LC System Specification Document

Milestone	System Specification Document
Authorized by	P. Defourny (UCL)
Authors	M. Boettcher (BC); O. Danne (BC); S. Bontemps (UCL), C. Lamarche (UCL), P. Defourny, (UCL)
Distribution	ESA: O. Arino, V. Kalogirou

Document Change Record

Issues

ISSUE	DATE	REASON
0.8	26/10/2012	Version for LC CCI internal review
0.8.2	13/12/2012	Updates for first two RIDs done
0.8.3	27/12/2012	Integration of internal UCL review results
1.0	11/12/2012	Version 1.0 sent to ESA

Detailed Changes

ISSUE	RID	PROBLEM DESCRIPTION	SECTION	CHANGE
0.8	SSD/1	Dangling document links	several	links to sections and to reference documents corrected
0.8	SSD/2	Incomplete sentence	page 56	sentence reworded
0.8	SSD/3	TBCs in cost estimation	page 64	

Summary

This Land Cover system specification document defines the Land Cover system for the 3 years of the second phase with potential for the time beyond. It proposes a cluster solution based on Apache Hadoop for pre-processing combined with an infrastructure optimised for classification. The processing infrastructure is an evolutionary development based on the Land Cover prototype system. It is supplemented by services for users providing information and data access, and options for interaction and involvement. The elements of this system are functionally designed. A hardware infrastructure is proposed and the costs are estimated.

Symbols and acronyms

AD	Applicable Document
ATBD	Algorithm Theoretical Basis Document
BRDF	Bidirectional Reflectance Distribution Function
CCI	Climate Change Initiative
CDR	Climate Data Record
DARD	Data Access Requirements Document
ECSS	European Committee for Space Standardisation
ECV	Essential Climate Variable
ESA	European Space Agency
EO	Earth Observation
FR	MERIS Full Resolution
JRC	Joint Research Centre
LC	Land Cover
LCCS	UN Land Cover Classification System
MERIS	Medium Resolution Image Spectrometer
NDVI	Normalised Difference Vegetation Index
PFT	Plant Functional Types
PSD	Product Specification Document
RD	Reference Document
RR	MERIS Reduced Resolution
SR	Surface Reflectance
SRD	System Requirements Document
SoW	Statement of Work
UCL	Université catholique de Louvain
URD	User Requirements Document
SPOT-VGT	Spot Vegetation

Table of content

1 Introduction

1.1 Purpose and scope

This System Specification Document (SSD) defines the Land Cover (LC) system for the 3 years of the second phase with potential for the time beyond.

The SSD proposes a design for the operational LC system of ESA's Climate Change Initiative (CCI) project. The system shall be developed and used in phase 2 of the CCI project running from 2013 to 2016. LC is one of 13 Essential Climate Variables (ECV) currently studied by CCI. The system covers production, user services, and - dependent on funding or volunteers - long term stewardship for the LC contribution to the Climate Data Record (CDR) to be generated and continuously updated and improved by CCI.

The SSD is a response to the LC System Requirements Document (SRD) [AD 4] and is a deliverable of the LC CCI project requested in the Statement of Work (SoW) [AD 1]. In addition to a structured description of the system the SSD analyses several trade-offs, among them the choice of an infrastructure (cluster, or deployment in a cloud) and the sharing with other ECVs (a system per ECV, a system per group of ECVs, one CCI system for all ECVs). Important aspects are the system life cycle and growth over time, and the cost-to-performance relation.

The system design is based on experience with a prototype developed for LC CCI phase 1. The prototype is documented in a System Prototype Description (SPD, [RD 6]), an Input Output Data Definition (IODD, [RD 4]), and a Detailed Processing Model (DPM, [RD 5]). While IODD and DPM will be applicable to phase 2 with their main content with repeated updates of these documents, the SPD is replaced by this SSD for phase 2. The degree of re-use from the prototype is one of the topics in this SSD.

1.2 References

The following documents are applicable to this document:

ID	Title	Issue	Date
[AD 1]	ESA Climate Change Initiative Phase I - Scientific User Consultation and Detailed Specification Statement of Work (SoW)	1.4	09.11.2009
[AD 2]	Land Cover ECV Proposal	1.0	05.03.2010
[AD 3]	Land Cover CCI User Requirements Document	2.1	23.11.2010
[AD 4]	Land Cover System Requirements Document	1.1	05.01.2012
[AD 5]	Land Cover Product Specification Document	1.3	18.07.2011

The following documents are referenced in this document:

ID	Title	Issue	Date
[RD 1]	ESA CCI Project Guidelines, EOP-DTEX-EOPS-TN-10-0002	1.0	05.11.2010
[RD 2]	Land Cover Data Access Requirements Document	1.4	03.11.2010
[RD 3]	Land Cover Algorithm Theoretical Basis Document	0.4	28.09.2011
[RD 4]	Land Cover Input Output Data Definition	2.0	12.10.2012
[RD 5]	Land Cover Detailed Processing Model	2.0	12.10.2012
[RD 6]	Land Cover System Prototype Description	1.0	12.10.2012

1.3 Document overview

After this formal introduction,

section 2 provides an overview of the LC CCI system, describing its purpose and intended use, its main requirements, its context, its main functions and components

section 3 describes main operational scenarios and use cases of the system

section 4 is a trade-off analysis looking at the alternatives for system distribution, infrastructure, and sharing with other ECVs

section 5 provides a functional architecture with components and interfaces, ordered by the three aspects of user's views to the system, the system operator's view for reliable production, and the algorithm developers' view for continuous improvements

section 6 is a collection of further analyses regarding re-use of components, system life cycle, and cost and performance

section 7 Traces system requirements to sections of this document

2 Land Cover CCI system overview

This section provides an overview of the LC CCI system, describing its purpose and intended use, its context, its main functions and components.

2.1 Purpose and intended use

The LC CCI system generates LC products and supports the process of algorithm improvement, reprocessing and validation, making it easy to test new ideas. It hosts the data necessary for this. It will be used by the LC CCI development team. It provides data and services to climate users and the LC community.

The LC CCI system aims at providing a large scale land cover classification as part of the climate data record (CDR) derived from the best satellite and auxiliary data using the best retrieval and classification algorithms available and to improve them. The CDR contribution consists of global LC state products, global LC condition products, and a global surface reflectance (SR) composite time series. The products are provided with associated metadata, documentation, and validation reports. Figure 2-1 shows a land cover classification map as an example of what is generated by LC CCI.

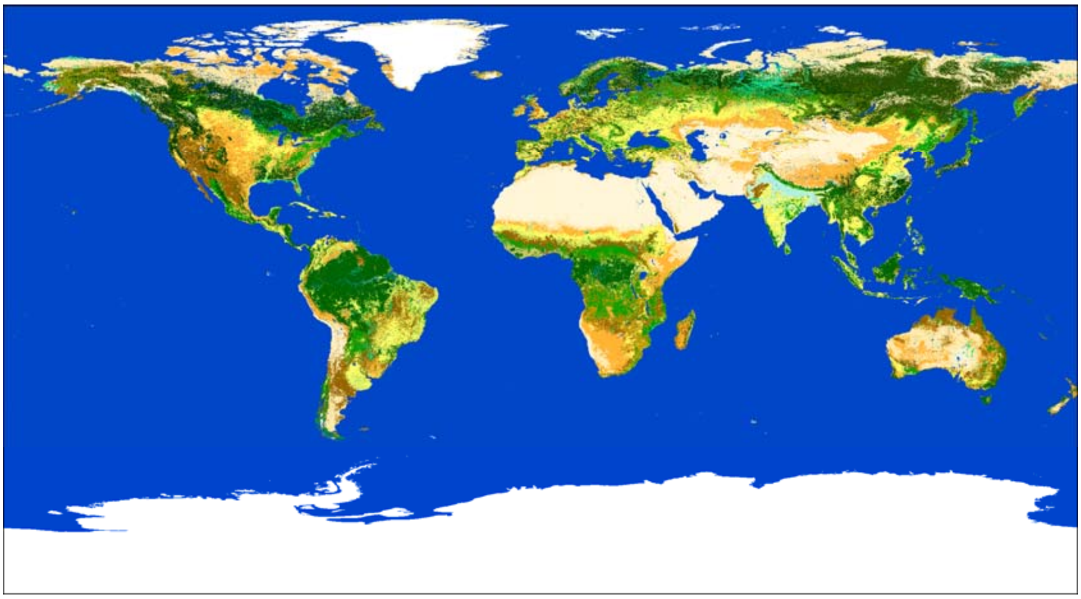


Figure 2-1: The CCI Global Land Cover Map V1 from the 2010 epoch (2008-2012)

LC maps are used in climate models as proxy for land surface parameters based on Plant Functional Types (PFTs) and the United Nations (UN) Land Cover Classification System (LCCS) for thematic characterization. They are further used as proxy for tracking human activities, i.e. land use affecting land cover. And they serve as datasets for validation of model outcomes or to study feedback effects. Features critical for the use in climate modelling are the classification system used, the spatial and temporal resolution, and the accuracy and consistency over time of the LC products.

2.2 Main system requirements

The LC-CCI SRD lists about sixty functional, operational and performance requirements for the system. There are some high level requirements, and some performance and sizing requirements that have an impact on the system design.

- High level requirements are to generate the LC CCI products (LC-SR-0010, LC-SR-0100), to implement LC processing workflows and scenarios (LC-SR-0400), to ensure the stability of the outputs important for climate research (LC-SR-0200), and to be reactive to improvements (LC-SR-0300). This SSD foresees measures for stability by proper versioning and at the same time provides openness to support improvements.
- The quantitative requirements are more on the performance and sizing side than on the reliability and security side because the main scenario is reprocessing. It is more important to have a good overall performance than to be highly reliable in the short term. Regarding sizing, the system shall handle the MERIS dataset as well as Sentinel 3 and optionally Sentinel 2 data, and it shall be able to reprocess all its input data within three months (LC-SR-1100, LC-SR-3310, LC-SR-3335). This leads to the sizing and cost analysis in section 6.
- The functional scope of the system is not restricted to the reprocessing, validation and improvement cycle, though this is its main purpose. Also functions to make available output products and documentation to users are in the scope. This is addressed starting with the following subsections and finally in section 5.

The system requirements addressed by each section are listed in the beginning of the sections and subsections of this document for backward tracing. The traceability matrix in section 7 provides the complete forward tracing from requirements to sections.

2.3 Land Cover CCI system context

The LC-CCI system is a value adder between data providers and users with interfaces to validation experts and other ECV systems.

The LC system is related to its users and its data providers according to Figure 2-2.

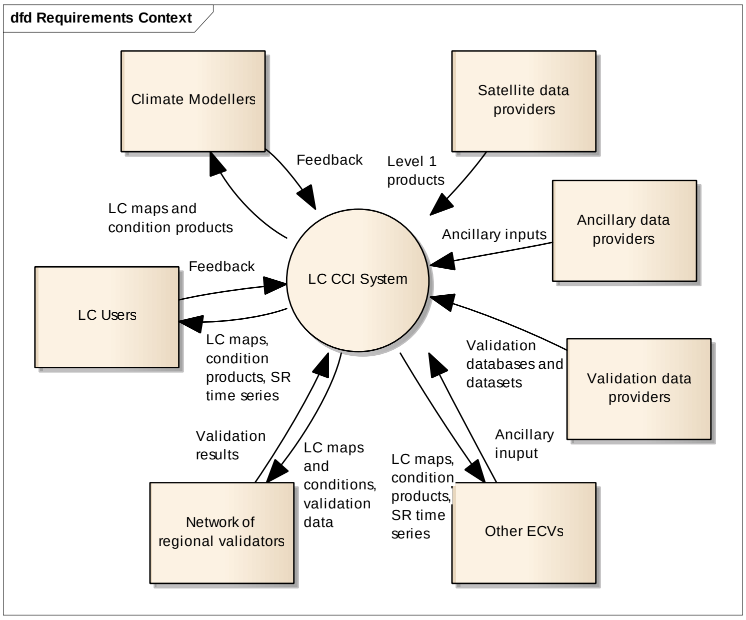


Figure 2-2: LC system context between input data providers and users

- Climate modellers and users from the LC community receive products (and information about them) and optionally provide feedback.
- The interface to satellite data providers usually is the level 1 product.
- Ancillary datasets and datasets used for validation are provided from different external sources and projects.
- Other ECVs may provide inputs to LC, e.g. cloud, aerosol or fire products and may be receivers of LC products for consistency checks.
- The Earth Observation (EO) science team of LC is considered part of the LC CCI system and therefore does not show up in this diagram.

2.4 Main functions

The LC CCI system hosts input data, performs pre-processing, classification processing, supports validation, and serves the output to users. It supports the interaction between development team and users by information services and e.g. a managed forum. Processor interfaces, configurable data management, and version control with easy transfer to operations support test and development of new algorithms and continuous improvement.

To fulfil its purpose in such a context the LC CCI system provides three high level functions:

- production in a broader sense, i.e. the generation of the CDR contribution
- dissemination in a broader sense, i.e. serving the user with data, documentation, services
- improvement, i.e. the chain from feedback handling via updates to versions of the CDR

The Statement of Work [AD 1] asked for the "fundamental operations" of the system. For production the focus is on (repeated) offline reprocessing of complete missions. Functions for this are:

- storage to make available and hold inputs, intermediates, outputs, auxiliary data
- data processors for the processing steps to transform inputs into outputs

- control of processing workflows, and of massive parallel bulk production in particular
- quality checks, automated and visual
- validation, matchups, comparison with reference data
- documentation, metadata to distinguish versions
- ingestion of new data and corresponding auxiliary data for the extension of the CDR

How bulk production is managed is of highest importance here as it is crucial for the agility of the cycles and thus the quality of the outputs.

For dissemination the focus is on the service for climate users. Functions for this are:

- project information, data discovery, catalogue service
- data access, online and optionally on media, bulk access to complete data records
- data customisation, tools available as service and optionally as installable software
- validation support, access to reference data, reports
- documentation, access to documents on products and algorithms, example products
- feedback handling, issue tracking, forum and email communication
- long-term preservation of the data and its representation information

Whether the climate users are served rapidly and with the data and formats they need is a key while to serve the LC community is essential, too, to get support for the program and feedback on data.

For improvement the focus is on a suitable environment with a small effort to exchange processor versions and configurations. Besides the feedback function of user services the main functions are:

- test of new prototype processor versions, provision of pluggable interfaces for processors
- validation of results, web-based tool for external validators
- access to full mission data, reprocessing capability for large sets of inputs
- version management for processors, configurations, and data, with documentation of what has been tested, updated, released

The LC CCI system shall be attractive for the development team to use it for their scientific and development work instead of other local environments. It shall be accessible, easy to use and powerful. This is a key for an agile development process for LC CCI.

2.5 High level decomposition

A processing environment, user services and data stewardship are the three subsystems of the LC CCI.

Requirements addressed by this section are:

→ LC-SR-1110 Backup archive

→ LC-SR-3320 Optimised archive access

The functions listed in the previous subsection are implemented by functional components. These functional components can be grouped in subsystems. This subsection provides an overview of the components of the LC CCI system as starting point for the operational scenarios in section 3 and design trade-offs in section 4. The detailed specification with all components, functions and interfaces follows in section 5.

Starting with a very coarse structure, Figure 2-3 distinguishes three subsystems: production, user services, and data stewardship. They do not completely correspond one-to-one with the functional aspects of production, dissemination, and improvement. They are instead distinguished by the ways they encapsulate earth observation data.

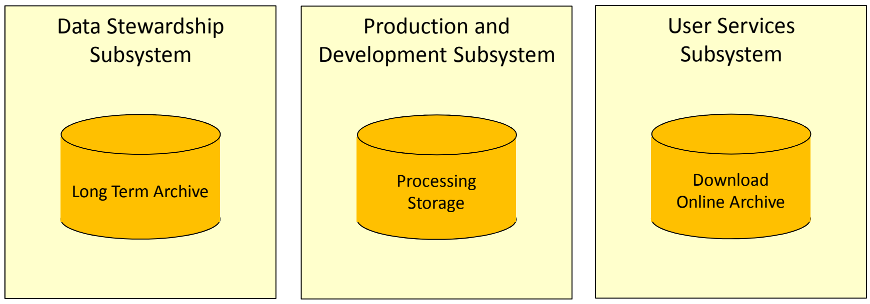


Figure 2-3: Subsystems for production, user services and long term archiving

- Processing storage of the production subsystem in the middle is accessed by processors with high throughput. This shall be protected against direct and concurrent access by users.
- Therefore a second archive accessible online for data download is part of the user services subsystem on the right.
- While the online archive is optimised for external access the long term archive is optimised for reliable long term storage of the data and all representation information required to use it. Typical access to the long term archive is in large chunks of e.g. a complete product set of a certain version. The long term archive is part of a subsystem for data stewardship on the left.

Different uses imply different features and optimisations. This does not exclude that e.g. two of the subsystems are integrated into one or hosted by one organisation. But the interface between them also allows separating them. It is mainly a data interface of earth observation products.

Stepping down to the component level of detail, Figure 2-4 shows components of all three subsystems.

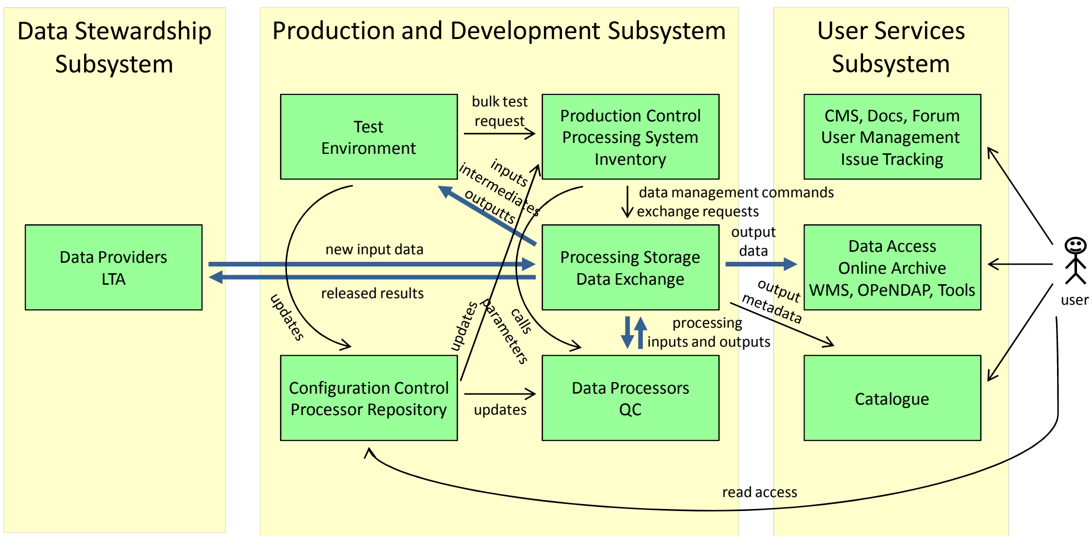


Figure 2-4: Components of the LC CCI system for production, user services and long term archiving

- The components of the two columns in the middle constitute the production and development subsystem. Production control, processing storage and the processors provide the basic infrastructure for processing. A test environment with read access to all data and the option to use the production infrastructure for bulk tests serves the development needs.

A middleware shall implement main functions of the processing system and of processing storage to support bulk production. A middleware is a software service layer above the operating system that makes it easier for developers to develop applications. An example is cluster management software. Depending on the middleware selected the two functions of processing control and data storage can be tightly coupled.

- The user services subsystem consists of at least three components: web presentation, data access by users, and a catalogue. The web presentation includes user forum and issue tracking. Data access offers different protocols and supports online regridding and aggregation. The catalogue is used for product search and metadata access. In addition, users can get read access to the processor repository for documentation and optionally for external use and validation.
- The data providers play a similar role for the inputs as a component for data stewardship plays for the output: long term data preservation and bulk data provision on request. Depending on whether this function is provided externally a component for the preservation of inputs and outputs may have to be foreseen within the system.

2.6 Hardware infrastructure

The hardware of the LC CCI system comprises machines for massive parallel processing, optimised for high i/o for pre-processing, and optimised for multi-threading for classification. All inputs and pre-processing outputs are stored online, preferably distributed for data-local processing. There are devices for offline delivery of the large amount of surface reflectance products and network capacity for online provision of the classification outputs, optionally via a cloud service. Machines for user services and for online data provision are exposed to the internet while the processing environment is accessible via VPN.

Requirements addressed by this section are:

→ LC-SR-3360 Cluster for data-intensive processing

→ LC-SR-3380 Machine for multithreading-intensive processing

The functional components of the LC CCI system are deployed on an infrastructure of machines. Figure 2-5 shows the infrastructure for a Calvalus/Hadoop cluster with several LC CCI additions.

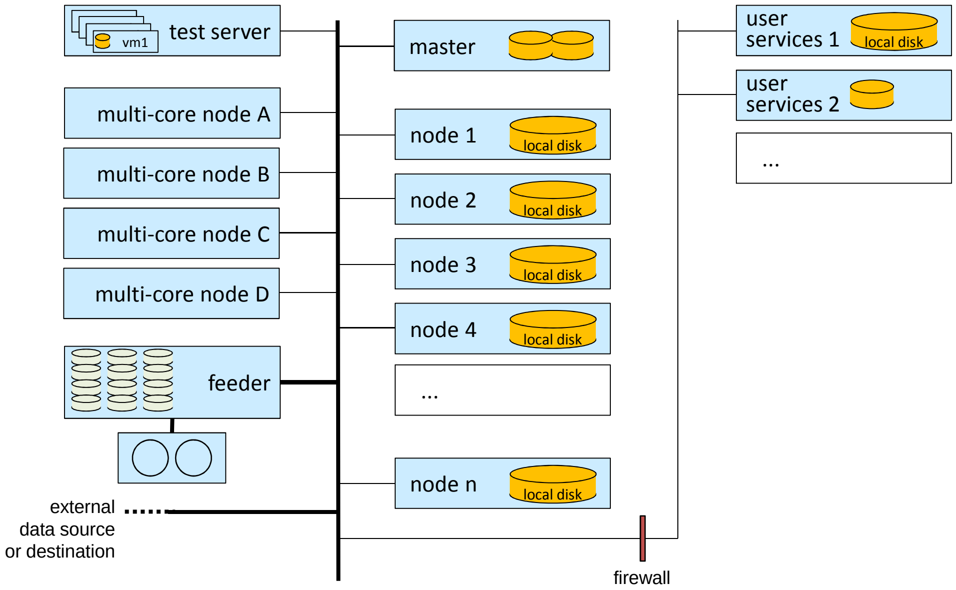


Figure 2-5: LC cluster with two types of nodes for data-intensive pre-processing and for computing-intensive classification

- A cluster of nodes made of standard computers run Calvalus/Hadoop. The machines can be standard rack-mounted computers with 16 GB RAM and a quadcore CPU. Local disks at nodes form the distributed archive managed by the Hadoop Distributed File System HDFS. A master node controls HDFS and Hadoop job and task scheduling. Master and nodes together host the Production Control component, the Processing Storage, and the Data Processors for pre-processing.
- A set of multi-core nodes with 16 to 32 cores each host the Data Processors and Production Control for the LC classification chain. The multi-core nodes have access to the distributed file system of the pre-processing cluster to access the surface reflectance inputs.
- A test server with virtual machines hosts the Test Environment. Virtual machines can be created when necessary. They have read access to the distributed file system for inputs.
- An input output server called feeder provides slots for 24 disks and a tape robot for concurrent ingestion of input data and for writing of output data to media. It is connected with higher bandwidth (10 Gbit/s) to the nodes cluster (switch) to allow for the necessary concurrency.
- User services machines host the different functional components of user services like portal, catalogue, online data server. The file server of the user services hosts LC map and condition products as well as selected examples of intermediate surface reflectance products. There are cold stand-by machines for the user services, and a RAID configuration for online data storage.

While the processing cluster is in a private network accessible by the LC development team via a VPN, the user services hosts are outside of the firewall and accessible by external users.

3 LC CCI processing workflows and operational scenarios

The following sections describe the main LC CCI operational scenarios, in particular: user data access and information, processing and validation of the full or partial CDR, and algorithm improvement. The specification of the elements implementing these functions is elaborated in detail in Section 5.

Requirements addressed by this section are:

→ LC-SR-0010 Generate and disseminate LC products

→ LC-SR-0400 Implement processing workflows and scenarios

3.1 Roles

The development team comprises scientists, operators, system integrators that together manage production and continuous development.

Actors in the operational scenarios are different users. The users are grouped in roles depending on how they use the system.

- climate users
  - interested in datasets consistent over time with infrequent updates
  - data format compatible with their models and other ECVs (NetCDF, projection)
- LC community users
  - interested in best existing land cover products
  - skilled in land cover classification, provide feedback and proposals
  - data format compatible with their uses (GeoTiff, projection)
- development team with scientists, operators, and system integrators
  - has mandate to push forward LC CCI, decide about requirements to analyse and implement, algorithms to test in agreement with ESA
  - is in dialog with users to collect feedback and requirements
  - incrementally improves the system and issues CDR versions
- external experts for LC validation
  - international community
  - support development team, provide local expertise and feedback on the products
- ESA
  - supervises the project, decides about overall direction of the project

Other actors not considered as users of the system are e.g. data providers.

3.2 User information and data access

Users access the LC CCI using the web site is the entry point for metadata and data access, documentation, a catalogue, and data services using various protocols. For questions users interact with LC CCI using a forum and an issue tracking system.

Requirements addressed by this section are:

→ LC-SR-0100 Output products and access

→ LC-SR-1330 Bulk Dissemination

The first operational scenario describes how a user of the data products interacts with the LC CCI system. This conceptually is at the end of the chain of production, validation and dissemination. So, it is assumed that the output data products have been generated and validated. The operational scenario may have the following steps:

- The development team announces a new version of the LC CCI products on the LC CCI web site.
- A user accesses the LC CCI web site as an entry point. The web site offers general information on LC CCI and provides links to services and other resources.
- The user follows links to example products and views them online.
- The user optionally registers to be known by name by the project.
- The user uses the catalogue to find LC CCI products (tiles, regional subsets or global files) that correspond to products from other ECVs. The catalogue also provides metadata and a quicklook.
- The user downloads LC map and condition products in the default format. The user may also use an online aggregation and reprojection tool, or OPeNDAP, or WMS to get subsets of products in a certain form.
- The user in addition downloads the documentation on the product including the Product User Guide.
- The user enters the LC CCI forum and reads contributions by other users and answers from the development team. The user adds a question to the forum that is answered by the development team the following day.
- To get the full dataset or to get more than a few surface reflectance products the user may request for bulk delivery of a dataset. For the map and condition products the actual version is available in a cloud service for immediate download. For surface reflectance products offline delivery is agreed between the user and the development team.

3.3 Pre-processing

The pre-processing workflow comprises radiometric correction, geometric correction, cloud detection, atmospheric correction, and mosaicing. Operators re-process subsets for tests and the complete dataset for the generation of a new version of the surface reflectance product.

Requirements addressed by this section are:

→ LC-SR-3340 Reprocessing support

→ LC-SR-4100 Production control
→ LC-SR-4110 Automated pre-processing
→ LC-SR-2130 Manual failure handling
→ LC-SR-2120 Automated failure handling

Figure 3-1 shows the processing workflow of the LC pre-processing chain. Pre-processing comprises the following steps:

- geometric correction of the L1B inputs
- smile correction and TOA reflectance processing
- cloud and snow identification
- land/water re-classification
- atmospheric correction
- map projection, mosaicing, and temporal aggregation

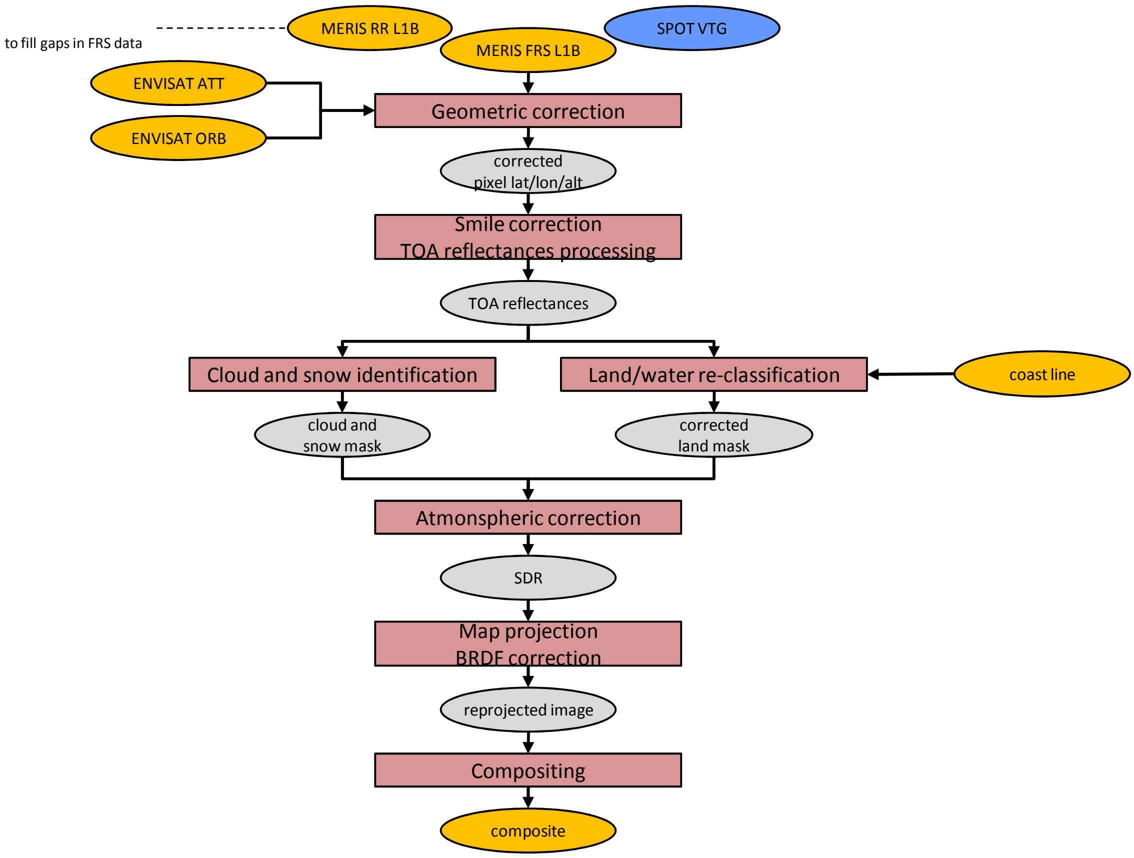


Figure 3-1: LC pre-processing chain inherited from prototype

To run the pre-processing workflow, an operator

- gets the instructions on temporal coverage, processor versions, processing parameters from the development team
- sets up a bulk processing task with the relevant parameters and starts it

- monitors the bulk processing task and the jobs and processing steps derived from it. Processing is controlled automatically.
- analyses errors if any and restarts failed jobs. If tasks fail they are automatically retried by the system.
- provides the results to the development team for QA and reports on the production
- re-runs certain parts of the production if necessary to fix data issues
- manages data storage and if necessary stores or deletes output products

To release the pre-processed dataset as a new version, an operator

- ensures that the processor software is freezed and processing parameters are version-controlled
- tags the output product with the product version

The process of quality checks for pre-processing is depicted in

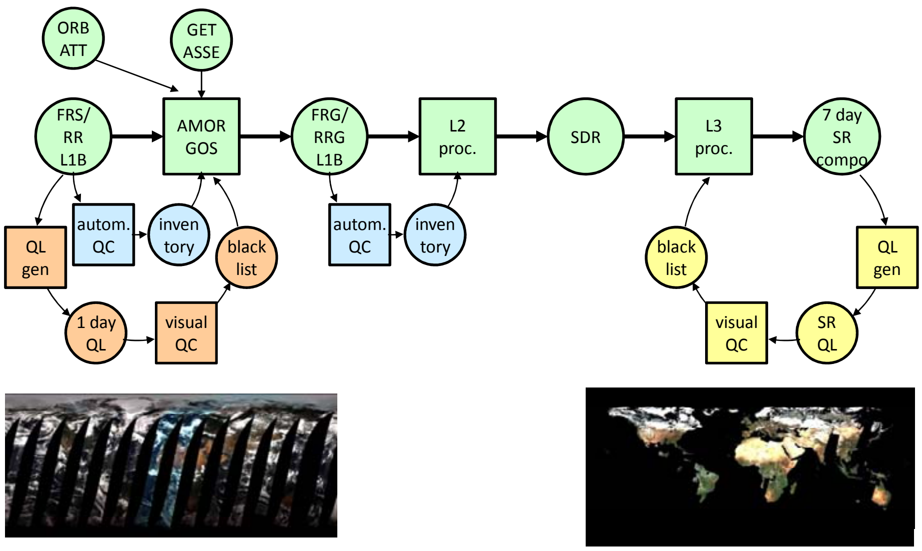


Figure 3-2: Pre-processing quality checks and re-processing cycles for selected products

- An automated quality check is performed on the input. All products with issues and overlaps between inputs are listed in an inventory used during processing.
- Quicklooks of all inputs and if necessary the inputs themselves are visually inspected. Products with issues are removed.
- The quicklooks of Level 3 products are screened again. Issues found here lead to investigation of Level 2 inputs for the respective period, exclusion of the product causing the issue, and re-processing of the Level 3 for the period.

This process is automated up to quicklook generation and operator-driven in the screening and exclusion cycle. Re-processing is automated again.

3.4 Classification processing

The classification system produces global LC databases made of LC maps for three epochs and of LC condition products, using as input the global SR composite time series for the period 1998-2012 generated by the pre-processing.

Requirements addressed by this section are:

- → LC-SR-3340 Reprocessing support
- → LC-SR-4100 Production control
- → LC-SR-4120 Automated classification
- → LC-SR-4130 Classification expert operating
- → LC-SR-2130 Manual failure handling
- → LC-SR-2120 Automated failure handling

The classification process aims at transforming the time series generated by the pre-processing steps into meaningful global LC maps for the climate modellers' point of view.

The classification workflow comprises preparatory steps including format conversion, mosaicking, compositing and stratification, spectral and temporal classification processes with unsupervised and supervised algorithms, and rule based labeling.

In addition to the global land cover maps, the CCI-LC project will deliver four global land cover condition (LC-condition) products: the NDVI, snow, burnt area (BA) and water condition products over the 1998-2012 epoch. The LC-condition workflow comprises preliminary steps (which may include – depending on the product – spatial filtering, format conversion and mosaicking), compositing, combination of products, gap filling or smoothing and temporal aggregation.

Figure 3-3 shows the processing workflow of the LC classification processing chain. Classification comprises the following steps:

- preparatory steps including format conversion, mosaicking, compositing and stratification
- spectral classification
- temporal classification
- classification merging
- auxiliary data integration

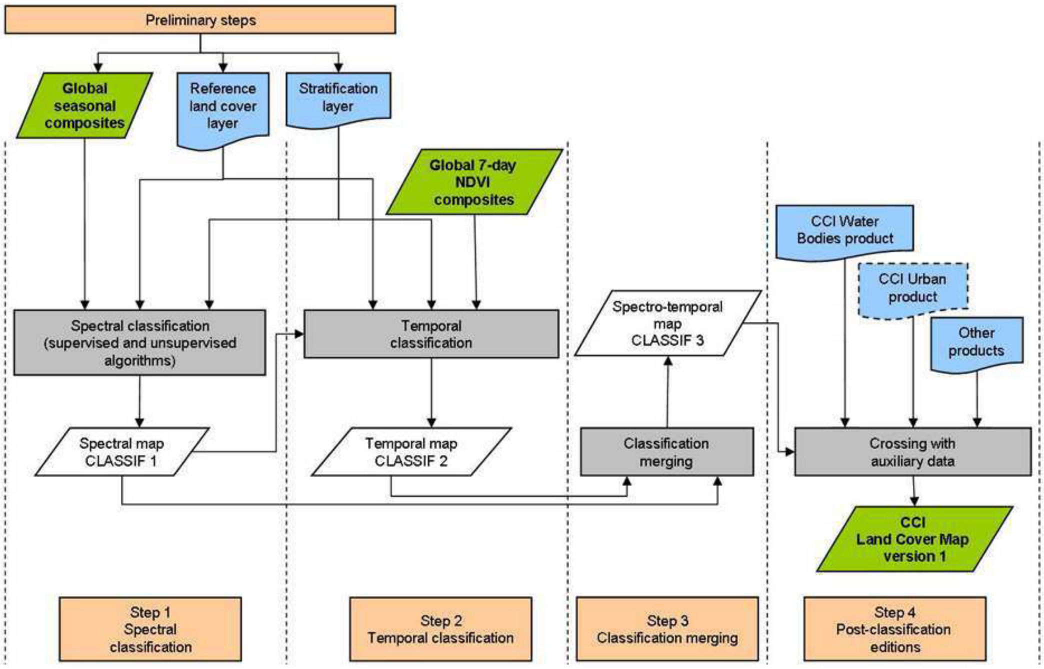


Figure 3-3: LC classification processing chain inherited from prototype

Figure 3-4 shows the processing workflow of the LC-condition processing chain. It comprises the following steps:

- preparatory steps (including spatial filtering for the NDVI only and format conversion and mosaicking for the other three variables)
- NDVI computation (for the NDVI only)
- compositing (for the snow, BA and water products only)
- combination of products
- smoothing (for the NDVI only)
- gap filling for the snow, BA and water products only)
- temporal aggregation.

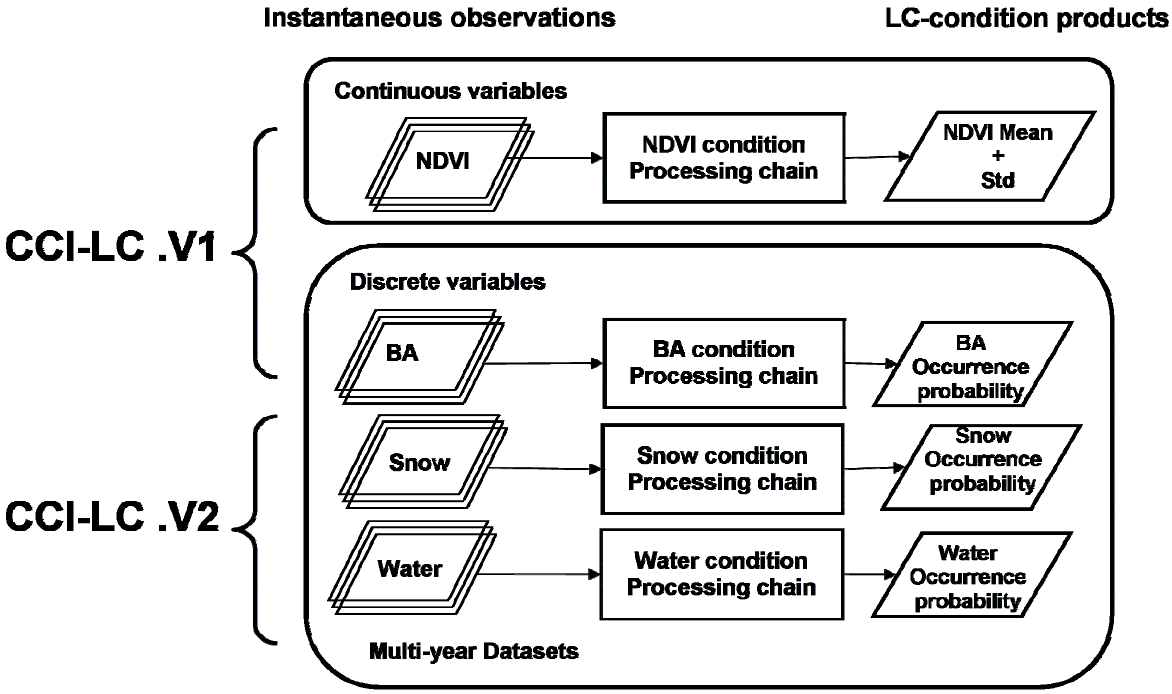


Figure 3-4. LC-condition production chain for continuous (NDVI) and discrete variables (snow, BA and water), inherited from prototype.

Operators re-process subsets for tests and the complete dataset for the generation of a new version of the LC CCI map and condition products.

To run the classification and the LC condition workflow, an operator (like for pre-processing) with support by a classification / LC condition expert

- gets the instructions on temporal coverage, processor versions, processing parameters from the development team
- sets up a bulk processing task with the relevant parameters and starts it
- monitors the bulk processing task and the jobs and processing steps derived from it
- analysis intermediate results and optionally adapts strategies, auxiliary data or parameters of the classification chain
- analyses errors if any and restarts failed jobs
- provides the results to the development team for QA and reports on the production
- re-runs certain parts of the production if necessary to fix data issues
- manages data storage and if necessary stores or deletes output products

To release the map and condition dataset as a new version, an operator

- ensures that the processor software is freezed and processing parameters are version-controlled
- tags the output product with the product version, optionally with a label "to be validated"

The development team asks external validation experts to check it and once validation completed, it announces the new version in the LC CCI web.

3.5 Validation

The validation team is in charge of the LC map validation. The team is in close relationship with a network of remote sensing specialists with recognised expertise over specific regions. These experts access an online validation tool via the LC CCI portal and interpret samples over their regions using high and very high spatial resolution imagery.

Requirements addressed by this section are:

→ LC-SR-4140 Expert-driven validation

External validation experts

- use the online validation tool
- use their knowledge of the region and interpret (i.e. characterize in terms of land cover classes) a sample of random points using high and very high spatial resolution imagery over several epochs

Those samples will be used later by the validation team, as a reference database to perform an independent statistical assessment of the LC maps. The validation team also compares the LC maps with other existing global land cover products.

3.6 Algorithm improvement

The development team in agreement with ESA decides about features or processes to be improved in order to meet user requirements. The development team implements the improvements as new versions of processors and if necessary as modified workflow. The team tests and validates new algorithms. The development team decides about new versions to be released.

Requirements addressed by this section are:

→ LC-SR-3110 Transfer to operations

There is a model behind the LC CCI system of how improvement is achieved. It supports testing of new ideas, rapid prototyping, and immediate feedback for a short loop. Figure 3-5 depicts this model.

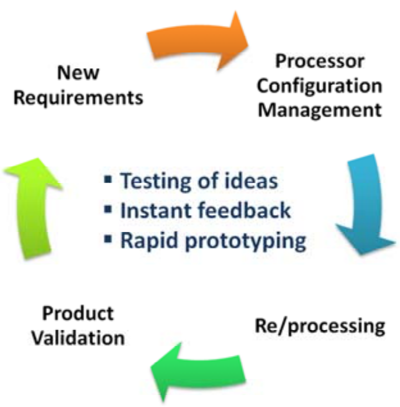


Figure 3-5: Model of continuous improvement with frequent re-processing

This model identifies four activities that contribute to the progress:

- New requirements are identified and analysed.
- This leads to new solutions in form of a modified configuration and new or modified processor implementations.
- Re-processing applies the new configuration. For LC as for other ECVs usually longer time series and global processing is required to assess the difference. Fast re-processing in the development cycle is essential.
- Validation shows the difference to former versions and may reveal new issues that may lead to new or refined requirements. The requirement selection process starts again.

Not every cycle leads to an improvement, and not at all every cycle leads to a new version of the CDR being released. But the availability of frequent reprocessing capabilities allows for immediate feedback on new ideas. To release a new version is a step optionally taken after the validation step.

Functions essential for such an approach are an implementation environment to improve operational processors, version control for software and parameterisations/configurations, structured versioned storage of data products, capabilities to test processing with single examples and with larger datasets, and tools for the inter-comparison and validation of results.

If re-processing in this cycle re-processes the complete dataset in order to assess the output, then the approach for "transfer to operations" is no longer a transfer of software into a different environment. But it is simply to decide and to tag the new output with a label for its release. Else, if the complete dataset has to be re-processed to generate the complete CDR then "transfer to operations" is an optional optimisation of the version-controlled processor software that can immediately be used for re-processing of the complete dataset.

4 Infrastructure trade-off

There are some fundamental decisions that determine how the LC CCI system will look like.

- To what extend shall the system use and build on the prototype?
- Shall the system be implemented in an existing infrastructure of an existing data centre?
- Shall the system run completely virtualised in a cloud?
- Shall a middleware be used, and which one?
- Which functions or subsystems are candidates for sharing with other ECVs?

This section describes the trade-off for these questions. It may not be complete in its alternatives, but they are considered the most relevant for LC CCI.

The questions are not completely independent of each other. The existing infrastructure of a dedicated data centre will exclude the deployment in a cloud. Also the middleware question may be decided already in this case. The following subsections describe for each alternative how it will look like, what has to be done in order to realise it, and the pros and cons.

4.1 Distributed or central LC infrastructure?

Advantage of the distributed system with pre-processing at one site and classification at another is that the corresponding experts have local access to processing and outputs. Advantage of a central LC infrastructure is that it avoids data transfer of the SR products.

Requirements addressed by this section are:

→ LC-SR-4400 Distribution

Figure 4-1 shows the alternative of a distributed system and of a centralised system for LC CCI. The distributed system has pre-processing at one site and classification at another.

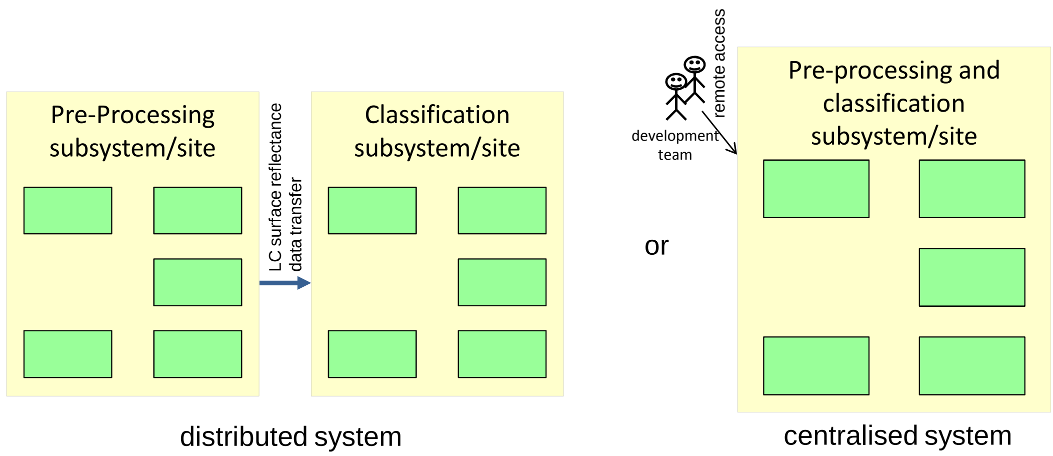


Figure 4-1: Distributed or co-located LC processing infrastructure

4.1.1 Distributed system as enhancement of the prototype

The LC CCI prototype mainly covers development, production and validation functions. Production in the prototype is distributed over two sites. The pre-processing chain runs on a Calvalus cluster at Brockmann Consult GmbH. Pre-processing experts supervise production, analyse results, and support the quality control. The classification chain runs on an infrastructure at UCL. Classification experts supervise production, analyse intermediate results and optionally adapt strategies, auxiliary data or parameters.

The subsystems interact with each other using data interfaces. LC surface reflectance product files are exchanged offline, writing them to disk at the pre-processing site and reading the disks at the classification site. Small amounts of data are exchanged via FTP. Note that besides this data transfer there is also data access by the local experts in a large extend. They need access to the data products for inspection, analysis and quality check.

This constellation is one option for the CCI phase 2 system for LC. As it only covers functions for production it must be supplemented by functions for long term data stewardship and by functions for user services.

To use this for LC CCI phase 2 the following actions are required:

- agreement with the involved operational entities with respect to maintenance, sustainability, conformity and reliability and data and services
- software development for improved operations, automated data exchange, processor versioning, concurrent algorithm development and test
- extension of data volume and processing power for Sentinel data requirements
- either extension by or interfacing to user services with user access to LC CCI data
- either extension by or interfacing to long term archiving facility

4.1.2 New deployment in a central LC infrastructure

In order to make this alternative comparable with the distributed one, only the production and development subsystem is considered. In the central LC infrastructure the processors for pre-processing and for classification are deployed and integrated. The data is available, and data streams for new input data are routed to this system.

The central LC infrastructure is hosted by a service provider. This service provider may offer services on different levels ranging from network and power over computing and storage hardware to an earth observation processing, data management and user service infrastructure. The LC system may have to be adapted if being integrated on one of the higher levels.

The critical point is how this centralised system is kept up-to-date and whether it is the primary development platform for new algorithms. If not, one of the purposes of the system is not fulfilled, and the migration of new processor versions will require additional steps. To make it the development platform, the centralised system provides a virtual private network (VPN) access without barriers for the different contributors of the development team.

To use this for LC CCI phase 2 the following actions are required:

- find and decide about a central LC infrastructure provider
- agreement with the contributing entities
- VPN setup for distributed development team
- software development for improved operations, processor versioning
- integration with service provider infrastructure
- extension for Sentinel data volume and processing power requirements
- either integration of or interfacing to user services with user access to LC CCI data

- either integration of or interfacing to long term archiving facility

4.1.3 Pros and cons

Arguments for the decentralised approach are

- no integration effort as it is already done in the prototype
- short cycle with production close to respective experts for the processing level, processing system is kept up-to-date with minimal effort
- Interactive parts of the classification process can be integrated without remote access.
- data with license restrictions (e.g. SPOT VGT) can remain in the respective institutions, data stream for new data and auxiliary data is already established
- Robust reliable production (in a dedicated data centre) is less important for bulk reprocessing than in an NRT environment because small delays due to disturbances can be recovered with relatively small margin of additional resources.

Arguments for the central approach are

- no data transfer required from site to site for intermediate products
- improvement cycle that comprises the complete processing chain can be faster
- operational experience and available broadband network connection of a data centre can be used, optionally other services can be used
- sharing infrastructure with other ECVs is possible with this configuration
- hosting the system close to the Sentinel data inputs is possible

4.2 Cluster or cloud?

Advantage of the cluster is that no overhead is introduced and the infrastructure can be optimised for the LC re-processing task. Advantage of a cloud solution is that flexible capacity can be rented. The cloud requires that the data is in the cloud, too, which is (prohibitively) expensive for the amounts of EO data.

Requirements addressed by this section are:

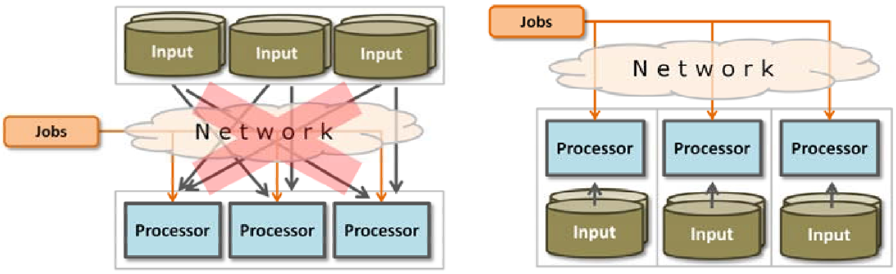
→ LC-SR-3330 Processing Concurrency

→ LC-SR-3335 Scalability

→ LC-SR-3360 Cluster for data-intensive processing

How will the LC CCI system look like and what has to be done to run the LC CCI system on a cluster or in "the cloud"? There are only a few experiences with the cloud approach, but there are practical experiences with different clusters, among them the Calvalus/Hadoop cluster used for the LC CCI prototype and clusters running Grid Engine as batch job scheduler.

One of the key issues of the infrastructures is scalability. For this the Hadoop middleware comes with a data-local processing approach. Figure 4-2 depicts the traditional archive-centric architecture on the left, and a cluster of standard computers on the right with local disks for a file system distributed over the processing nodes.



central data storage
(data are transferred on the network)

data-local processing
(tasks are transferred on the network)
avoids data access bottleneck

Figure 4-2: Network access versus data-local processing

The left architecture the location of data is transparent to the system. Data is provided via the network. In the right architecture the knowledge of where the data is stored is used for scheduling, and data usually is accessed locally within a machine. Grid Engine clusters as well as the cloud follow the approach on the left. Calvalus/Hadoop follows the approach on the right.

4.2.1 Running the LC CCI system on a cluster using Grid Engine

A cluster is a composition of computers, storage and network that can be used together for concurrent processing in a coordinated way. A cluster may be dedicated to a project or it may be shared. Clusters provide:

- computing power, physical or virtual machines, homogeneous or different classes, same operating system
- storage, optionally hierarchical with slower background storage and fast disks, or distributed to the processing nodes
- network, between machines, to storage, optionally with different bandwidths
- front end / master, dedicated machine(s) to control jobs on the cluster
- deployment, a way to install software on the cluster to be used from all machines
- staging, a way to make accessible large data volumes in case they are stored in background storage
- I/O to/from cluster, a way to add input data to the cluster and to retrieve output data, either via a network connection or offline on media
- quota regarding space and processing time

As an example the Edinburgh University ECDF cluster used by Sea Surface Temperature CCI consists of 286 machines with together 2912 CPU cores, high performance disks accessible from all nodes, and NAS background storage, Gigabit connection of nodes with 10 Gigabit backbone, and Infiniband to a subset of nodes.

Oracle Grid Engine, previously Sun Grid Engine, is a batch queuing system for computer clusters. It schedules and manages the execution of large numbers of concurrent jobs. A typical grid engine cluster consists of a master node and execution nodes.

The main command provided by the Grid Engine middleware is qsub. Example job submission:

qsub -l h_rt=24:00:00,sages_1ppn=1 -j y -cwd -o log/amorgos-fr-2009-05-01.out \
-N am_fr_20090501 **bin/amorgos-run.sh fr 2009 05 01**

It submits a job to the cluster. The command line argument of the qsub command will be executed on some node in an environment identical to the one of the qsub execution. The submitted job gets attributes with node capabilities and queue features.

To run the LC CCI system using a Grid Engine cluster the following activities are required:

- write pairs of start and run scripts for each processing step. The start script splits the processing request into qsub calls to be handled on a single host each. The run script is executed on some execution node of the cluster. Both the start and the run script may contain loops to iterate over the inputs. The border between these two iterations can be moved towards more concurrency or towards larger jobs on a single node.
- integrate production workflows for the LC CCI processing chain and the iteration over the input time period in production control
- deploy processor software
- ingest input data

4.2.2 Running the LC CCI system on a cluster using Calvalus/Hadoop

Apache Hadoop is an open-source cluster management software framework for data-intensive distributed computing. It is based on ideas from Google for a distributed file system and the MapReduce programming model. Hadoop is implemented in Java and can be extended with this programming language. There are several contributors to the framework, among them Yahoo. In a short form:

Hadoop = HDFS + scheduler + MapReduce

The Hadoop Distributed File System manages data on local disks of processing nodes, cares for automated replication, is used for automated software deployment to the distributed cache at nodes, and makes accessible all files homogeneously and transparently from all nodes.

Hadoop introduces the concepts of *job*, *task* and *attempt* to support the robust parallel processing of large sets of inputs. A job with a set of inputs is split into tasks for a single input each. A task is executed by a node in an attempt that is automatically repeated on a different node in case of failure. Hadoop provides job queues, different scheduling strategies, priorities, fairness. Scheduling is aware of where the data is stored to schedule tasks to nodes that have (a replica of) the data.

Hadoop MapReduce is a programming framework that allows concurrent aggregation of data. MapReduce supports such jobs by sorting, partitioning and streaming of intermediate data between nodes.

Calvalus is a software system that extends Hadoop by

- processor adapters for Unix executables in any programming language, processor adapters for the ESA BEAM Graph Processing Framework
- EO data processors, all BEAM operators, Level 3 binning and mosaicing algorithms, matchup with in-situ data, and trend analysis
- a data directory structure for inputs and outputs, auxiliary data, processor software bundles, in-situ data, and user areas
- a Hadoop configuration, a job submission client, a portal for job submission
- production monitoring and control for processing workflow management and bulk production

As a formula:

Calvalus = EO data processing with Hadoop

To run LC CCI on Calvalus/Hadoop the following activities are required:

- Use a cluster of standard computers with local disks instead of network storage

- integrate processors, build processor bundles as .tar.gz of processor installations of the different LC CCI processors (or integrate them as BEAM operators)
- integrate the LC CCI Level 3 as MapReduce job
- prepare request templates and configure production control for the LC CCI bulk workflows
- ingest the input data into HDFS

This has been done already for the LC CCI pre-processing prototype.

4.2.3 Running the LC CCI system in a cloud

A cloud infrastructure abstracts from the physical hardware and provides services like storage and virtual machines. Cloud infrastructures are available commercially from different providers (Amazon as an example). They provide:

- computing power in form of virtual machines, selectable classes regarding CPU power and RAM (Amazon EC2)
- storage as virtual disks, mountable on virtual machines (Amazon EBS)
- network storage accessible via the Web (Amazon S3)
- a Hadoop MapReduce implementation (Amazon EMR) based on network storage (S3)

Virtual disks are more expensive (magnitude of 100 Euros per TB and month) than network storage (magnitude of 50 Euros per TB and month) such that network storage will be used for the processing archive. Computing power is either paid on an hourly basis or with an advanced payment and a smaller hourly rate (magnitude of 2500 Euros per year for a 16GB/8 core machine). (All numbers are very rough estimations to understand the relations!)

To run LC CCI in a cloud the following actions are required:

- cost estimation
- set up virtual machines and storage
- optionally decide about middleware
- deploy processor software and install production control, integrate with middleware
- ingest input data into network storage

In order to use the flexibility of a cloud the ready-to-use configuration of a virtual machine template for LC CCI can be stored in network storage and can be launched on demand.

4.2.4 Pros and cons

Arguments for a cluster solution are

- infrastructure is available for frequent reprocessing
- structure can be selected suitable for the relatively large data volumes of EO data processing
- features like the degree of reliability/availability can be adjusted to the offline processing task
- price can be expected to be lower than commercial cloud

Arguments for Grid Engine as cluster middleware are

- low complexity because lower level of service, simple interface with qsub
- simple integration of executables and shell scripts, NFS for software access (no deployment necessary) and data access from all nodes
- scheduling that considers job attributes like CPU and memory demand
- widely used, thus easily shared with other projects

Disadvantages of Grid Engine as middleware are:

- shared storage is always a bottleneck for large EO data sets
- staging required for hierarchical storage

Arguments for Calvalus/Hadoop as cluster middleware are

- uses data-locality for scaleable massive-parallel full-mission processing

- distributed file system HDFS with standard disks on standard machines, automated replication for data security and continued service in case of node or disk failure
- scheduling that considers data-locality, data-local processing avoids network bottleneck
- higher level of service in job scheduling that cares for parallelisation on product sets (or file splits if required), with jobs, tasks, attempts, automated failover and repetition (the Grid Engine job corresponds to a Hadoop task)
- bulk production control
- MapReduce implementation with automated partitioning and streaming, avoids file I/O, supports concurrent distributed aggregation, used for EO level 3 processing, matchup computation, trend analysis
- distributed cache for automated software deployment on nodes

Disadvantages of Calvalus/Hadoop as middleware are:

- HDFS does not fully support a random-access read/write file interface. A local data copy is therefore necessary for NetCDF files which can only be accessed through a dedicated (random-access requiring) library. (Note that this data copy usually is not a copy over the network, but just a constant overhead.)

Arguments for a cloud solution are

- flexible use of processing power possible
- output data can be distributed to users via cloud network storage without copying
- no hardware maintenance

4.3 Sharing with other ECVs

Advantage of sharing is the re-use of input products and the option for the respective use of outputs as auxiliary data or for inter-comparison. Disadvantage of sharing may be that the computing platform may not be available to LC at any time, and that a data policy other than free and open access are more difficult to implement, e.g. to protect third party input data.

Is it more important to keep the ECVs independent of each other or to use synergies? Figure 4-3 shows a proposal for sharing functional parts while leaving others to the specific ECV.

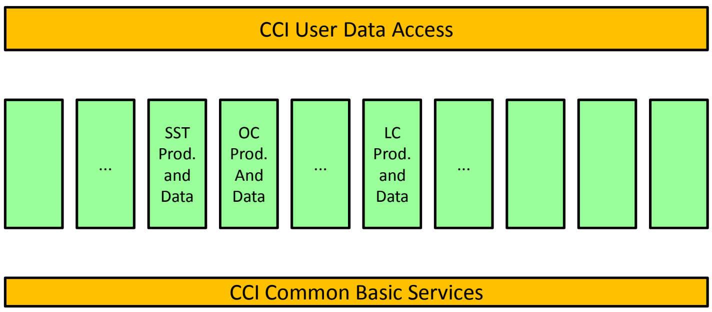


Figure 4-3b: Sharing user services and data stewardship with other ECVs

User services are shared to provide a common harmonised data and information access to (climate) users. In addition, basic services can be shared without introducing much dependency since they can easily be exchanged by different implementations if later independence is an issue. Basic services are e.g. a common backup archive for input data or a software repository offered for use by all ECV systems.

For the production and development subsystem an environment shared by several ECVs, among them LC CCI, provides common resources on different levels usable by all ECVs:

- production storage
- processing hardware (virtualisation)
- processing middleware (cluster)
- production control (multi-ECV production system)

Sharing may be done on different levels:

- If the same inputs are used sharing production storage saves space. In addition every ECV can immediately use the results of ECVs it shares a production storage with. If sharing is on the level of the hardware LC will get virtual machines in such an environment to deploy the LC CCI system.
- If sharing is on the level of the processing middleware then there will be e.g. a Grid Engine instance or a Calvalus/Hadoop instance (see previous subsection) with quotas and fair scheduling for LC and other ECVs. If one ECV does not use its share the resources can be used by other ECVs temporarily. Depending on the policy for scheduling a reprocessing campaign for one ECV may restrict the use of the infrastructure by other ECVs.
- Sharing or not sharing the production subsystem is in one direction obviously related to the decision whether to continue the prototype configuration from section 4.1. To use sharing means to migrate the prototype to the shared platform.

Even if sharing requires too much harmonisation among ECVs there may be common multi-ECV elements provided by CCI that can be configured for LC. They are then completed by LC-specific elements and plug-ins to compose the LC CCI system.

4.3.1 Pros and cons

Arguments for sharing are

- Saves space for inputs used by several ECVs
- Operations and maintenance required only once for the shared functions
- Peak power for LC is higher
- Synergy by immediate availability of outputs and data exchange among ECVs, e.g. aerosol for LC.
- Sharing provides functions to ECVs with less infrastructure or an infrastructure the single ECV does not have in its prototype (user services for LC CCI)

Arguments against sharing are

- Harmonisation is work, some prototypes are already derivates of existing elaborated systems
- Dependencies increase the risk of failure
- User communities and the interaction with users are different for different ECVs. A common user service may be not flexible enough
- Data with license restrictions (e.g. SPOT VGT) could not be shared, as well as licensed software from a local environment.

4.4 Recommendation for LC CCI

There is a weak preference regarding the first trade-off between distributed and centralised pre-processing and classification. The two strong arguments are "close to the experts" (in favour of a decentralised infrastructure) and "avoiding data transfer" (in favour of a centralised infrastructure). If "close to the experts" can be implemented by a central LC infrastructure with low barriers then this may be the best choice, knowing that there could be particular agreement for data with licence restriction.

The Calvalus/Hadoop cluster, combined with a dedicated hardware for classification, is the recommendation for the second trade-off. Reasons are the experience from the prototype and the expectation that it scales well and allows to master the Sentinel data. Cloud storage for dissemination of LC map and condition products may be borrowed from the cloud approach.

There is a preference for an independent LC production and development environment. The strong argument for it is "harmonisation is work". Resources shall better be used for LC development. The strongest argument against it and for a shared infrastructure is the sharing of inputs in the early years of Sentinel where the amount of data in relation to the technological progress is an issue for every ECV.

5 Functional design

5.1 Services for the users

The LC CCI web site is the entry point for external users and validators for information, metadata and data services including online subsetting and aggregation. A managed forum and an issue tracking system is provided for two-way communication.

Requirements addressed by this section are:

- → LC-SR-1220 User relations management
- → LC-SR-1230 User subscription
- → LC-SR-1240 Subsetting, aggregation, re-projection
- → LC-SR-1200 Web access
- → LC-SR-1300 FTP access
- → LC-SR-1310 NetCDF output format
- → LC-SR-1320 GeoTIFF output format
- → LC-SR-4140 Expert-driven validation
- → LC-SR-4142 Long-term validation data collection

User services are the functions and interfaces that an external user perceives and uses to interact with the system, and that the LC CCI project uses to present itself to the users and interact with them. A set of public resources allows users to gather information about the project, the goals, the data provided and algorithms used to produce the results. In addition to this, of course, the resulting dataset has to be made available using access mechanisms so that users can choose their preferred way of obtaining the data.

The same set of web-resources shall allow registered users to easily up- and download data, tools, and documentation. In addition, anonymous and registered users shall have access to information exchange tools like a managed forum, news feeds and other. Users access information and catalogue services through a central web-portal that bundles links to web-resources implemented independently from the central portal.

The LC CCI user services mainly comprise two functional aspects:

- Access to the data generated by the processing system, to matchup datasets and online subsetting and processing resources
- A community interface for general information exchange, document management, user registration and interaction.

5.1.1 Components and interfaces

These two high level functional aspects are implemented using a number of dedicated software packages that implement specific web functionality (Figure 5-1).

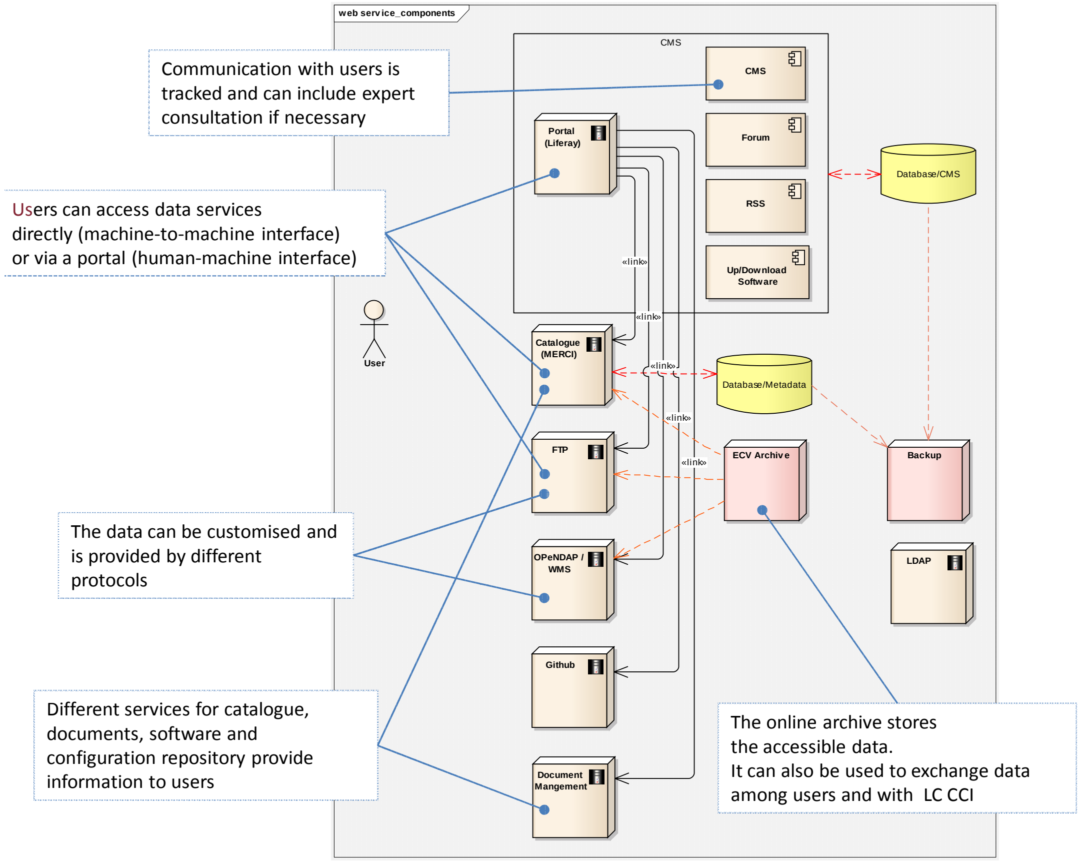


Figure 5-1: Portal and backend services for LC CCI users

A central LC CCI web-portal bundles all the resources available to the users. The portal is preferably implemented using a Content Management System (CMS). Some of the desired functionality is directly available in the CMS portal software (forum etc.) while other components act as stand-alone or even remote web-resources. All of the distributed functional components are connected using links from the central portal.

The other user-accessible services are catalogue, data access via FTP and other protocols including functions for customised data by subsetting, regridding, regional averaging, a document server and access to the software repository to publish a subset of the project documents and algorithm implementations.

For functions that allow write access to resources like contributions to the forum or provision of inter-comparison data it is desirable to identify users. For this purpose users can register in a user management service based on LDAP. It ensures that users have the same credentials for all the services.

5.1.2 Structured output data storage

The online archive is the backbone of all data services of the LC CCI system. Users access the online archive directly via FTP and HTTP. Other data services use the data and customise it according to the user's request. The production subsystem feeds new data and new versions into it for publication.

The online archive is structured by a directory tree where files are organised according to archiving rules which identify each item in this structure. There is exactly one place for each existing or newly created item or file. Knowing the archiving rules a function can identify its inputs and determine and distinguish locations for its outputs.

The directory structure is consequently very closely related to the processing storage structure (section 5.3.2) which has been used for the system prototype ([RD 6], section 3.3). The archiving rules and thus the structure of the tree distinguishes type information, version, and time information. As long as all LC products are global, it is not required to add spatial coverage in the rules.

Besides the different output products, the directory tree also contains external validation data. Additional categories, types and sensors are added as needed.

5.1.3 Data access to ECV products

Based on the online archive, users access data using one of the interfaces provided by the system. Besides plain FTP and HTTP, an OGC Web Map Service serves images of the data, an OPeNDAP server serves subsets, and a catalogue service (see section 5.1.4) allows search and retrieval via metadata.

Depending on the data volume distributed and the available resources, for some of the protocols user access constraints and load-balancing mechanisms are taken into account.

5.1.3.1 FTP

The complete data archive of output products is accessible to the users via FTP. The user has reading access to the archive of output products. The output archive root-directory is a symbolic link to each user's home directory so that all users share the same directory structure.

Depending on the bandwidth available and the download traffic actually observed it may be necessary to introduce access restrictions. These can be realized by introducing user accounts, i.e. connecting the FTP server to the LDAP-based user management system, and by introducing bandwidth restrictions on anonymous access.

5.1.3.2 WMS and OPeNDAP

Access to the data shall also be implemented using protocols that concentrate on the exploitation of the datasets, in contrast to the file-based mechanisms described in the previous subsection. Protocols implemented for this purpose are OGC WMS and OPeNDAP.

Both access protocols are implemented by the open source THREDDS server software. This data server implementation works with the output file format of the LC CCI products directly as it builds on top of the NetCDF library. The THREDDS server can be configured to use the output data archive file structure as catalogue base structure. It follows the suggested structure described in section 5.1.2.

The product metadata is derived from the NetCDF files by an automated archive scanning and parsing mechanism. The archive scanning mechanism also detects new product files or the removal of files from the archive and reflects these changes immediately in the web-interface.

The user can navigate through the archive using the directory structure of the archive files the same way used for browsing through the FTP archive. The THREDDS server software supports the user by offering a simple HTTP based user interface that implements simple directory browser functionality as shown in Figure 5-2. This way, specific files in the archive can be located in a very intuitive and natural way.

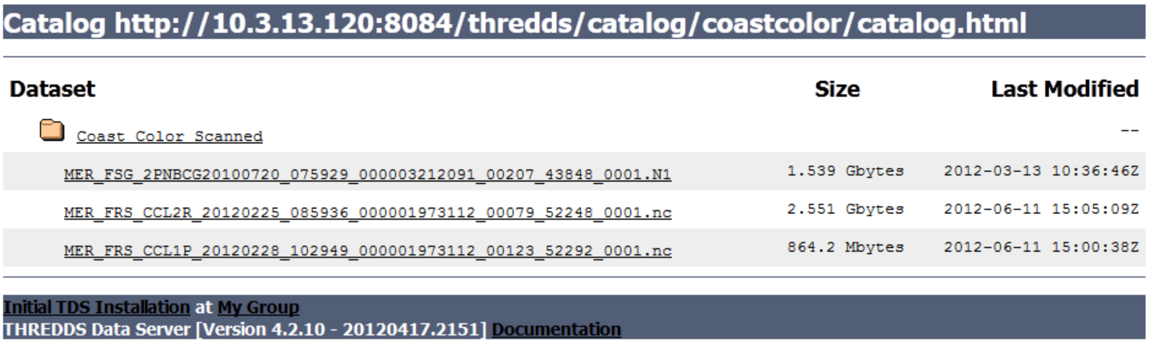


Figure 5-2: OPeNDAP user interface (example)

Having located a file of interest using the graphical web interface the user can access this specific file using the complete set of OPeNDAP operations as defined by the specification ([RD-1]).

The same input path also serves as target for the WMS interface. This interface serves images according to the OGC specification using HTTP as protocol.

As there are no restrictions planned (yet) on the availability of these protocols, no specific login to the services is required. If, during the operations phase it is necessary to restrict the access to these services, this can be implemented by using an Apache HTTP server that bridges the access to the resources. The Apache HTTP server then uses the central user management facilities to grant/deny the access to the OPeNDAP and WMS services.

5.1.4 Online catalogue

The catalogue service is the metadata interface to the LC CCI system with a web-based graphical user interface for search and retrieval. Functions of the catalogue are

- discovery of available datasets
- search using product name, spatio-temporal search criteria, version, quality and statistics search criteria, presentation of results as list, map footprints on a zoomable map, thumbnail images
- inspection of metadata and quicklook images
- collection of products in a shopping basket
- download of collected products via HTTP streaming or as package via FTP
- subsetting of outputs by the search region if the product format supports this

In addition to the access functions the catalogue service registers new products. Product registration comprises input format and quality check, metadata extraction, calculation of statistical parameters, quicklook and thumbnail generation.

The following figures show examples for these functions on the basis of the MERCI catalogue system. MERCI is a basic modular catalogue implementation that covers the catalogue functions required for LC CCI. Figure 5-3 shows the catalogue search interface.

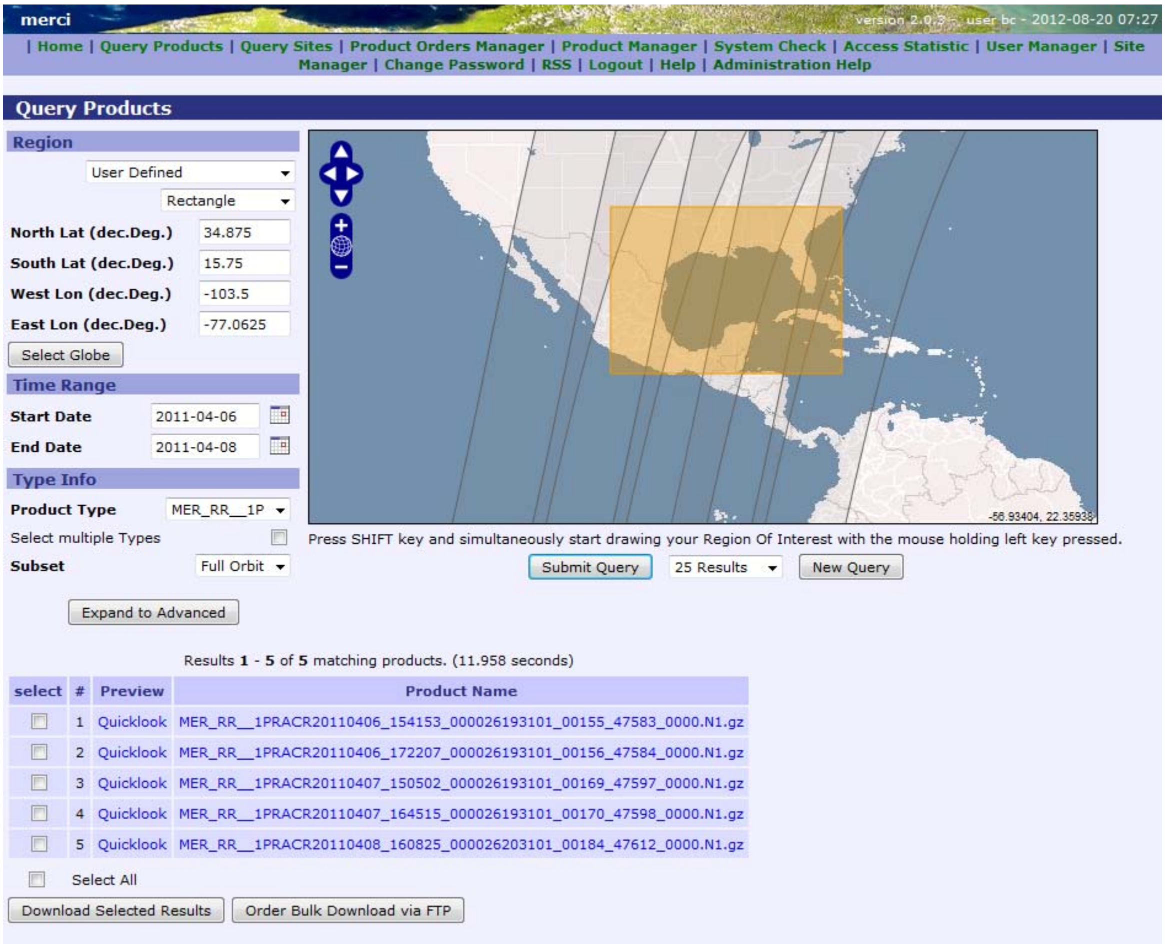


Figure 5-3: MERCI catalogue search interface

Each product registered in the system can be viewed in detail on the product detail page (Figure 5-4). This details web-page lists the metadata of the product, the quicklook image and the geo-boundary of the product on top of a map. The page also gives access to further details like the statistics, the results of the quality checks and a facility that allows generating images of each band of the product (Quicklook–on-the-fly functionality).

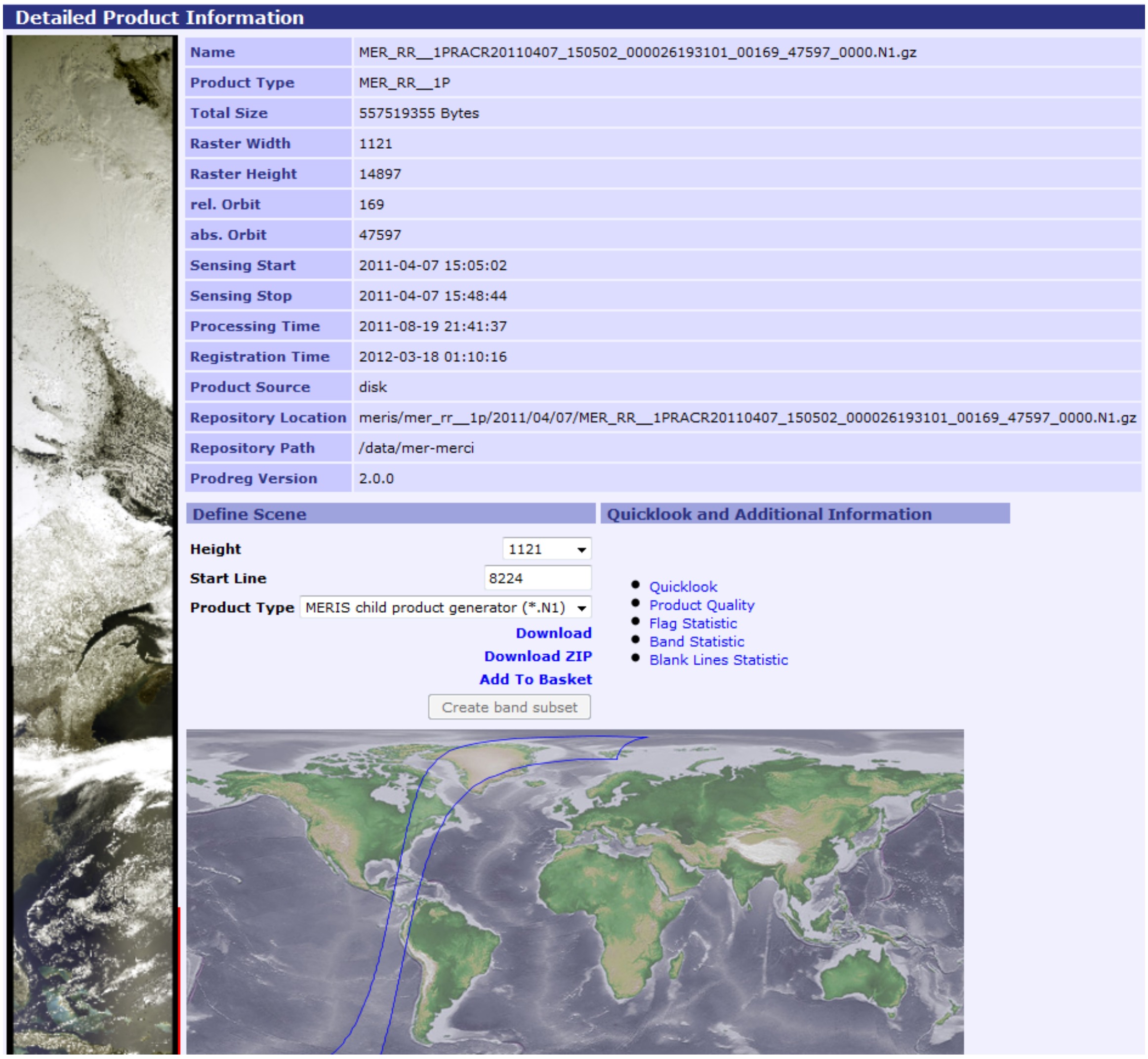


Figure 5-4: MERCI product details page

Products can be further inspected and downloaded using the links provided in search results and details pages.

The product registration function is an application that is applied to new products. Configurations and product-type-specific plug-ins identify the product, generate statistics and quicklooks, and enters the product into the catalogue database.

The catalogue supports product versions. Products are identified by name and version. The name is a logical identifier of the product usually composed of type and time information. Together with the version information it is unique. The catalogue supports search by a configured "current" version per type. Users can select a different version in the search mask.

5.1.5 Web presentation and community interaction

In addition to data access user services provide information on different levels. The web site shall act as a central starting point to explore achievements, data and other resources available to the users. It shall provide basic information for users not familiar with the project, in depth access to resources for the users of the data with static and dynamic content, and finally access to administrative pages and tools for the system operators.

The approach for the LC CCI system is to configure a Content Management System (CMS) as information service front end. The CMS proposed for LC CCI is the open-source solution Liferay. Liferay is used successfully within ESA, e.g. for the CalValPortal (http://calvalportal.ceos.org). Liferay provides

- Separation of content and layout, corporate web site layout using CSS
- Support for authoring by separated creation and publishing, dedicated approval step
- Management of links independent from web pages, links to services and data access
- User rights management, LDAP interface (see user management below)
- Document database
- News feeds
- Forum

Functionality not provided by the CMS shall be realised using separate, dedicated tools that are integrated into the CMS. These are especially the issue tracking and the extended document management functionality.

The basic functionality of any CMS is to separate the design of the web-site from the content. This allows keeping a consistent look-and-feel that is automatically applied to all new pages added to the web-site. In Liferay, the page design can be customised, using a specific set of CSS that is integrated into the system and applied to all web-content belonging to a specific domain.

The content creation is supported from any remote computer for any user that has the necessary access rights. The CMS separates the editing and formatting process from the publishing. This simplifies the editing process as the new web-content can be created in its final external form (style, layout, color-scheme), reviewed by any other administrative user and finally published.

Beyond this functionality, Liferay includes a large number of web-applications out of the box. For LC CCI this provides a toolbox of building blocks that allow to create a modern web-site.

The standard Liferay distribution contains a full fledge forum functionality. It is successfully used e.g. to implement the ESA-BEAM user forum (http://www.brockmann-consult.de/cms/web/beam/forum) and the CalValPortal discussion forum (http://calvalportal.ceos.org/cvp/web/guest/forum).

Questions and issues discussed in a public forum help to spread the information to the community. The forum as a web resource allows users to follow and participate to discussions about specific issues or questions. This helps in understanding in more depth the evolution of an idea, a solution of a problem. To operate a forum that is valuable to the users, it is required that the forum is managed by a skilled administrator that can both directly answer basic questions and get in contact with specialists to request answers to complex problems. A skilled administrator is the key to an informative and valuable forum.

In addition to the forum, the web-portal also includes an e-mail based access mechanism to the administrators. This opens a more private feedback channel, in contrast to the forum where every comment or question posted is per definition publicly visible.

Liferay includes a RSS implementation that supports ATOM 1.0, RSS 1.0 and RSS 2.0 protocols. Any announcement published in Liferay can be fed to a RSS channel. Despite this standard functionality, RSS channels can also be attached to Message Board Threads, Blog

Entries and Wiki Pages. The user is offered a pre-defined RSS subscription button at several places in the portal.

Issue tracking

To track the interactions with a user that raised a problem, problems with products or software issues, the web-portal contains an issue-tracking module. This approach ensures that issues and follow-up actions are tracked and assigned to a developer.

For each issue entered into the system, the issue tracker records at least

- A problem area
- Title and description
- Submitter and date of entry
- Responsible person and status information
- A history of comments

A system of scaled user rights allows editing, creation and deletion of all issues or restrictions on specific problem areas or sub-projects. The reading access to the issue tracking system shall be open to the public without any restrictions. This approach creates confidence and clarity about the LC CCI project and software status.

For the use within the LC CCI web portal, it is proposed to use Atlassian Jira as software implementing the functionality required. Although this is commercial software, Atlassian provides free software licenses for Open Source Projects. Jira fits into the context of the web portal as it allows customisation to implement a corporate design and it supports LDAP user authorisation.

As an example, Figure 5-5 shows the Atlassian Jira issue tracker for the BEAM toolbox.

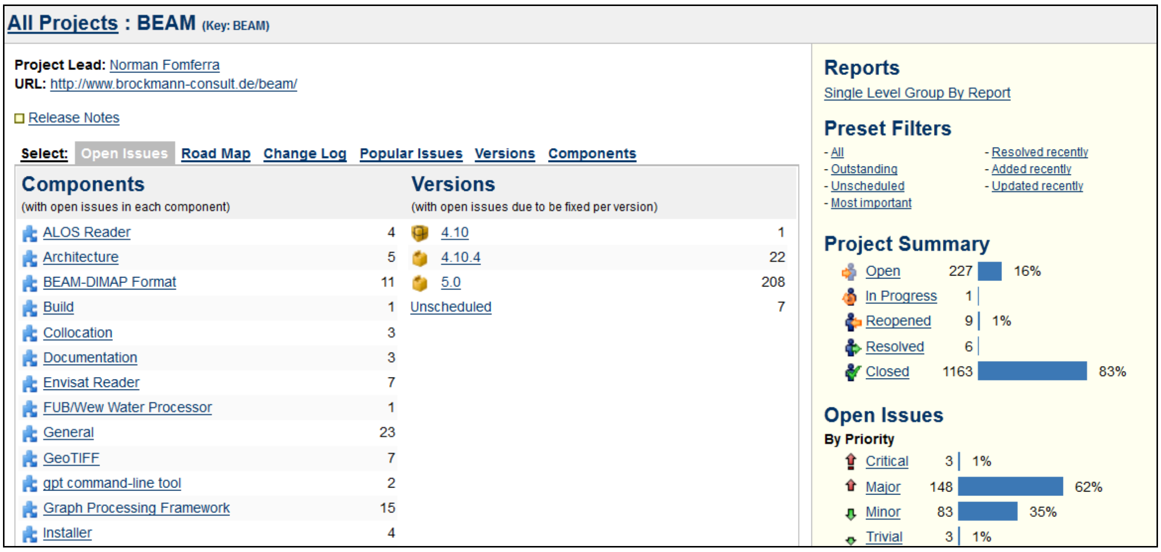


Figure 5-5: Atlassian Jira issue tracking system for the BEAM toolbox

Community interaction

Despite the responsibility for interaction with the users based on the services available in the web-representation, the development team is responsible for a community process.

The development team contributes actively to the international scientific dialogue about LC variables, initiating and contributing to discussions about

- accuracy of retrieval
- strengths and weaknesses of algorithms
- calibration and validation methods
- product formats and metadata
- exploitation of the ECV dataset

For this purpose the LC CCI user services provide a platform to promote the use of the LC variables, to announce updated datasets and to obtain feedback on limitations or possible or required improvements.

5.1.6 Access to tools, documentation and algorithm implementations

In addition to data and metadata the LC CCI project provides tools, documentation and processor implementation source code to interested users. Tools are software that can be downloaded and used locally to work with the LC CCI data. Documentation intended for users are the descriptions of the data (Product User Guide, validation reports including degree of compliance with URD) and of the algorithms (Algorithm Theoretical Basis Document). The Product User Guide also includes sample code in different programming languages of how to read LC CCI data.

Access to processor implementations for some algorithms, in particular those developed within the LC project, is provided with the open source idea in mind. Well documented public code can be reviewed, and that the operational code is available for possible improvements. This avoids the disadvantages of the previous ESA approach for IPF development with the time-consuming and inflexible detour via Detailed Processing Model document, and full re-implementation.

There are three ways how this is made available:

- via CMS by offering final versions of the documents and of tools for download
- in an optional document management system for the exchange, collaborative authoring and versioning of documents
- in a software repository with read access to processor and tool implementations that are open source

The tools that may be provided are :

- Re-gridding tool to extract data on different grids
- Visualisation tool
- Data reading and subsetting tools
- Data inter-comparison tools
- Regional and temporal averaging tool

These tools are offered as desktop applications suitable to execute on all major operation systems. A platform-specific standard installer software installs the tools at the user's computer.

Document management system

The optional document management system (DMS) supports collaborative editing and exchange of intermediate document versions for the development and operations team. This system shall allow a structured view to a complete document tree, with all the version history included. The DMS is integrated with the CMS and with LDAP user management. Candidates that allow for this are KnowledgeTree (https://www.knowledgetree.com/) or other DMS solutions.

The alternative to a DMS is a directory tree with versioned documents like the one for the data products and a rule how to add or update documents and issue new versions. What is missing in such an approach is document metadata and efficient cross-document search capabilities.

Processor software repository

The software of the LC CCI processing system and the processing algorithm code are under configuration control. Section 5.4.2 below describes the project-internal version control approach. Part of the version control system with open source processor and tools implementations are publicly readable. This enables confidence in the final data products as each step of the processing chain can be reproduced by any user. Additional to the raw source code, the repository should also contain the information required to build the software from the source-code.

The software repository contains the actual processing code and all prior versions. Whether the current development branch is made accessible is decided by the development team. All of these versions can be accessed using a simple mechanisms to chose a tag and download the source-code of the selected version.

The write access to the processor repository is restricted to the development team including or extended by the algorithm developers (if external), and the operations put configurations under version management. As all software changes are updated directly in the repository, the software changes are published almost immediately and are made available for review in a short time.

The version control system proposed for the code is git. There are public code repositories like GitHub that host open and closed projects.

5.1.7 User management

User management of the LC CCI system is a service that stores information about users and their roles. Roles are related to access rights. This is useful even though the data policy of CCI is free and open access:

- to restrict administrative access to web resources, e.g. to the CMS
- to distribute the available bandwidth in a fair way to concurrent users
- to identify an author of a contribution in the forum
- to provide a drop-down area to a provider of data

Roles

The role-based access control (RBAC) of the LC CCI system is based on user roles with different responsibilities and use cases:

- Climate users access the complete dataset and updated versions of it. They are one of the main user groups. Feedback is expected from them.
- LC science community users are also interested in parts of or the complete dataset. They may be interested in the algorithms, inter-comparison with other sensors, matchups for validation. Feedback is expected from them, too.
- The interested public may be interested in the goals and achievements of LC CCI, example data products, visualisation, explanation.
- The development team comprises algorithm and system developers and validators. This role has intersecting interests with the science community, but has the mandate to push forward the system. This team uses the system for the algorithm

improvement cycle. It is responsible for the data being generated, its quality, the scientific and technical documentation, and the web content.

- The operations team comprises system operators that are responsible to maintain the system and to run production and reprocessing. The user forum is managed by a member of either the development team or the operations team. Questions are delegated to the experts from either of the teams.

There are several tasks related to user services and web resource that make use of user management. These tasks include

- User management itself
- Web content creation and update
- Email support
- Management of the forum
- Update of software, tools and data provided by the user services
- Issue tracking
- Delegation of in-depth questions to the appropriate specialists

While the system supports named individual users also anonymous access shall be possible. Technically, 'anonymous' is a common external user that can be used by anyone to access the system and to communicate. The only restrictions are that if e.g. bandwidth restrictions apply all anonymous users will share a contingent. In case of many concurrent anonymous users named users have an advantage.

Registry

To allow a convenient user experience with the web-interfaces exposed by the LC CCI system, each user shall be able to use the same access credentials to log in to any of the services. This functionality is implemented using an LDAP authentication mechanism.

A central LDAP server hosts all user data required by the several system components i.e.

- user name
- password
- user role(s)
- optionally a way to contact the user, on the user's choice

Figure 5-6 shows how different front end user services access the LDAP server for authentication.

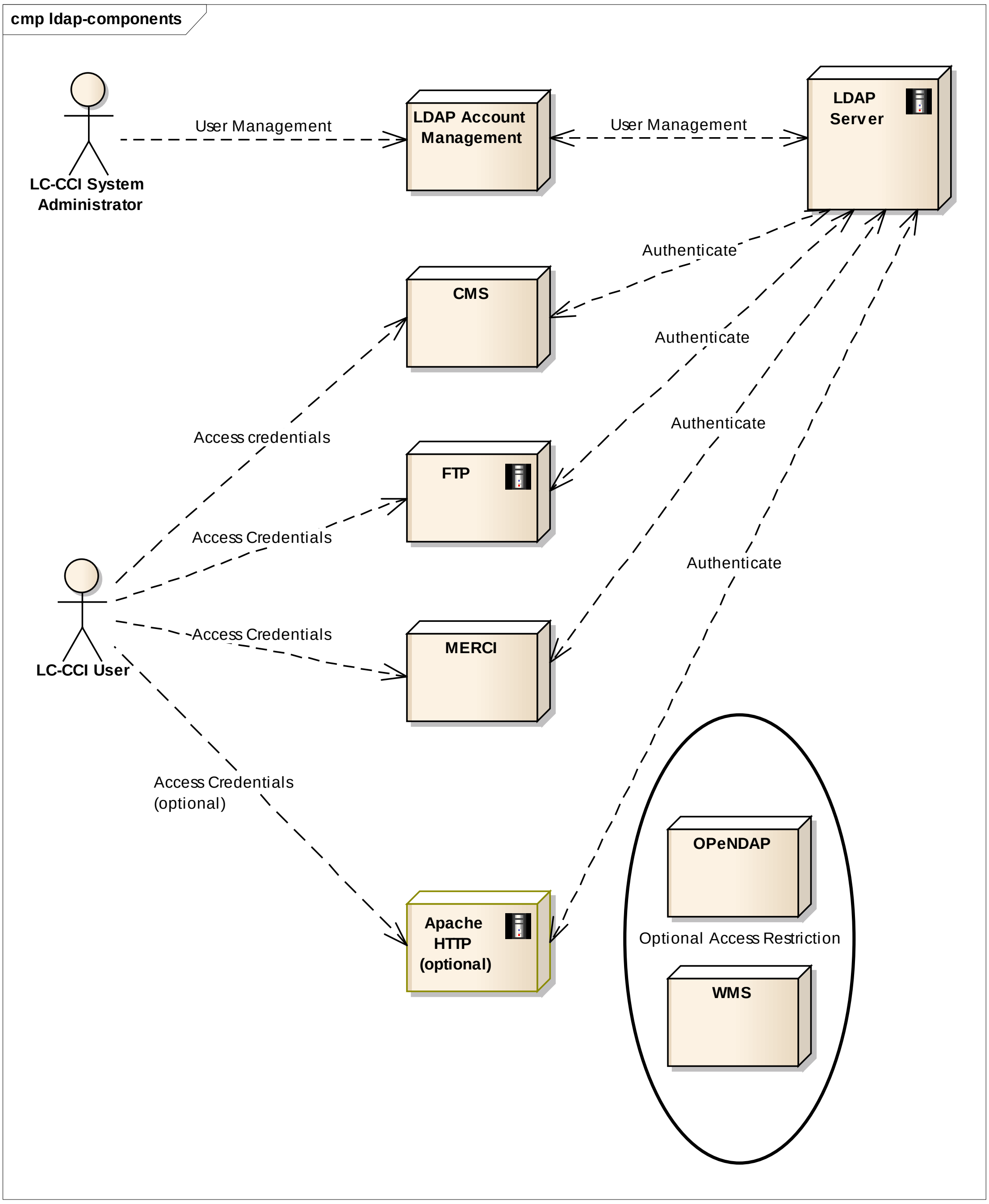


Figure 5-6: Central user management for web-resources

In addition, the resources that should initially be accessible without restrictions (WMS, OPeNDAP) can optionally be integrated into the centralized user access rights system. In the case of extensive use of these resources, the web access can be piped through an Apache HTTP server that forwards the incoming request to the WMS/OPeNDAP implementing resources. The Apache HTTP server also connects to the LDAP server and verifies the user credentials and grants access.

The user management administration functions are realized using a LDAP Account Management utility for the operations team. This functionality will be provided using a GUI based account management systems for LDAP, e.g. LDAP Account Manager (http://www.ldap-account-manager.org/) or Webmin (http://www.webmin.com/).

5.2 LC CCI data processors

The LC CCI data processors are developed from the processors of the prototype. For pre-processing they comprise AMORGOS, several operators implemented in the BEAM GPF framework, and mosaicing aggregation. For classification LC CCI provides processors for unsupervised and supervised spectral classification, temporal classification, merging, and post-classification editing tools.

Requirements addressed by this section are:

- → LC-SR-4200 LC processor integration
- → LC-SR-4210 AMORGOS processor
- → LC-SR-4215 SPOT-VGT geocoding improvement
- → LC-SR-4220 LC L2 Processor
- → LC-SR-4230 LC L3 Processor
- → LC-SR-4235 LC Merging Processor
- → LC-SR-4310 Stratification processor
- → LC-SR-4315 Compositing processor
- → LC-SR-4320 Spectral classification processor
- → LC-SR-4330 Temporal classification processor
- → LC-SR-4340 Classification merging processor
- → LC-SR-4350 Manual editing tool
- → LC-SR-4360 Land cover condition processor
- → LC-SR-4370 Land cover map and condition integration processor

The LC CCI data processors are developed from the processors of the prototype. For pre-processing they comprise AMORGOS, several operators implemented in the BEAM GPF framework, and mosaicing aggregation. For classification LC CCI provides processors for unsupervised and supervised spectral classification, temporal classification, merging, and post-classification editing tools.

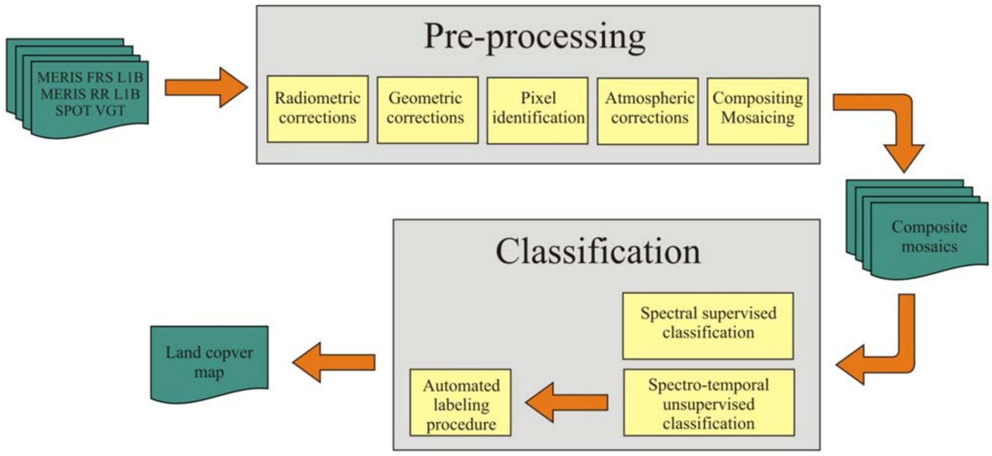


Figure 5-7: Components of the production and development subsystem involved in the LC CCI processing workflow (taken from [RD 3]*). A more detailed breakdown is shown in Figure 3-1 and Figure 3-2.*

These processors together build the production and development subsystem (Figure 5-7). Their role and functionality in the processing chain is summarized below in this section. The underlying algorithms of these processors are described in detail in [RD 5], their output is described in [AD 5].

5.2.1 Processors in Preprocessing Subsystem

Table 5-1: Processors in Preprocessing Subsystem

Processor	Role and Functionality
Geometric Correction	applies the AMORGOS processor for improved geolocation accuracy
Radiometric Correction	corrects MERIS 2nd reprocessing products to 3rd reprocessing radiometric quality
TOA Reflectance Processing	converts input TOA radiances to reflectances
Land-Water Delineation	introduces an improved map of land/water distribution
Pixel Identification	classifies pixels by distinct features
Cloud and Snow Screening	improves cloud/snow pixel distinction
SDR Retrieval	provides surface directional reflectances from TOA reflectances by an inversion algorithm
SDR Projection and Mosaicing	reprojects SDR data onto a Plate Carrée grid, mosaics tile products to a global map

5.2.2 Processors in Classification Subsystem

Table 5-2: Processors in Classification Subsystem

Processor	Role
Format conversion and mosaicking	do format conversion of L3 products, as well as tiles mosaicking
Compositing	generate composites over specific periods, using L3 products from a single or several years
Stratification	prepare the stratification layer and split the global dataset into separate strata to guide the classification processing
Spectral Classification	apply supervised and unsupervised classification, using the spectral properties of seasonal composites as input
Temporal Classification	apply a complementary classification algorithm which makes benefit from the temporal content of the 7-day NDVI time series
Classification Merging	merge spectral and temporal land cover maps
Post-classification Editions	apply various post-classification editions, i.e. adding the CCI Water Bodies product
Land cover condition processor	compile, for each variable, an average profile (discrete or continuous values) over a multi-year period starting from annual time series
Land cover map and condition integration processor	integrate the LC maps and condition products in characterizing the consistency between them

5.3 Data management and processing management

The components for data management and processing management conceptually comprise an archive and an inventory, a production control component, and the LC CCI data processors. Bulk production requires automated workflows and bulk requests with status and reporting to the operators. Workflows include product quality control steps.

Requirements addressed by this section are:

- → LC-SR-3340 Reprocessing support
- → LC-SR-3350 Processing granularity
- → LC-SR-4150 Operator tools
- → LC-SR-4110 Automated pre-processing
- → LC-SR-4120 Automated classification
- → LC-SR-1130 Level 1 input

→ LC-SR-1120 Input ingestion
→ LC-SR-3200 Reprocessing performance
→ LC-SR-3360 Cluster for data-intensive processing
→ LC-SR-3370 Map-reduce for level 3 binning and aggregation
→ LC-SR-2130 Manual failure handling
→ LC-SR-2120 Automated failure handling
→ LC-SR-2110 Processing quality control
→ LC-SR-2100 Quality of outputs
→ LC-SR-4144 Automated validation
→ LC-SR-4146 Validation datasets
→ LC-SR-1210 Data inventory
→ LC-SR-2200 Structured data storage
→ LC-SR-4410 Internal data exchange

This section defines the structures and functions that implement the operational systematic production and reprocessing in the LC CCI system. Focus of this is consistency, completeness, traceability, and efficiency of the processes.

5.3.1 Components and interfaces

The decomposition shown in Figure 5-8 results from further breaking down the production and development subsystem.

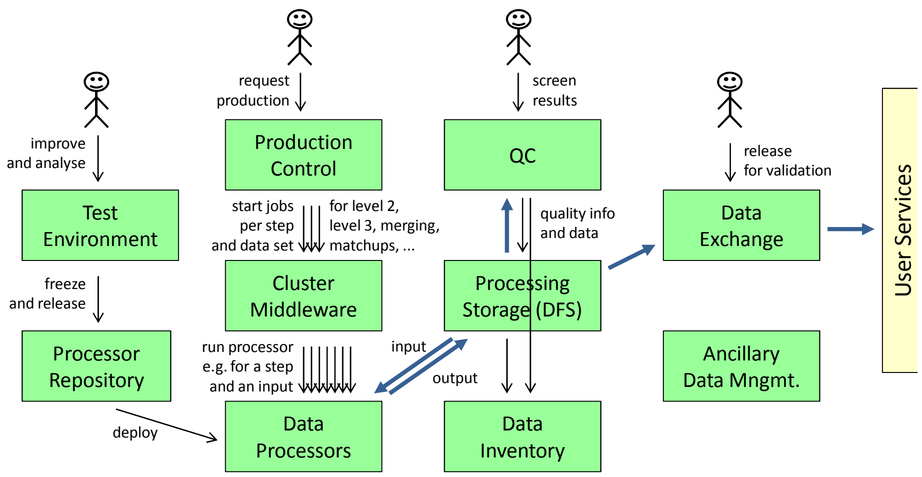


Figure 5-8: Components of the production and development subsystem involved in a workflow for LC CCI processing

Table 5-3 describes the components of the production and development subsystem by their name, their purpose and function, the local data stored and managed, and the implementation approach.

Table 5-3: Components of the production and development subsystem

Component	Purpose and Function	Data	Implementation
Data processors	Generates level 2, level 3 and level 4 LC CCI products	Auxiliary data	Extensions of existing processors
Processing storage	Stores input data products, intermediates and outputs as well as auxiliary data, validation data and processor software bundles in a structured directory tree, makes them available to processors	Data product files directory tree	HDFS file system managing local disks of nodes for Calvalus/Hadoop, (NFS file system and centralised fast storage for Grid Engine)
Cluster middleware	Handles processing jobs and tasks (Calvalus/Hadoop) or jobs (Grid Engine), uses configuration and plug-ins to generate tasks and to call processors	job queue status of each job	Hadoop job tracker (or Grid Engine job control)
Production control	Handles production requests, manages workflows, manages resources processing capacity and storage space	Workflow definitions request queue status of requests	Calvalus production monitor
Data inventory	Handles product entries and collections, attributes of products like QA information, extensional collections (lists) of product entries and intentional collections (logical selection criteria like type and time)	Product entries collections	HTTP REST service, PostgreSQL database, PostGIS
Ancillary data management	Systematically ingests auxiliary data from external sources, stores aux data in processing storage, triggers production waiting for consolidated aux data, implements strategies of auxiliary data selection for processors (temporal coverage, proximity)	Aux data in processing storage Ingestion configuration Trigger rules Aux data selection rules	Combination of data exchange modules, processing storage, processor wrapper plug-ins
Quality check	Checks product integrity and content with specialised data	QA working area in	Data product readers, tests for

Component	Purpose and Function	Data	Implementation
	processors (with quality flags/report as output), adds quality attributes to inventory entries, computes data de-duplication in case of overlapping inputs, generates quicklooks, provides tools for systematic visual screening and for product inspection	processing storage inventory attributes of product entries	missing data and for geo-coding issues, ESA BEAM framework
Data exchange	Systematically ingests data from different sources, transfers outputs on release of a version or systematically, re-formats data if required, registers inputs in inventory, triggers QC and production	Ingestion configuration, transfer configuration trigger rules	Ingestion modules for different protocols
Test environment	Provides sandboxes with full (read) access to input data, processor installations, tools installed, deployment tool and request client to run bulk tests via production control, and local storage for outputs	Software installations	VirtualBox or VMware virtual machines
Processor repository	Stores code of processor implementations, versioning with branches, authorship, simplified user interface to store and to retrieve versions, supports automated deployment of processor bundles	Code repository	GitHub with public repository, git software package, wrapper tools for simplified access

The components of the production subsystem interact with each other, with other subsystems and with system users and operators. For this the components provide and use interfaces. All of the components have interfaces, but some of the interfaces characterise the LC CCI system most. Table 5-4 describes the interfaces by the exchanged items, their content and format, and the protocol used.

Table 5-4: Main interfaces of the production and development subsystem

Interface	Interface items content and format	Data exchange protocol
processor call interface(s)	Calls with parameters (command line), commands (interrupt) for control input data as files or streams working directory	Executables, Unix process, inter-process communication, wrapper scripts, working directory,

Interface	Interface items content and format	Data exchange protocol
	output data as files or streams progress and error messages for monitoring, return code, result specification (output product file names) log files	file interface for inputs, outputs, parameters, reports Plug-in software libraries, Unix threads, ESA BEAM Graph Processing Framework, function interface, in-memory operator chaining, tile cache, product readers and writers, can be extended
distributed file system interface	File system of directories and files, accessible from all nodes of the cluster File objects and data streams for Hadoop Unix file descriptors and file handles for Grid Engine	HDFS protocol for Hadoop, hadoop client for file system functions (mkdir, cd, ls, put, get, rm, cp), Java interface for data streams NFS for Grid Engine, Unix commands (mkdir, cd, ls, put, get, rm, cp), Unix file open method
repository and bundle deployment interface	Version lists, version specification Processor source code tree, source packages Processor bundle for deployment including software libraries, bundle descriptor Auxiliary data packages associated with processors	git protocol for software repository command line and web client to upload and manage processor versions sftp (and hdfs for Hadoop) for bundle deployment
request submission and monitoring interface	Production requests in text file representation (OGC WPS XML) or as HTTP request/response HTTP URLs for status retrieval Web pages with lists of production jobs and their status, control functions	Human-machine interface: Web pages Request submission, monitoring and control command line client Machine to machine interface: (RESTful) HTTP service, OGC WPS
inventory interface (within production subsystem, used by production service and by processors in	Queries (for collections, with constraints on attributes (QA) and time) Collection lists Product entries with metadata	Human-machine interface: Web pages Machine-to-machine: (RESTful) HTTP service, command line client

Interface	Interface items content and format	Data exchange protocol
wrapper scripts)		
data import and export interface	Retrieval commands Product files Transfer commands Delivery packages	sftp, scp, ftp, http media (disks, tapes)

Other interfaces exist that are used internally, e.g. the one used by the data exchange component to notify production control about new products, or the one between production control and cluster middleware to start and monitor processing jobs.

5.3.2 Structured processing storage

The structure definition for the processing storage file system is the basis for all processing functions. The processing file system hosts earth observation inputs, outputs, ancillary data, reference data, and also processor software bundles.

The hierarchy, versioning, and naming schema - called archiving rules - identifies each item in this structure. There is exactly one place for each existing or newly created item or file. Knowing the archiving rules a function can identify its inputs and determine and distinguish locations for its outputs.

A suitable directory structure which has been used for the system prototype is described in detail in [RD 6], section 3.3. However, the structure proposed there is not necessarily the final one, and concrete implementations may deviate from it, especially by introducing more layers or different orderings of the directory hierarchy. The principles of archiving rules and how functions use them remain the same.

5.3.3 Input ingestion

This function is required to initially stage satellite input data and ancillary data from the long-term archive or an external source to the processing storage, and also to systematically retrieve and ingest newly acquired data and corresponding ancillary data and in-situ reference data from an external provider for the continuous extension of the CDR. Figure 5-9 shows the function and the elements involved.

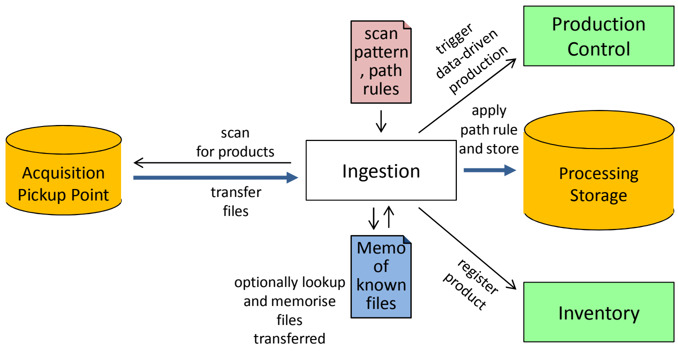


Figure 5-9: Ingestion function to pull input data from a pickup point of a data provider

- The ingestion function accesses a local or remote pickup point and scans it for new files.
- A local pickup point can be used to scan media for ingestion.
- The scan descends a directory tree and uses filters to detect data products.
- The function repeatedly scans the pickup point using a memory of known files to detect new ones for data-driven processing. Errors are handled by retry and reporting.
- The function uses path rules and version information to store the products in the processing storage directory structure.
- The function initiates registration of the product in the inventory.
- The function may trigger data-driven production of the newly ingested product. Also quality checks for new products are initiated this way. Optionally, input data is reformatted.

Implementation hints:

- For Calvalus/Hadoop with distributed local storage ingestion uses the HDFS interface to store files in the processing storage. The interface controls replication and block size of the files.

The ingestion function shall be implemented by the Data Exchange component.

5.3.4 Workflow and production control

The overall LC workflow has been described in more detail in chapter 3 of this document. The corresponding processing chain shall be set up by single processors being developed from the prototype processors which are described in section 5.2.

The production control is a function that initiates and controls data processing activities of the system. The approach for production control is to use some generic cluster middleware but to augment it with the

- handling of processing workflows
- bulk production and managed data-driven processing
- constraint handling like the availability of auxiliary data to be waited for
- integration of manual activities like quality checks
- resource management

The production control function shall be implemented by services that the operator interacts with and that allows for adaptations of workflows.

5.3.5 Data product quality checks

The quality of the LC output products obviously depends on the quality of the input data. Because the LC workflow contains an aggregation of inputs a single corrupted input that remains undetected can compromise a daily composite. The quality check function supports automated and operator-performed quality checks and the integration of their results into the LC workflow.

- First, all input products are screened automatically for product consistency (format, file size, geo-coding, data content). Optionally, all input products are converted to quicklooks for visual screening. Corrupted products are marked in the inventory and they are removed from processing storage.

- Second, the L2 and L3 products are optionally screened before they are used for L3 or L4 aggregation. Again, corrupted products are removed from processing storage and from the set of inputs for L3 and L4 processing in the production workflow.
- Third, the L3 and L4 output products are quality checked and validated before release.

For data-driven short-delay processing quality checks can be postponed to after generation of the products. In case QC detects issues products are retracted, reprocessed, and replaced if necessary.

The collaboration diagram in Figure 5-10 finally shows how Quality Check is integrated into its context by data and control flows.

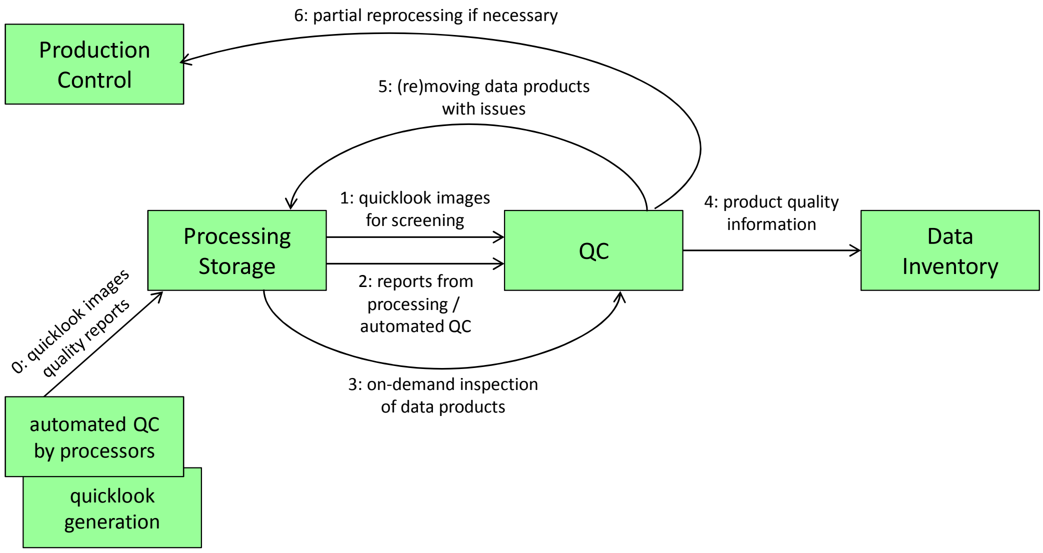


Figure 5-10: Quality check and its interfaces

- QC retrieves new quicklook images and quality reports from automated QC processors via processing storage.
- Operators inspect data products from processing storage if necessary. Product quality information is updated in the inventory.
- In case of quality issues partial reprocessing may be triggered or the products with quality issues are removed from the collection.

5.4 Concepts for continuous improvement

Version control, defined interfaces and flexible adapters allow for reliable exchange of algorithms, test, and transfer to operations with low barriers.

Requirements addressed by this section are:

→ LC-SR-2300 Processor Versioning

→ LC-SR-0200 Stability of outputs

→ LC-SR-0300 Reactive to improvements

→ LC-SR-3110 Transfer to operations

→ LC-SR-2400 Configuration control

→ LC-SR-2405 Modularity

→ LC-SR-3100 Continuous development

→ LC-SR-4142 Long-term validation data collection

This section defines the structures and functions that extend the production and reprocessing environment for continuous improvement. Focus is on flexibility, rapid testing and prototyping, and an interface to full mission data and reprocessing capabilities also for test and validation. The concepts described are processors, versioning, and a test environment. The concept of processors and versioning contribute to the modularity of the system. Within LC CCI they are most relevant for algorithm developers and validators.

5.4.1 Processor interfaces

Data processors are one of the means for modularisation in earth observation processing infrastructures. A processor is a software component that can be parameterised and that generates usually one higher level output product of a certain type from one or several input products of certain types in one call of the processor.

The LC DPM [RD 5] defines distinct processors for preprocessing and classification. They come with individual interfaces. As it is one of the goals of the project to simplify the transfer to operations of science code, a strict standardisation of processor interfaces like the ESA IPF interface is not required in preference for the approach to wrap the existing processors into adapters. For this to work processors have to provide as a minimum:

- parameterisation - parameters to specify inputs and other parameters
- data access - read access to inputs and auxiliary data, write access to outputs and optional runtime space (working directory),
- feedback - intermediate status, success or failure, identification of the result(s)
- packaging and versioning - structured package of software, identified by name and version

Depending on the processing middleware used the processor wrapper is an adapter between the middleware task/job interface and the respective processor.

- In a Grid Engine environment 'start' and 'run' scripts care for job generation and job execution with processor calls. Processors are executables installed in the shared network file system.
- In a Calvalus/Hadoop environment and in case of a processor provided as Unix executable a 'call' script cares for input provision and output archiving. Processors are software packages automatically deployed on demand.
- A processor implemented in Java may either implement the BEAM operator interface or the Hadoop Mapper interface to be executed without additional adapter in a Calvalus/Hadoop environment. Processors are Java software library packages automatically deployed on demand.

Unix executables use a file interface to access inputs and write outputs. Parameters are transmitted as command line arguments, as environment variables, or in a parameter file. Unix executables can provide feedback with the return code and with messages on stdout/stderr and in log files. Figure 5-11 and Figure 5-12 show processor wrapper scripts for a Grid Engine environment and for a Calvalus/Hadoop environment respectively. Both of them provide the environment for the processor call and call it with the parameters it requires.

```
#!/bin/tcsh


#input parameters:

set l1bproduct = $1
set sensor = MERIS

set beamRootDir = /opt/beam-4.10.3

set l1bBaseName = `basename $l1bproduct .N1`

echo "time $beamRootDir/bin/gpt-d-l2.sh  lc.l2 -e -c 3000M -Psensor=$sensor -t \
${l1bBaseName}_SDR.dim $l1bproduct"

time $beamRootDir/bin/gpt-d-l2.sh  lc.l2 -e -c 3000M -Psensor=$sensor -t \
${l1bBaseName}_SDR.dim $l1bproduct
```

Figure 5-11: Wrapper script example (SDR processor call for single MERIS L1b input) for Grid Engine

```
#!/bin/bash

set -e


# sr-step.sh 2010-01-01 2010-01-07 notemporal /calvalus/projects/lc/sdr-fr/2010 \
/calvalus/projects/lc/2010-fr/sr/2010

start=$1
stop=$2
filter=$3
input=$4
output=$5

year=`basename ${input}`
input1=`dirname ${input}`
output1=`dirname ${output}`


if [ `basename $input1` = "sdr-rr" ]; then
  resolution="rr"
  RESOLUTION="RR"
else
  resolution="fr"
  RESOLUTION="FR"
fi

template=$LC_INST/etc/sr-${filter}-template.xml
request=$LC_INST/requests/sr-${resolution}-${filter}-${start}.xml

cat $template | sed -e "s,\${year},${year},g" -e "s,\${start},${start},g" -e \
"s,\${stop},${stop},g" -e "s,\${input},${input1},g" -e "s,\${output},${output1},g" –e \
"s,\${resolution},${resolution},g" -e "s,\${RESOLUTION},${RESOLUTION},g" > $request


cd $LC_INST

echo "java -jar $LC_PRODUCTION_JAR -e --beam $LC_BEAM_VERSION --calvalus
$LC_CALVALUS_VERSION $request"

java -jar $LC_PRODUCTION_JAR -e --beam $LC_BEAM_VERSION --calvalus $LC_CALVALUS_VERSION
$request
```

Figure 5-12: Wrapper script example (Sample aggregation processor call) for Calvalus

Java processors in BEAM can consist of a single operator or of a graph of operators applied to input products to produce an output. Figure 5-13 shows the interface of operator implementations in BEAM. One of the two methods computeTile or computeTileStack must be implemented.

```
package org.esa.beam.framework.gpf;
public class Operator {
    final OperatorContext context;
    protected MyOperator() {
        context = new OperatorContext(this);
        context.injectParameterDefaultValues();
    }
    public abstract void initialize() throws OperatorException;
    public void computeTile(Band targetBand, Tile targetTile, ProgressMonitor pm)
        throws OperatorException {
        ...
    }
    public void computeTileStack(Map<Band, Tile> targetTiles,
                                 Rectangle targetRectangle, ProgressMonitor pm)
        throws OperatorException {
        ...
    }
    public void dispose() {}
    ...
}
```

Figure 5-13: Operator interface for BEAM processors

There are objects for products, bands (variables), and tiles with access functions via the operator context for inputs and outputs. The objects abstract from the respective file formats by readers and writers. Processing parameters are provided as function parameters via the operator context, and a processor gives feedback by callback functions and - if necessary - exceptions. BEAM processors can be used as executables (gpf command), interactively (VISAT tool), and in Calvalus.

```
package org.apache.hadoop.mapreduce;
public class Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT> {
    public Mapper() { ... }
    protected void setup(Context context)
        throws java.io.IOException, java.lang.InterruptedException { ... }
    protected void map(KEYIN key, VALUEIN value, Context context)
        throws java.io.IOException, java.lang.InterruptedException { ... }
    public void run(Context context)
        throws java.io.IOException, java.lang.InterruptedException { ... }
    protected void cleanup(Context context)
        throws java.io.IOException, java.lang.InterruptedException { ... }
}
```

Figure 5-14: Mapper interface with for other Java processors in Hadoop

Java processors in Calvalus/Hadoop may choose to implement the Hadoop Mapper interface instead of the BEAM Operator interface. This interface requires to implement either a function map() or a function run() (Figure 5-14). Inputs are accessed as streams or as sets of generic key-value pairs that are supported by an efficient Hadoop-internal file format. Also outputs are streams or sets of generic key-value pairs.

While both Java interfaces can be more efficient because they can use streaming and operator chaining in BEAM, the use of the wrapper scripts allows adaptation to Unix executables without changing them. Both alternatives are available in the LC CCI system.

5.4.2 Processor version concept

Processors, or more precisely processor bundles including software and configuration are the units that are under configuration control in the LC CCI system. Each processor bundle has a version. A bundle may include one or several processors. There is a runtime structure of the processor software, and optionally there is a different development structure. The runtime structure is packed into an installation package for deployment. The packing procedure labels this installation package with the proper version in its file name.

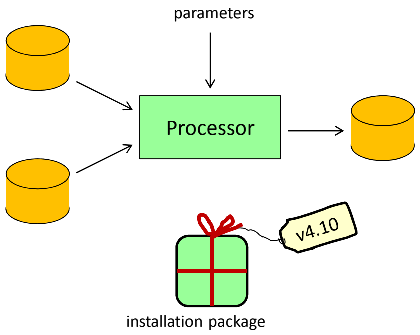


Figure 5-15: Processors and versioned processor installation packages for modularity

Processing jobs specify a bundle by name and version, and a processor to be run. A combination of bundles and their respective versions used for the generation of a dataset form an assembly. The LC CCI assemblies comprise a certain version of the the LC Preprocessing software, and a certain version of the LC Classification software. In return, both of these consist of various submodules. With the same assembly the same output is reproducible from the same input if required.

In order to simplify development, the schema for development versions is less strict. It follows the snapshot approach sometimes used with the 'git' version control system. The same version number can be used repeatedly to tag the development version as long as it is not freezed. Figure 5-16 shows the versioning schema and its principles.

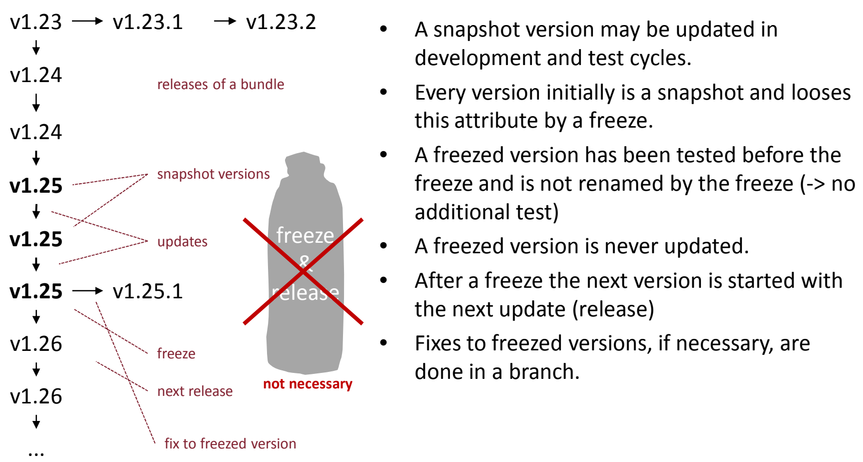


Figure 5-16: Processor and configuration versioning with freeze and release

The git version control system and a git server (e.g. GitHub) is proposed for the processor bundles of the LC CCI system. If there are parts that are not open source the corresponding bundles are kept in a non-public repository of the git server. All other bundles are in a public repository (public for reading) in order to allow interested users to review the software and configurations used and to allow developers to contribute.

The LC CCI version control provides the following convenience functions for processor developers:

- convenient commit: stores the current status of the directory tree of the development structure of the processor bundle, a convenience command; includes adding/deleting files from the directory tree, commit and push to the repository, tagging with the current version number, or a new branch version number in case of fixes
- convenient checkout: restores the directory tree of the development structure of the processor bundle; checks out main trunc or specified version. If a version is specified a simple commit starts or continues a branch.
- freeze and release: commits and freezes the current version and ensures that convenient commits will use the next version.
- deploy: generates and optionally installs the installation package of the current version. If the version is freezed the installation package contains this feature as a marker file. Only freezed versions shall be used for production and for bulk tests.

Concurrent development by experienced developers can still use the commands of the version control system (git) for merging and branching if required. The convenient functions can be used without detailed knowledge of version control systems when conflicts caused by concurrent changes of the same module are avoided.

5.5 System documentation

The system documentation comprises manuals, specifications and reports. It is supplemented by LC data oriented documentation like product specification, algorithm definitions, and validation reports. Access to an open-source repository replaces the DPM where applicable.

The system documentation comprises requirement documents, design and interface control documents, test documents, manuals, and maintenance information. Compared to this there are additional project documents related to the system, e.g. the LC CCI ATBD and the LC CCI PUG.

- The existing SRD and SSD define requirements and design of the system. They may get updates in CCI phase 2. An ICD for the main interfaces complements this for machine-to-machine user services, repository and deployment, request submission, processor integration, processing storage and inventory, and data exchange.
- An operations manual includes an instruction part with step-by-step descriptions of different use cases, and a reference part related to the functions and functional components and their capabilities, how to use them. There are different parts for the three subsystems user services, production environment, and development.
- An installation and administration manual describes the initial setup and configuration of the LC CCI system, how to upgrade the system to a newer version, how to do maintenance and backup, how to extend the system with additional hardware and for new sensors.
- The processor integration guide describes the most important internal interface of the system and is a form of an ICD. The focus is on how to integrate the LC processors.
- The system verification documents define a set of tests and report about their results for the versions of the system that have been provided.
- The software release notes describe valid combinations of versions of components and software packages and they identify the corresponding documentation. They identify the versions currently in use.
- The issue tracking system documents system issues, among others, and their status.

This set of documents has been selected because they are considered most relevant and most used.

6 Development, life cycle, cost and performance

This section is a collection of further analyses regarding re-use of components, system life cycle, and cost and performance.

6.1 Re-use and development

The LC CCI system re-uses components from the prototype, adds components for user services, configures and adapts them, and develops LC-specific extensions.

Table 6-1 lists software packages that can be adapted, configured and integrated for use in the LC CCI system.

Table 6-1: Re-used software and its integration effort

Software	Role	Adaptation	Integration and Configuration
AMORGOS	MERIS geocoding	None	wrapper scripts for the integration into the processing framework
LC L2 processors	SDR retrieval	development from prototype	processor bundle deployment, parameterisation
LC L3 processors	SR retrieval and aggregation	development from prototype	processor bundle deployment, parameterisation
Classification processors	LC classification and condition generation	development from prototype	processor wrapper scripts for the integration into the processing framework, parameterisation
User tools	Aggregation, reprojection, regridding	development from prototype	None for offline use Provision as service for user services
Apache Hadoop	processing system middleware	None	Hadoop setup
Calvalus	processing system	None	workflow definitions, parameterisation
BEAM	used in Calvalus, in tools, and in L2 and L3 processor	LC CCI data readers	none

Software	Role	Adaptation	Integration and Configuration
Liferay	portal framework	None	Web design, content, system configuration, forum setup, integration of other services (MERCI, THREDDS, Jira)
MERCI	catalogue	LDAP user management	CCI product discovery metadata definition CCI product quality check criteria CCI product quicklook and thumbnail generation
THREDDS	online data server	None	setup of THREDDS OPeNDAP, WMS
MapServer OpenLayers	Image server for validation	development from prototype	automated provision of LC results as MapServer inputs
vsftpd	Online data server	None	directory structure and access control configuration
Jira (optional)	Issue tracking	None	Setup of Jira, LC project configuration
KnowledgeTree (optional)	Document management	None	Setup of KT, tree structure configuration
OpenLDAP	User management backend	None	LDAP configuration

6.2 System life cycle drivers and considerations

Starting with the prototype migrated to the target platform, LC-CCI is operated and incrementally developed by functional extension, improved algorithms, and new input datasets. The large, growing amount of data for Sentinel 3 and optionally for Sentinel 2 requires a stepwise extension of the hardware infrastructure, too.

The LC CCI system life cycle will not be completely static and pre-planned because of the desired inclusion of new requirements over time. Driving forces for an evolutionary development are

- availability of the LC CCI prototype as a starting point for further development
- incremental functional extension of the system
- improvement of algorithms, addition of new workflows and processing chains
- Sentinel 3 and other missions, increased data volume, options for synergistic use
- continuous improvements in hardware and data centre environments
- concurrent development for other ECVs

To respond to the first two points the system is initially based on the prototype. It comprises only a subset of functions and components, but on the target platform. It may start with a core processing chain for ATSR and AVHRR using prototype processors and a core user service with the Web portal and an FTP data service for outputs. Incrementally, additional components and functions are added and interfaces to data providers and users are extended.

The third point of algorithm development requires the addition of validation capabilities, e.g. the online validation tool, user feedback e.g. via the forum, and configuration control for the software. The next stage of the system shall support these functions.

The increased data volume and the new products are a qualitative change, too. The existing methods need to be adapted to make use of new channels. The amount of data is a challenge to currently available hardware for storage, memory and throughput. Optimisation will be required to avoid new bottlenecks. Example: To keep all input data online the system may grow on a monthly basis. This would provide the necessary resources to keep reprocessing time constant. Before the new data is available the system includes the necessary extensions and optimisations.

For the next three years phase of LC CCI it is reasonable to decide for a certain hardware and data centre environment and to keep this constant. But for the longer perspective also renewal of hardware and optional change of software layers can be expected. The system is prepared for this by the modularity of its functional components.

Last, the concurrent development for other ECVs is kept independent initially to avoid delays. But if supported, ECVs may exchange results and use them at least for consistency checks once they are available. LC CCI intends to use Fire CCI products as condition products and for comparison.

6.3 Sizing and performance analysis

The data storage budget for inputs and outputs for historical data and for the yearly increase of acquired data is 720 TB initially, growing by 720 TB per year (TBC) considering Sentinel 3.

Full reprocessing of historical data requires about 700000 CPU core hours.

Requirements addressed by this section are:

→ LC-SR-1100 Archive capacity
→ LC-SR-1110 Backup archive
→ LC-SR-3310 Production performance
→ LC-SR-3335 Scalability

This section defines the budgets for data storage and processing capabilities. The budget for data storage mainly depends on the amount of input data to be managed. This comprises historical data and data acquired continuously.

Table 6-2: Data budget for historical EO input data and for new data of future missions

Type	Coverage	Historical Data	Acquired data
MERIS FR and RR	2002-2012	155 TB	-
SPOT VGT	since 1997	7 TB	0.6 TB per year

Type	Coverage	Historical Data	Acquired data
NOAA AVHRR HRIT	1992-98 (partial)	6 TB	-
S3 OLCI	from 2014 on	-	150 TB per year
S3 SLSTR (SWIR, TBC)	from 2014 on	-	subset of 232 TB per year (TBC)
S2 (optional)	from 2014 on	-	TBC (optional)

The volume of Level 2 intermediate products depends on the amount of inputs, while the volume of Level 3 and Level 4 output products is constant per year.

Table 6-3: Data budget for LC outputs

Type	Coverage	Historical Data	Acquired data
LC SR	1997 - 2012	60 TB	6 TB per year
LC Map and Condition	1997 - 2012	3 TB	1 TB per year

The hardware storage budget must foresee some spare for intermediates and test results, a factor for different versions to be kept concurrently, and a factor for redundancy.

- 100% spare (for intermediates)
- 100% redundancy (for inputs)
- 100% versions (for outputs)

This corresponds to a factor of 4 in the storage budget.

- The system storage is 720 TB initially
- and grows by 720 TB per year.

Table 6-4: Processing load budget for LC reprocessing

Type	Amount	CPU Core hours historical data	CPU Core hours per additional year
Pre-processing	168 TB	600000 h (90 days on 300 cores)	600000 h
Classification	60 TB	100000 h	50000 h
Sum		700000 h	650000 h

The allowed reprocessing time of 3 months leads to

- a concurrency of 400 for the historical data
- a concurrency of 760 and 1120 for subsequent years with S3 OLCI and S2

6.4 Cost estimation

Costs for storage, processing hardware, network, development and integration, operations, dissemination comprise about 500000 € per year of the three years second phase.

Requirements addressed by this section are:

→ LC-SR-1330 Bulk Dissemination

The costs for the system are composed of costs for storage, processing, network, development and integration, operations, dissemination. Looking into storage and processing using the Hadoop approach of combined CPU and data storage, a machine with a quadcore CPU, 16 GB RAM, and 9 TB disk space is about 1000 €.

- 80 machines are sufficient to keep the historical data, leading to 90 days for reprocessing. This is about **80,000 €**. The initial infrastructure for classification processing is estimated with **30,000 €**.
- 80 additional machines are sufficient to keep the data of one additional year and reprocess it in less than 3 month. This is another **80,000 € per year**. The additional infrastructure for classification is estimated with **15,000 €**.

The bottleneck is the amount of data to be stored and to be provided for processing, very much balanced by the CPU power.

The user services are hosted on machines with a cold stand-by spare (4 machines, ~**4000 €**). Electric power will consume a considerable amount of money for the processing provided (**20,000 €** per year initially, increasing by **20,000 € per year**).

Operations is about one person year per year (~**150,000 €**). Development and integration is about the same amount for the first year (~**150,000 €**) and a smaller amount in subsequent years (~**75,000 €**).

Table 6-5: Cost estimates per year

	First year	Second year	Third year
Hardware	114000 €	95000 €	95000 €
Development	150000 €	75000 €	75000 €
Operations	150000 €	150000 €	150000 €
Electric power	20000 €	40000 €	60000 €
Sum	434000 €	360000 €	380000 €

Since this is the "target" system there are options for reduction if the requirements are released. Cost drivers are:

- storage of OLCI data without data reduction, including redundancy
- storage of two complete output versions
- reprocessing time of 3 month

Dissemination via cloud storage

Due to the large volume and the high traffic generated by the dissemination to a significant number of users, this approach generates additional costs. Estimates based on the pricing information of two major suppliers (Amazon S3 and Google Cloud Storage), the costs for storage are in the order of 0.10 € /GB/month and download traffic costs are about 0.05 €/GB. For the LC CCI output dataset with a size of about 1 TB this results in

- storage: 0.1 €/GB x 1TB = 100 € per month, **1,200€ per year**
- traffic: 0,05 €/GB x 1TB = **50 € per user** downloading the full dataset

In case of 10000 users this is 500000 €.

7 Requirements traceability

ID	Title	Reference
LC-SR-0010	Generate and disseminate LC products	§3
LC-SR-0100	Output products and access	§3.2
LC-SR-0400	Implement processing workflows and scenarios	§3
LC-SR-0200	Stability of outputs	§5.4
LC-SR-0300	Reactive to improvements	§5.4
LC-SR-4100	Production control	§3.3, §3.4
LC-SR-4200	LC processor integration	§5.2
LC-SR-2100	Quality of outputs	§5.3
LC-SR-2110	Processing quality control	§5.3
LC-SR-4210	AMORGOS processor	§5.2
LC-SR-4215	SPOT-VGT geocoding improvement	§5.2
LC-SR-4220	LC L2 Processor	§5.2
LC-SR-4230	LC L3 Processor	§5.2
LC-SR-4235	LC Merging Processor	§5.2
LC-SR-4310	Stratification processor	§5.2
LC-SR-4315	Compositing processor	§5.2
LC-SR-4320	Spectral classification processor	§5.2
LC-SR-4330	Temporal classification processor	§5.2
LC-SR-4340	Classification merging processor	§5.2
LC-SR-4350	Manual editing tool	§5.2
LC-SR-4360	Land cover condition processor	§5.2
LC-SR-4370	Land cover map and condition integration processor	§5.2
LC-SR-4140	Expert-driven validation	§3.5, §5.1
LC-SR-4142	Long-term validation data collection	§5.1, §5.4
LC-SR-4144	Automated validation	§5.3
LC-SR-4146	Validation datasets	§5.3
LC-SR-1210	Data inventory	§5.3
LC-SR-2200	Structured data storage	§5.3
LC-SR-4410	Internal data exchange	§5.3
LC-SR-1220	User relations management	§5.1
LC-SR-1230	User subscription	§5.1
LC-SR-1240	Subsetting, aggregation, re-projection	§5.1

ID	Title	Reference
LC-SR-3200	Reprocessing performance	§5.3
LC-SR-3310	Production performance	§6.3
LC-SR-3330	Processing Concurrency	§4.2
LC-SR-3335	Scalability	§4.2, §6.3
LC-SR-3360	Cluster for data-intensive processing	§2.6, §4.2, §5.3
LC-SR-3370	Map-reduce for level 3 binning and aggregation	§5.3
LC-SR-3380	Machine for multithreading-intensive processing	§2.6
LC-SR-1100	Archive capacity	§6.3
LC-SR-1110	Backup archive	§2.5, §6.3
LC-SR-3320	Optimised archive access	§2.5
LC-SR-1330	Bulk Dissemination	§3.2
LC-SR-4400	Distribution	§4.1
LC-SR-1130	Level 1 input	§5.3
LC-SR-1120	Input ingestion	§5.3
LC-SR-1200	Web access	§5.1
LC-SR-1300	FTP access	§5.1, §6.4
LC-SR-1310	NetCDF output format	§5.1
LC-SR-1320	GeoTIFF output format	§5.1
LC-SR-2130	Manual failure handling	§3.3, §3.4, §5.3
LC-SR-2120	Automated failure handling	§3.3, §3.4, §5.3
LC-SR-3110	Transfer to operations	§3.6, §5.4
LC-SR-2300	Processor Versioning	§5.4
LC-SR-3340	Reprocessing support	§3.3, §3.4, §5.3
LC-SR-3350	Processing granularity	§5.3
LC-SR-4150	Operator tools	§5.3
LC-SR-4110	Automated pre-processing	§3.3, §5.3
LC-SR-4120	Automated classification	§3.4, §5.3
LC-SR-4130	Classification expert operating	§3.4
LC-SR-2400	Configuration control	§5.4
LC-SR-2405	Modularity	§5.4
LC-SR-3100	Continuous development	§5.4