APPENDIX 1: REFERENCES

- [1]: World Meteorological Organization: [Essential Climate Variables](#). © 2019 World Meteorological Organization (WMO).
- [2]: ESA CCI Water Vapour: Product Specification Document. M. Schröder, M. Hegglin, O.Danne, J. Fischer, A. Laeng, R. Siddans, C. Sioris, G. Stiller, and K. Walker. Issue 2.0, 1 November 2019.
- [3]: ESA CCI Water Vapour: Data Access Requirement Document. M. Schröder, M. Hegglin, H. Brogniez, J. Fischer, D. Hubert, A. Laeng, R. Siddans, C. Sioris, G. Stiller, T. Trent, and K. Walker. Issue 1.0, 11 December 2018.
- [4]: ESA CCI Water Vapour: Algorithm Theoretical Basis Document. J. Fischer et al., in progress, 2019.
- [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]: Data Standards Requirements for CCI Data Producers, CCI-PRGM-EOPS-TN-13-0009, Issue 2.0, date of issue 17/09/2018.
- [11]: Data Standards Requirements for CCI Data Producers. Issue 2.1, in preparation, 2019.
- [12]: ESA CCI Water Vapour: System Requirements Document. O.Danne and M. Hegglin, Issue 2.0, 8 May 2020.
- [13]: IT4Innovations National Supercomputing Center. VSB – Technical University of Ostrava, Czech Republic. <https://www.it4i.cz/?lang=en>
- [14]: JASMIN – Petascale storage and cloud computing for big data challenges in environmental science. <http://www.jasmin.ac.uk/>
- [15]: Google Earth Engine – A planetary-scale platform for Earth science data & analysis. <https://earthengine.google.com/>
- [16]: IBM Platform LSF documentation. [https://www.ibm.com/support/knowledgecenter/en/SSETD4/product\\_welcome\\_platform\\_lsf.htm](https://www.ibm.com/support/knowledgecenter/en/SSETD4/product_welcome_platform_lsf.htm)
- [17]: RACC introduction. <https://research.reading.ac.uk/act/knowledgebase/racc-introduction/>

- [18]: Boettcher, M., Zuehlke, M., and H. Permana; Cal/Val and User Services – Calvalus. Software User Manual. Brockmann Consult, Version 1.7, 19 December 2017.
- [19]: Batch Computing on LOTUS. CEDA help documentation, <https://help.ceda.ac.uk/category/107-batch-computing-on-lotus>
- [20]: 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](http://envisat.esa.int/pub/ESA_DOC/ENVISAT/MERIS/meris.ProductHandbook.2_1.pdf)
- [21]: Farr, T. G., et al., The Shuttle Radar Topography Mission, Rev. Geophys., 45, RG2004, doi:10.1029/2005RG000183. (2007).
- [22]: Shuttle Radar Topography Mission web site. <https://www2.jpl.nasa.gov/srtm/index.html>
- [23]: 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.
- [24]: ECMWF Public Datasets Archive web access. <https://confluence.ecmwf.int/display/WEBAPI/Access+ECMWF+Public+Datasets>
- [25]: 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>
- [26]: 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>
- [27]: 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](http://dx.doi.org/10.5067/MODIS/MOD35_L2.006).
- [28]: CEDA Archive: NEODC database. <http://data.ceda.ac.uk/neodc>
- [29]: CEDA Archive: MODIS temporal coverage in NEODC database. [http://data.ceda.ac.uk/neodc/modis/metadata/temporal\\_coverage/](http://data.ceda.ac.uk/neodc/modis/metadata/temporal_coverage/)
- [30]: 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>
- [31]: ESA CCI Water Vapour: Product User Guide. O. Danne, M. Hegglin, M. Schröder, R. Preusker, J. Fischer, C. Brockmann, Issue 1.0, 8 May 2020.
- [32]: ESA CCI Water Vapour: System Verification Report. O.Danne and M. Hegglin, Issue 2.0, 8 May 2020.
- [33] Hegglin, M. I., S. Tegtmeier, and the SPARC Data Initiative Team, SPARC Data Initiative: Overview, *ESSD*, submitted.
- [34] 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.



**[35]:** Aura MLS Level 2 and 3 data quality and description document. NASA, Jet Propulsion Laboratory. Version 4.2x-4.0. [https://mls.jpl.nasa.gov/data/v4-2\\_data\\_quality\\_document.pdf](https://mls.jpl.nasa.gov/data/v4-2_data_quality_document.pdf)

**[36]** Sommer, Michael; Dirksen, Ruud; Immler, Franz. (2012): RS92 GRUAN Data Product Version 2 (RS92-GDP.2). GRUAN Lead Centre. DOI:10.5676/GRUAN/RS92-GDP.2

## APPENDIX 2: GLOSSARY

Term	Definition
<i>ATBD</i>	Algorithm Theoretical Basis Document
<i>BC</i>	Brockmann Consult
<i>Cal/val</i>	Cal/Val and User Services
<i>CCI</i>	Climate Change Initiative
<i>CDO</i>	Climate Data Operators
<i>CDR</i>	Climate Data Record
<i>CEDA</i>	Centre for Environmental Data Analysis
<i>CEMS</i>	Climate and Environmental Monitoring from Space
<i>CF</i>	Climate and Forecast
<i>CM SAF</i>	Satellite Application Facility on Climate Monitoring
<i>CPU</i>	Central Processing Unit
<i>DARD</i>	Data Access Requirement Document
<i>DWD</i>	Deutscher Wetterdienst (German Weather Service)
<i>ECMWF</i>	European Centre for Medium Range Weather Forecast
<i>ECSAT</i>	European Centre for Space Applications and Telecommunications
<i>ECV</i>	Essential Climate Variable
<i>EO</i>	Earth Observation
<i>ERA</i>	European Re-Analysis
<i>ESA</i>	European Space Agency
<i>FTP</i>	File Transfer Protocol
<i>GFS</i>	Google File System
<i>GITZ</i>	Geesthacht Innovation and Technology Centre
<i>GFS</i>	Google File System (GFS),
<i>GWS</i>	Group Workspace
<i>HDFS</i>	Hadoop Distributed File System
<i>HOAPS</i>	Hamburg Ocean Atmosphere Parameters and Fluxes
<i>JASMIN</i>	Joint Analysis System Meeting Infrastructure
<i>KO</i>	Kick-off
<i>LSF</i>	Load Sharing Facility
<i>LTS</i>	Long Term Support
<i>MERIS</i>	Medium Resolution Imaging Spectrometer

<b>Term</b>	<b>Definition</b>
<i>MODIS</i>	Moderate Resolution Imaging Spectroradiometer
<i>NASA</i>	National Aeronautics and Space Administration
<i>NCAS</i>	National Centre for Atmospheric Science
<i>NEODC</i>	NERC Earth Observation Data Centre
<i>NERC</i>	National Environment Research Council
<i>NetCDF</i>	Network Common Data Form
<i>NIR</i>	Near Infrared
<i>OLCI</i>	Ocean and Land Colour Instrument
<i>PB</i>	PetaByte
<i>PSD</i>	Product Specification Document
<i>PUG</i>	Product User Guide
<i>RAL</i>	Rutherford Appleton Laboratory
<i>RR</i>	Reduced Resolution
<i>SCRIP</i>	Spherical Coordinate Remapping and Interpolation Package
<i>SDK</i>	Software development Kit
<i>SE</i>	Spectral Earth
<i>SNAP</i>	Sentinel Application Platform
<i>SoW</i>	Statement of Work
<i>SR</i>	System Requirement
<i>SRD</i>	System Requirements Document
<i>SSD</i>	System Specification Document
<i>SSM/I</i>	Special Sensor Microwave/Imager
<i>SSMIS</i>	Special Sensor Microwave Imager Sounder
<i>STFC</i>	Science and Technology Facilities Council
<i>TB</i>	TeraByte
<i>TCWV</i>	Total Column of Water Vapour
<i>TOA</i>	Top of Atmosphere
<i>UoR</i>	University of Reading
<i>VRWV</i>	Vertically Resolved Water Vapour
<i>VM</i>	Virtual Machine
<i>WV</i>	Water Vapour
<i>YARN</i>	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()

monthrang = calendar.monthrange(year, month)
start = datetime.date(year, month, 1)
stop = datetime.date(year, month, monthrang[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=$auxroot/era-interim-t2m-mslp-tcwg-u10-
v10/${date_before:0:4}/era-interim-t2m-mslp-tcwg-u10-v10-
${date_before:0:4}-${date_before:5:2}.nc
    erathisday=$auxroot/era-interim-t2m-mslp-tcwg-u10-v10/$year/era-
interim-t2m-mslp-tcwg-u10-v10-$year-$month.nc
    eratimestack=era-interim-t2m-mslp-tcwg-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-tcwg-u10-v10/$year/era-
interim-t2m-mslp-tcwg-u10-v10-$year-$month.nc
    eradayafter=$auxroot/era-interim-t2m-mslp-tcwg-u10-
v10/${date_after:0:4}/era-interim-t2m-mslp-tcwg-u10-v10-
${date_after:0:4}-${date_after:5:2}.nc
    eratimestack=era-interim-tcwg-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-tcwg-u10-v10/$year/era-
interim-t2m-mslp-tcwg-u10-v10-$year-$month.nc
fi

# interpolate temporally

echo "$(date +%Y-%m-%dT%H:%M:%S -u) interpolate temporally ..."
eratimeslice=era-interim-t2m-mslp-tcwg-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 client script on Calvalus:

#### **l2\_tcwg-meris.xml**

```
import glob
import os
import datetime
import calendar

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

```
from pmonitor import PMonitor

#####
#
# TCWV chain for MERIS:
# Includes steps:
# - Idepix: L1b + EraInterim --> Merged Idepix (seq + nc.gz)
# - TCWV L2: Merged Idepix seq --> TCWV L2 (seq + nc.gz)
# - TCWV L3 Daily: all TCWV L2 from day --> TCWV L3 daily global
#      mosaic (nc)
#
#####

def getMonths(year):
    if year == '2002':
        months = [ '04', '05', '06', '07', '08', '09', '10', '11',
'12' ]
    elif year == '2012':
        months = [ '01', '02', '03', '04' ]
    else:
        months = [ '01' , '07' ]

    return months

def getNumDaysInMonth(year, month):
    return calendar.monthrange(int(year), int(month))[1]

def getMinMaxDate(year, month):
    numDaysInMonth = getNumDaysInMonth(year, month)
    minDate = datetime.date(int(year), int(month), 1)
    maxDate = datetime.date(int(year), int(month),
int(numDaysInMonth))
    return (minDate, maxDate)

#####

#### main script: ####
```



```
years    = [ '2011' ]

l3_resolutions = [ '05' ]

inputs = ['l1b']
for year in years:
    for month in getMonths(year):
        for iday in range(1, getNumDaysInMonth(year, month)+1):
            # L1b + Era Interim are the starting point
            day = str(iday).zfill(2)
            inputs.append('/calvalus/eodata/MER_RR__1P/r03/' +
                           year + '/' + month + '/' + day)
            inputs.append('/calvalus/projects/wvcci/era-
                           interim/meris/' + year + '/' + month + '/' + day)

hosts    = [('localhost',32)]

pm = PMonitor(inputs, \
               request='tcwv_meris_chain', \
               logdir='log', \
               hosts=hosts, \
               script='template.py')

WVCCI_INST_DIR = os.environ['WVCCI_INST']
outputFormat = 'NetCDF4-WVCCI'

for year in years:
    for month in getMonths(year):

        (minDate, maxDate) = getMinMaxDate(year, month)
        lastDayOfMonth = getNumDaysInMonth(year, month)

        for iday in range(1, lastDayOfMonth+1):
            day = str(iday).zfill(2)

            # ===== 12 day =====

            # Idepix
```

```
l2_idepix_name = 'l2_idepix-meris-' + year + '-' +  
month + '-' + day  
params = ['sensor', 'MERIS',  
          'minDate', str(minDate),  
          'maxDate', str(maxDate),  
          'year', year,  
          'month', month,  
          'day', day,  
          'input', '/calvalus/eodata/MER_RR__1P/r03/'  
            + year + '/' + month + '/' + day,  
          'erainterim', '/calvalus/projects/wvcci/era-  
interim/meris/' + year + '/' + month + '/' + day,  
          'output',  
          '/calvalus/projects/wvcci/idepix/meris/' + year + '/' + month + '/' +  
day]  
  
pm.execute('l2-idepix-MERIS.xml',  
          ['l1b'],  
          [l2_idepix_name],  
          parameters=params,  
          logprefix=l2_idepix_name)  
  
# Merge Idepix with EraInterim  
l2_idepix_erainterim_merge_name = 'l2_idepix-  
erainterim-meris-' + year + '-' + month + '-' + day  
params = ['sensor', 'MERIS',  
          'minDate', str(minDate),  
          'maxDate', str(maxDate),  
          'year', year,  
          'month', month,  
          'day', day,  
          'idepix',  
          '/calvalus/projects/wvcci/idepix/meris/' + year + '/' + month + '/' +  
day,  
          'input', '/calvalus/projects/wvcci/era-  
interim/meris/' + year + '/' + month + '/' + day,  
          'output', '/calvalus/projects/wvcci/idepix-  
era-interim/meris/' + year + '/' + month + '/' + day]  
  
pm.execute('l2-idepix-erainterim-MERIS.xml',  
          [l2_idepix_name],  
          [l2_idepix_erainterim_merge_name],  
          parameters=params,  
          logprefix=l2_idepix_erainterim_merge_name)
```

```
# TCWV
l2_tcwv_name = 'l2_tcwv-meris-' + year + '-' + month +
              '-' + day

params = ['sensor', 'MERIS',
          'minDate', str(minDate),
          'maxDate', str(maxDate),
          'year', year,
          'month', month,
          'day', day,
          'input', '/calvalus/projects/wvcci/idepix-
era-interim/meris/' + year + '/' + month + '/' + day,
          'output',
          '/calvalus/projects/wvcci/tcwv/meris/l2/' + year + '/' + month + '/'
          + day]

pm.execute('l2-tcwv-MERIS.xml',
           [l2_idepix_erainterim_merge_name],
           [l2_tcwv_name],
           parameters=params,
           logprefix=l2_tcwv_name)

pm.wait_for_completion()
```

#### 4. Request template for MERIS TCWV L2 processing on Calvalus:

##### **l2\_tcwv-meris.xml**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>

<wps:Execute service="WPS"
  version="1.0.0"
  xmlns:wps="http://www.opengis.net/wps/1.0.0"
  xmlns:ows="http://www.opengis.net/ows/1.1"
  xmlns:xlink="http://www.w3.org/1999/xlink">

  <ows:Identifier>L2</ows:Identifier>

  <wps:DataInputs>
    <wps:Input>
      <ows:Identifier>productionName</ows:Identifier>
```

```
<wps:Data>
  <wps:LiteralData>wvcci-tcwv-MERIS-{$year}-{$month}-
    {$day}</wps:LiteralData>
</wps:Data>
</wps:Input>
<wps:Input>
  <ows:Identifier>processorBundleName</ows:Identifier>
  <wps:Data>
    <wps:LiteralData>snap-wvcci</wps:LiteralData>
  </wps:Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>processorBundleVersion</ows:Identifier>
  <wps:Data>
    <wps:LiteralData>1.0-SNAPSHOT</wps:LiteralData>
  </wps:Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>processorName</ows:Identifier>
  <wps:Data>
    <wps:LiteralData>ESACCI.Tcwv</wps:LiteralData>
  </wps:Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>calvalus.output.dir</ows:Identifier>
  <wps:Data>
    <wps:Reference xlink:href="hdfs://calvalus${output}" />
  </wps:Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>inputPath</ows:Identifier>
  <wps:Data>
    <wps:LiteralData>${input}/L2_of_MER_RR__1.*${yyyy}${MM}${dd}_*.
      seq$</wps:LiteralData>
```

```

        </wps:Data>
    </wps:Input>

    <wps:Input>
        <ows:Identifier>minDate</ows:Identifier>
        <wps:Data>
            <wps:LiteralData>${minDate}</wps:LiteralData>
        </wps:Data>
    </wps:Input>

    <wps:Input>
        <ows:Identifier>maxDate</ows:Identifier>
        <wps:Data>
            <wps:LiteralData>${maxDate}</wps:LiteralData>
        </wps:Data>
    </wps:Input>

    <wps:Input>
        <ows:Identifier>processorParameters</ows:Identifier>
        <wps:Data>
            <wps:ComplexData>
                <parameters>
                    <sensor>MERIS</sensor>
                </parameters>
            </wps:ComplexData>
        </wps:Data>
    </wps:Input>

    <cloudFilterLevel>CLOUD_SURE_AMBIGUOUS_BUFFER</cloudFilterLevel>
    </parameters>
    </wps:ComplexData>
    </wps:Data>
</wps:Input>

<wps:Input>
    <ows:Identifier>calvalus.resume</ows:Identifier>
    <wps:Data>
        <wps:LiteralData>true</wps:LiteralData>
    </wps:Data>
</wps:Input>
<wps:Input>

```

```
<ows:Identifier>calvalus.system.beam.reader.tileHeight</ows:Identifier>
```

```
    <wps>Data>
      <wps:LiteralData>16</wps:LiteralData>
    </wps>Data>
  </wps:Input>
<wps:Input>
```

```
<ows:Identifier>calvalus.system.beam.reader.tileWidth</ows:Identifier>
```

```
    <wps>Data>
      <wps:LiteralData>*</wps:LiteralData>
    </wps>Data>
  </wps:Input>
<wps:Input>
```

```
<ows:Identifier>calvalus.hadoop.mapreduce.job.priority</ows:Identifier>
```

```
    <wps>Data>
      <wps:LiteralData>NORMAL</wps:LiteralData>
    </wps>Data>
  </wps:Input>
<wps:Input>
```

```
<ows:Identifier>calvalus.hadoop.mapreduce.job.queueName</ows:Identifier>
```

```
    <wps>Data>
      <wps:LiteralData>other</wps:LiteralData>
    </wps>Data>
  </wps:Input>
<wps:Input>
```

```
<ows:Identifier>calvalus.hadoop.mapreduce.map.failures.maxpercent</ows:Identifier>
```

```
    <wps>Data>
      <wps:LiteralData>5</wps:LiteralData>
    </wps>Data>
  </wps:Input>

<wps:Input>
```

```
<ows:Identifier>calvalus.hadoop.mapreduce.task.timeout</ows:Identifier>

  <wps:Data>
    <!-- 1h in milliseconds -->
    <wps:LiteralData>3600000</wps:LiteralData>
  </wps:Data>
</wps:Input>

<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.map.java.opts</ows:Identifier>

  <wps:Data><wps:LiteralData>-Djava.awt.headless=true -
Xmx3072M</wps:LiteralData></wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.map.memory.mb</ows:Identifier>

  <wps:Data><wps:LiteralData>4096</wps:LiteralData></wps:Data>
</wps:Input>

</wps>DataInputs>

</wps:Execute>
```

## 5. Request example for MERIS TCWV L2 processing on Calvalus:

### **l2\_tcwv-meris-2011-07-29.xml**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>

<wps:Execute service="WPS"
  version="1.0.0"
  xmlns:wps="http://www.opengis.net/wps/1.0.0"
  xmlns:ows="http://www.opengis.net/ows/1.1"
  xmlns:xlink="http://www.w3.org/1999/xlink">

  <ows:Identifier>L2</ows:Identifier>
```

```
<wps:DataInputs>
  <wps:Input>
    <ows:Identifier>productionName</ows:Identifier>
    <wps:Data>
      <wps:LiteralData>wvcci-tcwv-MERIS-2011-07-
        29</wps:LiteralData>
    </wps:Data>
  </wps:Input>
  <wps:Input>
    <ows:Identifier>processorBundleName</ows:Identifier>
    <wps:Data>
      <wps:LiteralData>snap-wvcci</wps:LiteralData>
    </wps:Data>
  </wps:Input>

  <wps:Input>
    <ows:Identifier>processorBundleVersion</ows:Identifier>
    <wps:Data>
      <wps:LiteralData>1.0-SNAPSHOT</wps:LiteralData>
    </wps:Data>
  </wps:Input>

  <wps:Input>
    <ows:Identifier>processorName</ows:Identifier>
    <wps:Data>
      <wps:LiteralData>ESACCI.Tcwv</wps:LiteralData>
    </wps:Data>
  </wps:Input>

  <wps:Input>
    <ows:Identifier>calvalus.output.dir</ows:Identifier>
    <wps:Data>
      <wps:Reference
xlink:href="hdfs://calvalus/calvalus/projects/wvcci/tcwv/meris/12/
        2011/07/29" />
    </wps:Data>
  </wps:Input>

  <wps:Input>
```



```
<ows:Identifier>inputPath</ows:Identifier>
<wps>Data>
  <wps:LiteralData>/calvalus/projects/wvcci/idepix-era-
interim/meris/2011/07/29/L2_of_MER_RR__1.*${yyyy}${MM}${dd}_.*.seq
$</wps:LiteralData>
</wps>Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>minDate</ows:Identifier>
  <wps>Data>
    <wps:LiteralData>2011-07-01</wps:LiteralData>
  </wps>Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>maxDate</ows:Identifier>
  <wps>Data>
    <wps:LiteralData>2011-07-31</wps:LiteralData>
  </wps>Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>processorParameters</ows:Identifier>
  <wps>Data>
    <wps:ComplexData>
      <parameters>
        <sensor>MERIS</sensor>

<cloudFilterLevel>CLOUD_SURE_AMBIGUOUS_BUFFER</cloudFilterLevel>
      </parameters>
    </wps:ComplexData>
  </wps>Data>
</wps:Input>

<wps:Input>
  <ows:Identifier>calvalus.resume</ows:Identifier>
  <wps>Data>
```

```
<wps:LiteralData>true</wps:LiteralData>
</wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.system.beam.reader.tileHeight</ows:Identifier>

<wps:Data>
  <wps:LiteralData>16</wps:LiteralData>
</wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.system.beam.reader.tileWidth</ows:Identifier>
<wps:Data>
  <wps:LiteralData>*</wps:LiteralData>
</wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.job.priority</ows:Identifier>
<wps:Data>
  <wps:LiteralData>NORMAL</wps:LiteralData>
</wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.job.queueName</ows:Identifier>
<wps:Data>
  <wps:LiteralData>other</wps:LiteralData>
</wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.map.failures.maxpercent</ows:Identifier>
<wps:Data>
  <wps:LiteralData>5</wps:LiteralData>
</wps:Data>
```

```
</wps:Input>

<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.task.timeout</ows:Identifier>

  <wps:Data>
    <!-- 1h in milliseconds -->
    <wps:LiteralData>3600000</wps:LiteralData>
  </wps:Data>
</wps:Input>

<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.map.java.opts</ows:Identifier>

  <wps:Data><wps:LiteralData>-Djava.awt.headless=true -
Xmx3072M</wps:LiteralData></wps:Data>
</wps:Input>
<wps:Input>

<ows:Identifier>calvalus.hadoop.mapreduce.map.memory.mb</ows:Identifier>

  <wps:Data><wps:LiteralData>4096</wps:LiteralData></wps:Data>
</wps:Input>

</wps>DataInputs>

</wps:Execute>
```

## 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 bulk processing on Calvalus after successful execution:

### wvcci-l2-tcwv-meris.report

```
template.py l2-tcwv-MERIS.xml sensor MERIS minDate 2011-07-01 maxDate
2011-07-31 year 2011 month 07 day 02 input
/calvalus/projects/wvcci/idepix-era-interim/meris/2011/07/02
output /calvalus/projects/wvcci/tcwv/meris/l2/2011/07/02 l2_idepix-
erainterim-meris-2011-07-02 l2_tcwv-meris-2011-07-02
```

```
template.py l2-tcwg-MERIS.xml sensor MERIS minDate 2011-07-01 maxDate
2011-07-31 year 2011 month 07 day 01 input
/calvalus/projects/wvcci/idepix-era-interim/meris/2011/07/01
```

```
output /calvalus/projects/wvcci/tcwg/meris/l2/2011/07/01
l2_idepix-era-interim-meris-2011-07-01 l2_tcwg-meris-2011-07-01

template.py l2-tcwg-MERIS.xml sensor MERIS minDate 2011-07-01
maxDate 2011-07-31 year 2011 month 07 day 08 input
/calvalus/projects/wvcci/idepix-era-interim/meris/2011/07/08
output /calvalus/projects/wvcci/tcwg/meris/l2/2011/07/08
l2_idepix-era-interim-meris-2011-07-08 l2_tcwg-meris-2011-07-08

... and many more...
```

## 8. Log file example for MERIS TCWV L2 processing on Calvalus

### **l2\_tcwg-meris-2011-07-31.xml**

```
Loading Calvalus configuration
'/home/olaf/.calvalus/calvalus.config'...

Configuration loaded.

Loading production request '/home/olaf/wvcci-
inst/requests/l2_tcwg-meris-2011-07-31.xml'...

Production request format is 'WPS-XML'

Production request loaded, type is 'L2'.

Ordering production...

2019-05-03 15:20:40,355 INFO com.bc.calvalus: orderProduction: L2
(for olaf)

2019-05-03 15:20:40,356 INFO com.bc.calvalus:
calvalus.hadoop.mapreduce.job.priority = NORMAL

2019-05-03 15:20:40,356 INFO com.bc.calvalus:
calvalus.hadoop.mapreduce.job.queueName = other

2019-05-03 15:20:40,356 INFO com.bc.calvalus:
calvalus.hadoop.mapreduce.map.failures.maxPercent = 5

2019-05-03 15:20:40,356 INFO com.bc.calvalus:
calvalus.hadoop.mapreduce.map.java.opts = -
Djava.awt.headless=true -Xmx3072M

2019-05-03 15:20:40,356 INFO com.bc.calvalus:
calvalus.hadoop.mapreduce.map.memory.mb = 4096

2019-05-03 15:20:40,356 INFO com.bc.calvalus:
calvalus.hadoop.mapreduce.task.timeout = 3600000

2019-05-03 15:20:40,356 INFO com.bc.calvalus: calvalus.output.dir
=
hdfs://calvalus/calvalus/projects/wvcci/tcwg/meris/l2/2011/07/31

2019-05-03 15:20:40,357 INFO com.bc.calvalus: calvalus.resume =
true

2019-05-03 15:20:40,357 INFO com.bc.calvalus:
calvalus.system.beam.reader.tileHeight = 16
```

```
2019-05-03 15:20:40,357 INFO com.bc.calvalus:
calvalus.system.beam.reader.tileWidth = *

2019-05-03 15:20:40,357 INFO com.bc.calvalus: inputPath =
/calvalus/projects/wvcci/idepix-era-
interim/meris/2011/07/31/L2_of_MER_RR__1.*${yyyy}${MM}${dd}_.*.se
q$

2019-05-03 15:20:40,357 INFO com.bc.calvalus: maxDate = 2011-07-
31

2019-05-03 15:20:40,370 INFO com.bc.calvalus: minDate = 2011-07-
01

2019-05-03 15:20:40,370 INFO com.bc.calvalus: processorBundleName
= snap-wvcci

2019-05-03 15:20:40,370 INFO com.bc.calvalus:
processorBundleVersion = 1.0-SNAPSHOT

2019-05-03 15:20:40,370 INFO com.bc.calvalus: processorName =
ESACCI.Tcwv

2019-05-03 15:20:40,370 INFO com.bc.calvalus: processorParameters
= <parameters>

                                <sensor>MERIS</sensor>

<cloudFilterLevel>CLOUD_SURE_AMBIGUOUS_BUFFER</cloudFilterLevel>

                                </parameters>

2019-05-03 15:20:40,371 INFO com.bc.calvalus: productionName =
wvcci-tcwv-MERIS-2011-07-31

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

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

CREATING new JobClient for: olaf

2019-05-03 15:20:42,158 INFO com.bc.calvalus: Submitting Job:
wvcci-tcwv-MERIS-2011-07-31

[main] INFO org.apache.hadoop.mapreduce.JobSubmissionFiles -
Permissions on staging directory /tmp/hadoop-
yarn/staging/olaf/.staging are incorrect: rwxrwx---. Fixing
permissions to correct value rwxrwx---

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

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

2019-05-03 15:20:43,063 INFO com.bc.calvalus: Total files to
process : 14
```

```
[main] INFO org.apache.hadoop.mapreduce.JobSubmitter - number of
splits:14

[main] INFO org.apache.hadoop.mapreduce.JobSubmitter - Submitting
tokens for job: job_1553782106366_34729

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

[main] INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl
- Submitted application application_1553782106366_34729

[main] INFO org.apache.hadoop.mapreduce.Job - The url to track
the job:
http://master01:8088/proxy/application_1553782106366_34729/

2019-05-03 15:20:45,269 INFO com.bc.calvalus: Submitted Job with
Id: job_1553782106366_34729

2019-05-03 15:20:45,270 INFO com.bc.calvalus: -----
-----

2019-05-03 15:20:45,270 INFO com.bc.calvalus: remoteUser=olaf
mapreduce.job.user.name=null

2019-05-03 15:20:45,279 INFO com.bc.calvalus: calvalus.bundles =
snap-wvcci-1.0-SNAPSHOT

2019-05-03 15:20:45,279 INFO com.bc.calvalus:
calvalus.calvalus.bundle = calvalus-2.10-SNAPSHOT_for_CAWA

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.input.dateRanges = [2011-07-01:2011-07-31]

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.input.minHeight = 0

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.input.minWidth = 0

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.input.pathPatterns = /calvalus/projects/wvcci/idepix-
era-
interim/meris/2011/07/31/L2_of_MER_RR__1.*${yyyy}${MM}${dd}_.*.se
q$

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.input.regionName =

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.l2.operator = ESACCI.Tcwv

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.l2.parameters = <parameters>

                                <sensor>MERIS</sensor>

<cloudFilterLevel>CLOUD_SURE_AMBIGUOUS_BUFFER</cloudFilterLevel>

                                </parameters>

2019-05-03 15:20:45,280 INFO com.bc.calvalus:
calvalus.l2.processorType = OPERATOR

2019-05-03 15:20:45,280 INFO com.bc.calvalus: calvalus.output.dir
=
hdfs://calvalus/calvalus/projects/wvcci/tcwv/meris/12/2011/07/31
```

```
2019-05-03 15:20:45,285 INFO com.bc.calvalus: calvalus.processAll
= false

2019-05-03 15:20:45,285 INFO com.bc.calvalus:
calvalus.productionType = L2

2019-05-03 15:20:45,286 INFO com.bc.calvalus:
calvalus.regionGeometry =

2019-05-03 15:20:45,286 INFO com.bc.calvalus: calvalus.resume =
true

2019-05-03 15:20:45,286 INFO com.bc.calvalus:
calvalus.snap.bundle = snap-wvcci-1.0-SNAPSHOT

2019-05-03 15:20:45,286 INFO com.bc.calvalus:
calvalus.system.beam.reader.tileHeight = 16

2019-05-03 15:20:45,286 INFO com.bc.calvalus:
calvalus.system.beam.reader.tileWidth = *

2019-05-03 15:20:45,286 INFO com.bc.calvalus:
calvalus.system.snap.dataio.reader.tileHeight = 64

2019-05-03 15:20:45,286 INFO com.bc.calvalus:
calvalus.system.snap.dataio.reader.tileWidth = *

2019-05-03 15:20:45,286 INFO com.bc.calvalus: calvalus.user =
olaf

2019-05-03 15:20:45,287 INFO com.bc.calvalus: -----
-----

Production successfully ordered. The production ID is:
20190503132040_L2_6fcc13dd4c91ad

Production remote status: state=SCHEDULED, progress=0.0,
message=''

Production remote status: state=RUNNING, progress=0.0, message=''

Production remote status: state=RUNNING, progress=0.0, message=''

Production remote status: state=RUNNING, progress=0.45,
message=''

Production remote status: state=RUNNING, progress=0.45,
message=''

Production remote status: state=COMPLETED, progress=1.0,
message=''

Production completed. Output directory is
olaf/20190503132040_L2_6fcc13dd4c91ad
```

## 9. Jobs definition and registration script on JASMIN:

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

```
#!/bin/bash
```

```
#. ${WVCCI_INST}/bin/wvcci_env/wvcci-env-l2-tcwv-modis.sh
. ${WVCCI_INST}/bin/wvcci-env.sh # this script shall now be used
for everything!
```

```
echo "entered wvcci-l2-tcwv-modis-step..."
idepixPath=$1
idepixFile=$2
cloudMaskPath=$3
year=$4
month=$5
day=$6
hhmm=$7
wvcciRootDir=$8

task="wvcci-l2-tcwv-modis"
jobname="${task}-${year}-${month}-${day}-${hhmm}"
command0="./bin/${task}-bash.sh"
command="${command0} ${idepixPath} ${idepixFile} ${cloudMaskPath}
${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}"

echo "calling read_task_jobs()..."
read_task_jobs ${jobname}

if [ -z ${jobs} ]; then
    timelim=180
    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-snap.sh
#!/bin/bash
```



```
## TCWV

if [ -f $idepixEraInterimMerge ]; then
    auxdataPath=/gws/nopw/j04/esacci_wv/software/
    dot_snap/auxdata/wvcci

    tcwv=$tcwvDir/${modisstem}_tcwv.nc

    if [ -f $cloudMaskPath ]; then
        echo "$SNAP_HOME/bin/gpt ESACCI.Tcwv -e -
        SsourceProduct=$idepixEraInterimMerge -
        Smod35Product=$cloudMaskPath -PauxdataPath=$auxdataPath -
        Psensor=MODIS_TERRA -PprocessOcean=true -f NetCDF4-WVCCI -t

        $SNAP_HOME/bin/gpt ESACCI.Tcwv -e -
        -SsourceProduct=$idepixEraInterimMerge -
        Smod35Product=$cloudMaskPath -PauxdataPath=$auxdataPath -
        Psensor=MODIS_TERRA -PprocessOcean=true -f NetCDF4-WVCCI -t

    else
        echo "$SNAP_HOME/bin/gpt ESACCI.Tcwv -e -
        -SsourceProduct=$idepixEraInterimMerge -
        PauxdataPath=$auxdataPath -Psensor=MODIS_TERRA -
        PprocessOcean=true -f NetCDF4-WVCCI -t $tcwv"

        $SNAP_HOME/bin/gpt ESACCI.Tcwv -e -
        SsourceProduct=$idepixEraInterimMerge -
        PauxdataPath=$auxdataPath -Psensor=MODIS_TERRA -
        PprocessOcean=true -f NetCDF4-WVCCI -t $tcwv

    fi
fi
```

## 11. LSF job submission script on JASMIN:

```
# wvcci-env.sh

#!/bin/bash

# WVCCI function definitions

set -e

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

WVCCI_TASKS=${WVCCI_INST}/tasks
WVCCI_LOG=${WVCCI_INST}/log
#export PM_LOG_DIR=${WVCCI_LOG}
```

```
#export
PM_PYTHON_EXEC='/gws/nopw/j04/esacci_wv/software/miniconda3/envs/wvcci/bin/python'

read_task_jobs() {
    echo "entered read_task_jobs()..."
    jobname=$1
    echo "jobname: $jobname"
    jobs=
    echo "WVCCI_TASKS/jobname.tasks:
        ${WVCCI_TASKS}/${jobname}.tasks"
    if [ -e ${WVCCI_TASKS}/${jobname}.tasks ]
    then
        for logandid in `cat ${WVCCI_TASKS}/${jobname}.tasks`
        do
            echo "logandid: $logandid"
            job=`basename ${logandid}`
            log=`dirname ${logandid}`
            echo "job: $job"
            echo "log: $log"
            #if grep -qF 'Successfully completed.' ${log}
            #if ! grep -qF 'Status: 1' ${log}
            # also make sure that terminated jobs are not
            # interpreted as successful:
            #if [ ! grep -qF 'Status: 1' ${log} ] && [ ! grep -qF
            # 'TERM_RUNLIMIT' ${log} ]
            if [ grep -qF 'Status: 0' ${log} ] && [ ! grep -qF
            'TERM_RUNLIMIT' ${log} ]
            then
                if [ "${jobs}" != "" ]
                then
                    jobs="${jobs}|${job}"
                else
                    jobs="${job}"
                fi
            fi
        done
    fi
}
```

```
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

    echo "WVCCI_INST: ${WVCCI_INST}"
    echo "WVCCI_LOG : ${WVCCI_LOG}"
    echo "jobname: ${jobname}"
    echo "command: ${command}"
    echo "timelim: ${timelim}"
    echo "memlim: ${memlim}"

    args=("$@")
    echo "$# arguments passed"
    numMoreArgs=$(( $#-2 ))

    # L2 TCWV MODIS:
    bsubmit="bsub -q short-serial ${timelim} ${memlim} -P
ga_qa4ecv -cwd ${WVCCI_INST} -oo ${WVCCI_LOG}/${jobname}.out
-eo ${WVCCI_LOG}/${jobname}.err -J ${jobname}
${WVCCI_INST}/${command} ${@:3:$mynumargs}"

    echo "bsubmit: $bsubmit"

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

}
```

***End of Document***