




fire
cci

ESA Climate Change Initiative – Fire_cci D3.1 System Specification Document (SSD)

Project Name	ECV Fire Disturbance: Fire_cci
Contract N°	4000126706/19/I-NB
Issue Date	23/05/2023
Version	2.4
Authors	Thomas Storm, Martin Böttcher, Grit Kirches
Document Ref.	Fire_cci_D3.1_SSD_v2.4
Document type	Public

*To be cited as T. Storm, M. Boettcher, G. Kirches (2023) ESA CCI ECV Fire Disturbance:
D3.1 System Specification Document, version 2.4.*

Available at: <https://climate.esa.int/en/projects/fire/key-documents/>

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4	
			Issue	2.4	Date
				Page	2


Project Partners

Prime Contractor/ Scientific Lead & Project Management	UAH – University of Alcalá (Spain)
Earth Observation Team	UAH – University of Alcalá (Spain)
	UPM – Universidad Politécnica de Madrid (Spain)
	CNR-IREA - National Research Council of Italy – Institute for Electromagnetic Sensing of the Environment (Italy)
System Engineering	BC – Brockmann Consult (Germany)
Climate Modelling Group	CNRS - Laboratory for Sciences of Climate and Environment (France)
	VU – Vrije Universiteit Amsterdam (Netherlands)



Distribution

Affiliation	Name	Address	Copies
ESA	Clément Albergel (ESA)	clement.albergel@esa.int	electronic copy
Project Team	Emilio Chuvieco (UAH)	emilio.chuvieco@uah.es	electronic copy
	M. Lucrecia Pettinari (UAH)	mlucrecia.pettinari@uah.es	
	Amin Khairoun (UAH)	amin.khairoun@uah.es	
	Magí Franquesa (UAH)	magin.franquesa@uah.es	
	Consuelo Gonzalo (UPM)	consuelo.gonzalo@upm.es	
	Ángel Mario García Pedrero (UPM)	angelmario.garcia@upm.es	
	Dionisio Rodríguez Esparragón (UPM)	dionisio.rodriguez@ulpgc.es	
	Daniela Stroppiana (CNR)	stroppiana.d@irea.cnr.it	
	Matteo Sali (CNR)	sali.m@irea.cnr.it	
	Thomas Storm (BC)	thomas.storm@brockmann-consult.de	
	Martin Boettcher (BC)	martin.boettcher@brockmann-cons...	
	Grit Kirches (BC)	grit.kirches@brockmann-consult.de	
	Florent Mouillot (CNRS)	florent.mouillot@ird.fr	
	Philippe Ciais (CNRS)	philippe.ciais@lscce.ipsl.fr	
	Guido van der Werf (VU)	g.r.vander.werf@vu.nl	

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4	
			Issue	2.4	Date
				Page	3

Summary

This document is the version 2.4 of the System Specification Document for the Fire_cci project. It introduces the structure, workflows and mode of operation of the processing system used in the Fire_cci project.

	Affiliation/Function	Name	Date
Prepared	BC	Thomas Storm Martin Böttcher Grit Kirches	11/05/2023
Reviewed	UAH – Project Manager	M. Lucrecia Pettinari	23/05/2023
Authorized	UAH - Science Leader	Emilio Chuvieco	23/05/2023
Accepted	ESA - Technical Officer	Clément Albergel	06/06/2023

This document is not signed. It is provided as an electronic copy.

Document Status Sheet


Issue	Date	Details
1.0	24/02/2016	First document for Phase 2 of Fire_cci
1.1	16/03/2016	Addressing ESA comments according to CCI_FIRE_EOPS_MM_16_0034.pdf
1.2	06/09/2016	Updated release of the document with minor corrections
1.3	25/10/2016	Addressing ESA comments according to CCI_FIRE_EOPS_MM_16_0109.pdf
1.4	30/09/2017	Update for MODIS / SFD processing
1.5	30/01/2018	Addressing ESA comments according to CCI-EOPS-FIRE-MM-17-0098.pdf
2.0	27/03/2020	First document for Phase 1 of CCI+
2.1	22/04/2020	Addressing ESA comments according to Fire_cci+:SSD_v2.0_RID.doc
2.2	22/10/2021	System extension for SFD 2019 and SYN
2.3	12/11/2021	Addressing ESA comments according to Fire_cci+_D3.1_SSD_v2.2_RID.doc
2.4	23/05/2023	Update according to Phase 2 of CCI+

Document Change Record

Issue	Date	Request	Location	Details
1.1	16/03/2016	ESA	Version number Section 1 Section 2.1 Section 2.2 Section 2.3 Section 3.1 Figure 3.1 Section 3.2 Section 3.4 Table 3.3 Section 5.1 Section 5.3 Annex	Change of previous version number from 2.0 to 1.0 to address this document being independent from the one in Phase 1. Minor changes in the text. Re-ordering of sentences. Minor changes in the text. Addition of new reference documents. Minor changes in the text. Updated Minor changes in the text. Inclusion of text detailing the SoW requirements. Changes in the requirements. Minor changes in the text. Minor changes in the text. Inclusion of additional information in the list of steps. Addition of new acronyms.



Issue	Date	Request	Location	Details
1.2	06/09/2016	BC	Section 1 Section 2.1 Section 2.2 Section 2.3 Section 2.4 Section 3.1 Section 3.3 Section 3.4 Section 4 Section 4.2.3 Section 5.1 Section 5.2	Updated. Paragraph removed. Minor changes in the text. Last paragraph shortened. Reference documents updated. Minor changes in the text. Figure 3.1 updated. Minor changes in the text. Added information about Calvalus MapReduce Requirements text resumed. Reference to the MERIS ATBD updated. Removed sub-section "Docke", as it is not applicable anymore. The rest of the sub-sections were re-numbered. Figure 4.4 updated. Added information about MapReduce and formatting tool implementation, eliminated information about Docker container. Figure 5.1 updated. Fully re-written according to system updates.
1.3	25/10/2016	ESA	Naming convention All document Section 3.1 Section 3.2 Section 5.1 Section 5.2	Reference to the year of the project added to the name of the document. The reference to the MERIS data as FSG was changed to FRS. Figure 3.1 updated. New entity in the text added. Figure 3.2 updated. Small changes in the text. Added information on versions of auxiliary data.
1.4	30/09/2017	BC	Whole document	Added system overview, technical methods and workflows for SFD and MODIS processing
1.5	30/01/2018	ESA	Sections 2.1, 3, 3.2, 3.3, 5.2, 5.2.3 Section 3.1 Section 4 Section 6	Small changes in the text Figure 3.1 updated and text expanded Section restructured, including previous sub-section 3.4 New section added
2.0	27/03/2020	BC	Sections 1, 2.1, 2.2, 2.3, 3.3, 4.1 Sections 3.1, 4.2, 4.2.1, 4.3 Section 5.1	Sections updated Small changes in the text Section referring to MERIS deleted, and now refers to the MODIS processing. Section and sub-sections updated.
2.1	22/04/2020	ESA	Section 3.1 Section 3.3.	Figure 3.1 updated Table 3.1: added definition to acronym
2.2	22/10/2021	BC	All text Section 5.2 Section 5.3	Minor updates Section updated to reflect workflow for SFD 2019 New section on workflow for SYN BA processing
2.3	12/11/2021	ESA	Section 5.2.2 Section 5.2.3 Section 5.2.4 Section 5.3.3 All text	Figure 5.3 eliminated Figure 5.4 eliminated Figure 5.8 expanded Figure 5.10 eliminated Small changes in the text
2.4	11/05/2023	BC	Sections 1, 2.2, 3.1, 5.2, 5.2.3	Small changes in the text

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4		
			Issue	2.4	Date	23/05/2023
			Page		5	

Issue	Date	Request	Location	Details
			Section 2.3 Sections 4, 4.1, 5.1.3, 6 Section 4.2.1 Sections 4.3, 4.3.1 Section 5.1 (MODIS)	Documents updated Sections updated Merged with Section 4.2 Sections removed Shifted to Section 5.3 and updated



 fire cci	Fire_cci		Ref.:	Fire_cci_D3.1_SSD_v2.4		
	System Specification Document		Issue	2.4	Date	23/05/2023
				Page	6	

Table of Contents

1. Executive Summary	8
2. Introduction	8
2.1. Background.....	8
2.2. Purpose and scope	8
2.3. Applicable and reference documents.....	9
3. System overview	10
3.1. Fire_cci system context	10
3.2. Main function and processing chain	11
3.3. System requirements.....	11
4. Fire_cci system architecture	12
4.1. High-level system decomposition	12
4.2. Calvalus system	12
5. Workflows	18
5.1. SFD workflow	18
5.1.1. Sentinel-2 pre-processing	18
5.1.2. Burned Area processing	19
5.1.3. ADP/PSD product production	19
5.1.4. Processing quality assurance	21
5.1.5. Tools and modules.....	23
5.2. SYN BA workflow	23
5.2.1. SYN pre-processing.....	24
5.2.2. Burned Area processing	24
5.2.3. Pixel product and Grid product formatting	24
5.2.4. Tools and modules.....	25
5.3. MODIS workflow.....	25
5.3.1. Data acquisition	26
5.3.2. Data processing	26
5.3.3. Data repackaging	26
5.3.4. ADP/PSD product production	26
6. System outputs	26
Annex: Acronyms and abbreviations	28

	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4	
			Issue	2.4	Date
				Page	7

List of Tables

Table 3.1: System requirements in the SoW 11

Table 3.2: System requirements specified in the ADP/PSD and CCI data guidelines... 11

Table 3.3: System requirements specified in the ATBDs 12

Table 4.1: Software items for Fire_cci processing 17

Table 6.1: Output products 27

List of Figures

Figure 3-1: System context of the Fire_cci system with team, data providers, and other entities..... 10

Figure 4-1: Fire_cci system architecture layers with specific elements, Calvalus, and Hadoop 13

Figure 4-2: Archive-centric 14

Figure 4-3: Data-local..... 14

Figure 4-4: Calvalus architecture 15

Figure 4-5: Software architecture 16

Figure 5-2: Formatting 20

Figure 5-2: Quicklooks of JD for one tile..... 21


Figure 5-3: ‘mini tile effect’ – mini tiles of the first line appear at wrong locations 22

Figure 5-4: Different types of artefacts found and resolved..... 22

Figure 5-5: Systematic QA based on quicklook inspection and tracking of issues: example of an intermediate stage of the QA. 23

Figure 5-6: The relation of inputs, intermediates, and outputs in SYN BA processing. 24

Figure 5-7: Sequence of MODIS processing actions 25

 fire cci	Fire_cci			Ref.:	Fire_cci_D3.1_SSD_v2.4		
	System Specification Document			Issue	2.4	Date	23/05/2023
						Page	8

1. Executive Summary

This document is the Fire_cci System Specification Document (SSD). The Fire_cci system is employed in order to generate the Fire_cci dataset, which comprises several global and regional sub-datasets (see below).

This document describes the Fire_cci system, the workflows it defines and uses, and the environment in which it exists.

In the present version, the SSD describes the processing systems that have been used for computing the MODIS global time series (FireCCI51), the Small Fire Dataset (FireCCISFD20) based on Sentinel-2 MSI data, and the system currently being used to generate the Sentinel-3-based global dataset (FireCCIS311).

2. Introduction

2.1. Background

The ESA Climate Change Initiative (ESA CCI) stresses the importance of providing a higher scientific visibility to data acquired by ESA sensors, especially in the context of the IPCC reports. This implies to produce consistent time series of accurate Essential Climate Variables (ECV) products that can be used by the climate (primarily) and other scientists for their modelling efforts. Long-term observations and the international links with other agencies currently generating ECV data are keys to this activity.

The fire disturbance ECV includes Burned Area (BA) as the primary variable. The project includes the development of algorithms for the BA retrieval of MODIS, Sentinel-2 MSI, and Sentinel-3 OLCI/SLSTR data.

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

2.2. Purpose and scope

The purpose of the Fire_cci system is the generation of the Burned Area products, the exchange of the results with project partners and delivery of the final products to the end users. There are various large satellite input datasets to be processed in a performant way in the course of the project. The algorithms used for processing will be continuously improved and adapted to new sensors with the need of several test and improvement cycles before full-scale processing. This shall also be supported by the processing system.

This SSD exists in the context of other Fire_cci documents: The Fire_cci ADP [RD-1] describes the content and format of the BA products to be generated, which are complementary to the PDS [RD-2] specifications, and the ATBDs [RD-3 to 5] define the algorithms that are used for BA processing. This SSD describes the design of the system that generates these BA products, including the integration of the processors that implement the algorithms defined in the ATBDs, the handling of the input datasets, and the control of the production workflow for the Fire_cci processing chains. The Fire_cci processing system is based on several layers of generic components of the Brockmann Consult's (BC) Calvalus processing system and its deployment on various shared infrastructures. The main concepts of the elements used by Fire_cci and the Fire_cci-specific configuration and extension are also in the scope of this document.

This document currently covers the workflows of Fire_cci used to produce the datasets FireCCI51, FireCCISFD20, and FireCCIS311. It will be extended in the course of the project for the specifics of each sensor that will be included.

2.3. Applicable and reference documents

The following documents are applicable to this document:

[AD-1]	Climate Change Initiative Extension (CCI+) Phase 2 - Statement of Work, prepared by ESA Climate Office, Reference ESA-EOP-SC-AMT-2021-55ESA-EOP-XXXXX, Issue 1.0, date of issue 28/01/2022
[AD-2]	ESA Climate Change Initiative – CCI Project Guidelines. Ref. EOP-DTEX-EOPS-SW-10-0002, Issue 1.0, date of issue 05 November 2010.
[AD-3]	Data Standards Requirements for CCI Data Producers, CCI-PRGM-EOPS-TN-13-0009, Issue 2.3, date of issue 26/07/2021

The following documents are further referenced in this document:

[RD-1]	Pettinari M.L., Chuvieco E. (2021) ESA CCI ECV Fire Disturbance: D1.2 Algorithm Development Plan, version 3.1. Available upon request.
[RD-2]	M.L. Pettinari (2022) ESA CCI ECV Fire Disturbance: D1.2 Product Specification Document, version 7.1. Available at: https://climate.esa.int/en/projects/fire/key-documents/
[RD-3]	Lizundia-Loiola J., Pettinari M.L., Chuvieco E., Storm T., Gómez-Dans J. (2018) ESA CCI ECV Fire Disturbance: D2.1.3 Algorithm Theoretical Basis Document-MODIS, version 2.0. Available at: https://climate.esa.int/en/projects/fire/key-documents/
[RD-4]	Bastarrika A., Roteta E. (2018) ESA CCI ECV Fire Disturbance: D2.1.2 Algorithm Theoretical Basis Document-SFD, version 1.0. Available at: https://climate.esa.int/en/projects/fire/key-documents/
[RD-5]	Khairoun, A., Lizundia-Loiola J. Chuvieco, E. Pettinari, M.L., Storm, T., Boettcher, M., Kirches, G. (2023) ESA CCI ECV Fire Disturbance: D2.2 Algorithm Theoretical Basis Document-SYN, version 1.1. Available at: https://climate.esa.int/en/projects/fire/key-documents/
[RD-6]	MapReduce: Simplified Data Processing on Large Clusters, Jeffrey Dean and Sanjay Ghemawat, 2004

3. System overview

This section provides an overview of the Fire_cci system with its system context, its main function and processing chain, and its architecture. Also, a set of system requirements are derived from the requirements in [AD-1] as starting point of the system design and later verification.

3.1. Fire_cci system context

The Fire_cci system's two main counterparts are the Fire_cci team, and on the other side data providers. There are a few more entities the Fire_cci system interacts with as shown in Figure 3-1.

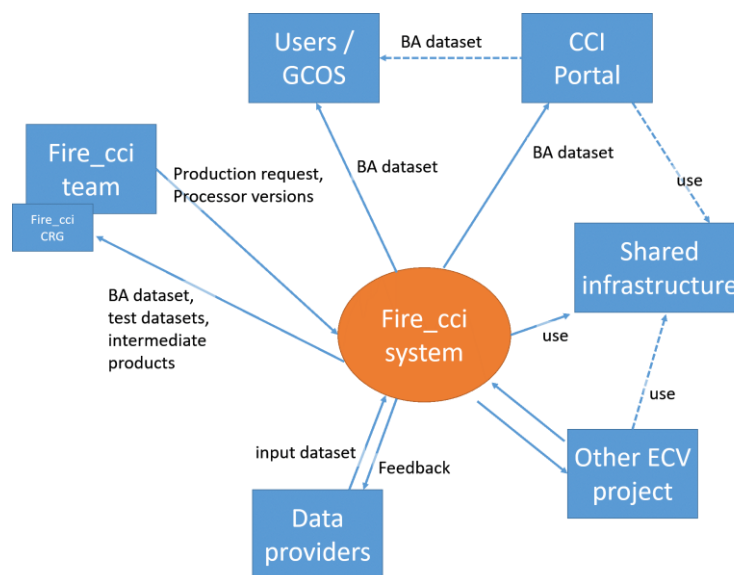



Figure 3-1: System context of the Fire_cci system with team, data providers, and other entities

The entities of the context are:

- The Fire_cci team as provider of processors, high level requests to produce certain results and feedback, and as first consumer of results, among them test datasets and intermediate results for assessment. Note that for this version of the system it is assumed that the Fire_cci team provides the validated products to users (not the processing system). The Fire_cci CRG, as part of the Fire_cci team, contributes to product quality control and assessment by intercomparisons.
- Data providers of the required satellite input data, in particular ESA.
- Other ECV projects as both providers of certain auxiliary datasets, among them the CCI Land Cover product (LC_cci) for a background land classification map.
- The CCI portal as a distinct receiver of the validated BA product.
- Other data users, such as GCOS or ECMWF, especially in the context of the Copernicus Climate Change Service.
- The common part of the processing infrastructure (i.e. the Calvalus system of Brockmann Consult, and CreoDIAS).

The interfaces towards these entities partially determine the Fire_cci system in addition to its main function that is subject of the next section.

	Fire_cci System Specification Document	Ref.:	Fire_cci_D3.1_SSD_v2.4		
		Issue	2.4	Date	23/05/2023
		Page			11

3.2. Main function and processing chain

The main function of the Fire_cci system is the repeated generation of the BA products with different input datasets and algorithms that undergo phased updating.

Functions of data management as well as sharing help to cope with the large amount of input data. Functions for processor integration, configuration control and versioning help to support the continuous development of the Fire_cci project.

3.3. System requirements

System requirements are usually derived from use cases, user requirements, and other inputs based on a first decomposition of the system into elements. In the CCI projects, the focus is mainly on the products to be generated, rather than on the system, which is only the means to produce the product but not a deliverable itself. Therefore, the main requirement for the system is to generate the product according to the ATBD and the ADP.

This leads to a first set of system requirements, as detailed in Table 3.1.

Table 3.1: System requirements in the SoW

ID	Title	Requirement
010	Reprocessing capability	The Fire_cci system shall be able to (re)process complete missions of Earth Observation (EO) data.
020	Processor improvement cycle	The Fire_cci system shall support the improvement cycle of processors and provide configuration control for processor versions
030	Sentinel data handling	The Fire_cci system shall ingest and process Sentinel data
040	Initial capacity and resources	The Fire_cci system shall provide sufficient space for the input and output data, and provide sufficient 200 compute cores for concurrent processing.
050	System scalability	The Fire_cci system shall be scalable by adding new hardware, without change in the architecture.
060	Additional missions	The Fire_cci system shall be extendable for additional missions by adding the corresponding processors

The ADP [RD-1] and PSD [RD-2] identify and specify two user products to be generated:

- The “Pixel BA product”
- The “Grid BA product”

The products shall further be compliant with the CCI data guidelines [AD-3] regarding format (NetCDF-CF), naming, and metadata. This leads to another set of system requirements, detailed in Table 3.2.

Table 3.2: System requirements specified in the ADP/PSD and CCI data guidelines

ID	Title	Requirement
110	Pixel BA product	The Fire_cci system shall generate the Pixel BA product according to the specification in the ADP/PSD.

ID	Title	Requirement
120	Grid BA product	The Fire_cci system shall generate the Grid BA product according to the specification in the ADP/PSD.
130	CCI data guideline compliance	The Fire_cci system shall generate NetCDF products compliant in naming, metadata and format to the CCI data producer recommendations and to the CF convention.

The ATBDs [RD-3 to 5] identify algorithms to be integrated as processors into the Fire_cci processing system:

- Pre-processing processors
- BA processors

This leads to another set of system requirements, detailed in Table 3.3.

Table 3.3: System requirements specified in the ATBDs

ID	Title	Requirement	Source
210	Pre-processing processors	If necessary, the Fire_cci system shall integrate processors for pre-processing to generate surface reflectance (SR) products from Level 1 inputs according to the ATBD.	[RD-3]
220	BA processors	The Fire_cci system shall integrate BA processors to generate BA products from SR products according to the ATBD.	[RD-3]

4. Fire_cci system architecture

This section describes the Fire_cci system architecture. It first provides an overview over the high-level decomposition (Section 4.1), and an in-depth description of the Calvalus sub-system in Section 4.2.

4.1. High-level system decomposition


In the previous contracts, the BC Calvalus system has been used to fully produce the MERIS-based BA dataset (FireCCI41) and the Sentinel-2 based BA dataset (FireCCISFD20), in both cases starting from the L1 input data, generating the final user products. Also, because the MapReduce approach (see [RD-6]) employed by Calvalus is an excellent fit to the formatting of the BA data to the final user products, this last step has been performed on Calvalus for all of the Fire_cci datasets processed by BC, which are FireCCI41, FireCCI50, FireCCI51, FireCCISFD20, FireCCIS310, and FireCCILT11.

Within the current contract, the 2021 dataset of the FireCCI51 time series and the FireCCIS311 datasets are completely being produced on Calvalus.

The following sections address the architectures and technical concepts and methods used for each sub-system.

4.2. Calvalus system

The Fire_cci system is largely based on Calvalus and the underlying infrastructure, with certain elements specifically configured and extended for Fire_cci (Figure 4-1). In

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4		
			Issue	2.4	Date	23/05/2023
			Page		13	

In addition to the software stack with many functions of the Fire_cci system provided by Calvalus, Hadoop or other frameworks, there is a shared cluster infrastructure that runs the corresponding services.

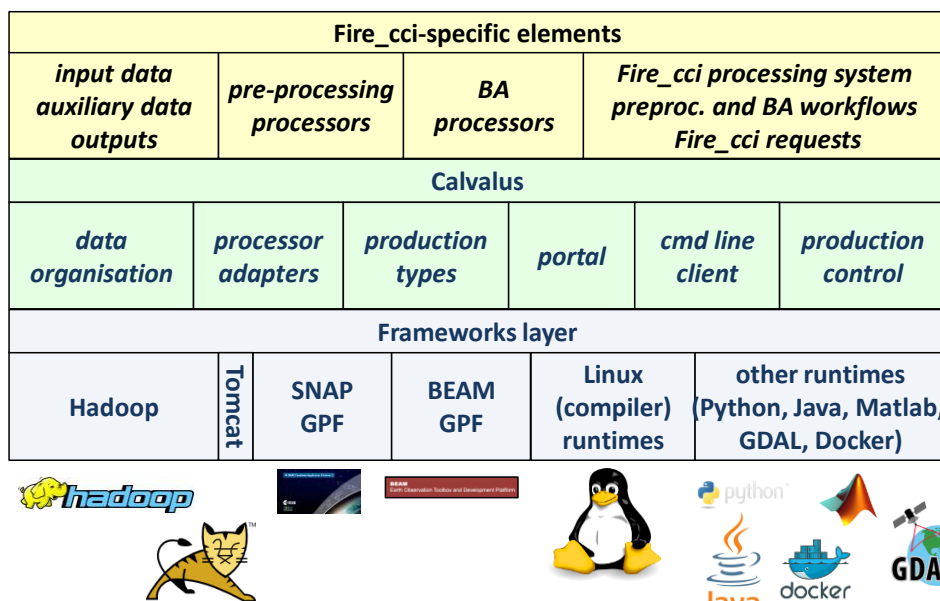


Figure 4-1: Fire_cci system architecture layers with specific elements, Calvalus, and Hadoop

The main elements that are specific to Fire_cci are:

- Access to the shared input data, space for auxiliary data and Fire_cci project intermediates and outputs.
- Processors or Fire_cci-specific processor configurations of existing processors for pre-processing.
- The BA processors and their wrappers for integration into Calvalus.
- The Fire_cci processing system instance on a virtual machine for the control of all Fire_cci workflows, with rules and request templates for pre-processing, BA processing, and formatting and all the respective dependencies.

In addition to this, there are several Fire_cci-specific configurations of shared elements, in particular:

- A “fire” queue in the Calvalus Hadoop scheduler for the adequate and fair sharing of the cluster processing resources.
- space on the shared BC’s FTP server

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

- The Calvalus data organisation for EO data, auxiliary 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 Fire_cci processors, in particular SNAP for the processors for pre-processing, 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.

Calvalus is proprietary software developed by Brockmann Consult GmbH since 2010. It employs the Apache Hadoop software 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 Sentinel 1-2-3.

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 4-2 and Figure 4-3). 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.

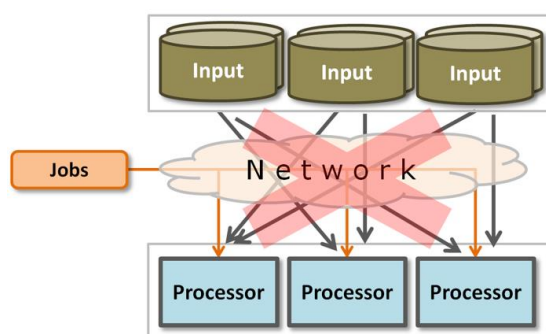


Figure 4-2: Archive-centric

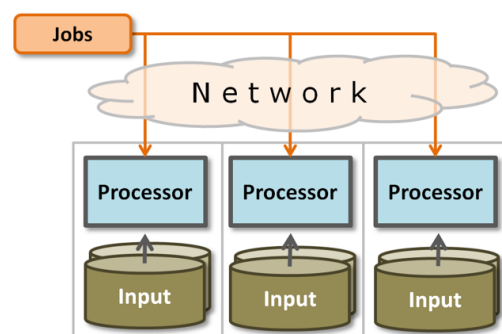


Figure 4-3: Data-local

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 Fire_cci, which are provided by project partners, are integrated into software bundles, and as such integrated into the system.

Hardware Infrastructure

The infrastructure used for Fire_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 113 processing nodes and a storage capacity of about 2.4 PB.

Figure 4-4 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.

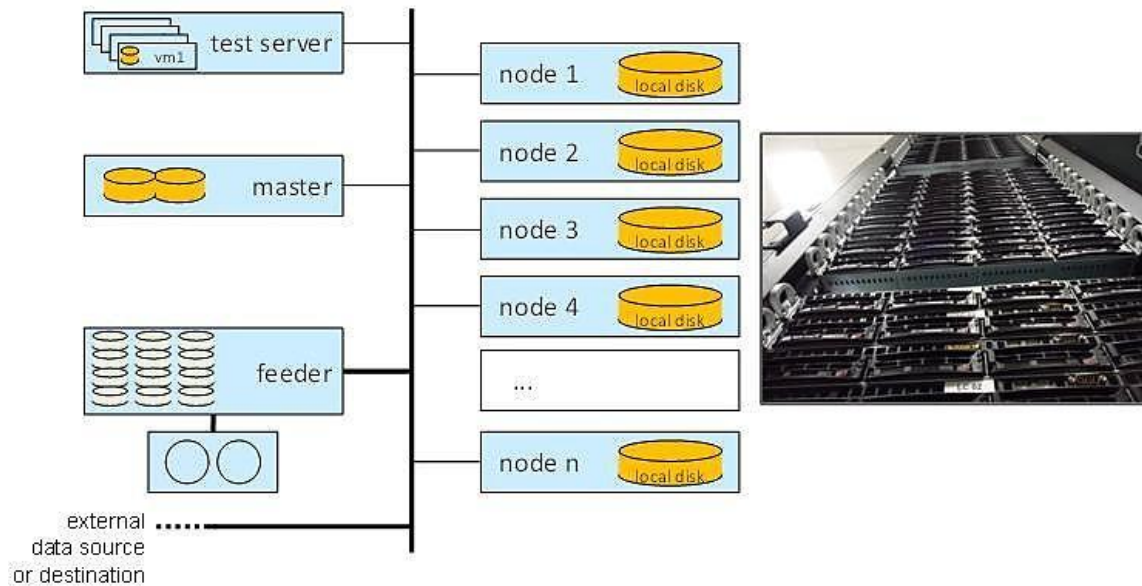


Figure 4-4: Calvalus architecture

This approach of Hadoop as middleware and with computing nodes that are at the same time storage nodes has been selected for Fire_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 pre-processing. The approach is scalable to several petabytes which is an advantage regarding the considerable data volumes needed for Fire_cci processing.

Software infrastructure

The software system for Fire_cci processing deployed on the infrastructure above is shown in Figure 4-5. The different layers of the software system start from the operating system up to the individual processors.

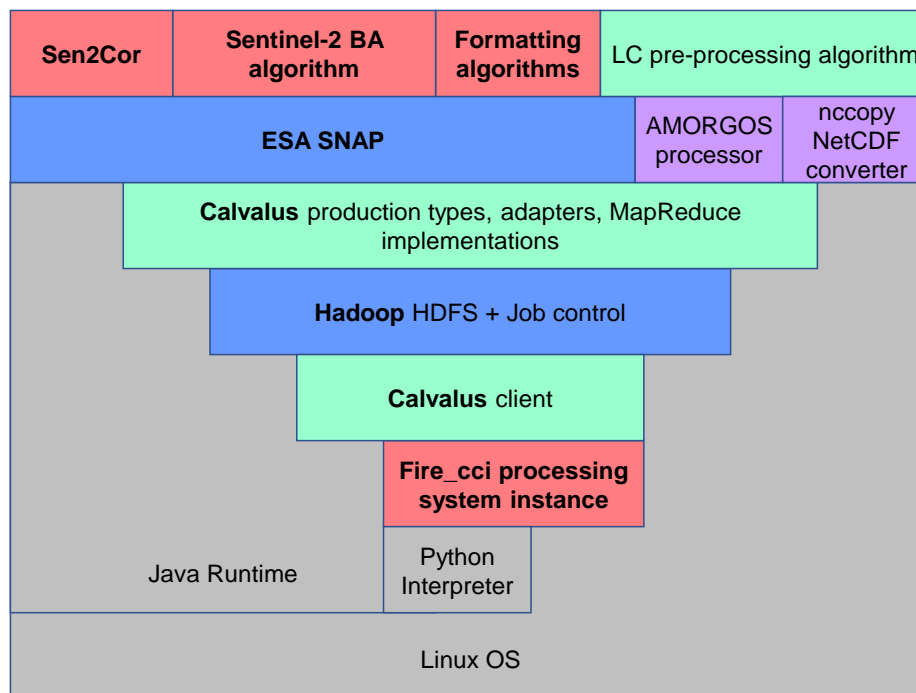


Figure 4-5: Software architecture

The layers are:

- Basic software (shown in grey) comprises the Linux operating system, the Java runtime, and the Python interpreter.
- 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 processors used within the Fire_cci processing are from NetCDF tool shown in pink. They are integrated as Unix executables into Calvalus for parallel execution.
- Specific elements developed or assembled for Fire_cci (specifically for FireCCISFD20) are Sen2Cor (used for Sentinel-2 pre-processing), the Sentinel-2 BA algorithm, the formatting software, and the Fire_cci processing system instance. They are shown in red. They will get updates to cover additional sensors.

Like the hardware, the software is also for the most part shared with other projects. But this needs more precise consideration. 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 Fire_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 ESA 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. Examples are Apache Hadoop maintained as open source by Apache Foundation, the programming language runtimes.

Table 4.1 lists the software items with their role in the Fire_cci processing subsystem and its configuration control.

Table 4.1: Software items for Fire_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_202), formerly available from http://www.oracle.com/technetwork/java/index.html ²
Python interpreter	Basic software used for pmonitor	Version 2.7 and 3.9, available from www.python.org
Apache Hadoop	Cluster software with data management HDFS, job scheduling, command line tools and Web operating interface, several APIs: client side for job submission and monitoring, server side for plug-in of map and reduce tasks.	Version 3.2.1, versions maintained by Apache, available from hadoop.apache.org ² (open source)
Calvalus	Earth Observation application layer on top of Hadoop with HDFS archive directory structure with archiving rules, client tool for request submission, software deployment, data ingestion processor adapters for various programming languages (among them for SNAP operators processor deployment as versioned software bundles). User portal for on-demand processing. Processing system instances for controlled bulk production.	Version 2.19, version control on GitHub in repository of Brockmann Consult GmbH
ESA SNAP	Framework for processor development and execution, concepts for product, reader, writer, operator, operator chaining, tile cache and many more, basic software for LC pre-processor, Level 3 aggregator, and overlap determination.	Version 7.0, provided by ESA, maintained and further developed by Brockmann Consult GmbH, version control in public repository (open source).
Daily quicklook aggregation	Generates quicklooks with global extent of the input products of a day, uses L3 aggregation with sub-sampling, and final JPEG formatting. Used for input screening for quality checks.	Maintained as part of Calvalus.

	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4		
			Issue	2.4	Date	23/05/2023
			Page		18	

Software item	Purpose, use	Configuration control
Fire Level 3 aggregator (production type “FireL3ProductionType”)	Aggregation of surface directional reflectance (SDR) values, reprojection, application of a temporal cloud filter, tiling and NetCDF-formatting of the result.	Maintained as part of Calvalus.
Fire SR product writer	This SNAP product writer implementation is a modification to the LC_cci product writer, which in turn extends the NetCDF 4 CF writer to set band names and attributes, global attributes, and the file name according to the LC_cci product specification. It is used in the generic Calvalus product formatting step.	Maintained as part of Calvalus.
Fire Formatting Tool	Uses the burned area data in order to create ADP/PSD-compliant, formatted final user data sets.	Maintained as part of Calvalus.
nccopy NetCDF conversion tool	Used to convert SNAP-generated NetCDF 4 files into classic format as required by CCI product convention	NetCDF library version 4.1.3, available for Ubuntu via apt-get.
Quicklook generator	Uses the generic L3 workflow to mosaic the tiles into one quicklook JPEG image.	Maintained as part of Calvalus.
fire-inst	Processing system instance with workflow control and processing progress and status.	Maintained in BC private repository, state backed up daily
fire-1.x	Software bundle containing the BA processor and the environment it uses.	Versioned; each update gets a new version, while old versions are kept and can be re-used.

5. Workflows

This section describes the processing workflows in detail, the data they use, the techniques employed, and the outputs they create.


5.1. SFD workflow

This section describes the processing workflow for the Small Fires Database (FireCCISFD20) product. It uses Sentinel-2 L1C data as input and generates a tiled 20-meter pixel product and a 0.05-degree grid product for sub-Saharan Africa for 2019. The complete workflow runs on a system deployed at CreoDIAS. CreoDIAS is also the source of the Sentinel-2 input data.

5.1.1. Sentinel-2 pre-processing

Sentinel-2 pre-processing is performed with the ESA standard processor Sen2Cor. 2904 granules with about 425000 granule items have been pre-processed to L2A products. Sen2Cor processing has been performed by the CreoDIAS team.

During QA (see below) several broken L2A products have been detected incrementally. They have been re-processed from L1C on Calvalus using a Sen2Cor processor bundle and individual Json requests.

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4	
			Issue	2.4	Date
				Page	19

5.1.2. Burned Area processing

The Burned Area retrieval for Sentinel-2 is done using the Sentinel-2 BA algorithm developed by EHU (see [RD-4]). The four steps of BA processing are:

- pre is the BA detection. It uses VIIRS active fire as ancillary data input. Each call gets a granule item and a predecessor of the same granule and platform up to 40 days in the past. These are either 8 calls per granule item, or 4 calls, depending on how many overpasses per repeat cycle we have for that granule. In sum, these are ~2,200,000 pairs of a granule item and a predecessor
- post merges all pair results with the same successor, ~425,000 steps. This narrows down the date of burn.
- fuse verifies and confirms BA by comparing S2A and S2B detection. Both platforms mutually confirm detected BA. If only one platform reports it, it is dropped.
- tile reprojects to geographic lat-lon and generates composites of monthly 5x5-degree tiles. This finally determines the date of burn with the resolution of the temporal cycle considering S2A and S2B, under non-cloudy conditions between 2 and 5 days.

To get larger jobs and to avoid too many jobs (in the order of 2,000,000), pre, post and fuse is requested with one job per granule, not per granule item pair. Each job processes the complete time series and is controlled by a generated table with all pairs possible and available for that granule between November 2018 and December 2019. The table of pairs is pre-generated by a catalogue query, extensive sorting, and the identification of all pairs considering the time constraint of 40 days. Each job spawns tasks, one for each pair. A typical number of tasks per job is ~1000. All tasks can be executed concurrently. post has to wait for pre and spawns concurrent tasks to merge the pairs with the same successor. Fuse waits for post in turn and spawns tasks, one for each (successor) granule item.

Tile jobs are generated per month and tile. Each job collects all granule item outputs of the fuse step overlapping with the tile area. It collects them for a period covering more than the month itself to exclude that a BA is reported again after it has been reported in the previous month already. The earliest detection counts.

The design of the algorithm, in particular the independent consideration of pairs of consecutive S2 granule items, allows for a very high degree of concurrency. The disadvantage of this method is that each granule item is read several times, usually 5 to 8 times, to be accurate. The advantage is that it can handle arbitrary long time series that do not need to fit in memory altogether. A pair at a time, and later the up to 8 pair results for one post step are enough.

5.1.3. ADP/PSD product production

The ADP/PSD-compliant formatting of the BA data has been implemented with the goal to exploit the massive parallelity of the Calvalus system. This implementation has been employed to generate the user data compliant to the ADP/PSD. The formatting step works on single months, and takes as input all global tiles for each month.

Both steps consider and merge a land cover (LC) layer into the data. The algorithm distinguishes burnable and non-burnable areas with LC, and it is added to the output. The LC layer used is CCI LC 2018 with 300m resolution. For the pixel product the LC layer is up-sampled to 20m.

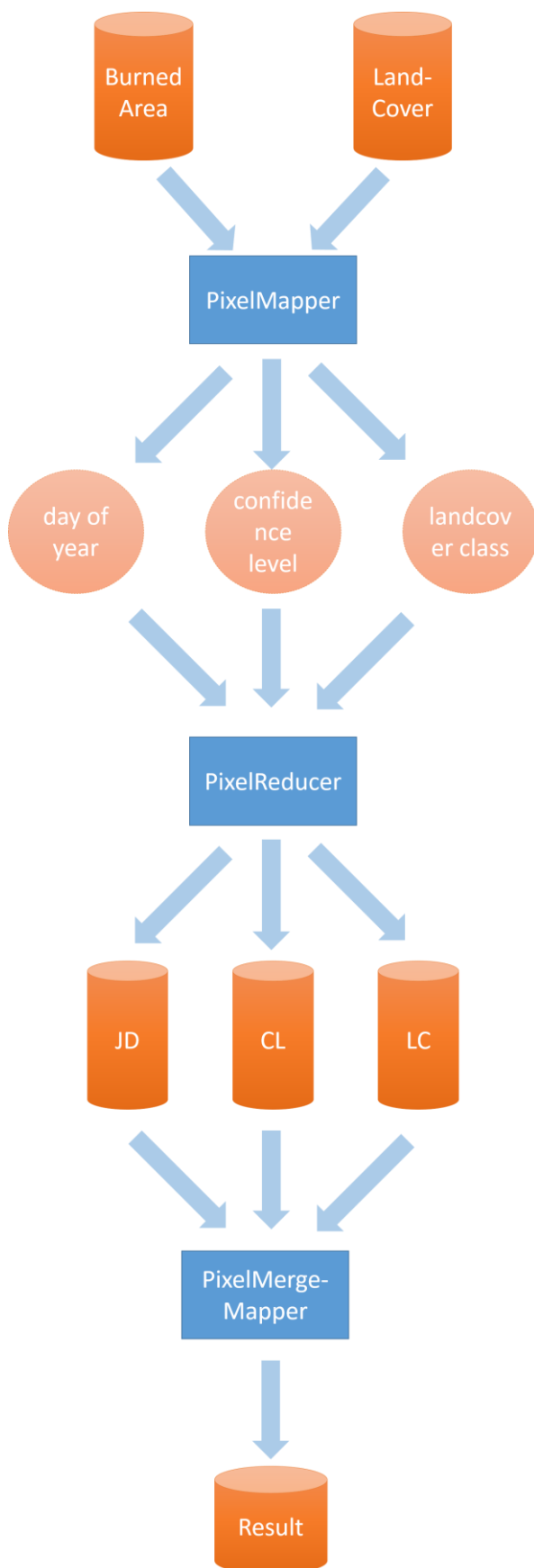


Figure 5-1: Formatting

Pixel processing

Due to the higher resolution of the resulting products, the pixel processing has higher demands on memory consumption. So, the process has been split up twice: the first step computes the output variables separately in a MapReduce step, and the second step puts them together (Figure 5-1). The process produces data for a single month and one of the six target areas defined in the ADP/PSD.

In the first step, a mapper implementation takes each tile and creates the respective output for one of the three output variables day of year, confidence level, and LandCover class. So, it is run three times for each tile in parallel. The pixel output is generally equivalent to the input.


The PixelReducer runs once for each output of the mapper. It creates actual files in the file system (as opposed to the PixelMapper, which keeps the data in memory). Finally, the PixelMergeMapper collects these data, and creates the result product and the respective metadata.

Error handling in the pixel processing is done using the Calvalus system's error handling protocol. Generally, no errors are accepted, as the process is expected to successfully use all available burned area data. Consequently, the overall failure rate of the process is 0%.

Pixel products are 5x5 degrees as the tile products. There are 116 tiles and 12 months.

Grid processing

The grid processing basically consists of both a mapper and a reducer implementation. The mapper implementation works in parallel on each available tile; it does the aggregation of the burned area and computes the errors as well as the patch

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4	
			Issue	2.4	Date
				Page	21

numbers, the fractions of observed area, and the burned area per Land Cover class. Since there are two grid products for each month, the mapper aggregates the data accordingly.

The reducer implementation retrieves the output of all the mappers, creates the two result files (one for each half of the respective month), fills them with metadata, and writes the mapper's output into them. These result files are archived afterwards, ready to be delivered to the users. This approach allows for a high degree of parallelisation.

Error handling in the grid processing is done using the Calvalus system's error handling protocol. Generally, no errors are accepted, as the process is expected to successfully use all available burned area data. Consequently, the overall failure rate of the process is 0%

Grid products are 0.05-degree global user products.

5.1.4. Processing quality assurance

Right after the start of production during the inspection of the first tile column it became obvious that there are unpredictable artefacts in the products. As it was unknown initially which types of errors will appear all outputs are screened. Quicklooks (QL) have been generated for all tiles during production. They have been inspected progressing with the production. Figure 5-2 shows the 12 monthly QL of one tile.

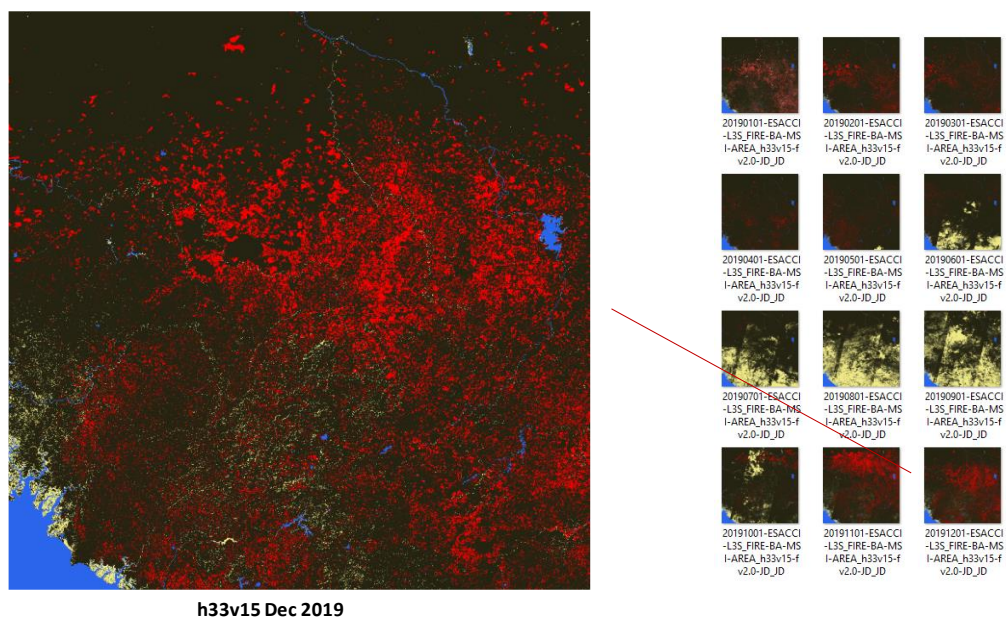


Figure 5-2: Quicklooks of JD for one tile

One of the issues found is a displacement of parts of an image as shown in Figure 5-3. It could not be finally distinguished whether this behaviour is caused by the data reader of the Sentinel-2 Toolbox or the data itself. However, it can fortunately be solved by re-generating the L2A from a L1C with Sen2Cor on Calvalus. More than 10 granules were affected by this issue.

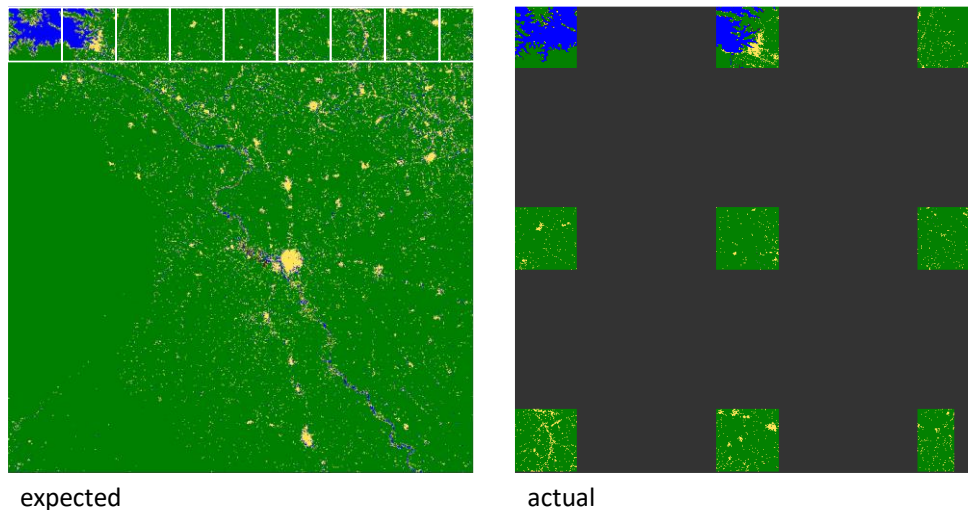


Figure 5-3: 'mini tile effect' – mini tiles of the first line appear at wrong locations

Figure 5-4 shows other types of artefacts observed.

- a) LIC artefacts
- b) Missing tiles
- c) Missing tiles caused by processing region
- d) Mini-tiles effect
- e) Land cover artefacts
- f) Water mask 'artefacts'

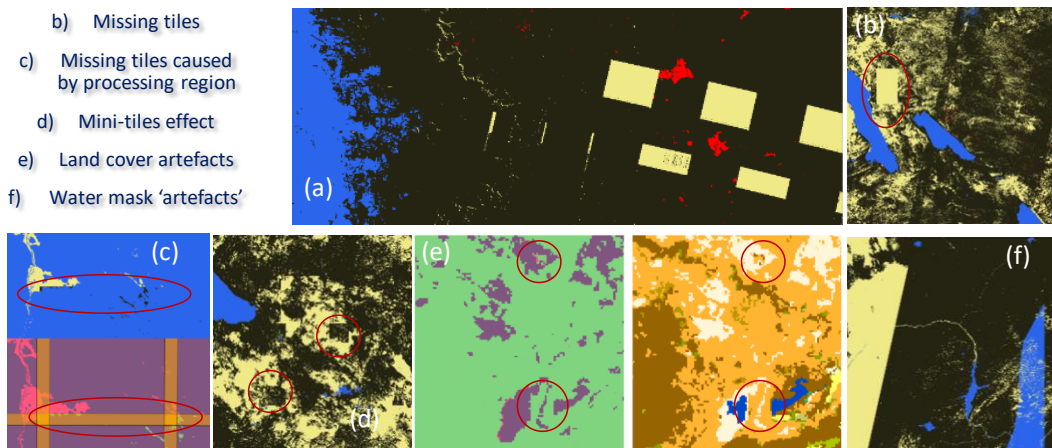


Figure 5-4: Different types of artefacts found and resolved

Some of them, e.g. missing granules, can be resolved by re-processing with an updated (fixed) configuration. Others, e.g. LIC artefacts, can only be fixed by masking the respective inputs.

QA is performed for the complete dataset. Fixes are verified by inspecting re-generated quicklooks for the respective tiles. A table tracks which tiles have issues and which of them are fixed, as shown in Figure 5-5. Systematic QA avoids typical artefacts that else would have been in the automatically generated result.

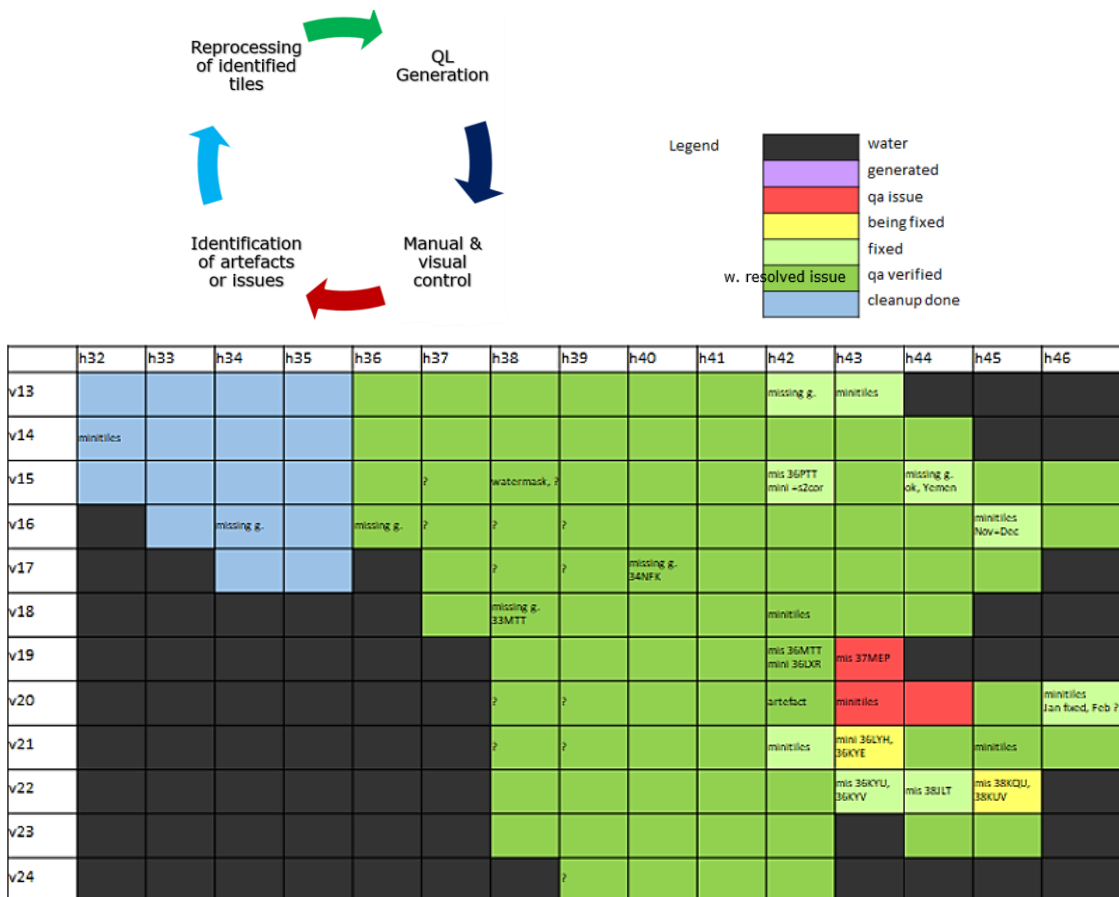


Figure 5-5: Systematic QA based on quicklook inspection and tracking of issues: example of an intermediate stage of the QA.

5.1.5. Tools and modules


The processing steps are implemented as SNAP operators, SNAP aggregators (for the tile step), and as Calvalus plug-ins (pixel and grid formatting, QL generation).

Production is controlled by a Calvalus processing system instance. It uses Calvalus PMonitor for workflow control, with one yaml request for the complete BA processing chain and a workflow implementation that uses PMonitor as job generator and workflow dependency monitor. For Sentinel-2 several tables pre-computed from catalogue query results control workflow generation and parameterisation.

The system makes use of CreoDIAS, in particular its openstack interface for virtual machine creation and scaling up and down the system. It also uses its S3 cloud storage interface to access inputs and store intermediates and results. Calvalus runs on S3 as distributed file system. Finally, the system makes use of the CreoDIAS finder catalogue. It automatically generates and submits catalogue queries and analyses responses to determine which inputs to process.

5.2. SYN BA workflow

This section describes the processing workflow for the FireCCIS311 product. It uses Sentinel-3 Synergy L2 data as input and generates a global tiled 300-metre pixel product and a 0.25-degree grid product for 2020. The complete workflow runs on the Calvalus system described in sections 4.2 and 5.1. CreoDIAS has been used as source for the Synergy input data.

	Fire_cci System Specification Document	Ref.:	Fire_cci_D3.1_SSD_v2.4			
		Issue	2.4	Date	23/05/2023	
				Page	24	

5.2.1. SYN pre-processing

The SYN Level 2 input data is still in a satellite viewing geometry raster. Pre-processing generates daily 10-degree tiles in geographic lat-lon projection. The transformation preserves resolution, i.e. selects a global grid that has ~300m in North-South direction.

The transformation spatially resamples using a nearest neighbour method. Temporally, if there are several overflights a day, the smallest zenith angle shall be selected. This criterion is implemented by an equivalent condition based on the distance to the nadir line in the data product. A pixel closer to the nadir line is preferred. In addition to reflectances pixel state (flags) is preserved during transformation as it is used by the BA algorithm.

Input data from November 2019 to mid-March 2021 is pre-processed. 273 daily tiles covering land are generated.

5.2.2. Burned Area processing

The Burned Area retrieval for SYN is done using the SYN BA algorithm developed by UAH [RD-5]. This algorithm operates tile-wise and monthly. It considers the past time series and requires lookahead to future data to determine BA of a particular month. This requires a temporal scheme to be followed, as shown in Figure 5-6.

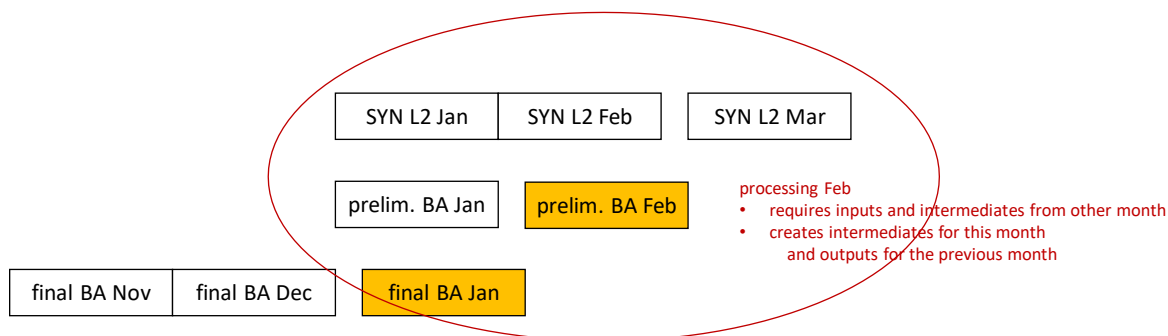


Figure 5-6: The relation of inputs, intermediates, and outputs in SYN BA processing

- For each tile the months are processed one after the other in ascending order.
- We start two months before the first output month, i.e. in November 2019 to get final results from January 2020 onwards.
- Processing a certain month requires SYN L2 input data from before the month, the month itself, and a period of the following month. (Processing February 2020 uses SYN inputs from January to March 2020).
- BA-processing for a month generates preliminary BA results. When the next month is processed these preliminary results are consolidated to get the final results for the month. Example: BA processing March 2020 consolidates results of February 2020. As a consequence, the algorithm uses two months look-ahead: one for the input data and one for consolidation.

5.2.3. Pixel product and Grid product formatting

For the creation of the ADP/PSD-compliant products, the same approach is used for SYN BA as it is used for the SFD described in section 5.1.3. The two final steps of BA formatting are:

- pixel formats tiles into user products. The pixel product aggregates the 10-degree tiles into six continental subsets, as specified by the ADP/PSD.
- grid aggregates and formats tiles into 0.25-degree global user products.

Both steps consider and merge a land cover layer into the data. The LC layer to be used is C3S LC 2018, i.e. the year immediately prior to the BA year being formatted.

5.2.4. Tools and modules

The BA processing steps are implemented in Python and integrated into Calvalus using Anaconda as light-weight container. SNAP aggregators (for the pre-processing step) and Calvalus plug-ins (pixel and grid formatting, QL generation) implement the other steps.

Production is controlled by a Calvalus processing system instance. It uses Calvalus PMonitor for workflow control, with one yaml request for the pre-processing chain, and one yaml request for the complete BA processing chain. The main parameters of the requests are the tiles and the time period. Additional parameters are the directory entry points for inputs, intermediates, and outputs. The workflow uses PMonitor as job generator and workflow dependency monitor.

5.3. MODIS workflow

There is a sub-section for each processing step: Section 5.3.1 deals with the data acquisition, Section 5.3.2 explains the MODIS BA retrieval, Section 5.3.3 describes the data repackaging and transfer, and Section 5.3.4 describes the final step, the formatting.

The basis for the processing is already pre-processed data, which means that no pre-processing is done within the context of Fire_cci. Second, the input dataset has not been available on the system at the start of the processing, meaning that it had to be acquired first. Two datasets needed to be acquired to the system for processing: MOD09GQ (daily surface reflectance data, global, 250m, 1200*1200 km tiles), and MOD09GA (daily surface reflectance data, global, 1km and 500m SIN Grid).

Figure 5-7 shows the logical ordering of the workflow; the single steps are explained in the upcoming subsections.

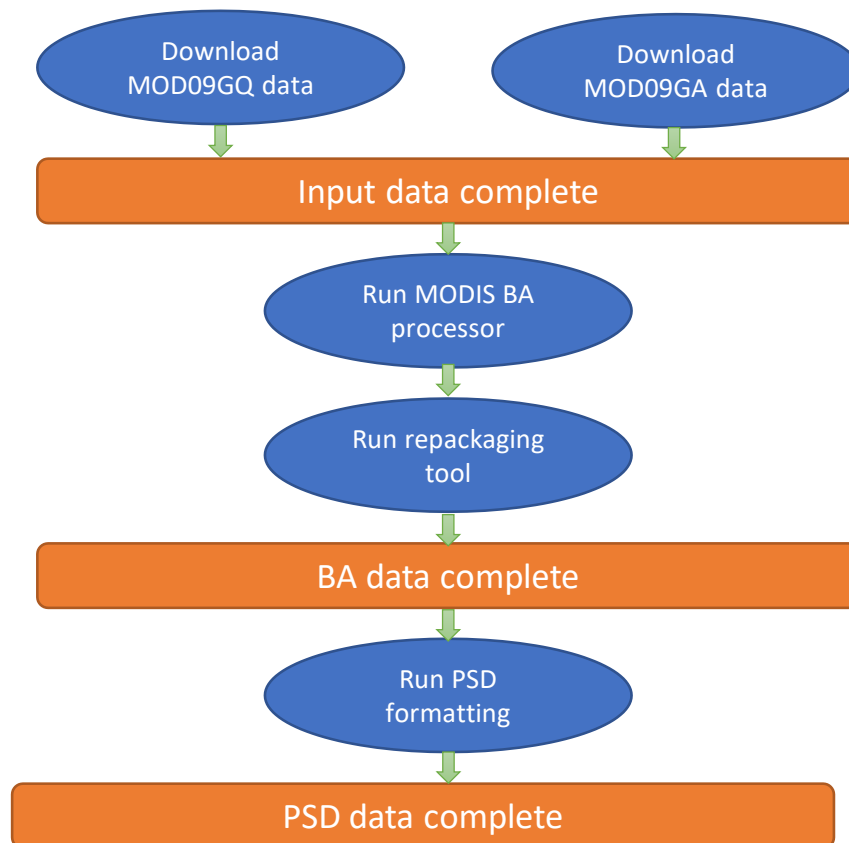



Figure 5-7: Sequence of MODIS processing actions

 fire cci	Fire_cci System Specification Document		Ref.:	Fire_cci_D3.1_SSD_v2.4		
			Issue	2.4	Date	23/05/2023
					Page	26

5.3.1. Data acquisition

As stated before, there are two datasets that needed to be acquired to the system for processing: MOD09GQ (daily surface reflectance data, global, 250m, 1200*1200 km tiles), and MOD09GA (daily surface reflectance data, global, 1km and 500m SIN Grid).

In order to reliably download those datasets, a dedicated script has been set up, which allows to start multiple (typically ~40) download jobs in parallel, tracks successfully downloaded files, compares checksums of the downloaded files with the checksums provided by the download site, and reports failing download attempts. Thus, it is able to handle broken connections, incomplete downloads, and the regular and irregular system downtimes. This script has been run on a dedicated transfer node of the Calvalus system, which allowed for high download rates.

5.3.2. Data processing

The MODIS BA processor has been described in [RD-3].

In order to run the MODIS BA processor on Calvalus, the environment had to be prepared accordingly. The MODIS BA processor has been written in Python2, and has a set of dependencies that are not all fulfilled by the Python version already installed on the Calvalus nodes, such as `gdal`, `ogr`, `scipy.spatial`, or `cv2`. Thus, another version of Python2 has been prepared, which contains all the needed dependencies. Prior to the processing execution, a dedicated activate-script has to be called, which sets up the environment and ensures that the correct python version is being used.

The processing has been split up into MODIS tiles and into years, which leads to a large number of processing tasks. These larger number of processing tasks are orchestrated, monitored, and controlled using a dedicated pmonitor workflow as described in section 5.1.5.

5.3.3. Data repackaging

The data produced by the MODIS BA processor contains a high number of image files, which are not used for ADP/PSD product creation, but serve as temporary files or are used for further analysis. The outputs of the algorithm are repackaged; this means that the image files relevant for ADP/PSD-compliant product creation (= burned area, uncertainty, and fraction of observed area) are put together into an internally compressed NetCDF-file for each month and tile. In order to achieve this repackaging, a Java tool has been written, based on the SNAP library. It accepts as input the result of the BA processor for a given tile, year, and month, and writes as output the internally compressed NetCDF-file. Also, a bash script has been developed, which acts as wrapper around that Java tool and ensures that all results are considered.

5.3.4. ADP/PSD product production

For the creation of the ADP/PSD-compliant products the same approach is used in FireCCI51 as for FireCCISFD20, described in section 5.1.3.

6. System outputs

The system, in the version described in this document, produces the datasets stated in Table 6.1.

Table 6.1: Output products

	MODIS	
	Pixel product	Grid product
Geographic extent	Continental tiles divided in 6 zones: Africa, Asia, Australia, North+Central America, South America and Europe.	Global
	Sentinel 2	
	Pixel product	Grid product
Geographic extent	Sub-Saharan African area corresponding to Zone 5 of the continental tiles, provided in 5x5-degree tiles.	Global, but data only provided over the Sub-Saharan African landmass
	Sentinel-3 Synergy	
	Pixel product	Grid product
Geographic extent	Continental tiles divided in 6 zones: Africa, Asia, Australia, North+Central America, South America and Europe.	Global

Annex: Acronyms and abbreviations

AD	Applicable Document
ADP	Algorithm Development Plan
API	Application Programming Interface
ATBD	Algorithm Theoretical Basis Document
BA	Burned Area
BC	Brockmann Consult GmBH
CCI	Climate Change Initiative
CEDA	Centre for Environmental Data Analysis
CF	Climate Forecast
CPU	Central Processing Unit
CRG	Climate Research Group
DEM	Digital Elevation Model
ECMWF	European Centre for Medium-range Weather Forecast
EHU	University of the Basque Country
EO	Earth Observation
ESA	European Space Agency
ECV	Essential Climate Variables
FTP	File Transfer Protocol
GCOS	Global Climate Observing System
GDAL	Geospatial Data Abstraction Library
GFS	Google File System
HDFS	Hadoop Distributed File System
IPCC	Intergovernmental Panel on Climate Change
JPEG	Joint Photographic Experts Group
JD	Julian Day
LC	Land Cover
LC_cci	Land Cover CCI project
Lat	Latitude

Lon	Longitude
LSF	Load Sharing Facility
MERIS	Medium Resolution Imaging Spectrometer
MODIS	Moderate Resolution Imaging Spectroradiometer
MSI	MultiSpectral Instrument
NetCDF	NETwork Common Data Format
OLCI	Ocean and Land Colour Instrument
PB	Peta Byte
PSD	Product Specification Document
QA	Quality Assessment
QL	Quick Look
RD	Reference document
SDK	Software Development Kit
SDR	Surface Directional Reflectance
SFD	Small Fire Dataset
SIN	Sinusoidal
SLSTR	Sea and Land Surface Temperature Radiometer
SNAP	Sentinel Application Platform
SoW	Statement of Work
SR	Surface Reflectance
SSD	System Specification Document
STFC	Science and Technology Facilities Council
SYN	Synergy Sentinel-3 data
TB	Tera Byte
UAH	University of Alcala
VIIRS	Visible Infrared Imaging Radiometer Suite
YARN	Yet Another Resource Negotiator