IS-ENES3 Deliverable D10.1

Architectural document of the ENES CDI software stack

Reporting period: 01/01/2019 – 30/06/2020

Authors: S. Fiore (CMCC), P. Nassisi (CMCC), F. Antonio (CMCC), L. Barring (SMHI), A. Ben Nasser (IPSL), K. Berger (DKRZ), D. Hassell (UREAD-NCAS), M. Juckes (UKRI), P. Kershaw (UKRI), S. Kindermann (DKRZ), G. Levavasseur (IPSL), A. Nuzzo (CMCC), C. Pagé (CERFACS), A. Stephens (UKRI), W. Som de Cerff (KNMI), A. Spinuso (KNMI), M. Stockhause (DKRZ), T. Weigel (DKRZ)

Reviewers: P. Kershaw (UKRI), S. Kindermann (DKRZ), M. Lautenschlager (DKRZ)

Release date: 31/07/2020

ABSTRACT

This document provides a comprehensive description of the design activities performed through the first 18 months of the IS-ENES3 project. Taking into account the requirements collected with the milestone M10.1, it proposes an overall architectural view of the ENES Climate Data Infrastructure software stack and it goes into the details of each single component. This work will lead the development of the ENES CDI software stack architecture during the project timeframe and it will be periodically revised to include emerging needs and ensure full compliance with the users' requirements.

Dissemination Level		
PU	Public	X
CO	Confidential, only for the partners of the IS-ENES3 project	

Revision table			
Version	**Date**	**Name**	**Comments**
Initial draft	07/04/2020	S. Fiore, P. Nassisi	First draft of the document
Release for review	30/06/2020	WP10 Team	All sections completed
Final review	31/07/2020	S. Fiore, P. Nassisi	Reviewer comments integrated

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 824084

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

One of the most important needs of the climate modeling community, climate impact community as well as interdisciplinary research domains is to rely on stable and consistent services to access and process the high volume of climate model data from CMIP and CORDEX simulations hosted in distributed repositories across Europe.

To this end, the IS-ENES project will evolve and consolidate the ENES Climate Data Infrastructure (CDI) software stack to i) provide a sustainable, streamlined and scalable climate model data distribution, ii) support interoperability for automated data processing, iii) provide an interoperable and flexible computing layer, supporting scientific data analysis and processing, iv) maintain an international documentation infrastructure to support the future Model Intercomparison Projects.

Through this document, a comprehensive description of the design performed so far will be presented as well as some preliminary implementation activities, with the aim to evolve and consolidate the ENES Climate Data Infrastructure software stack. This work will lead to the development of the CDI components for the project timeframe, laying the groundwork and providing the European contribution to the future ESGF architecture.

1. INTRODUCTION

This report aims to provide a complete architectural design of the ENES CDI software stack, according to the overall IS-ENES3[1] project objective #3, which fosters the *support for the exploitation of model data by both the earth system science community and the climate change impacts community*. The software components involved are developed and maintained as open source efforts by the IS-ENES partner institutes with contributions from the international Earth System Grid Federation[2] (ESGF) developers community.

The document addresses the deliverable **D10.1 "Architectural document of the ENES CDI software stack"** of the IS-ENES3 project, within the WP10/JRA3 "ENES Climate Data Infrastructure software stack developments". Starting from the technical requirements collected in the context of the milestone **M10.1 "Technical requirements on the software stack"** [1], it provides an overall view of the ENES CDI software stack with a detailed design of each component; some preliminary implementation activities performed during the first reporting period are also described. The architectural design, which is at the core of the document, makes use of diagrams describing the structure and the behaviour of each component, thus highlighting how every single module contributes to meet the requirements of the overall software stack.

The ENES CDI software stack releases will constitute the European contribution (developed during the IS-ENES3 project lifetime) to the long-term view provided by the international ESGF Future Architecture[3]. The different releases of the ENES CDI in D10.2, D10.3 and D10.5 reflect this development. Technical constraints and side effects are discussed in this deliverable, the ESGF Future Architecture Report as well as M5.1 "Draft architecture design" and M5.2 "ESGF CMIP6 summary" from NA4/WP5; cultural and resourcing constraints are part of the sustainability process carried out in NA1/WP2. Indeed, the transition to a new, sustainable ENES-CDI is a longer process throughout the project's runtime and it also requires international coordination and consideration of non-technical side effects.

The rest of the document is structured as follows: Section 2 proposes an introduction to the IS-ENES3 project, its main goals, stakeholders and correlation with ESGF and other European data ecosystems. A general overview of the current status of the ENES CDI is presented, introducing the list of software components such as the core data distribution and compute services, vocabulary management, documentation and impact study tools. Section 3 resumes the results achieved through the M10.1 document, listing the main functional and non-functional requirements for the overall software stack. Section 4 proposes an overall ENES CDI design, emphasizing physical aspects through a multi-tier architecture diagram, while Section 5 goes into the details of the design for each component anticipating some preliminary

[1] https://is.enes.org/

[2] https://esgf.llnl.gov/index.html

[3] http://doi.org/10.5281/zenodo.3928223

developments. Section 6 proposes a plan for the future releases. Finally, Section 7 drives the main document conclusions and makes a brief analysis of the outcomes.

2. ENES CLIMATE DATA INFRASTRUCTURE

In this section, an overview of the current status of the ENES CDI is described, with a focus on its role in the European landscape and its correlation with international ESGF context. A synthetic list of the CDI software components is also presented.

2.1. Goals of the ENES CDI and EU landscape

Users from the climate modelling community, the climate impact community as well as interdisciplinary research domains rely on stable and consistent services to access and process the high volume climate model data of WCRP[4] reference simulations from CMIP[5] and CORDEX[6], hosted in distributed repositories across Europe. To this end, the ENES CDI is the primary source for climate model data in Europe (originating from both European and international modeling groups). It provides access to data from ESGF, from the WDCC[7] archival system and from the Climate4Impact portal[8], as well as processing services, documentation and standards. From the perspective of the software stack, the ENES CDI consists of a set of services related to core data distribution/management, compute and analytics, vocabulary management, documentation and impact studies. The software developed in WP10 provides the building blocks for the setup of the ENES CDI service infrastructure managed in WP7; its long-term sustainability plan is part of WP2.

The IS-ENES3 project will evolve and consolidate the ENES Climate Data Infrastructure software stack according to the overall project objective #3 (*IS-ENES3 will support the exploitation of model data by both the earth system science community and the climate change impacts community*).

In IS-ENES3, the high-level objectives with respect to the ENES CDI software stack aim to:

- design, improve, extend and consolidate the ENES CDI software stack as a basis for a sustainable, streamlined and scalable climate model data distribution solution for users in the climate modeling and the climate impact research and modelling communities.
- Support interoperability of data files and archives for automated data processing through improved and extended standards and metadata.
- Provide an interoperable and flexible computing layer supporting scientific data analysis and processing within the infrastructure, by evolving existing solutions

4 https://www.wcrp-climate.org/
5 https://www.wcrp-climate.org/wgcm-cmip
6 https://cordex.org/
7 https://www.dkrz.de/up/systems/wdcc
8 https://climate4impact.eu/

towards an integrated set of service offerings for end users (from climate researchers to the climate impact research and modelling community including long-tail end users).

- Evolve the Climate4Impact (C4I) platform towards a climate data analytics portal for impact scientists. This is done by providing advanced data processing services and data access services in the C4I portal. The functionality will be made available through user friendly interfaces (e.g., tailored search interfaces, guided wizards, Jupyter notebooks with use case examples).
- Maintain and develop the ES-DOC international documentation infrastructure to support CMIP6 and other MIPs as well as expand the scope of documentation to new areas for the climate modelling process, including model evaluation.

The Earth System Grid Federation is based on a software architecture that provides access to a federated data archive distributed across multiple sites (called ''Nodes'') that are geographically distributed around the world but can interoperate due to the adoption of a common set of services, protocols and APIs [2].
Through the ENES-CDI, IS-ENES3 provides the European contribution to ESGF in terms of softwares (WP10) and services (WP7). However, the ENES-CDI also provides room for (i) EU-level/national specific developments, which do not target ESGF directly, and (ii) early software developments, which would need a pre-production phase for testing and validation before moving into a wider production environment.
Additionally, IS-ENES3 activities include (i) the contribution to the future ESGF architecture (WP5), which spans across a longer timeframe with respect to IS-ENES3 and (ii) the IS-ENES Sustainability plan (WP2), which of course includes the ENES CDI too.
Table 1 summarizes and clarifies the link between the aforementioned activities in the project.

Table 1. IS-ENES3 contribution to the ENES-CDI and ESGF

IS-ENES3 WP	**Context**	**Target**	**Timeframe**
WP2 “Governance, Sustainability and Innovation”	ENES CDI	Sustainability of the ENES CDI	Long term (decade)
WP5 “Networking on data and model evaluation”	ESGF	Software stack architecture	Long term (decade)
WP7 “Data standards, distribution and processing services”	ENES CDI	Operational Services	Short/Medium (IS-ENES3 timeframe)
WP10 “ENES Climate Data Infrastructure software stack developments”	ENES CDI	Software stack architecture and development	Short/Medium (IS-ENES3 timeframe)

From the table above, it can be easily inferred that ensuring the proper link between the ENES CDI software stack architecture and the future ESGF architecture is essential in terms of design perspective and towards a common long-term, sustainable objective. In this perspective, through the ENES CDI contribution, IS-ENES3 will contribute to short/medium term actions in the long-term ESGF roadmap with resources as in the past that are balanced with the US contribution.
The ENES CDI will be designed by also considering the ongoing efforts in the wider European *data ecosystem* and it will look forward to the EOSC[9] roadmap and evolution as well as to the Copernicus[10] landscape. In particular, the design of some specific components, like the compute service, will take into account the link with EOSC e-infrastructures (EGI[11], EUDAT[12]) and the Copernicus C3S tools and platform.

9 https://www.eosc-portal.eu/
10 https://climate.copernicus.eu/
11 https://www.egi.eu/tag/eosc/
12 https://www.eudat.eu/

3. REQUIREMENTS ANALYSIS

This section summarizes the results of the M10.1, by reporting the table of the functional requirements (Table 2) and the mapping between the ENES CDI services and the non-functional requirements (Table 3). For a more detailed description, please refer to the document M10.1.

3.1. Brief summary of the results coming from M10.1

The document "M10.1 - Technical requirements on the software stack" of the IS-ENES3 project has provided a wide list of technical requirements for the ENES CDI software stack, driven by the work done in WP5/NA4 "Networking on data and model evaluation" and WP3/NA2 "Community engagement" as well as by previous meetings and workshops in the community (i.e. ESGF F2F Conferences). It represents the foundation of the software stack architecture design, addressed through this document, and will drive the implementation of the ENES CDI during the IS-ENES3 project.
The report extracts functional and non-functional requirements starting from a comprehensive set of user stories and use cases for each service of the ENES Climate Data Infrastructure.
As a result, preliminary architectural insights have been also proposed in the milestone document, though presenting the final ENES CDI software stack architecture is the main goal of this deliverable.

3.2. Technical requirements specification

3.2.1. List of functional requirements

Table 2. List of functional requirement codes and names for each ENES CDI service

Requirement code	Requirement name	Service
[DATAFR#1]	Data publication	ESGF Data
[DATAFR#2]	Data search	ESGF Data
[DATAFR#3]	Data download	ESGF Data
[CITFR#1]	CMIP6 compliance	Data Citation
[CITFR#2]	Insertion or update of citation information	Data Citation
[CITFR#3]	Access to citation information	Data Citation

[PIDFR#1]	PIDs assigned for CMIP6 files and datasets	Persistent Identifier (PID)
[PIDFR#2]	Essential metadata registered for CMIP6 file and dataset PIDs	Persistent Identifier (PID)
[DDCFR#1]	Discovery	IPCC DDC at DKRZ
[DDCFR#2]	Download	IPCC DDC at DKRZ
[ERRFR#1]	Data publication	Errata
[ERRFR#2]	Issue viewer	Errata
[ERRFR#3]	Issue list	Errata
[ERRFR#4]	Issue management	Errata
[STATSFR#1]	Data publication statistics	Data Statistics
[STATSFR#2]	Cross-project data download statistics	Data Statistics
[STATSFR#3]	Project-specific data download statistics	Data Statistics
[STATSFR#4]	Data statistics export	Data Statistics
[STATSFR#5]	IS-ENES3 KPIs	Data Statistics
[REPLICFR#1]	Persistent status tracking and recording	Data Replication
[REPLICFR#2]	Multiple transfer protocols and parallel transfers	Data Replication
[REPLICFR#3]	Faceted data characterization	Data Replication
[REPLICFR#4]	Automatic retry	Data Replication
[COMPFR#1]	Server-side data analysis	Compute & Analytics
[COMPFR#2]	Subsetting	Compute & Analytics
[COMPFR#3]	Simple operations	Compute & Analytics
[COMPFR#4]	Complex analysis	Compute & Analytics
[COMPFR#5]	Access control	Compute & Analytics
[COMPFR#6]	API	Compute & Analytics

[COMPFR#7]	Standard interface	Compute & Analytics
[COMPFR#8]	High-level user environment for data analysis	Compute & Analytics
[COMPFR#9]	Compute Service discovery	Compute & Analytics
[C4IFR#1]	Search Wizard Interface	Climate4Impact
[C4IFR#2]	Visualization Service	Climate4Impact
[C4IFR#3]	Documentation and Guidance	Climate4Impact
[C4IFR#4]	User Space Storage (Basket)	Climate4Impact
[C4IFR#5]	Data Subsetting Service	Climate4Impact
[C4IFR#6]	Download Service	Climate4Impact
[C4IFR#7]	Standard Climate Indices and Indicators Calculation Service	Climate4Impact
[C4IFR#8]	Data repackaging Service	Climate4Impact
[C4IFR#9]	Provenance/Lineage Service	Climate4Impact
[ESDOCFR#1]	Documentation publication	ES-DOC
[ESDOCFR#2]	Web-based documentation access	ES-DOC
[ESDOCFR#3]	API-based documentation creation and access	ES-DOC
[ESDOCFR#4]	Further Info URL	ES-DOC
[CFFR#1]	Discussion board	Climate Forecast (CF)
[CFFR#2]	Publication workflow	Climate Forecast (CF)
[DREQFR#1]	Database (DR climate indices)	Data Request
[DREQFR#2]	Programmatic access	Data Request
[DREQFR#3]	Versions	Data Request
[DREQFR#4]	Request ingestion	Data Request
[DREQFR#5]	Database (DR CMIP)	Data Request

[DREQFR#6]	Web Interface	Data Request
[DREQFR#7]	Software	Data Request
[DREQFR#8]	Component versions	Data Request
[DREQFR#9]	Schema	Data Request
[IDFR#1]	Delegation	Identity Management and access entitlement
[IDFR#2]	Delegation with service chaining	Identity Management and access entitlement
[IDFR#3]	Role based access control	Identity Management and access entitlement
[IDFR#4]	Tokens	Identity Management and access entitlement
[IDFR#5]	IdP trust	Identity Management and access entitlement

3.2.2. Non-functional requirements matrix

Table 3. Mapping between services and set of non-functional requirements [3]

	NFR# 1	NFR# 2	NFR# 3	NFR# 4	NFR# 5	NFR# 6	NFR# 7	NFR# 8	NFR# 9	NFR# 10	NFR# 11	NFR# 12	NFR# 13
ESGF DATA	M	M	M	M	M	M	M	M	M	M	M	-	-
DATA CITATION	O	M	-	-	M	-	O	-	M	O	-	M	-
PERSISTENT IDENTIFIER	O	M	M	-	M	O	O	O	-	-	M	-	-
IPCC DDC AT DKRZ	O	M	M	-	M	-	-	O	M	O	-	M	-
ERRATA	M	M	O	M	M	M	O	M	M	O	-	-	-
DATA STATISTICS	M	M	M	O	M	-	M	M	M	O	-	-	M
DATA REPLICATION	-	M	M	-	-	-	-	-	M	-	-	-	-
COMPUTE & ANALYTICS	-	M	M	O	M	-	M	-	M	-	M	-	M
CLIMATE4 IMPACT	M	O	M	O	M	M	-	-	M	M	-	-	O
ES-DOC	M	M	M	M	M	M	M	M	M	M	O	-	-
CLIMATE FORECAST	M	M	-	-	-	-	-	-	-	M	M	-	-
DATA REQUEST	M	M	-	-	-	-	M	-	O	-	O	-	-
IDENTITY MNG & ACCESS ENTITLEMENT	-	-	-	-	-	-	-	-	-	-	R	R	M

Legend:

M = MUST
O = OPTIONAL
R= RECOMMENDED

NFR#1: Transparency
NFR#2: Robustness
NFR#3: Scalability
NFR#4: Efficiency
NFR#5: Security
NFR#6: Reusability
NFR#7: Extensibility
NFR#8: Flexibility
NFR#9: Usability
NFR#10: Look & feel
NFR#11: Interoperability
NFR#12: Access control
NFR#13: Deployability

4. DESIGN OF THE ENES CDI SOFTWARE STACK ARCHITECTURE

This section provides the architectural design of the ENES CDI software stack. The preliminary architectural sketch of the ENES Climate Data Infrastructure software stack already proposed in the milestone M10.1 has been validated and confirmed, unless minimal variations, in this section. An additional diagram providing an in-depth view of the overall architecture has also been created, based on the work done by the WP10 team during the first Reporting Period (see Section 5 of this document for a detailed description of each component).

4.1. Software stack architecture

The ENES CDI stack (Fig. 1) is part of a multi-tier architecture organized according to the following Tiers:

- **Platform Tier**: it includes the ENES CDI stack, a suite of software components that can be selectively deployed according to specific needs and goals. It consists of the following layers :
  - **Fabric** *(pink)*: it provides basic data, metadata and compute services;
  - **Federation** *(cyan)*: it federates and integrates different services at the Fabric layer across multiple collaborating organizations, thus providing federation-level capabilities (unified view) as well as a single entry point to the user;
  - **Application** *(yellow)*: it provides end-users applications to (i) perform data analysis, (i) get access to documentation, (iii) run/visualize climate indicators, (iv) report data usage/publication metrics, etc.
  - **Security** *(purple)*: it is an orthogonal layer of the stack that goes across the Fabric, Federation and Application layers. It includes, among the others, firewall settings, OS/applications/services security updates, etc.
  - **Monitoring** *(purple)*: as in the case of Security, it represents a cross-architectural layer that addresses monitoring aspects at different levels (e.g. from infrastructure to services up to applications, with different sets of metrics).

  The picture also highlights the main protocols by which each layer exposes its services; they are very diverse on the *Fabric* layer and more convergent towards Web Service API and HTTP respectively on the *Federation* and *Application* layers.

  The Platform Tier relates to any service of the ENES CDI that *could* be deployed according to a PaaS or SaaS approach in a public or private virtualized environment.

- **Infrastructure Tier** (*green*): it consists of compute, storage and network physical resources and, on top of it, the *Data*, which relate to sensors and to the output of numerical simulations.
  The Infrastructure Tier *could* be virtualized, thus providing access to the resources according to a IaaS approach either in a public or private cloud environment.

Finally, in grey, a set of ESGF services exploited in the ENES-CDI that are associated with collaborative development efforts carried out with partners outside Europe.

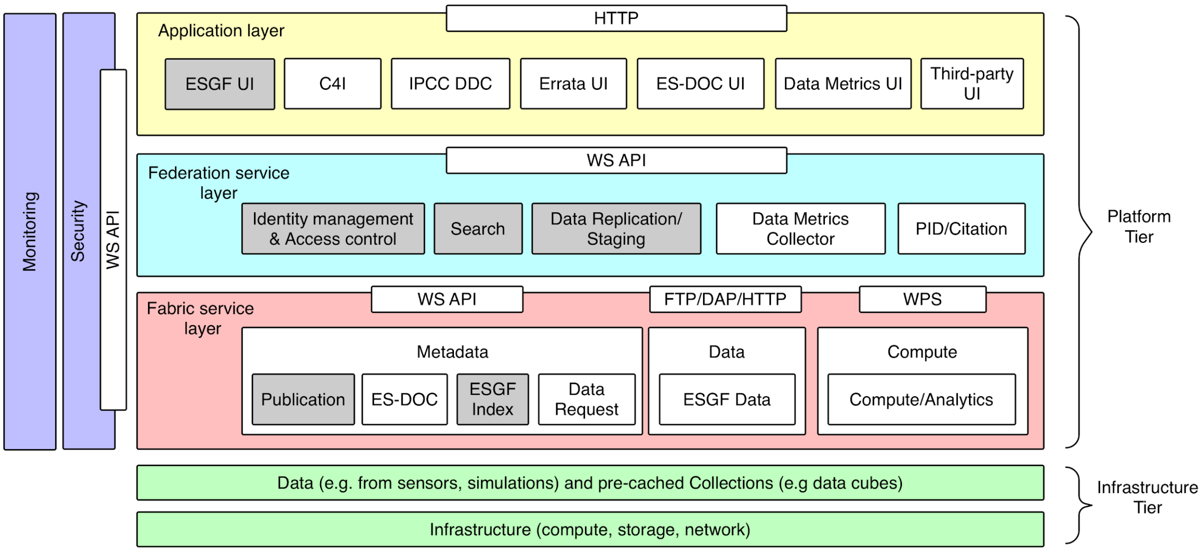

Fig. 1: ENES CDI software stack architecture

4.2. Detailed view of the ENES CDI architecture

In Fig. 2 the same architecture described above is detailed in a component diagram, highlighting the complexity of each component and the connection between each of them.

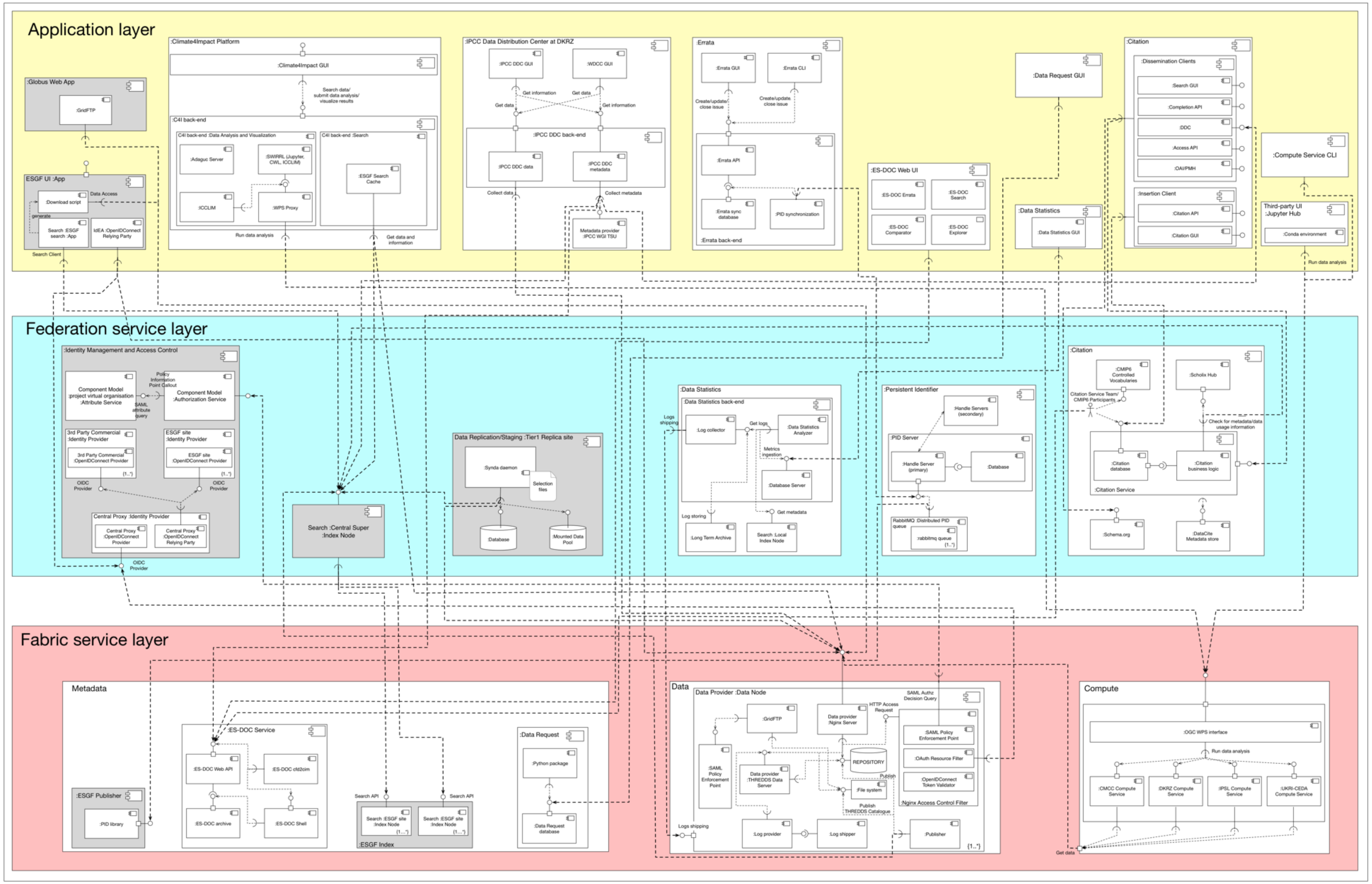

Fig. 2: Detailed component diagram of the ENES CDI software stack

4.3. List of the ENES CDI services

The overall goal of the architectural design phase is to define the ENES CDI software stack architecture. In this section, we provide the list of the ENES CDI services. For a brief description summarizing their current status and the main goal, please refer to the section 2.6 of the document M10.1 "Technical requirements on the software stack".
Table 4, reported here from the M10.1 document, highlights the link between the ENES CDI services and the list of software components associated with them, with a special focus on those developed in the context of the IS-ENES3 project. For each module, a reference to the GitHub repository is also provided.

Table 4. ENES CDI software components

ENES CDI Service	Software components
ESGF Data	*esg-publisher* doc: https://esgf.github.io/esg-publisher/index.html repo: https://github.com/ESGF/esg-publisher
	esgf prepare repo: https://github.com/ESGF/esgf-prepare
	esgf-pyclient repo: https://github.com/ESGF/esgf-pyclient
	CoG repo: https://github.com/EarthSystemCoG/COG
Data Citation	doc: http://cmip6cite.wdc-climate.de repo: internal gitlab software versioning at DKRZ
Persistent Identifier (PID)	*ESGF data publication pid library* repo: https://github.com/IS-ENES-Data/esgf-pid
	RabbitMQ federation doc: https://acme-climate.atlassian.net/wiki/spaces/ESGF/pages/107708573/PID+Services+Working+Team+esgf-pidwt (restricted access)
	PID consumer library doc and repo: https://gitlab.dkrz.de/esgf/handlequeueconsumer (restricted access)

IPCC Data Distribution Centre at DKRZ	*IPCC Data Distribution Centre at DKRZ:* http://ipcc.wdc-climate.de DDC web pages on server hosted at CEDA: http://www.ipcc-data.org/sim/
Errata	*ESGF Errata Service*: https://errata.es-doc.org/ doc: https://es-doc.github.io/esdoc-errata-client/ repos: ● Web-Service: https://github.com/ES-DOC/esdoc-errata-ws ● Front-end: https://github.com/ES-DOC/esdoc-errata-fe ● CLI: https://github.com/ES-DOC/esdoc-errata-client
Data Statistics	*ESGF Data Statistics UI:* http://esgf-ui.cmcc.it:8080/esgf-dashboard-ui-2020/ doc: ● https://acme-climate.atlassian.net/wiki/spaces/ESGF/pages/1043464194/Federated+data+usage+statistics+ESGF+Dashboard ● https://acme-climate.atlassian.net/wiki/spaces/ESGF/pages/1054113816/Proposed+ESGF+Usage+of+Filebeat+and+Logstash repo: https://github.com/ESGF/esgf-dashboard
Data Replication	*Synda replication software package* doc: http://prodiguer.github.io/synda/ repo: https://github.com/Prodiguer/synda
Compute	*ECAS* service: https://ecaslab.cmcc.it/jupyter/hub/login doc: https://ecaslab.cmcc.it/web/home.html repo: https://github.com/ECAS-Lab
	Ophidia doc: http://ophidia.cmcc.it/ repo: https://github.com/OphidiaBigData
	Birdhouse WPS framework doc: https://birdhouse.readthedocs.io/en/latest/ repo: https://github.com/bird-house security proxy: https://github.com/bird-house/twitcher
	ESGF-specific WPS framework under development for C3S prototype repos under development at: https://github.com/roocs underlying library: https://github.com/pydata/xarray

	Third party components: - *JupyterHub:* https://jupyter.org/hub - *xarray:* *http://xarray.pydata.org/en/stable/*
Climate4Impact	*C4I front-end, C4I backend, C4I storybook, C4I errorhandler, C4I front-end content, C4I search portal backend, C4I map preview, C4I frontend dataset preview* service: https://climate4impact.eu/impactportal/general/index.jsp repo: https://gitlab.com/is-enes-cdi-c4i
ES-DOC	*ES-DOC service and documentation*: http://es-doc.org CIM repo: https://github.com/ES-DOC/esdoc-cim-v2-schema pyesdoc repo: https://github.com/ES-DOC/esdoc-py-client CMIP6 content repos: https://github.com/ES-DOC-INSTITUTIONAL cdf2cim repo: https://github.com/ES-DOC/esdoc-cdf2cim
Climate Forecast (CF)	service: http://cfconventions.org/ doc: http://cfconventions.org/ repo: https://github.com/cf-convention/
CMIP Data Request	service: http://clipc-services.ceda.ac.uk/dreq/ doc: http://w3id.org/cmip6dr repo: https://pypi.org/project/dreqPy/
Metadata for Climate Indices	Proposed metadata specification for climate indices, repository: https://bitbucket.org/cf-index-meta/cf-index-meta/src/master/
Identity Management and Access Entitlement	doc: https://github.com/ESGF/esgf.github.io/wiki/Security%7CInterfaceControlDocument service: Attribute and Authorisation Services repo: http://esgf.org/esgf-security/ service: OpenID Provider and Relying Party repo: http://esgf.org/esg-orp/ service: OAuth 2.0 and Short-lived Credential Service repo: https://github.com/ESGF/esgf-slcs-server

5. DESIGN OF THE ENES CDI SOFTWARE STACK COMPONENTS

In the following, a detailed description of the design activities carried out for each ENES CDI software component is presented. The aim of each subsection is to propose a suitable solution matching the requirements collected in the milestone M10.1 and summarized in Section 3.
For each ENES CDI module, a UML component diagram is proposed, highlighting its main physical blocks, the required and the provided interfaces and the connection with the other components; additionally, some preliminary development activities are also reported as part of the work done during the first period. The colors adopted refer to how the component is distributed on the three levels of the general architecture of the ENES CDI software stack reported in Section 4.1: (i) pink for the Fabric service layer, (ii) cyan for the Federation service layer and (iii) yellow for the Application layer. The white blocks are external modules involved in the component workflow.

5.1. ESGF Data

The core ESGF data services are provided by a set of interconnected yet separately [NFR#6] deployable components which provide the basic functionalities for data search [DATAFR#2] (search API and Solr[13] data catalog) and data access [DATAFR#3] (THREDDS Data Server[14] for HTTP access as well as GridFTP server for Globus[15] access). Based on a Solr slave master deployment architecture and the associated index replication, the search service provides a consistent view for the whole available ESGF hosted data collections. As envisioned in the ESGF Future roadmap, future work will relate to a single centralized index node architecture to be deployed in the cloud.
To provide a consistent AAI (Authentication and Authorization Infrastructure) service layer, currently different components are involved: MyProxy servers, OpenID[16] identity providers as well as access filters securing applications [NFR#5] and interacting with these components. The AAI layer will undergo a complete update and redesign based on state of the art technology (using OpenID Connect and OAuth 2.0) during the IS-ENES3 project, see Section 5.14 for details. This will also affect the portal component (CoG, though this is planned to be completely replaced), which provides a unified web access point to search for data and trigger data downloads from the data servers. For data managers, a dedicated data publisher component [DATAFR#1] is responsible for making data visible and accessible in the ESGF infrastructure. All these components are summarized in the following component diagram (Fig. 3) also showing the core interfaces and component interplay. A special data node setup is used to

13 https://lucene.apache.org/solr/
14 https://www.unidata.ucar.edu/software/tds/current/
15 https://www.globus.org/
16 https://openid.net/

integrate the IPCC Data Distribution Center at DKRZ (Section 5.4). In this case the data is stored on tape and a virtual file system layer enables the integration of the data as part of an ESGF data node, and thus no THREDDS based data access layer is provided.

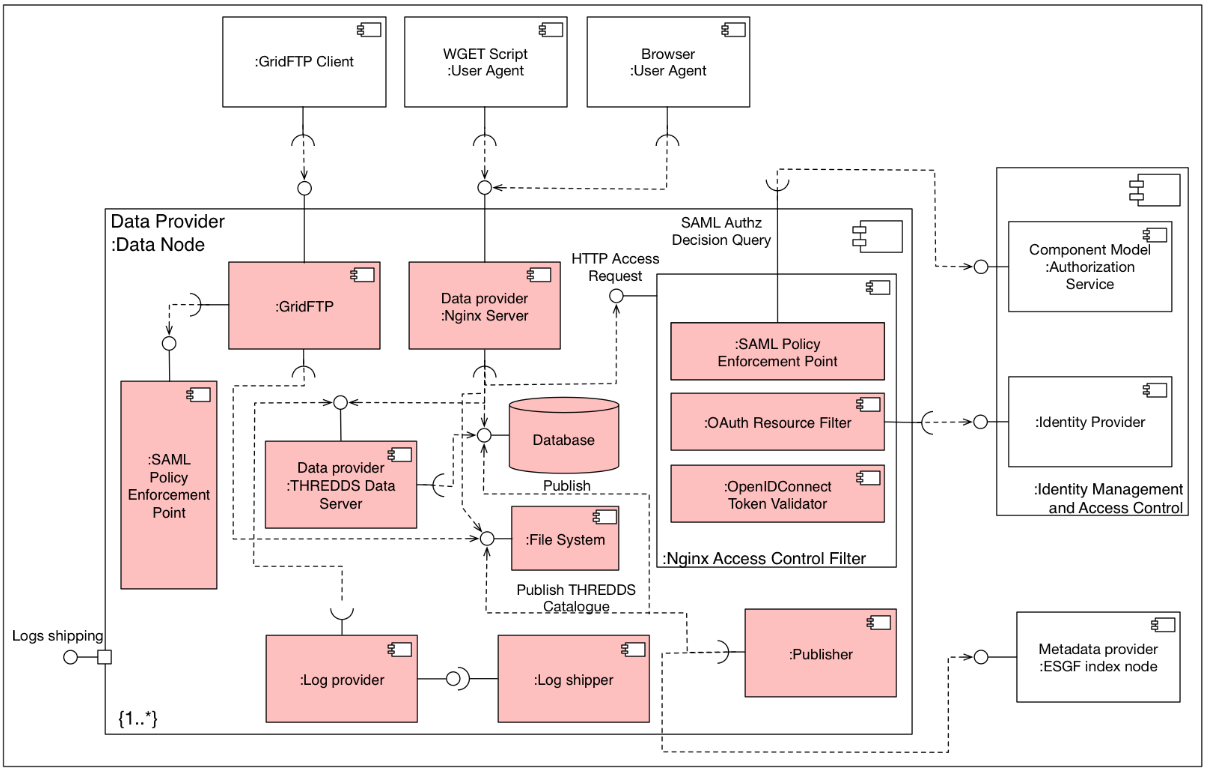

Fig. 3: ESGF Data component diagram

The ESGF data distribution service acts as a relay service connecting the ESGF data providers (climate modeling centers) with the data users (other infrastructures, climate modeling community, climate impact community and other downstream communities). The basic interaction scenario is illustrated in the following flow chart (Fig. 4):

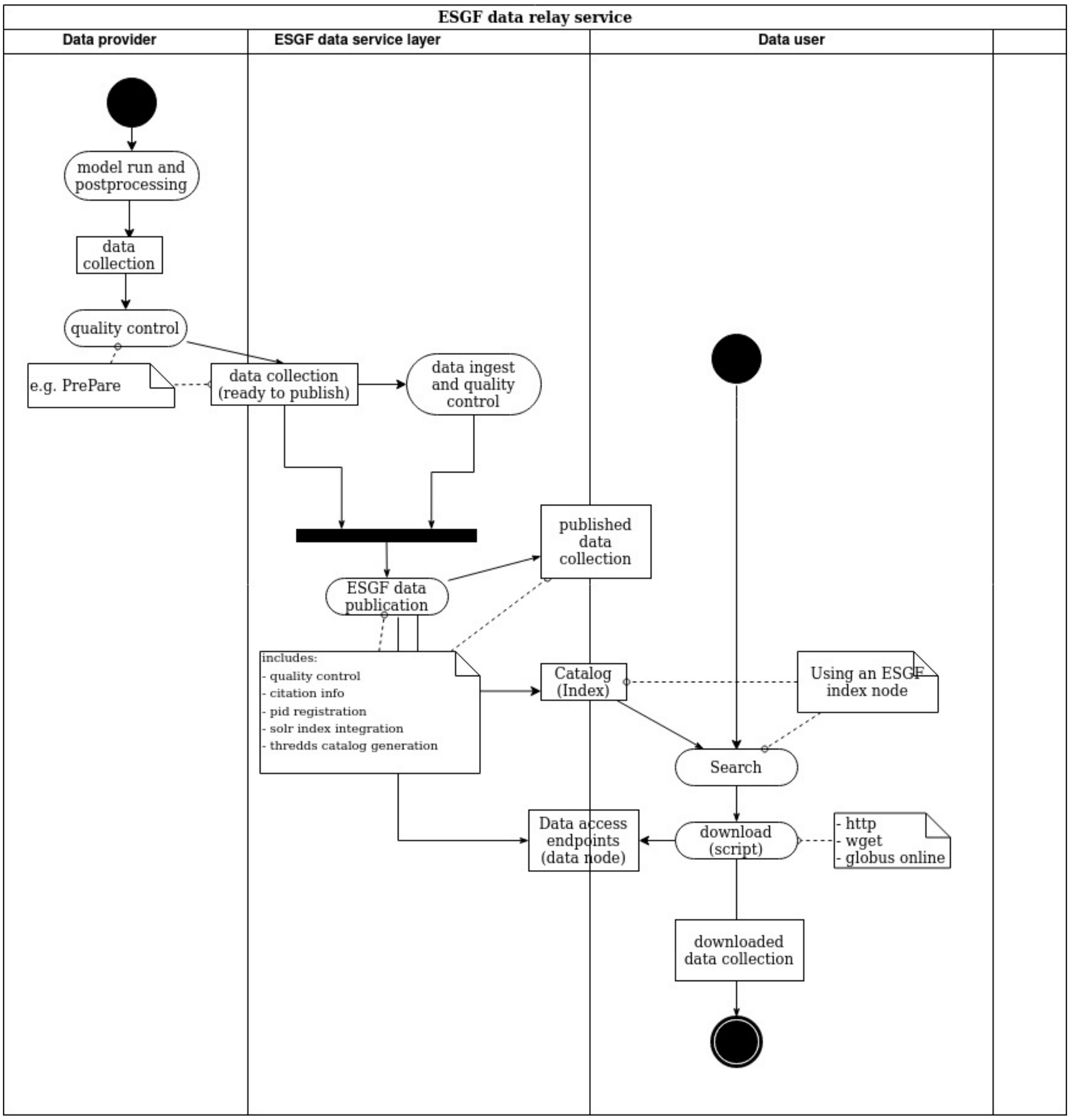

Fig. 4: ESGF data relay service flow chart

The ESGF data publication (in the center of the flow chart) is preceded by mandatory data preparation steps (including data quality control) on the data provider side and is thus included in this chart. For CMIP6 ESGF data publication, for example, data producers have to adhere to the requested CMIP6 data format standards and requirements [NFR#11] (e.g. data has to pass all the tests of the PrePare data quality control software).

In the ESGF data publication step, data gets ingested in the search catalog and thus made accessible via different supported access mechanisms (HTTP, Wget and Globus) [NFR#9]. In addition (for specific projects like CMIP6), the data publication step includes additional data indexing and information association steps:

- associating data with data citation information
- registering persistent identifiers (which also enables the association of datasets with errata information)

Therefore, the data publication needs to be configured specifically for the individual data project to be supported by ESGF.
The basic end user interaction with the ESGF service stack is illustrated on the right-hand side: users access the catalog, select data based on predefined search facets and then can directly access the distributed data access endpoints involved via different supported protocols and access mechanisms.

5.2. Data Citation

Data Citation has become an integral part of scholarly publications. Initiatives like COPDESS[17], ESIP[18], FORCE11[19] or Scholix[20] work on standardizations and guidelines for data citations. IPCC WGI[21] of the current IPCC cycle integrates data citations in the AR6[22] to improve the traceability and transparency of the key findings of the climate assessment. In order to enable the citation of CMIP6 data, the data has to be provided for humans as well as for machine-access [NFR#12]. Key is also to disseminate the information about CMIP6 data references outside the project context[23].

17 https://copdess.org/
18 https://www.esipfed.org/
19 https://www.force11.org/
20 http://www.scholix.org/
21 https://wg1.ipcc.ch/
22 https://www.ipcc.ch/assessment-report/ar6/
23 http://cmip6cite.wdc-climate.de

5.2.1. Functionality

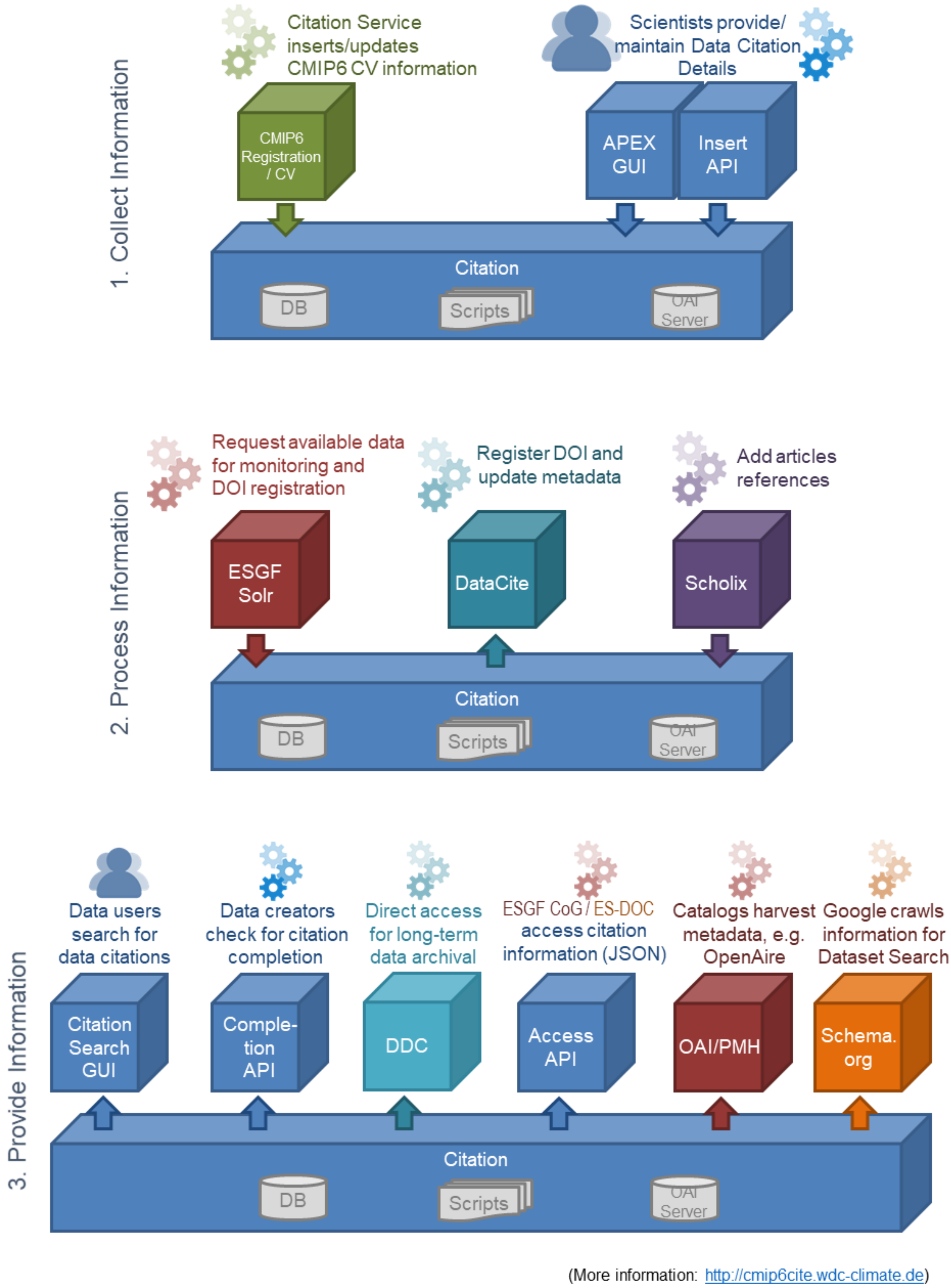

Fig. 5: Data Citation functionalities

Fig. 5 shows three different functionalities for the Data Citation service:

- **Collect Information:** Information on data citation and general CMIP6 information are provided by the Citation Service team and the CMIP6 participants.
    - Registration Information (Citation Service team): The DRS entries in the citation database together with information on the registered models are inserted based on the registered information in the CMIP6 Controlled Vocabulary[24] [CITFR#1]. A Climate and Environmental data Retrieval and Archiving (CERA) user account is provided for each citation manager.
    - Data Citation Information (CMIP6 Participants): CMIP6 participants provide information on authors (incl. affiliation and ORCID[25]), title and paper/data references for the citation entries. A GUI[26] based on Oracle APEX[27] and an API are provided for maintaining the citation information [CITFR#2].
- **Process Information:** Provided citation information is processed to register DataCite[28] DOIs and data DOI usage is added based on Scholix.
    - ESGF Solr/Search API: Requests of available CMIP6 data in the ESGF are used for metadata curation, quality checks and to check one of the preconditions for the data DOI registration.
    - DataCite Metadata Store (MDS): For completed data citation entries with available ESGF data a DOI is registered and for changes in the citation information the DataCite information is updated.
    - Scholix Hub: Data usage is regularly checked via a Scholix request and added to the metadata.
- **Provide Information:** Different users need different interfaces to access data citation information [CITFR#3].
    - Citation Search[29] for data users
    - Completed Citation API for ESGF data node managers
    - Access Citation API (ESGF CoG, ES-DOC furtherInfo): Complete data citation information as provided on the DOI landing pages is accessible in JSON via the Access API
    - OAI/PMH[30] (external catalogs): DataCite XML documents are accessible for harvesting on DKRZ's OAI server.
    - Schema.org/JSON-LD[31]: content is provided in the meta information of the DOI landing page
    - DDC: The DDC accesses the citation information directly from the database and adds this to the data documentation of the AR6 Reference Data Archive.

---

24 https://github.com/WCRP-CMIP/CMIP6_CVs
25 https://orcid.org
26 http://cera-www.dkrz.de/citeXA
27 https://apex.oracle.com/it/
28 https://datacite.org/index.html
29 http://bit.ly/CMIP6_Citation_Search
30 https://www.openarchives.org/pmh/
31 https://json-ld.org/

5.2.2. Standards/Consistency and Monitoring

The citation service uses DataCite's metadata standard and controlled vocabularies for the exchange of data citation information. For dissemination, the technical interface standards OAI-PMH and schema.org are used.
It is recommended to provide ORCIDs for every author and paper references for more in-depth information around the data.
The consistency with other information providers among the CMIP6 infrastructures will be granted by usage of the DRS to identify data citation entries and usage of the CMIP6 CV for the DRS components.
Basic automated curation of metadata content against citation requirements and completeness checks will be done on a regular basis.
Monitoring of basic technical components is done via icinga[32].
All automated procedures are logged. For operations changing database content the updated/inserted content will be logged. All metadata files registered at DataCite are stored within the database with identification information and timestamp [NFR#2].

5.2.3. Architectural design

The diagram in Fig. 6 shows the designed architecture in terms of components, provided and required interfaces, ports, and relationships between them, to highlight the service-oriented architecture of the system.

[32] https://icinga.com/

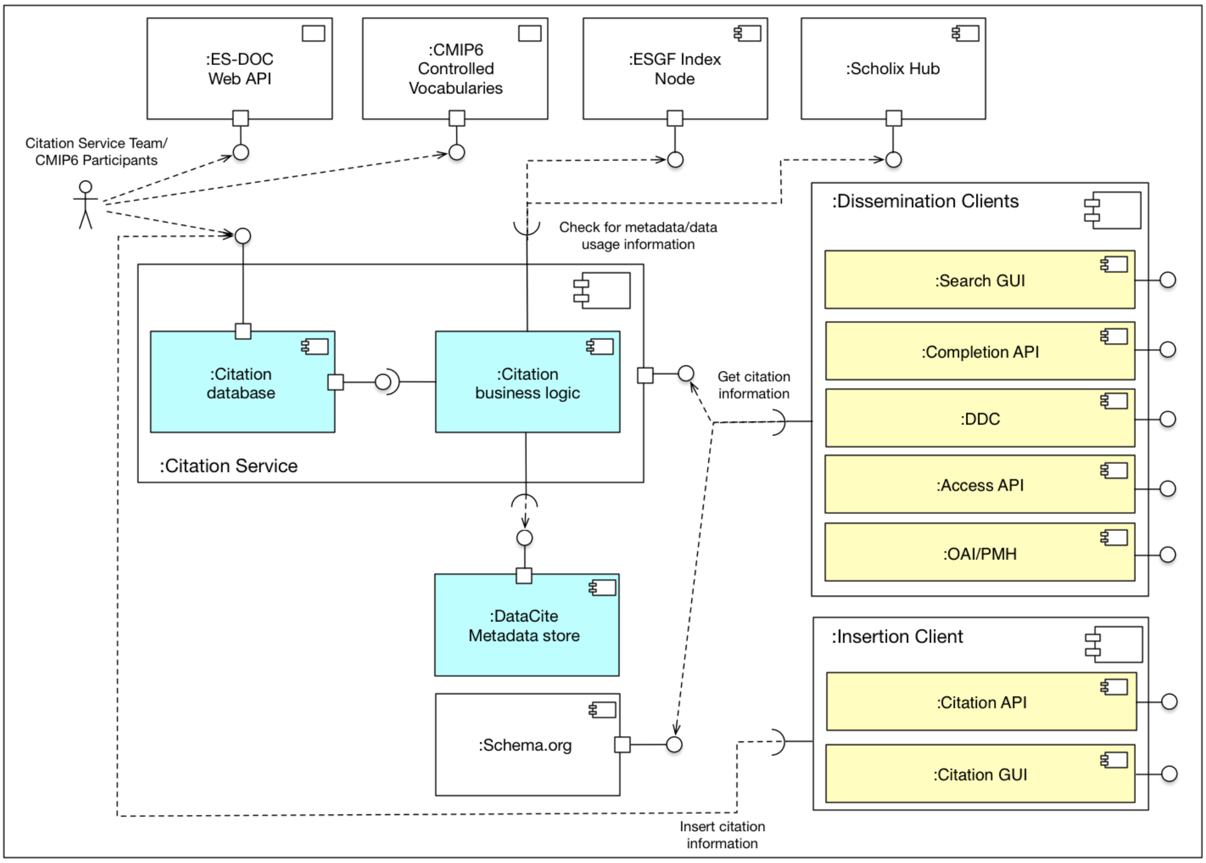

Fig. 6: Citation service component diagram

5.3. Persistent Identifier

The persistent identifier infrastructure aims to support the ESGF CMIP6 data management (and future CMIP projects) based on the experiences made for CMIP5. The overall concept and design was documented in the "CMIP6 PID implementation plan" WIP document[33].
The key requirements, additional to those listed in the Milestone 10.1 "Technical requirements on the software stack", and associated design decisions are summarized in the following Table 5 before describing the PID infrastructure architecture in detail:

Table 5: Persistent Identifier additional key requirements

Requirement	**Architectural design decision**
reliable PID registration delivery	Distributed RabbitMQ[34] message delivery infrastructure
highly scalable PID registration [NFR#3]	multiple relay queues set up in different

33 https://cog-cu.colorado.edu/site_media/projects/wip/CMIP6_PID_Implementation_Plan.pdf
34 https://www.rabbitmq.com/

	locations (continents)
robust [NFR#2], sustainable PID backend solution	Using handle system backend software stack[35]
easy integration with the ESGF publisher component [NFR#11]	Development of a modular python PID library component (pip python package)[36]
highly scalable PID resolution [NFR#3]	setup of PID replication as part of the European ePIC[37] consortium and EUDAT. Handle PID mirroring.

The PID infrastructure is composed of the following core components:

- a (python) client library, which is integrated into the ESGF publisher, which interfaces with the PID registration message delivery infrastructure [PIDFR#1];
- the PID registration delivery infrastructure, which is based on a distributed message queuing system (rabbitmq) consisting of interconnected message relay and queuing components (deployed at distributed ESGF sites around the world) as well as a central consumer hosted at DKRZ;
- the consumer component at DKRZ interacts with the handle server(s) hosting and persisting PID metadata information in SQL database backends [PIDFR#2]. These handle servers are integrated into the European ePIC infrastructure providing replication and handling resolution resilience.

The overall composition of PID ESGF publisher component, rabbit message queue and handle system backend is illustrated in the diagram in Fig. 7.

[35] https://www.handle.net/
[36] https://github.com/IS-ENES-Data/esgf-pid
[37] https://www.pidconsortium.net/

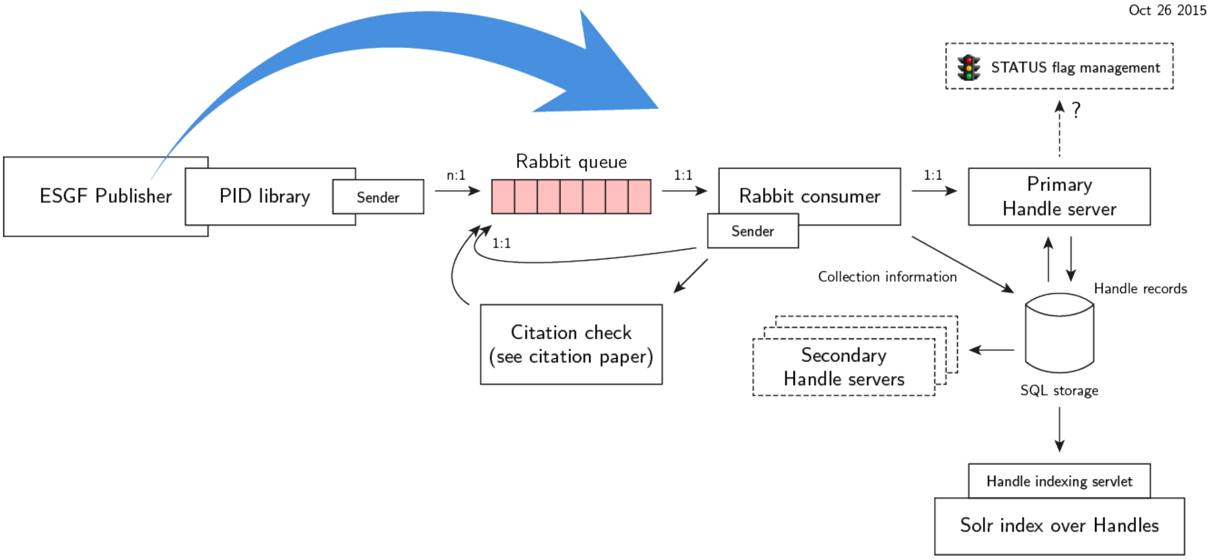

Fig. 7: Persistent Identifier architecture

The generic separation into client part (ESGF publisher integration), middleware part (distributed message queue) and backend part (handle system with message queue consumer integration) [NFR#6] is illustrated in the following component diagram (Fig. 8):

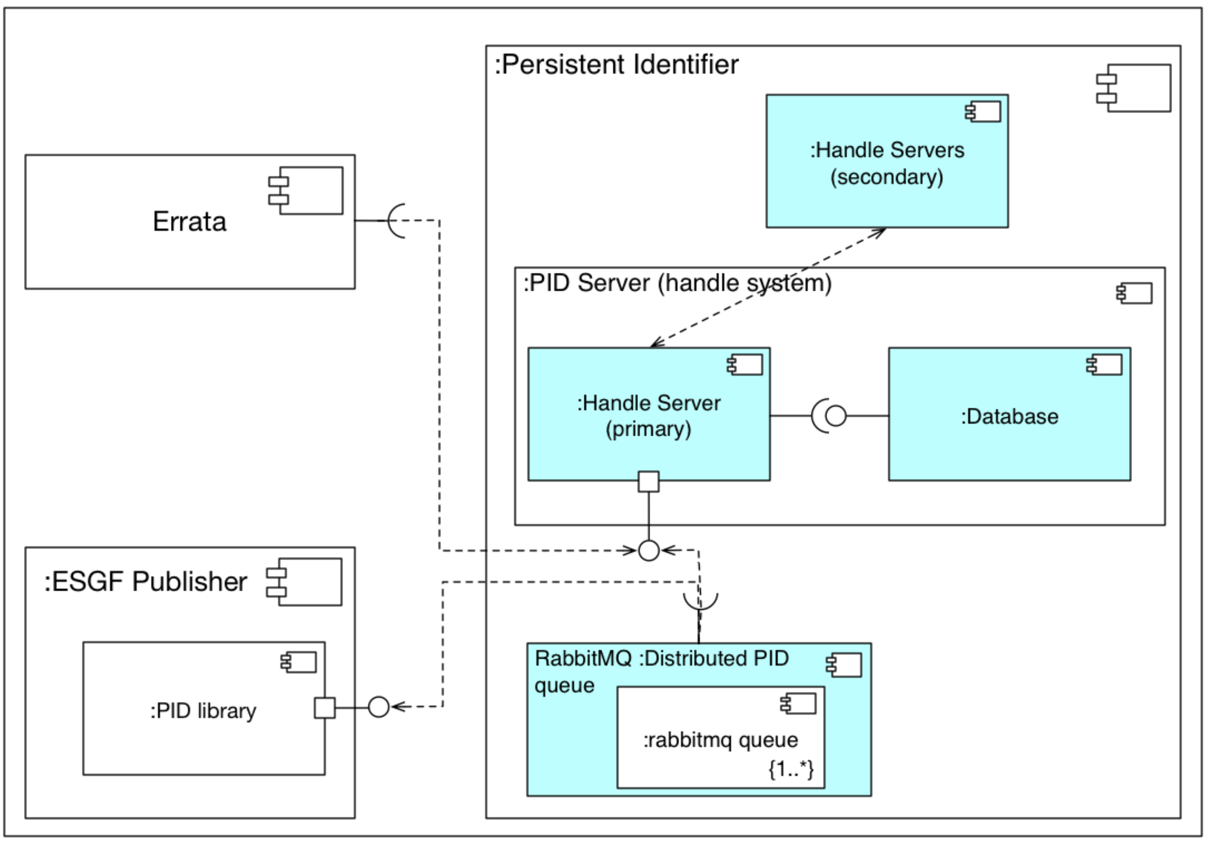

Fig. 8: Persistent Identifier component diagram

To provide stable and sustainable landing pages (for PID resolution) an additional component is involved, which is hosted as part of the DKRZ World Data Center for Climate (WDCC). It provides dynamically generated web pages ("landing pages") based on the content of the PID metadata information and additional (project specific) context information. As an example, the handle "hdl:21.14100/ebae2bd0-be37-3b1b-92a3-4d32672724e9" resolves via the generic handle resolver (see e.g. Handle.Net[38]) to the web page shown in Fig. 9 (supporting content negotiation, thus also supporting command line client interactions).

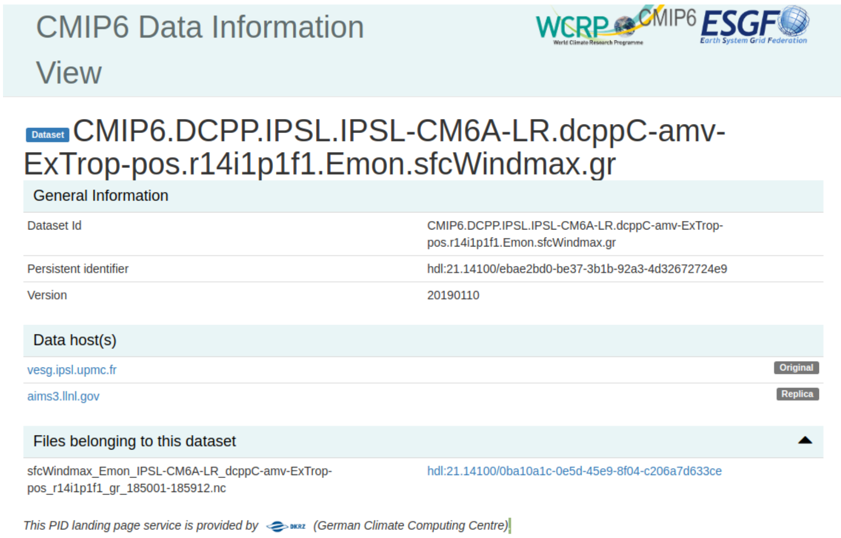

Fig. 9: Dataset landing page

5.4. IPCC Data Distribution Centre at DKRZ

The IPCC WGI is one of the early users of CMIP6 data. The IPCC DDC[39] supports the authors in the writing process especially in the analysis of data to derive key findings by providing Virtual Workspaces. The CMIP6 data subset underlying the AR6 will be long-term archived in the IPCC DDC AR6 Reference Data Archive as part of the traceability of AR6 key findings and as well as for data re-use. The quality requirements for the IPCC DDC data and metadata are high, complying to the TRUST principles (Transparency, Responsibility, User Focus, Sustainability, Technology) as implemented in e.g. the Core Trust Seal[40].

38 http://proxy.handle.net
39 http://ipcc-data.org
40 https://www.coretrustseal.org/

5.4.1. Long-Term Data Archival Workflow

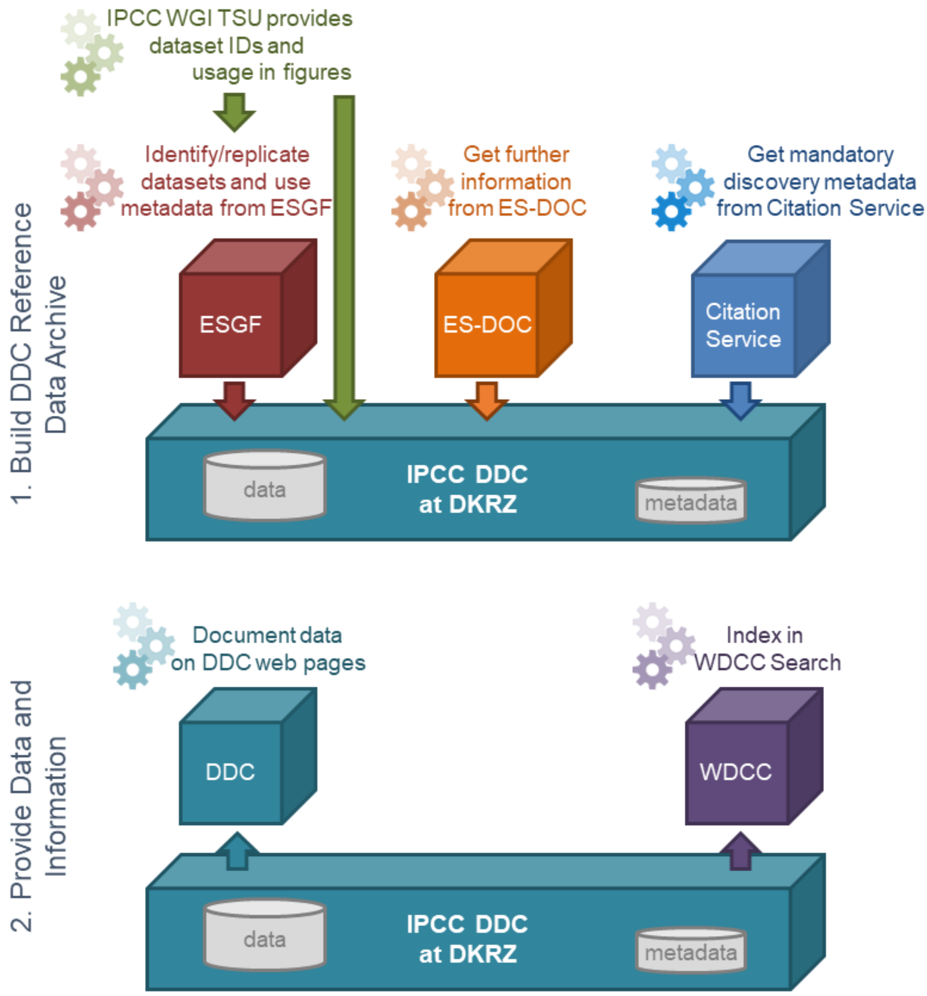

Fig. 10: CMIP6 data subset transfer process underlying the IPCC AR6 Part I

Fig. 10 shows the transfer process of the CMIP6 data subset underlying the IPCC AR6 Part I, in which data and metadata are gathered:

1. **Data workflow:**
   IPCC WGI Technical Support Unit (TSU) collects information on input data such as CMIP6, which will be used to identify the CMIP6 data subset for transfer in the IPCC DDC AR6 Reference Data Archive.
2. **Metadata workflows:**
   The basic documentation of CMIP6 data in the NetCDF data headers is complemented by documentation from other resources:
   2.1. ESGF index: ESGF index provides access to use metadata without processing every NetCDF file.

2.2. IPCC WGI TSU: The information about data usage in the figures of IPCC AR6 Part I will be added to the metadata for transparency.

2.3. Citation Service (see section 4.2): the Citation Service provides discovery metadata on project and model originating from the CMIP6 Registration[41] as well as information on data citation, author details, references to paper and data, and data licenses.

2.4. ES-DOC (see section 4.10): ES-DOC provides rich information on model, simulation, and experiment. It is planned to add this documentation to the metadata.

2.5. Further Metadata: the DDC investigates the inclusion of further information in the metadata, e.g. ESMValtool results.

All digital information will undergo a diligent quality assurance for completeness, consistency, and compliance to international standards (e.g. NetCDF/CF) and further project conventions[42]. As a final step, the data in the IPCC DDC AR6 Reference Archive are registered DataCite DOIs making the data citable.

5.4.2. Curation and Long-Term Data Provision

After long-term data archival, regular curation procedures are applied to ensure data FAIRness on the long-term and in compliance with the TRUST principles. These include:

- the technical maintenance of the infrastructure and
- the curation of metadata, e.g. check for newly published errata (section 4.5) for the AR6 Reference Data or check for new publications based on AR6 Reference Data via Scholix.

Continuous user support and regular modernization of hardware components will also be provided.

5.4.3. Architectural design

The Fig. 11 below shows the main blocks of the IPCC DDC at DKRZ in a component diagram, highlighting how data and metadata information gathered by the DDC is exposed through the IPCC DDC GUI [DDCFR#2] and the WDCC search [DDCFR#1], respectively.

[41] https://github.com/WCRP-CMIP/CMIP6_CVs

[42] The quality assurance content for the IPCC DDC AR5 Reference Data Archive is documented alongside the data and at: https://redmine.dkrz.de/projects/cmip5-qc/wiki/Qc_l3#Criteria-for-QC-L3DOI-publication

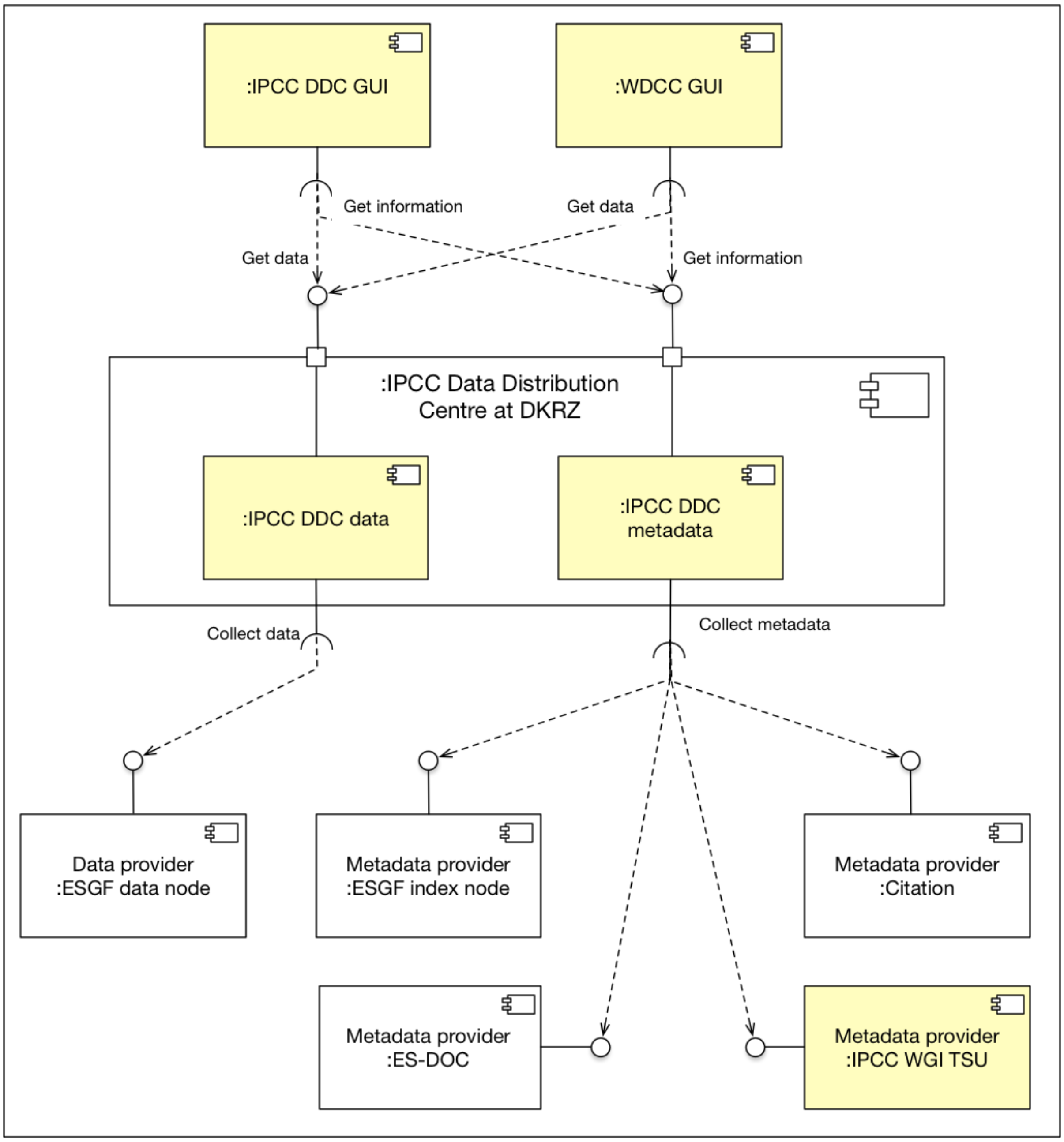

Fig. 11: Component diagram of the IPCC DDC at DKRZ

5.5. Errata

Because of the experimental protocol inherent complexity of projects like CMIP5 or CMIP6, it becomes important to record and track reasons for datasets version changes. Proper handling of errata information in the ESGF publication workflow has a major impact on the quality of data. Version changes should be documented and justified by detailing what has been updated, retracted and/or removed.

This chapter is an overview of the design and implementation of the Errata service, both on the back-end and front-end of the service, starting from a general view of the architecture and focusing on the issue mechanisms behind the front-end and the command-line interfaces.

5.5.1. Errata Service architecture

As a part of the ES-DOC ecosystem, the Errata Service offers a user-friendly front-end and a dedicated API to provide timely information about known issues affecting ESGF data (Fig. 12).

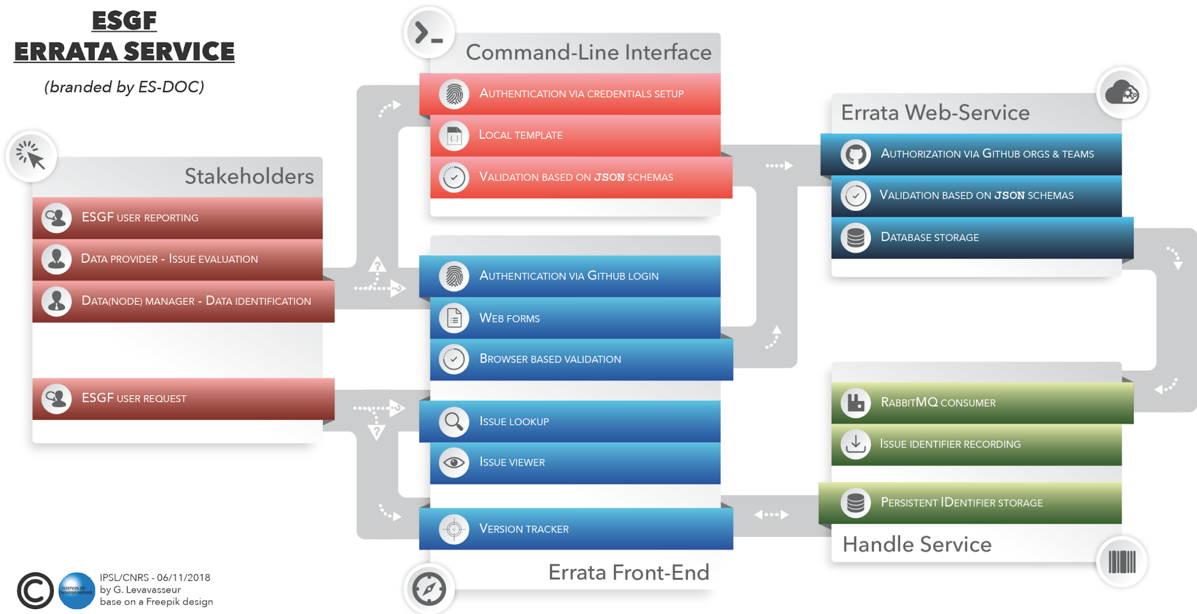

Fig. 12: Errata Service architecture

ESGF users can query about modifications and/or corrections applied to the data in different ways [ERRFR#2] [ERRFR#3]:

- through the centralized and filtered list of ESGF known issues[43];
- through the "PID lookup" interface[44] to get the version history of a (set of) file/dataset(s).

All the ESGF projects, with a pyessv (Python Earth Science Standard Vocabularies)[45] Controlled Vocabulary, are currently supported by the Errata Service.
Only identified and authorized actors of the corresponding modelling groups are allowed to create, update and close issues using either a lightweight CLI and/or a straightforward web-

43 https://errata.es-doc.org/
44 https://es-doc.github.io/esdoc-errata-client/lookup.html
45 https://github.com/ES-DOC/pyessv

form. In June 2018, the IPSL moved the ESGF Errata Service to production phase; the services are available at the official website[46].
The Errata Service makes good use of the Persistent IDentifier (PID) attached to each dataset and file during the ESGF publication process to persist the errata information [ERRFR#1]. The documentation[47] has been fully revised to guide users through the errata procedure [NFR#9].

5.5.2. Issue management

An issue on ESGF data can be detected by any user or any actor of the ESGF community. This issue has to be reported to the appropriate data provider. To that end, most of the ESGF NetCDF files have a contact attribute with the email of the corresponding climate modeling group. If this attribute does not exist, the usual ESGF users mailing list is used.
The issue will then be validated from a scientific point of view before its registration into the Errata Service. Thus, only identified and authorized actors of the corresponding modeling groups (see next section) can create, update and close issues.
The issue life-cycle can be managed in two ways [ERRFR#4]:

- a web form is directly accessible from the service home page once the user is logged onto the service;
- a command-line interface is available via the errata Python client. This can be installed with PyPI. The CLI requires two input files (i) describing the issue in a JSON syntax and (ii) listing the affected datasets.

Both rely on the Errata API described in Section 5.5.6.

5.5.3. Authentication and authorization

Authentication requires a verified (by email) GitHub account. Authorization is controlled using GitHub's organizations invitational based structure. The ES-DOC-ERRATA officers (designated in each group) and administrators are the only persons qualified to add GitHub users to the requested teams. For the authorization, a user needs to be part of the organization team specified for the institute and the project he/she on behalf of which wishes to publish issues [NFR#5].

5.5.4. Issue schema and requirements

Any given issue to be created on the Errata platform needs to adhere to a number of constraints/requirements (see Table 6). The idea behind these measures is to ensure a higher quality of the information stored and the scientific integrity of this important documentation.

[46] https://errata.es-doc.org/
[47] https://es-doc.github.io/esdoc-errata-client/

Table 6: Actions, parameters and requirements related to an issue

Action	**Parameter**	**Requirement**
Create issue[48]	Project name (eg CMIP6)	Mandatory & value controlled
	Title	Mandatory (16-255 characters)
	Description	Mandatory (16-1023 characters)
	Severity	Mandatory, possible values: [Low, Medium, High, Critical]
	Materials & urls	Optional, tested for validity
	Dataset list	Mandatory, cannot be empty
Update issue[49]	project name	Cannot be changed
	Description	cannot deviate more than 20% from original description
	Severity	Mandatory, possible values: [Low, Medium, High, Critical]
	Status	Mandatory and value controlled
	URLS & Materials	Optional, tested for validity
	Dataset list	Mandatory, can add or remove datasets. Cannot be empty.
Close issue[50]	project name	Mandatory

48 https://es-doc.github.io/esdoc-errata-client/create.html
49 https://es-doc.github.io/esdoc-errata-client/update.html
50 https://es-doc.github.io/esdoc-errata-client/close.html

	Title & Description	Mandatory
	Status	Mandatory, possible values: [Resolved, Wontfix]
	Severity	Mandatory, possible values: [Low, Medium, High, Critical]
	Materials & URLs	Optional, tested for validity
	Dataset list	Mandatory, cannot be empty

5.5.5. PID synchronization

When a new issue is created/updated through either the command line errata client or the web hosted form, the PID (Persistent Identifier) attached to every dataset declared with the issue is subsequently updated reflecting the changes.

This mechanism has been implemented in an asynchronous fashion as shown in Fig. 13. This leads to a fluid workflow for the errata creation/update, while the dirty work of making sure the errata information is properly persisted in the designated PID handles is performed independently through a sync cron job to ensure that it is failsafe [NFR#1]. This however also means there will be a slight delay between the errata information creation and when it will appear in the dataset and files handles. However, given the information will immediately be present on the index of the errata service, we deemed this trade-off is exactly what we need.

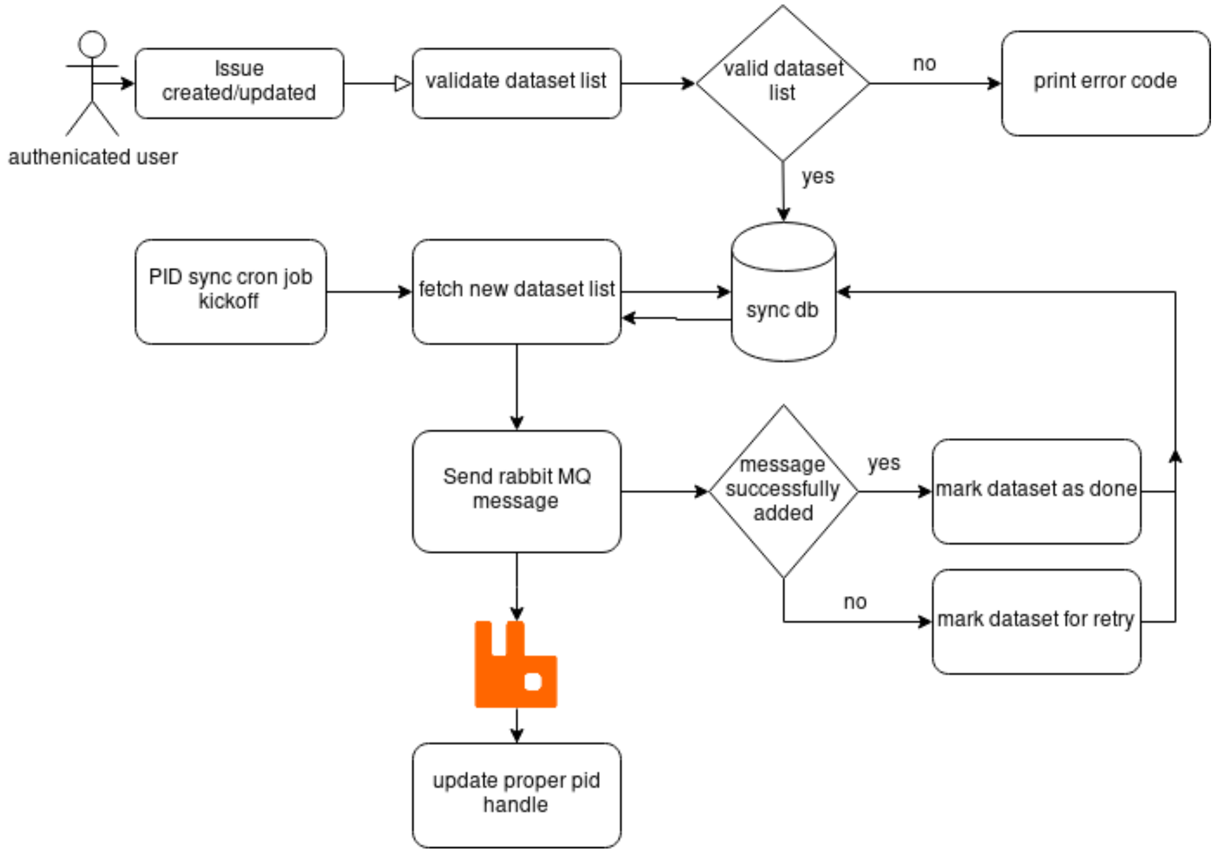

Fig. 13: Errata asynchronous PID integration via RabbitMQ

5.5.6. Application Programming Interface (API)

Most of the endpoints we have exposed to regular users/devs are the same used by the front-end and the command line interface. The errata API allows for interested parties to interact with the errata ecosystem programmatically [NFR#6], within the rules and safeguards mentioned above. Currently we offer the endpoints listed in Table 7.

Table 7: Errata service APIs

Endpoint	Consumes	Produces	Parameters	Authorization	HTTP codes
create issue path: **1/issue/create**	app/json	app/json	name: body desc: issue creation schema mandatory: true	org:read	200-401-403-405
update issue path: **1/issue/update**	app/json	app/json	name: body desc: issue update schema mandatory: true	org:read	200-401-403-405
close issue path: **1/issue/close**	app/json	app/json	name: body desc: issue json schema mandatory: true	org:read	200-401-403-405
retrieve	app/json	app/json	name: uid	-	405

issue path: **1/issue/retrieve**			desc: uids to retrieve mandatory: false		
pid search path: **1/issue/pid**	app/json	app/json	name: pids desc: pid handles mandatory: true	-	405
simple pid search path: **1/issue/simple_pid**	app/json	app/json	name: datasets desc: dataset string list mandatory: true	-	405

5.5.7. Architectural design

The component diagram in Fig. 14 shows how the main blocks are connected to each other and to the Persistent Identifier (described in Section 5.3) to properly handle the errata information.

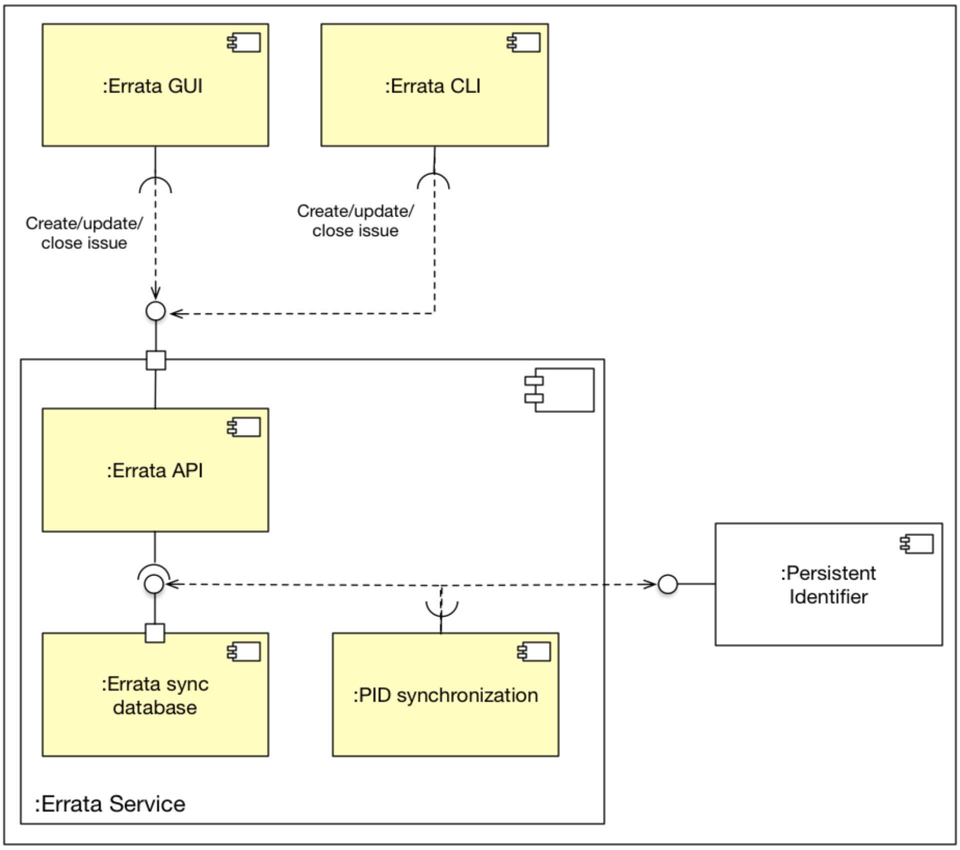

Fig. 14: Errata service component diagram

5.6. ESGF Data Statistics

In the following chapter an overview of the design of the ESGF Data Statistics system will be presented, both for back-end and front-end side, starting from a general view on the architecture and going down into the details of the data storage, the collection of local and federated statistics and the description of the front-end user interface.

5.6.1. Architecture in the large

In the following, Fig. 15 represents the ESGF Data Statistics general architecture.

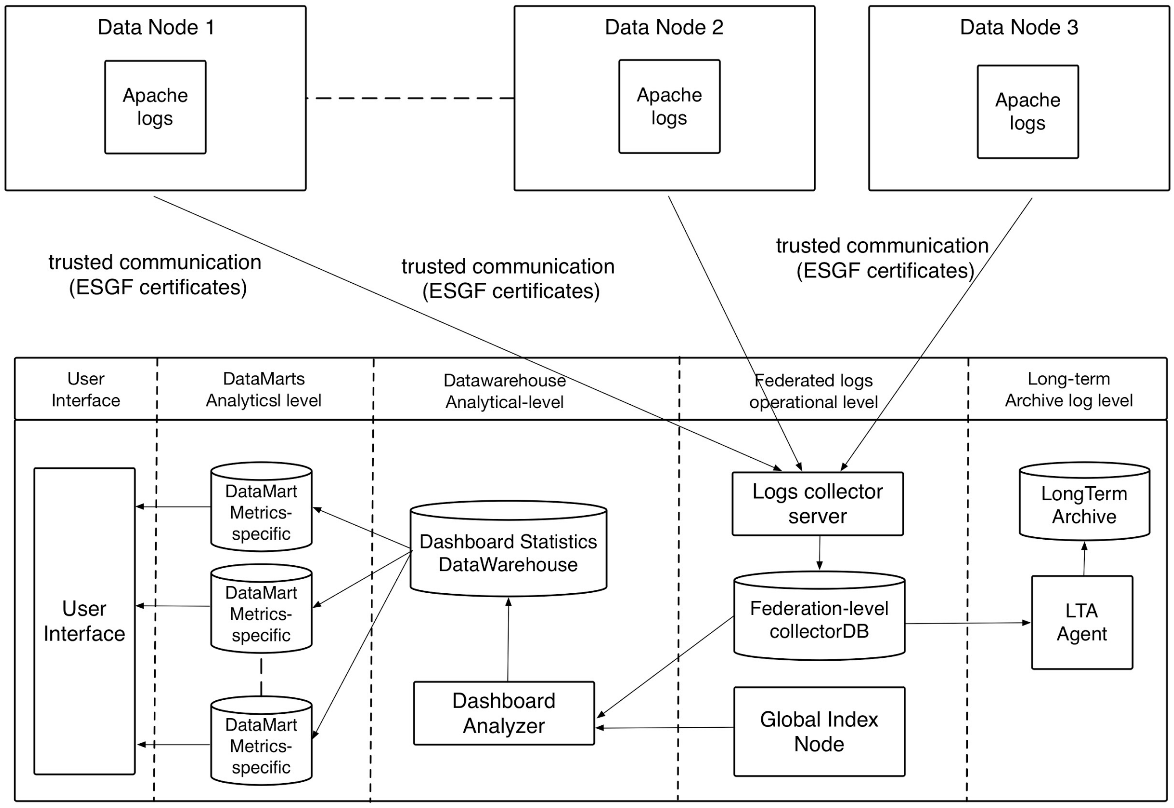

Fig. 15: The ESGF Data Statistics Architecture

The architecture relies on several components, starting from the Logging module to gather the basic logging information related to the downloads, to the Collector module, the Dashboard Analyzer and the data warehouse/data marts repository. The LTA Agent addresses the long term archival of the log info. The user interface is deployed at a central site and possibly mirrored, in the future, either in the cloud or at other ESGF sites [NFR#8] [NFR#13].

5.6.2. Architecture in the small

The sequence diagrams in Fig. 16 and Fig. 17 show the high-level interactions between the different components of the dashboard architecture both for the data usage and published data pipeline.

In particular, the first one describes how the system works starting from the download of a file by an end-user to the visualization of the download statistics through the user interface. When a download occurs, it is firstly logged on the data node and then sent to the collector node,

filtering out any sensitive information. On the collector node, the download logs will be properly processed through the interaction with the instance of Solr deployed on an index node and ingested into a data warehouse from which the user interface will get the statistics to be visualized.
The second diagram, instead, shows the workflow leading to the visualization of the metrics related to the data available through the ESGF catalog, starting from the publication of a file on the ESGF catalog. The dashboard analyzer periodically queries the Solr API to get information about the data published over the federation and ingests this information into a database from which the user interface will get the statistics to be visualized.

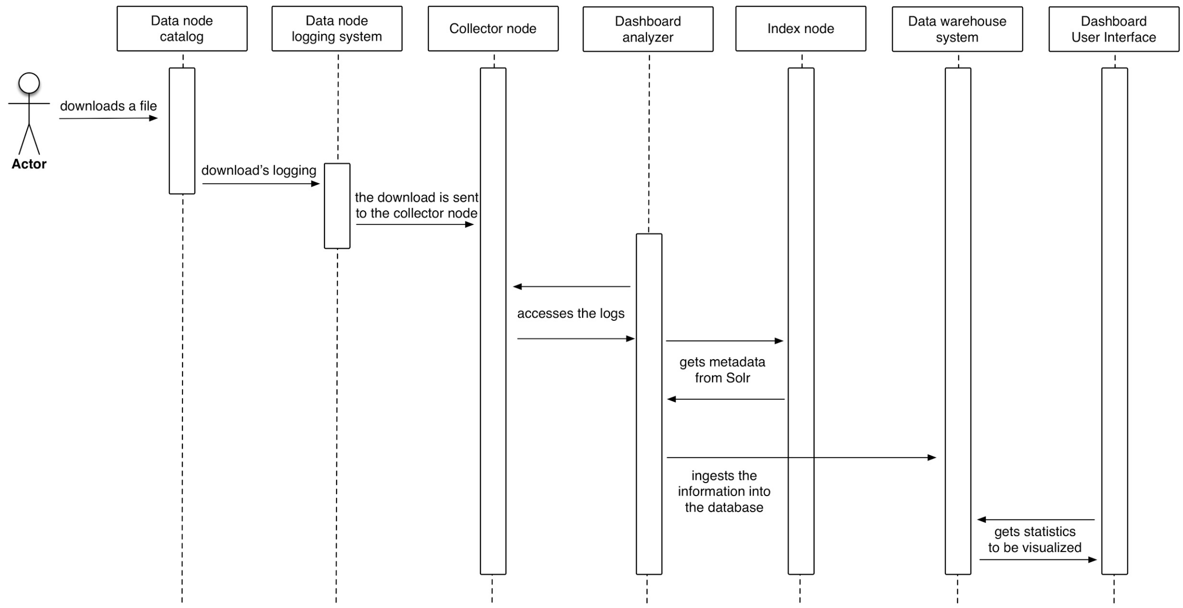

Fig. 16: Sequence diagram for the download workflow

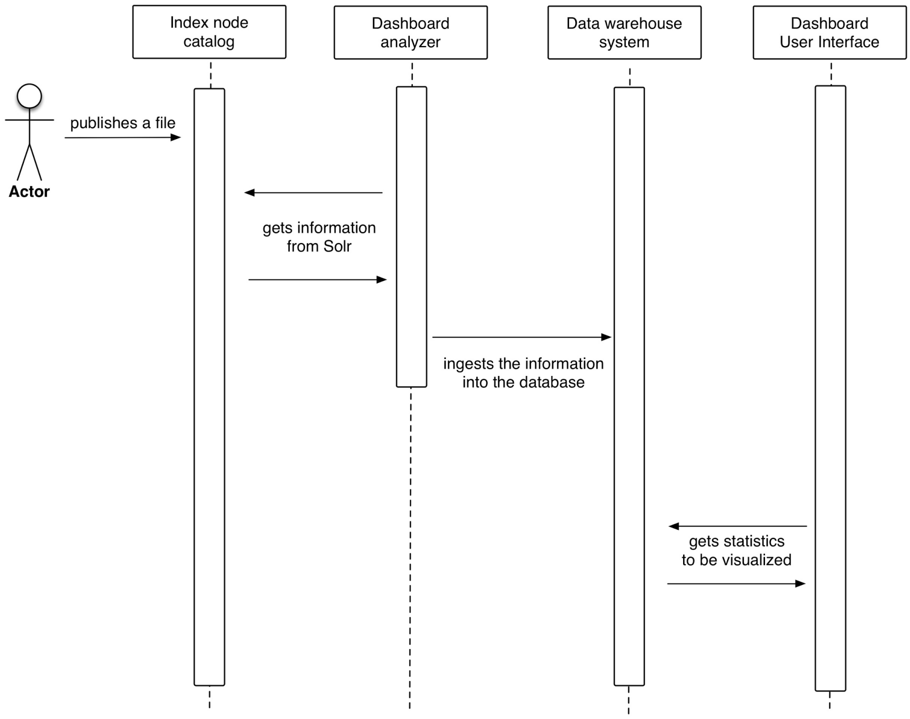

Fig. 17: Sequence diagram for the published data workflow

5.6.2.1. Logging and Collector modules

The logging information related to the downloads, gathered on each data node, will be sent by a log shipping software module to a centralized server acting as a log collector.
The access log contains information about requests coming to the web server. In this way, it is possible to track all the information about the downloads at each data node.
Sensitive information, such as IP addresses, cannot be transmitted to the central logs collector server. For this reason, a country lookup tool will be used to provide country-level information, which can be considered as an acceptable information level. Any communication between each data node and the logs collector server will rely on a secure connection to protect the information against network threats [NFR#5].
All the log entries sent by the log shipping agents installed on each data node will be gathered at the collector side. In particular, log events will be written to files on disk using a date-based log rotation and organized on a daily basis in the data archive.

5.6.2.2. Data warehouse system

The considerable amount of data usage information to be retrieved from each ESGF data node needs to be properly saved, managed and made available to be easily accessible in the future. To this purpose, a series of data warehouse systems have been designed to enable the collection

of an extended set of statistics about not only logging information but also project-specific download statistics and geolocation of the clients.
In particular, the following data management systems have been modeled:

a. a data warehouse containing general cross-project information;
b. a data warehouse for each relevant project in ESGF (e.g. CMIP5, CMIP6, Obs4MIPs[51], CORDEX, etc.);
c. a set of data marts for each data warehouse, which are simple specialized views on data, providing a more efficient access to key information required by the graphical user interface.

Cross-project Data Warehouse

The cross-project data warehouse contains general download information shared between different ESGF projects (Fig. 18).
The collection of cross-project download statistics addresses not only the requirement **[STATSFR#2]** but also the **[STATSFR#5]** one (Milestone 10.1 "Technical requirements on the software stack"), since a filter on the European data nodes will return the IS-ENES3 KPIs reflecting the status of the ENES infrastructure.

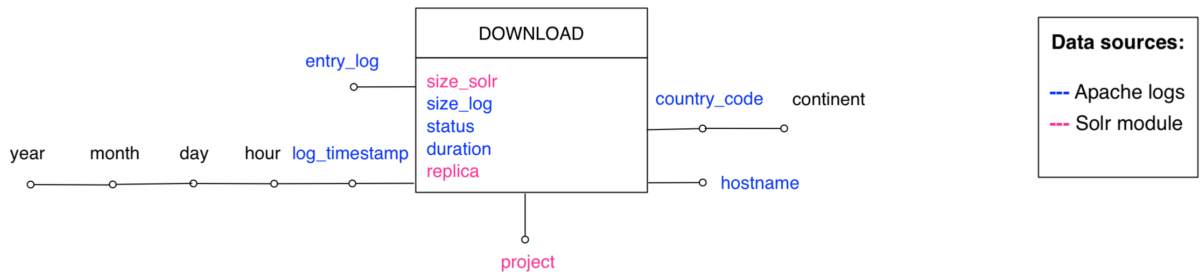

Fig. 18: Cross-project Dimensional Fact Model

The cross-project Dimensional Fact Model (DFM) highlights how the download operation is characterized by some relevant features. More specifically, the data warehouse keeps track of the following metrics:

- *size_solr*: the file size returned by the Solr instance;
- *size_log*: the size actually recorded by the logging operation;
- *status*: the status of the download (success or failure);
- *duration*: the time taken to download the file;
- *replica*: if the downloaded file is a replica of the original one or not.

These quantities can be considered from different points of views (for instance, over time, by project, by host or analyzed from a spatial perspective) with the aim to have a more accurate overview of the meaning of this information.

[51] https://esgf-node.llnl.gov/projects/obs4mips/

Project-specific Data Warehouses

The project-specific data warehouses aim to support the collection of project-related metadata bound to the data usage statistics. To this purpose, and to address the functional requirement **[STATSFR#3]**, with the collaboration of other ESGF partners, a number of relevant features have been defined for the projects CMIP5, CMIP6, CORDEX and Obs4MIPs.
The following diagram (Fig. 19) refers to the CMIP6 project, but the same approach has been followed also for the other project-specific data warehouses.

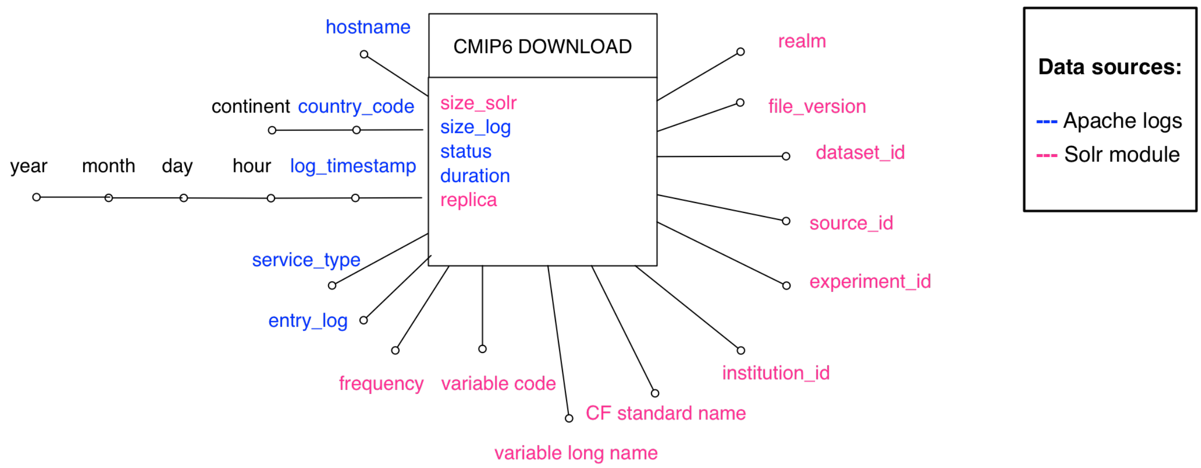

Fig. 19: CMIP6-specific Dimensional Fact Model

The main differences from the cross-project data warehouse rely on the information gathered from the ESGF Solr module, which allows the analyzer to enrich the data usage statistics with specific metadata (pink dimensions in the DFM diagram above).
These features increase the possibilities of data analysis providing the community with a helpful tool to better understand what is the most downloaded and, consequently, interesting data according to the most popular project-specific facets.

Data Marts

Data marts are aggregated views of the entire data warehouse system, focusing on some key information available and providing easy access to frequently needed data, in order to improve the response time of the web front-end [NFR#4].
A number of data marts have been modeled for each data warehouse system described above.
The metrics resulting from the aggregation process are:

- the total volume of the downloads;
- the total number of the downloads;
- the average duration of a download operation;
- the total number of downloads related to a replicated file.

In Fig. 20, a sample data mart of the CMIP6 project is shown, representing a view on the metrics analysed by variable over time. Similarly, other data marts are produced for CMIP5, CORDEX and other projects.

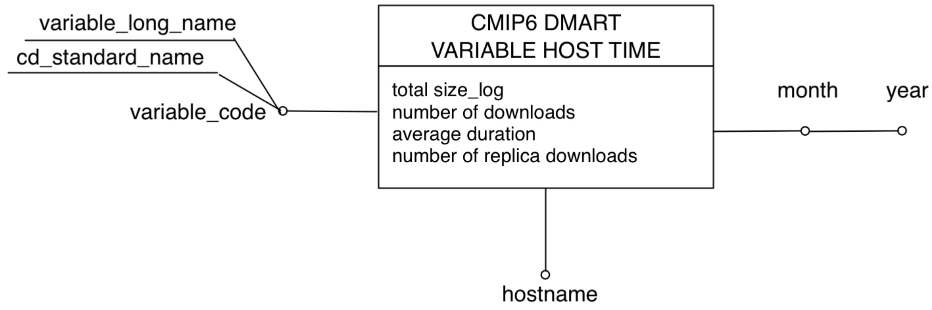

Fig. 20: CMIP6 data mart - view by variable over time

5.6.2.3. Data statistics analyzer

Metadata extracted from Solr

In order to deal with the considerable amount of log entries gathered from the ESGF data nodes, a data statistics analyzer software module will be developed to store the corresponding information in the proper data warehouse systems.

The following flowchart (Fig. 21) provides a diagrammatic representation of the main steps forming the processing algorithm.

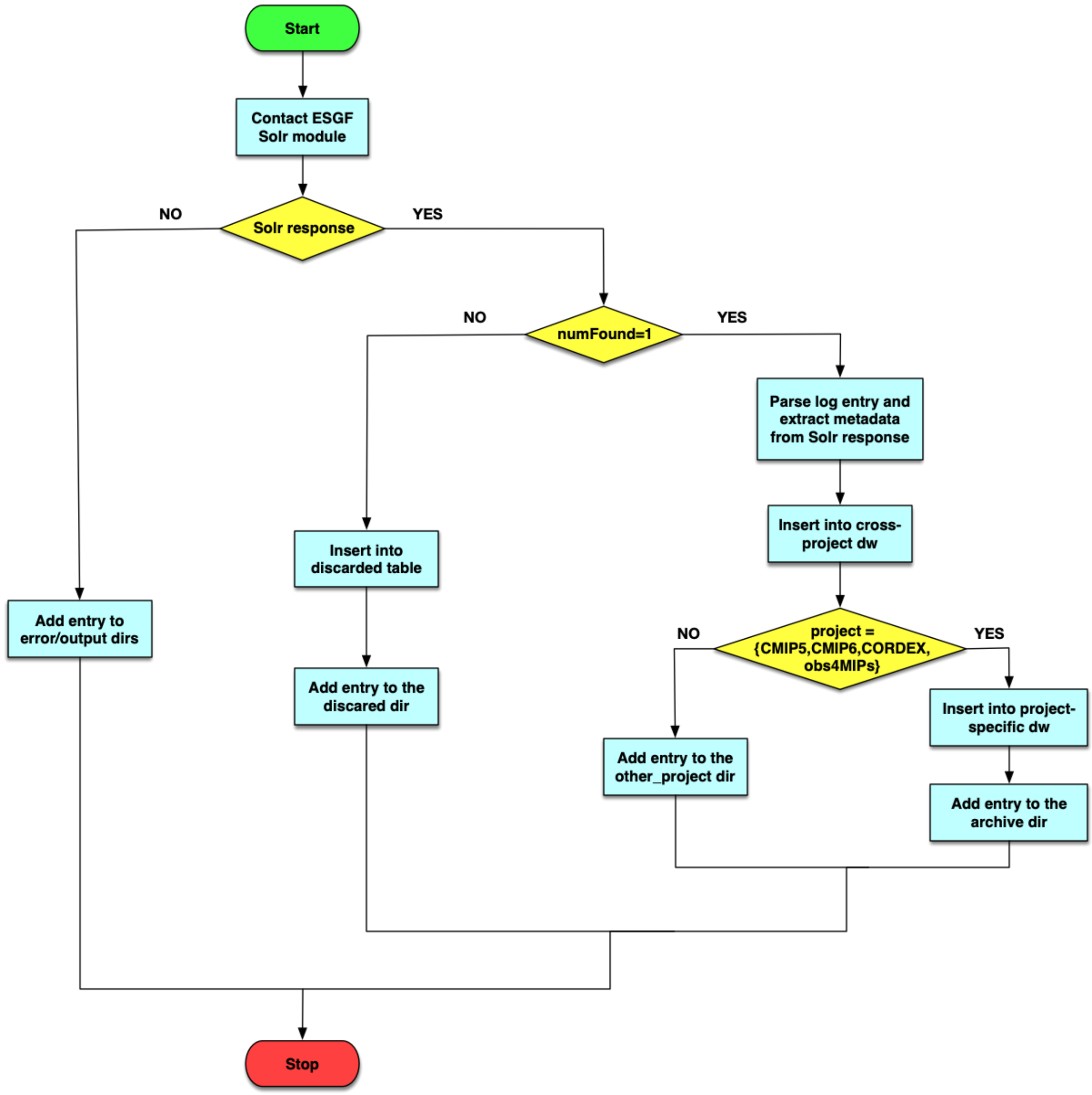

Fig. 21: Data statistics analyzer flowchart

For each log entry, the process will try to contact the ESGF Solr module in order to gather additional information such as the file size and other metadata closely related to the data which the file refers to (variable, experiment, model, …).
In order to face any possible failure on the storage or data warehouse systems, an appropriate backup mechanism will be provided to protect against data loss.

Data publication metrics from Solr

The metrics related to the data published are gathered directly from the Solr, available on the local Index Node instance, which indexes the whole ESGF data catalog.
An ad-hoc application will periodically query the Solr API to get information about the number of published datasets and the data volume, both for the whole federation and for the top

projects. The Solr response will be parsed by the application and the information will be stored into specific tables of the collector database.

5.6.2.4. Dashboard user interface

The Dashboard user interface [5] represents the last step of the whole Data metrics architecture and the main entry point for the final users who want to get information about the data usage and data publication metrics as well as the level of interest from the community in specific climate datasets, projects, variables, etc.
To meet the different user requirements, the data metrics service will provide an integrated point of access to visualize statistics at single site, ENES and federation level [NFR#1].
Different categories of statistics will be made available through a set of simple and attractive graphical widgets [NFR#10] with the main aim to provide a comprehensive view to the data collected and aggregated through the previous steps of the metrics' chain.
The web application will be structured as a three-tier architecture, meaning that its main components will be distributed over three logical levels and they can be upgraded or replaced independently and deployed over different physical layers, and it will be implemented through the MVC (Model-View-Controller) design pattern with:
- the data layer (the model) that is the level hosting the database server where information is stored. In the specific case of the metrics user interface, the data layer will be managed and populated by the statistics analyzer component described in the previous sections;
- the presentation layer (the view) standing on the top level and displaying information on the data usage and publication in the form of a graphical user interface (GUI);
- the application layer (the controller) that will be pulled by the presentation layer and perform the core functionalities of the process interacting with the data layer.
Being a centralized web-application, the design of a usable and effective user interface required a strict interaction with the final users of the community in order to meet the functional and non-functional requirements and provide helpful insights to the end users.
In the following, the main sections of the user interface, structured to fulfill the user requirements identified in the Milestone M10.1, will be described. A mockup of the main views is shown in Fig. 22.

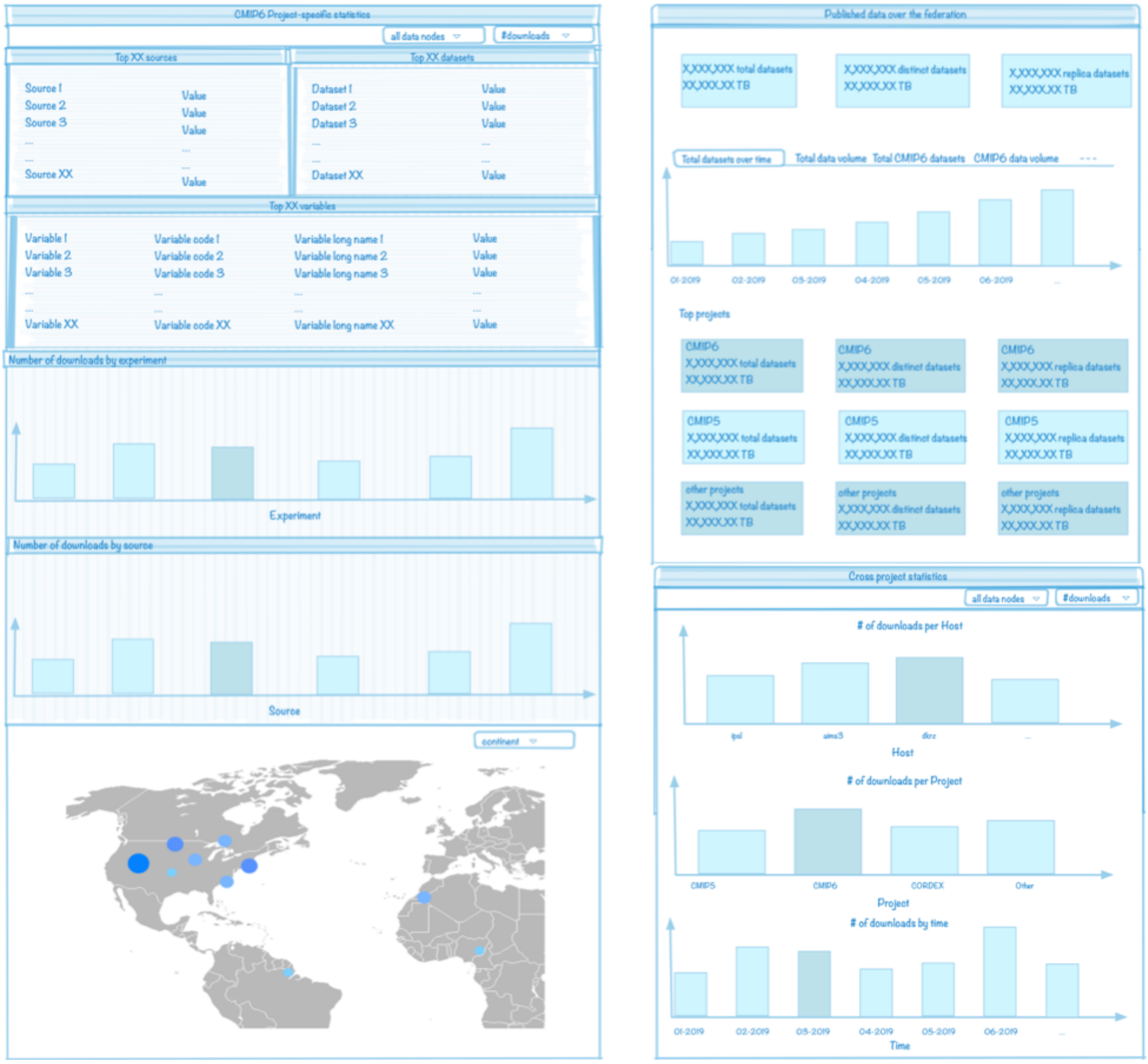

Fig. 22: Data Statistics main views mockups

Published data over the federation

The section will provide information about the ESGF data archive [STATSFR#1] in terms of number of available datasets and corresponding data volume, with a distinction between distinct and replicated datasets. Also, information for the main projects will be presented as well as a trend over time for the total amount of datasets and for specific projects datasets. Specific views will be provided to visualize data available for particular facets of the main projects.

Cross-project statistics

The Cross-project section will provide a set of views related to the cross-project download statistics [STATSFR#2] in terms of number of downloaded datasets and data volume over the whole federation and for specific data nodes. The statistics will be visualized in different widgets grouping the information by time and other relevant facets.

Project-specific statistics

Project-specific views will be available for the top projects [STATSFR#3] and will provide different analysis perspectives on the data, based on the most popular facets of a project.
Information as the number of downloaded datasets and the data volume will be available as an overview over the whole federation or for a specific data node. Also, geo referenced information about the projects' downloads will be provided, grouped by continents and countries.

IS-ENES3 KPIs

Information about the data usage statistics related to the IS-ENES3 data nodes will be available through a specific section [STATSFR#5]. Number of successfully downloaded files and the related data volume from European and non-European users will be visualized as well as the number of the European and non-European distinct clients that performed downloads.

Data statistics export

The system will provide the possibility for the user to easily export the data visualized through the different widgets in CSV format to be used for reports, documentation and further analysis [STATSFR#4].

5.6.3. Architectural design

The components diagram in Fig. 23 maps the designed architecture to a physical system that will execute it and, specifically, how the different tools and softwares will be distributed across the physical nodes.

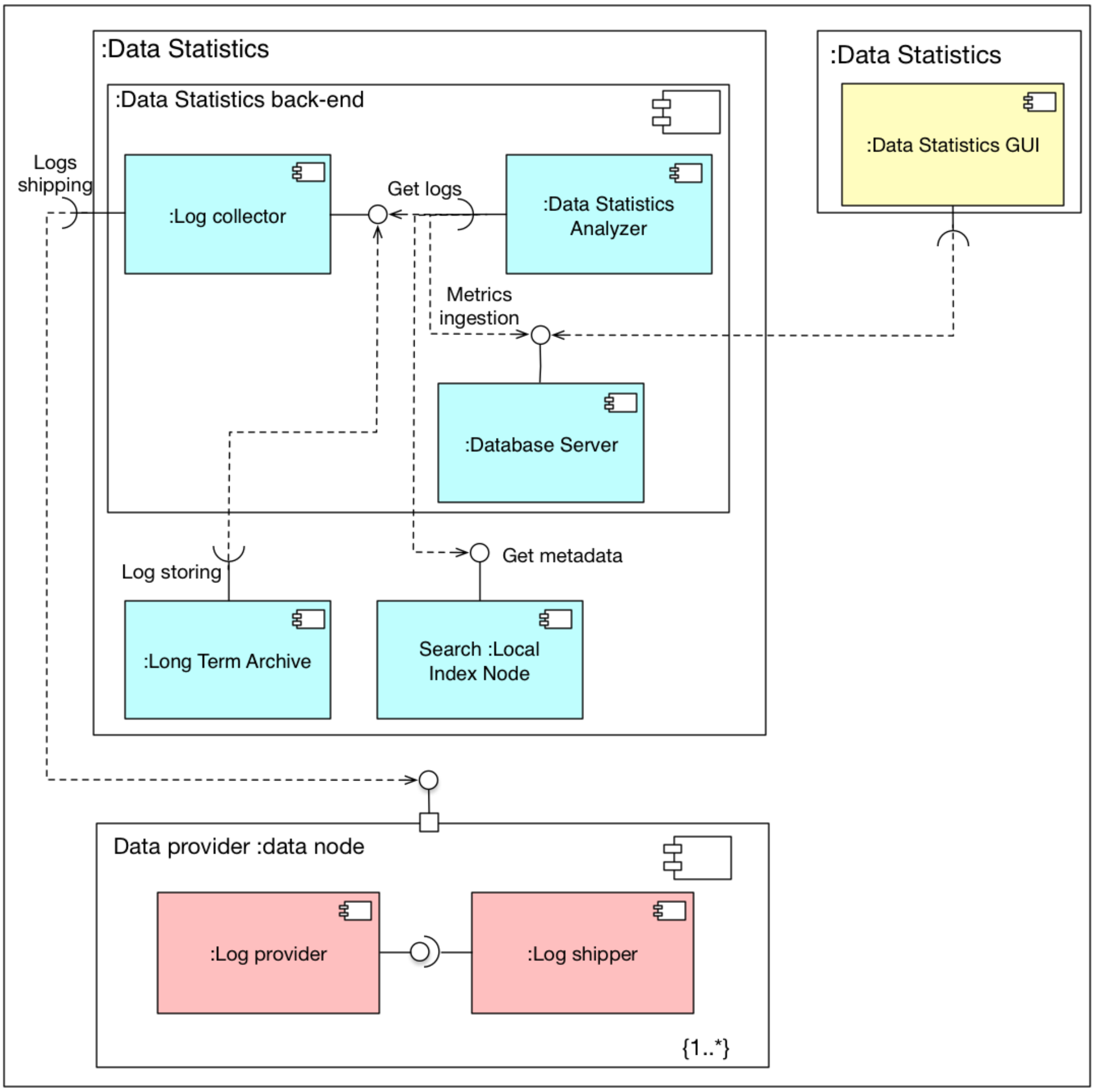

Fig. 23: Data statistics component diagram

5.7. Data Replication

Data replication involves the following architectural components:

- ESGF data nodes ("Tier 2") providing access to the originally published data collections from individual modeling centers;
- ESGF replica nodes ("Tier 1") providing access to original data as well as replicated datasets. These replica nodes are associated to larger data pools hosting replicated datasets and also to high performance data transfer nodes supporting Globus-based data transfer;

* a replication management software component ("Synda") hosted at replica nodes, which triggers and manages parallel data replication streams involving different data nodes and different transfer protocols [REPLICFR#2];
* a site-specific data ingest and publication workflow integrating the replica datasets in the local data pools and publishing these datasets via ESGF.

The interaction and deployment of these components are illustrated in the following diagram in Fig. 24:

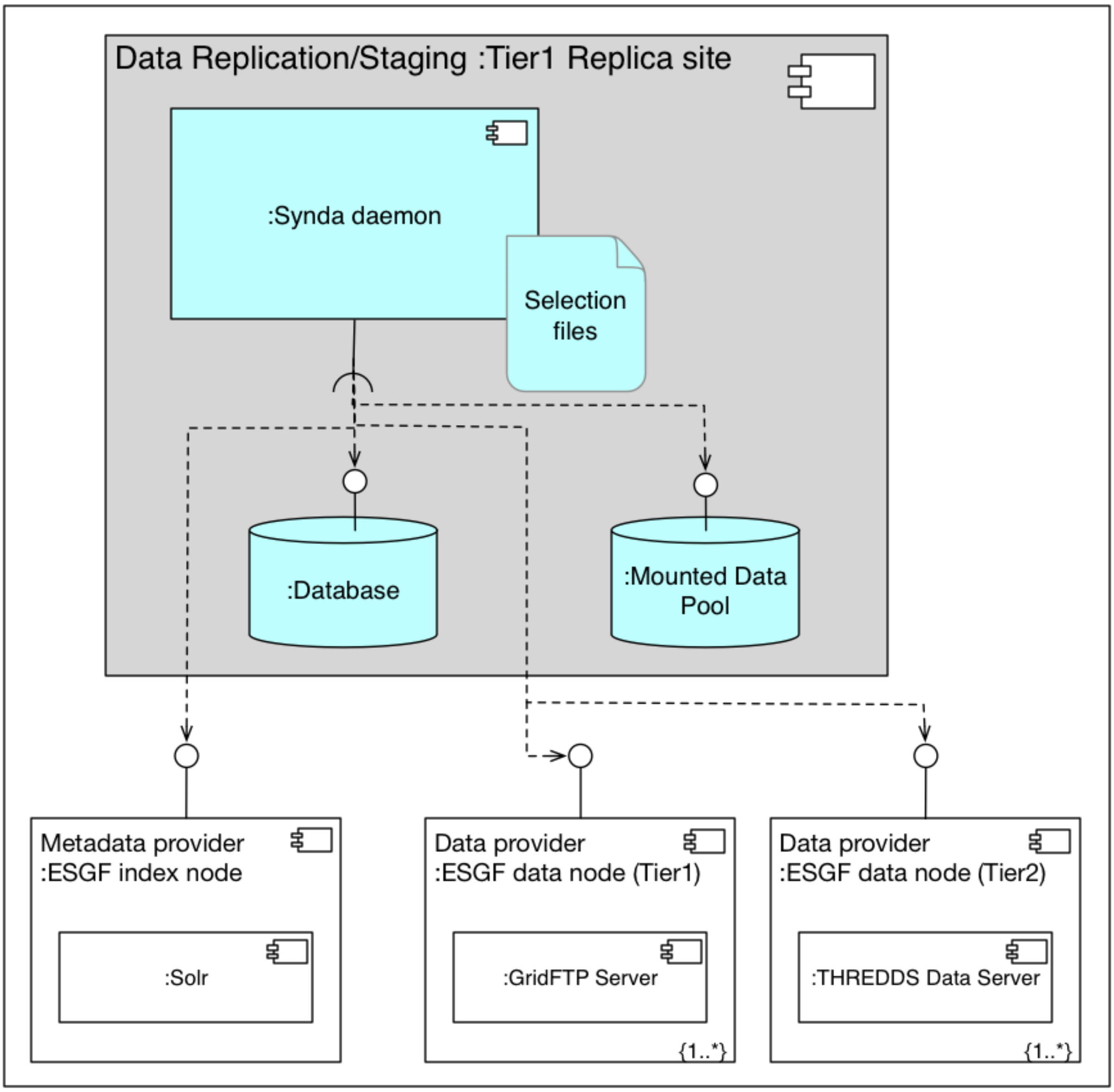

Fig. 24: Deployment and interaction of Data Replication components

The generic workflow involving these components can be summarized as follows:

* a replica center specifies the data collection to be replicated declaratively using so-called "selection files" characterizing the data collection based on specific search facet settings [REPLICFR#3];

- the selection file is provided to the component running the Synda transfer software component;
- Synda triggers parallel downloads from the involved data nodes (which are derived from interacting with the ESGF search interfaces based on the facet specifications given in the selection file);
- the status of the different downloads is persisted in a local database [REPLICFR#1]. Interrupted transfers are automatically retried [REPLICFR#4];
- the selection files are also used to keep the replicated data sets up to date in case of the availability of new versions;
- having a replica collection on disk, the local workflow then publishes this collection into ESGF, which makes this collection visible as “replicas” in the ESGF search interface.

For managing large replica collections, it is normally necessary to split the collections description (“Synda selection file”) into individual parts characterizing datasets belonging to specific organizational entities to enable a consistent overall organizational bookkeeping which is outside of the scope of the Synda software package.

Figures 25, 26 and 27 represent, respectively, the Synda software general view, the synchronous download workflow via Synda and the Synda asynchronous download workflow.

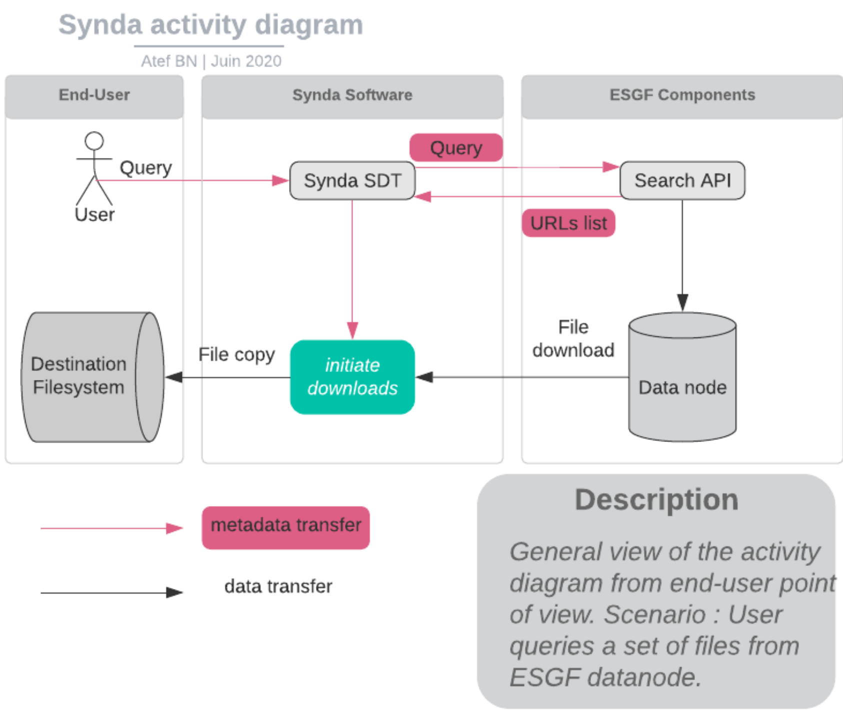

Fig. 25: Synda software general view

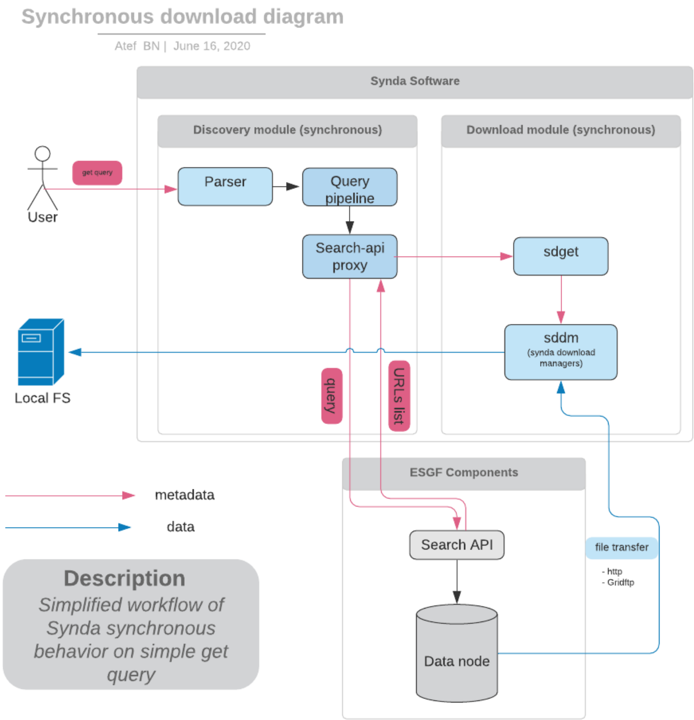

Fig. 26: Synchronous download workflow via Synda

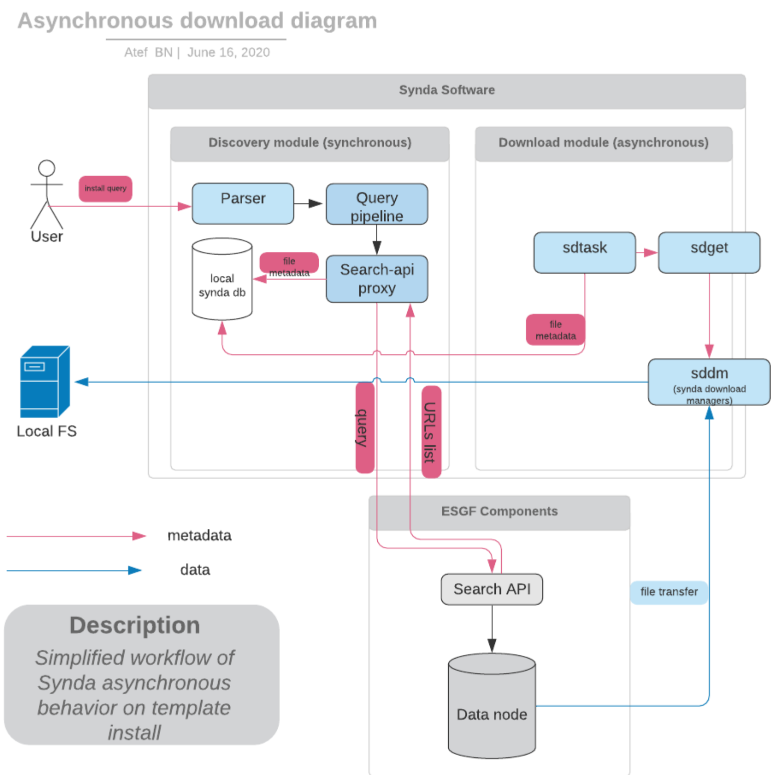

Fig. 27: Asynchronous download workflow via Synda

5.8. Compute & Analytics

A key activity of JRA3/WP10 is the development of a compute layer for processing and analytics of CMIP6 and CORDEX data. This activity targets the core part of the scientific data analytics and processing layer of the ENES CDI to fully address computing needs and move forward a sustainable and integrated data analytics and processing model.

From the users requirements gathering process, there is a wide spectrum of end-user needs to be addressed (e.g. multi-model analytics experiments, climate indicators, etc.) associated with a large user base of climate and impact studies researchers.

Over several telcos and also workshops (i.e. IS-ENES/ESGF Virtual workshop on Compute & Analytics, WP5), it was clear the design of the IS-ENES compute & analytics layer includes key aspects which should be part of any compute and analytics engine implementation and others which may be site-specific in order to address institutional/national needs.

In terms of common aspects, the compute service design must include:

- An interoperable and flexible server front-end based on the OGC-WPS interface [COMPFR#7] [NFR#11] [NFR#8];
- a programmatic client interface with a Python binding [COMPFR#6];
- a security infrastructure based on the work and roadmap defined with the ESGF IdEA WG activity [COMPFR#5].

The following diagram (Fig. 28) provides the architectural design of the IS-ENES3 compute services highlighting the three common aspects mentioned above, while site-specific aspects about the compute service solutions running at CMCC, DKRZ, UKRI and IPSL are reported in the following subsections.

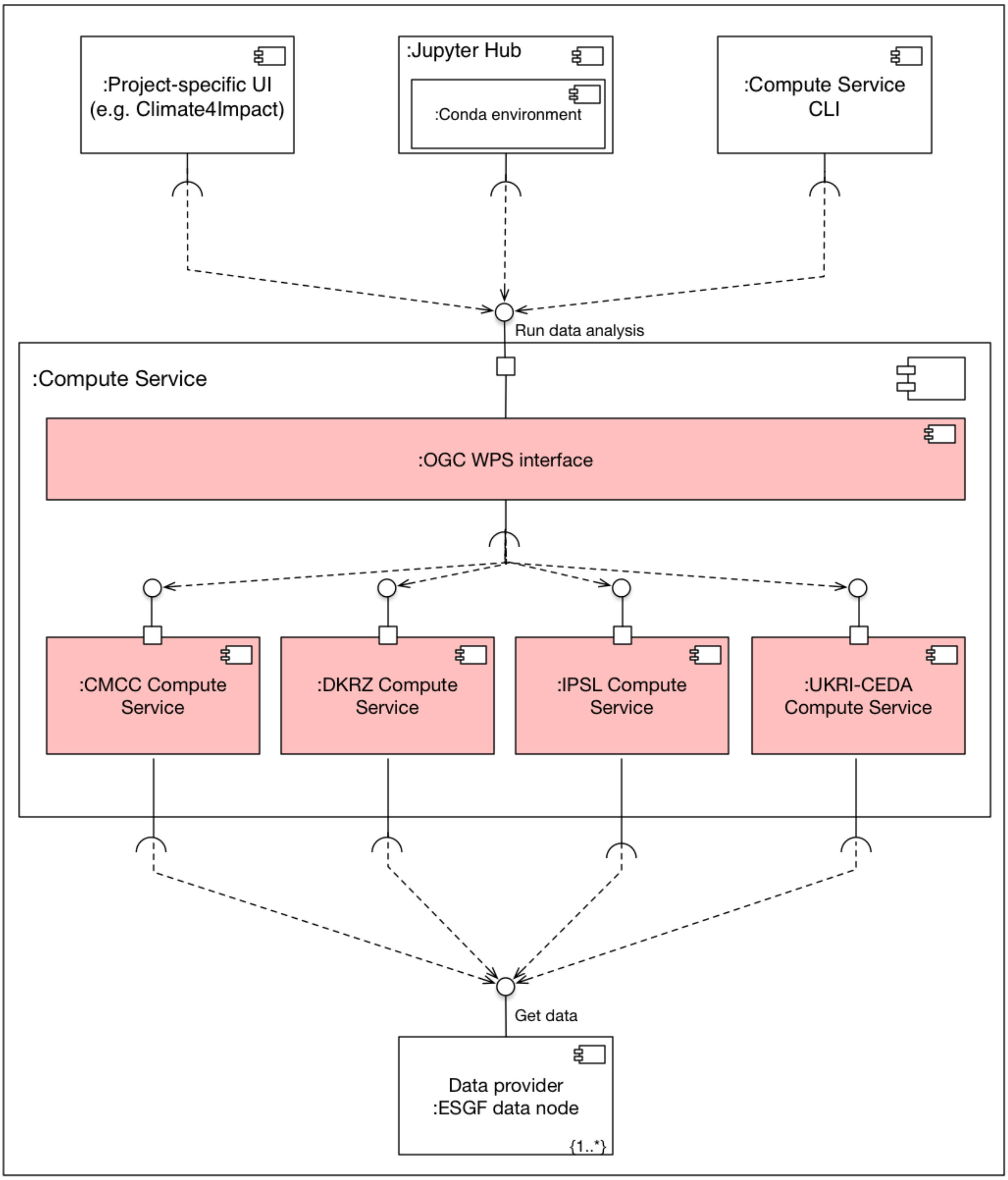

Fig. 28: Component diagram of the Compute service

5.8.1. Compute Service at CMCC

This section presents the architectural design of the Compute Service ("Analytics Hub") developed at CMCC. Such service has been in particular designed to meet the requirements from the climate scientists regarding multi-model analysis [COMPFR#4].

Multi-model data analysis requires access to a large amount of data (i.e. from CMIP experiments) available through the ESGF federated data archive, as well as running workflows

with tens/hundreds of data analytics operators. Examples of multi-model analysis are, among others: anomaly analysis, trend analysis and climate change signal analysis.
Such analyses consist of two main stages:

- the first part includes a number of identical sub-workflows, each associated with a specific climate model involved in the CMIP experiment and independent of the others; a future climate scenario must also be defined as input for this step;
- the second part considers a final workflow to perform statistical analysis on the set of output provided at the end of each sub-workflow at the first stage.

To fully understand some key challenges and very practical issues related to multi-model climate analysis, it is important to analyse the entire user's scientific workflow behind it. To perform multi-model climate analysis, the end-users should:

- download all the needed input datasets from the distributed ESGF data nodes to their local machines (local could mean the scientist's workstation or the user account on a HPC facility). Such a preparatory step represents a strong barrier for climate scientists, as the data download can take a significant amount of time (depending on the amount of data required by the analysis). Moreover, downloads can suffer from network instability, dropped connections, etc. which make the entire process even more painful.
- prepare a set of batch scripts that can properly process all the collected data. To this end, analyzing large datasets involves running multiple data operators, from a set of domain-oriented command line interface (CLI) tools (mostly sequential). This is usually done via scripts on the client side and requires climate scientists to take care of, implement and replicate workflow-like control logic aspects in their scripts, along with the expected application-level part. At this level, re-usability of scripts has never (or very poorly) been addressed.
- install and update all the required data analysis tools/libraries on their local machines. To this end, the proper setup of the ICT environment (which requires system management and technical skills) is key to run the analysis, as the user generally leverages a wide set of tools and the compatibility at ecosystem level (e.g. libraries), mainly related to software versions, can raise several issues.
- run the analysis taking into account the available computational and storage resources. This could lead to user-specific solutions about how to split the analysis, exploit parallelism, use the available resources, etc. In this regard, the large volume of data and the strong I/O requirements pose additional challenges related to performance as well as data handling.

5.8.1.1. Architectural design in the large

The proposed architecture in Fig. 29 implements an Analytics-Hub level on top of the existing ESGF data nodes backbone to allow the execution of multi-model climate analyses on a single location. The Analytics-Hub is responsible for providing computing and analytics capabilities on top of a data collection layer which both (i) pre-stages and caches the data relevant to the

analyses from the different ESGF data nodes and (ii) keeps the local copy of data synchronised with the remote copy available in the ESGF infrastructure.
Of course, a centralized storage location, like in the Analytics-Hub, cannot represent a scalable solution for the whole CMIP data archive (approximately 20 PB expected for CMIP6), but it can be considered as a suitable approach for the analysis of one or more selected variables (depending on storage availability). As a consequence, multiple, distributed Analytics-Hubs could serve the entire community by addressing the full spectrum of variables. Such scenario provides a centralised, variable-centric and Analytics-Hub-based infrastructural paradigm for multi-model climate analysis, on top of the distributed, model-centric and data nodes-based paradigm available through the ESGF infrastructure, mostly serving data access needs.
The design has been mainly driven by the data distribution requirement inherently coming from the legacy of the ESGF infrastructure as well as by the need to avoid large-scale data movement simply through the adoption of server-side analytics solutions [COMPFR#1].

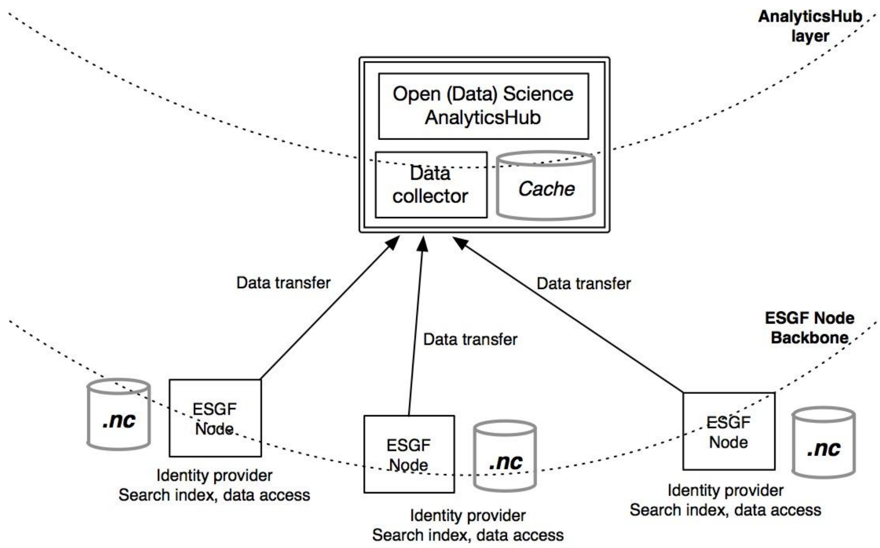

Fig. 29: Analytics-Hub architecture in the large

5.8.1.2. Architectural design in the small

The internal design of the Analytics-Hub consists of several components:

- an interface/GUI providing an Open (data) Science-ready environment where scientists can run their own Data Science applications, perform interactive and exploratory data analysis, run analytics workflows, perform data visualization, manage collaborative sessions, share analysis experiments, etc. [COMPFR#8];
- a workflow-enabled, secure, and interoperable Analytics-Hub front-end able to address the user's requests both in terms of single tasks and workflows;
- an analytics framework back-end able to perform data analysis at scale and support metadata management at different levels: datasets (e.g. data attributes), infrastructure (e.g. data partitioning & mapping onto the storage system, computational and data

resources, software ecosystem), and processing (e.g. provenance, logging and bookkeeping);
- a data collector and its local storage to gather the relevant datasets from ESGF and keep them in sync with the remote repositories.

As shown in Fig. 30, the proposed environment has relevant implications; actually, it also includes (i) publication services to enable sharing and re-usability of results (open data) across the community, and (ii) open development platforms to host, review, manage and share code (e.g. workflows, applications) by means of an open and community-oriented approach.

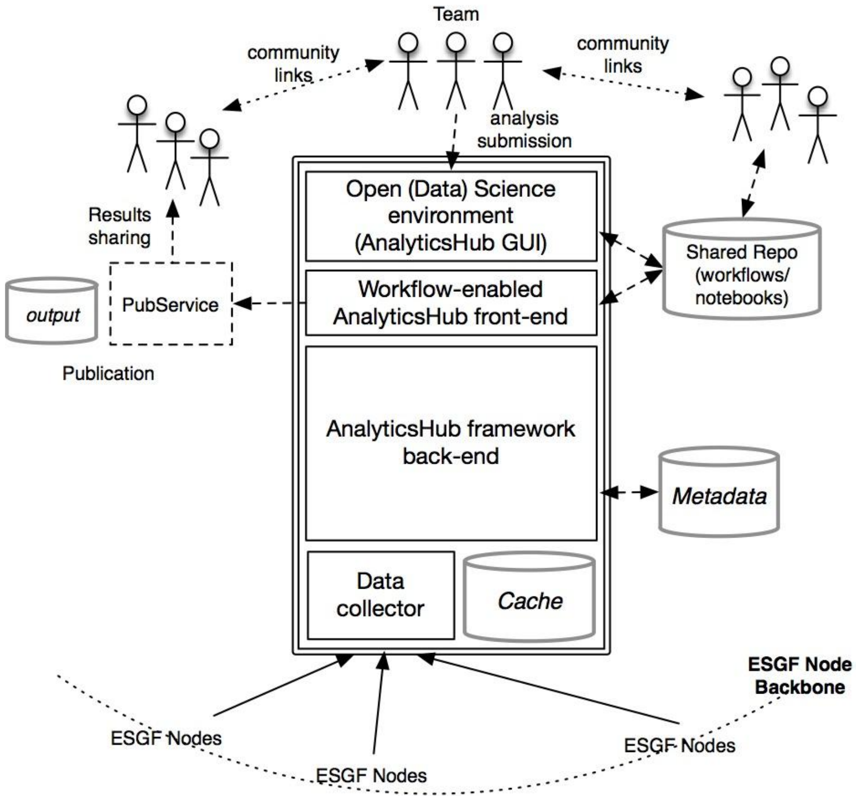

Fig. 30: Analytics-Hub architecture in the small

5.8.2. Compute Service at DKRZ

The compute service design at DKRZ integrated different installations with different, but complementary interfaces to be able to address the different end-users compute service use-cases and requirements:

- A flexible JupyterHub environment [COMPFR#8] interfacing the HPC system (see center column in Fig. 31). Individual users can select and reserve different compute resource profiles and then interactively use these resources, which are dynamically allocated on the HPC compute backend for the reserved time periods. This environment is closely connected to the large model data pool(s) hosted at DKRZ, currently hosting more than 6 PBytes of freely accessible climate model data collections. Parts of this pool also acts as a data cache, closely connected with high performance data transfer nodes and a software stack supporting data ingestion and export.

- An additional JupyterHub environment (right column in Fig. 31) is hosted interfacing a small compute cluster to flexibly support emerging e-science integration efforts [NFR#8] and developments (e.g. supporting the EOSC-hub ECAS service) with limited associated compute resources, but flexible authentication and data sharing integrations.
- To support compute web service endpoints interfacing with dedicated data analysis software hosted on data pool near resources (VM, dedicated physical servers or on HPC nodes exploited via Slurm) the deployment of OGC WPS services [NFR#11] [COMPFR#7] is supported (which are based on the Birdhouse open source framework initiated by DKRZ). These deployments are currently provided in pre-operational and test mode to support user groups in providing dedicated web service interfaces to their analysis code (e.g. for providing services to the Copernicus climate data store or the IS-ENES climate impact portal). A flexible AAI proxy [COMPFR#5] [NFR#5] is part of these WPS deployments supporting different authentication mechanisms.
- Additional access to compute resources is provided via dedicated login nodes. This can be flexibly used for end users to work remotely exploiting DKRZ resources. This can be exploited e.g. as part of ENES CDI Transnational Access (TA) service, otherwise dedicated project allocations need to be granted.

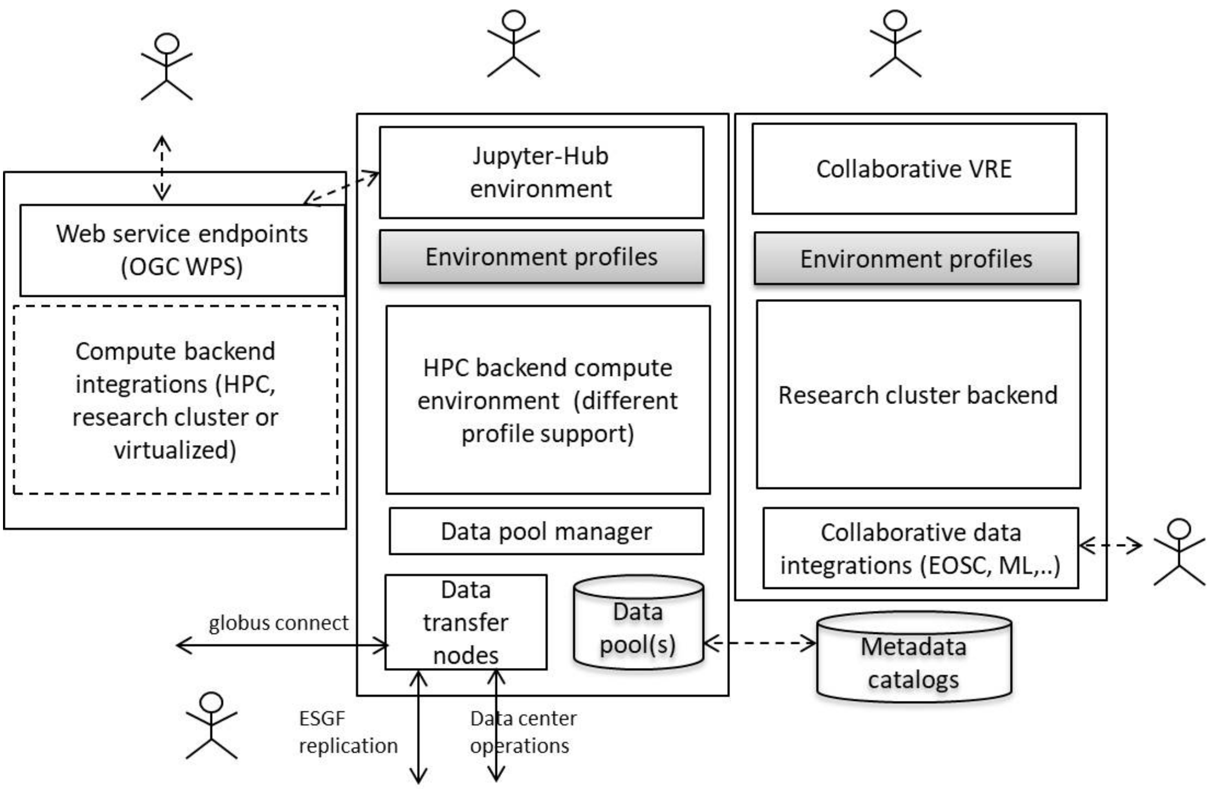

Fig. 31: DKRZ interactive compute environment

Current work concentrates on providing a flexible JupyterHub/WPS environment in which users can i) select and exploit different hardware and software environments and ii) use and deploy dedicated processing functionalities which need to be exposed as web services to remote parties [COMPFR#9] (e.g. portals).

5.8.3. Compute Service at IPSL

The compute service design at IPSL relies on generic remote access to dedicated login nodes (see Fig. 32).

The IPSL mesocentre includes several pre-configured Python environments that activate mutualized and useful tools for data quality check and analysis. Individual users can reserve different compute resources through a cluster with compute nodes (2000 cores, up to 256 GB RAM/node). The cluster uses a usual PBS resource manager. Current work concentrates on providing a flexible JupyterHub environment [COMPFR#8] that will interface with the HPC system through a Kubernetes instance to be able to address the different end-users compute service use-cases and requirements.

Each user can ask for a work space with dedicated storage that will be balanced with other requests. The IPSL mesocentre provides 50 TB shared storage for data analysis (Lustre), temporary and final results, alongside of a 4Po of specific CMIP and CORDEX and observational datasets (Reanalysis, Obs4MIPs, input4MIPS, etc.) with centralized access [COMPFR#1] (including the whole French climate modeling production).

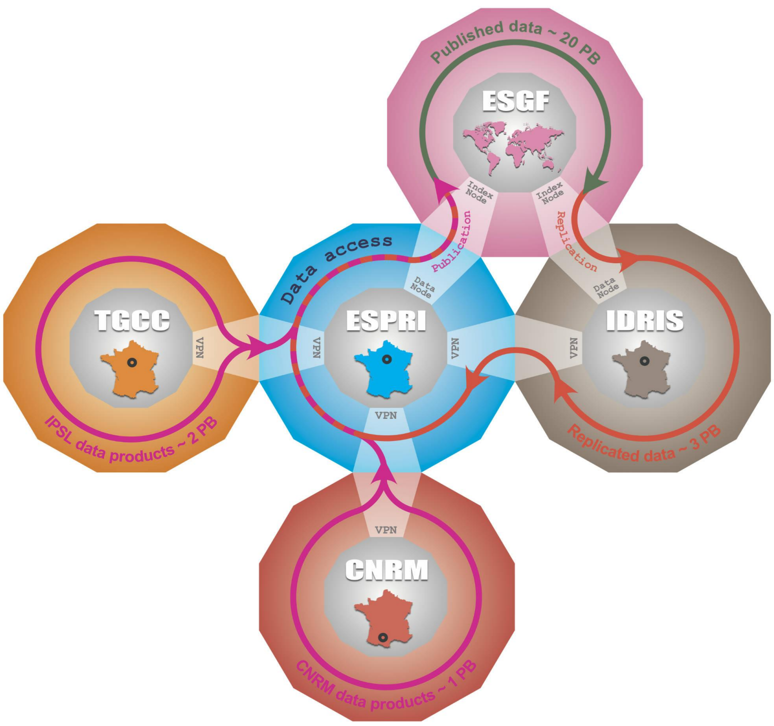

Fig. 32: IPSL multi-model data analysis platform

5.8.4. Compute Service at UKRI (CEDA)

The CEDA Team (now part of UK Research and Innovation, or UKRI) is developing a new Compute Solution that builds upon experience of deploying WPS services over a period of more than a decade for a range of projects and services. Having developed its own WPS (known as "cows", CEDA OGC Web Services), CEDA aims to unite its approach with that of DKRZ and IPSL to develop a common framework based upon the Birdhouse suite of tools.
The focus of this work, currently being undertaken with C3S funding, is to develop a basic set of common operations that are:

- Delivered through a WPS API [COMPFR#7]
- Provided through a set of open-source python libraries
- Robust [NFR#2] and reliable
- Deployable in a repeatable and automatable manner [NFR#13]
- Horizontally scalable [NFR#3]
- Tested against large volumes of ESGF data (i.e. CMIP6, CMIP5 and CORDEX) [NFR#4]

The design of the system reuses a number of pre-existing components in the common Python and PANGEO software stacks, with Xarray providing the main I/O and processing library. The initial design focuses on the following operations:

- Subsetting over the standard spatiotemporal domain [COMPFR#2]:
  - Temporal subsetting
  - Vertical dimension subsetting
  - Horizontal spatial subsetting (along linear axes)
- Averaging [COMPFR#3]:
  - Simple averaging over time, vertical and horizontal domains over a user-selected range
- Re-gridding:
  - Basic one-way regridding from the model domain to an agreed set of resolutions.

5.8.4.1. Robustness and reliability

CEDA has experience of putting up WPS services over the last decade. The common experience is that users are reluctant to take up the services unless they have been very well tested. It is typical that a WPS process will be tested with a small number of input files before being made available to the user community. When called with an arbitrary dataset from a large heterogeneous project (such as CMIP6 or CORDEX) the process has a high chance of failure. This failure leads to the user community not having confidence to base their analyses on remote processing services. This has multiple detrimental outcomes such as (i) lack of trust leading to development of multiple libraries/frameworks/solutions: duplication and waste of effort; (ii) scientists writing their own code: untested, less sharable; (iii) scientists downloading the data locally: expensive on bandwidth and inefficient.

The focus of the CEDA Compute solution is to capture the heterogeneity of the underlying data sets through a comprehensive system of *characterising* and checking samples of the data. The checks are used to indicate where some data sets can be considered as *atypical* when compared with others in the sample. The checks are then matched up with *fixes* that are used to correct the data sets on-the-fly when the user requests a given process. Fig. 33 shows an overall of the system architecture. The fixes are implemented using Xarray but are described in a language-agnostic format that can be used by any project/software.

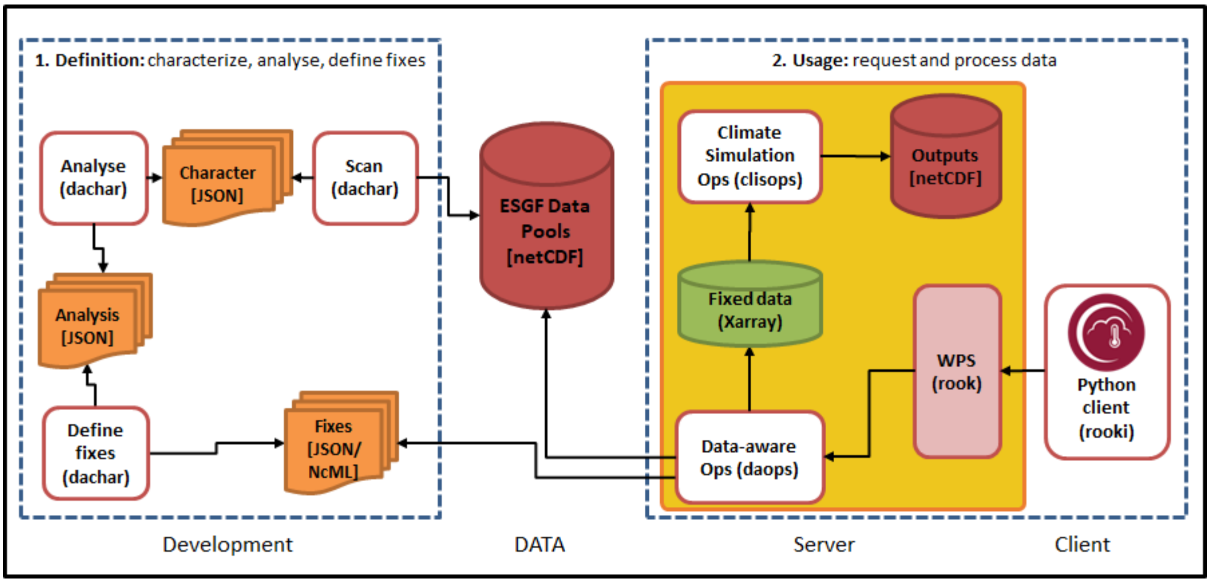

Fig. 33: CEDA compute service architecture

The CEDA Compute solution involves a "characterisation" component (on the left). This includes a number of information stores that will be made accessible outside the project. These will share the knowledge of the character and fixes with collaborators in the wider community. On the right side, the "usage" component shows how a number of software libraries interact to provide "data-aware" operations - i.e. those that can perform fixes on-the-fly.

5.8.4.2. Deployment and scalability

The CEDA Solution is deployed using Ansible playbooks so that the service components can be rolled out in a repeatable and automated manner. This includes a "single-server" test deployment with the WPS interface and execution of tasks all on one server. However, the production deployment includes an external load-balancer, capable of pointing at deployments across multiple sites. Each site deployment has a main node which houses the database, Slurm scheduler master node and the web-service interface. This is attached, via Slurm, to a set of batch nodes that can be added to as required depending on demand. Since the servers are built with Ansible, the cost of adding more servers is only the set-up cost of provisioning new virtual machines. Fig. 34 shows the deployment solution with load-balancing across multiple sites. The external load-balancing allows either site to go down without service interruption. The use of the Slurm scheduler and the batch nodes enables straightforward horizontal scaling.

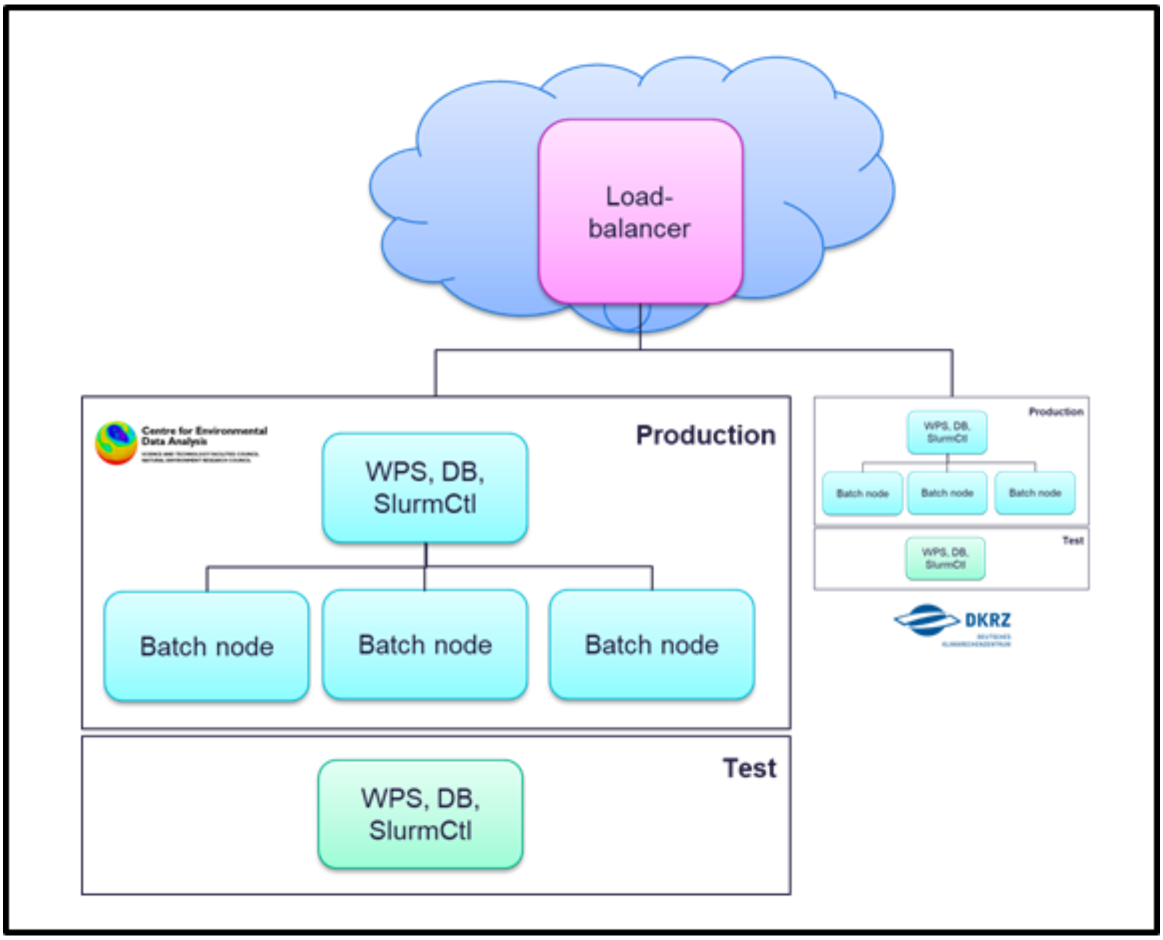

Fig. 34: An example of the load-balanced deployment across the CEDA and DKRZ sites.

5.9. Climate4Impact

The IS-ENES Climate4Impact (C4I) platform has been in operation since 2011. It provides easier access to climate simulations for end users, especially the climate change impact modelling community.

The characteristics of the C4I platform are as follows:

- Targeted at climate science researchers and the climate change impact community, to explore climate data [C4IFR#1], download required data, and perform on-demand analysis.
- Connected to ESGF web services.
- Visualization capability via ADAGUC[52] Software [C4IFR#2].
- Provides Analysis Services using (Py)WPS and reproducible notebooks (coming soon) to perform calculations.
- Runs an operational service.
- Promotes Users' engagement: climate research community, climate impact community

[52] http://adaguc.knmi.nl/

as well as interdisciplinary research community.

The C4I Platform provides a range of processing capabilities, from time and spatial subsetting [C4IFR#5] (with a GUI using OGC WCS[53] Standard) to simple statistics, such as time average, to more complex calculations, such as climate indices and indicators [C4IFR#7] (provided by the ICCLIM[54] software). More complex tools such as statistical downscaling are also provided through an interface to the University of Cantabria Downscaling Portal[55].
However, since 2011, new technologies have emerged. Those technologies help to better separate content and functionalities [NFR#2] [NFR#6], and they also improve user experience [NFR#9]. They also help developers from multiple partners to better work together on the same codebase.
The C4I Platform is being redesigned completely in order to take advantage of those new technologies, notably ReactJS, MarkDown, Birdhouse WPS. It is also structured using a micro-services architecture approach. The development environment has been revisited and completely changed, using Visual Studio Code, GitLab and Amazon S3 Bucket. The deployment must be easy and flexible [NFR#13], so each microservice component can be deployed using Docker, and the whole C4I Platform can be deployed using Docker-Compose. The architecture diagram shown below reflects this. Those micro-services can be deployed on several servers if needed, and the WPS component is external, running necessarily on another server to ensure scalability [NFR#3] and prevent overload of the UI platform server. Emerging technologies also motivated an updated design of the system. This includes the extension of the analysis capabilities of C4I with new functionalities to allow more flexibility and experimentation with the data. The portal will be able to stage and pre-process, by means of workflows, distributed raw data available at the ESGF nodes, onto dedicated workspaces running on Cloud. Here users can develop and document new analysis methods adopting Jupyter notebooks. They will be able to update the dataset of interest, as well as the underlying computational environment autonomously. All these actions will be recorded in a provenance model fostering reproducibility [C4IFR#9]. The use cases will be refined and implemented during the coming reporting period.
An overview of the current C4I Architecture sketch is shown in Fig. 35, using a layered traditional approach, while, in Fig. 36, the UML component diagram shows the physical organization of the platform, the interfaces and the real connections between the elements.

53 http://www.ogc.org/standards/wcs
54 https://icclim.readthedocs.io/
55 https://www.meteo.unican.es/en/portal/downscaling

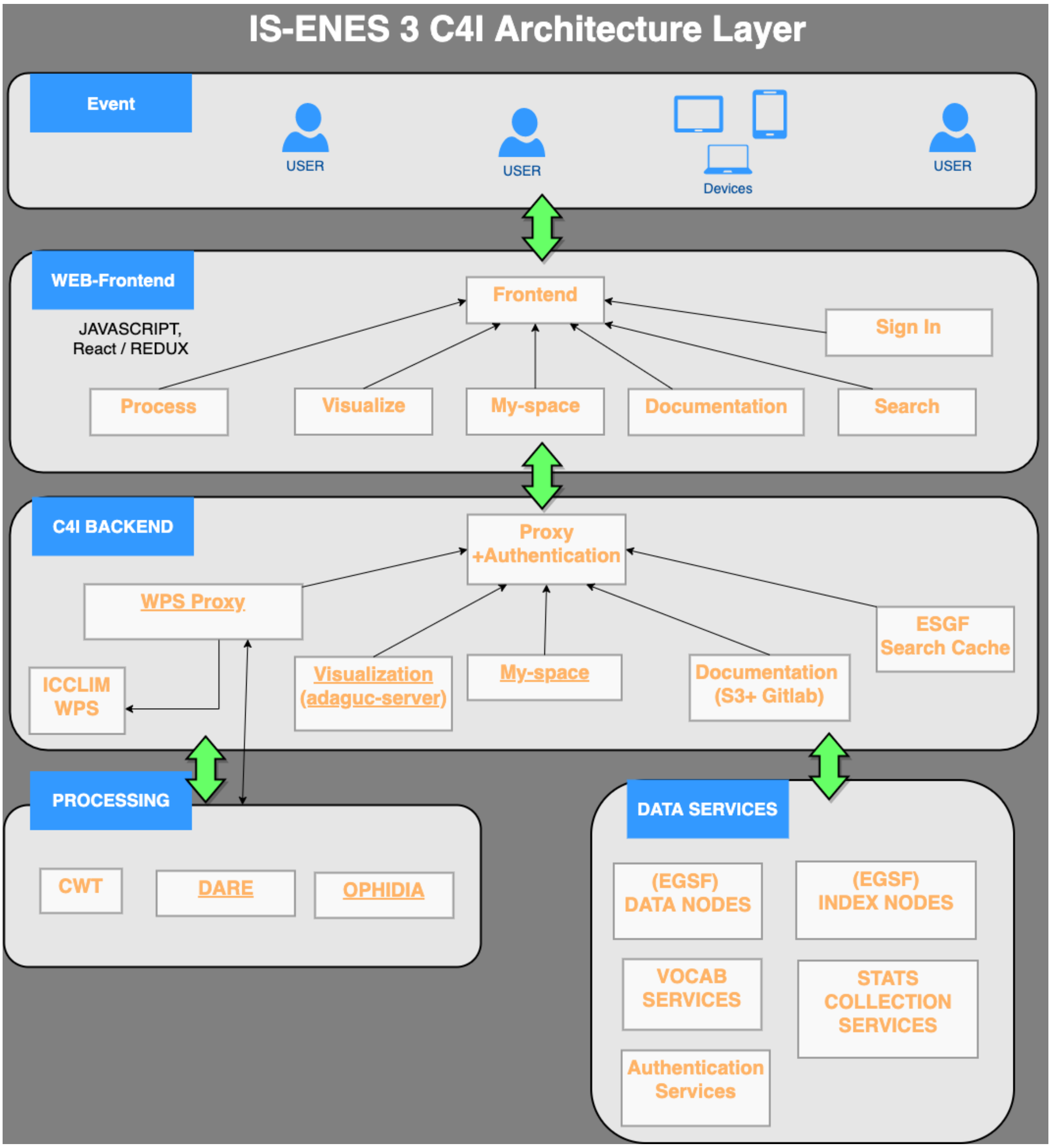

Fig. 35: Climate4Impact architecture diagram

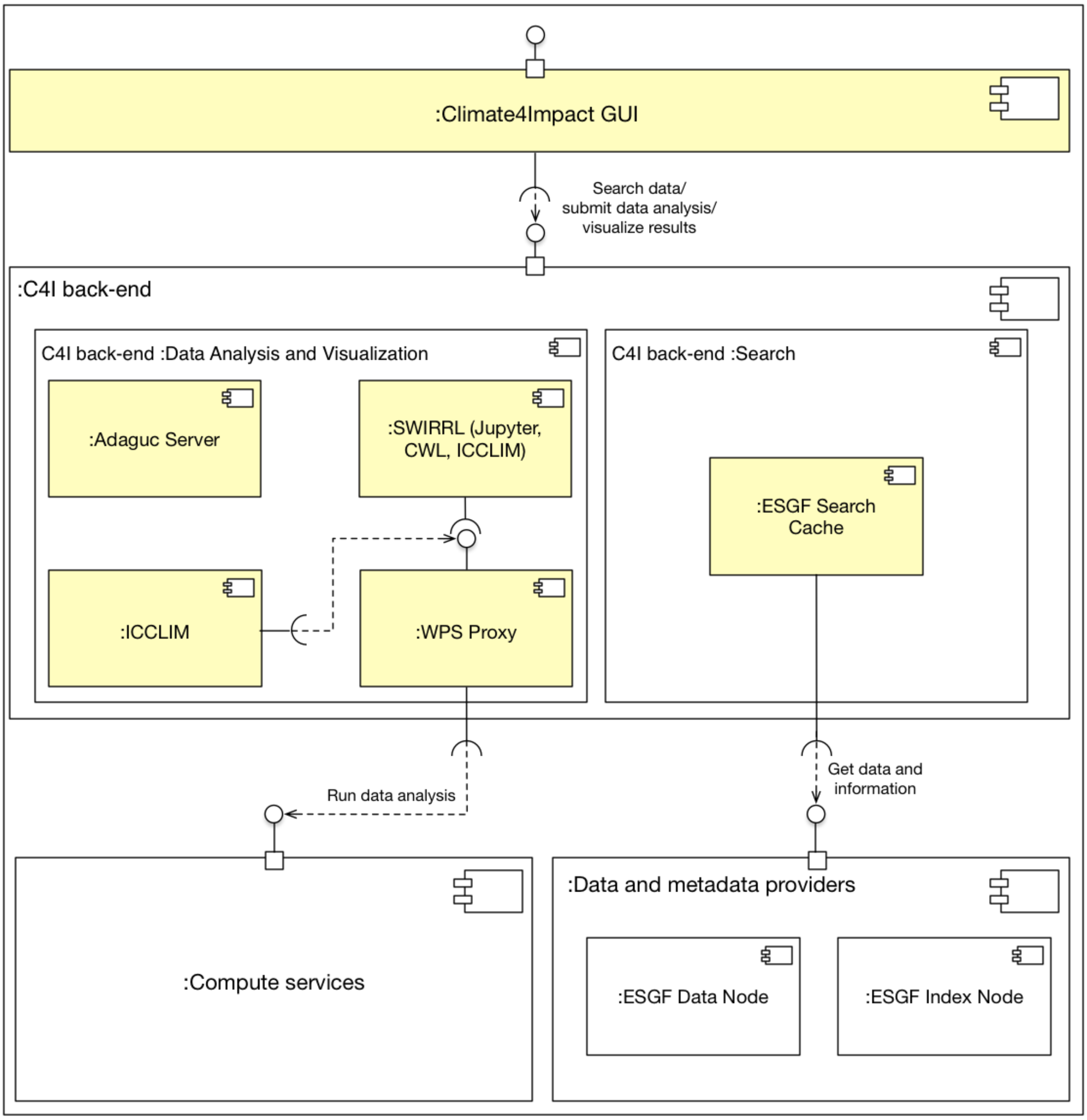

Fig. 36: Component diagram of the Climate4Impact platform

The Docker-Compose organization of components is shown in Fig. 37. A *nginx* web server is acting as a proxy to merge all separate subcomponents together.

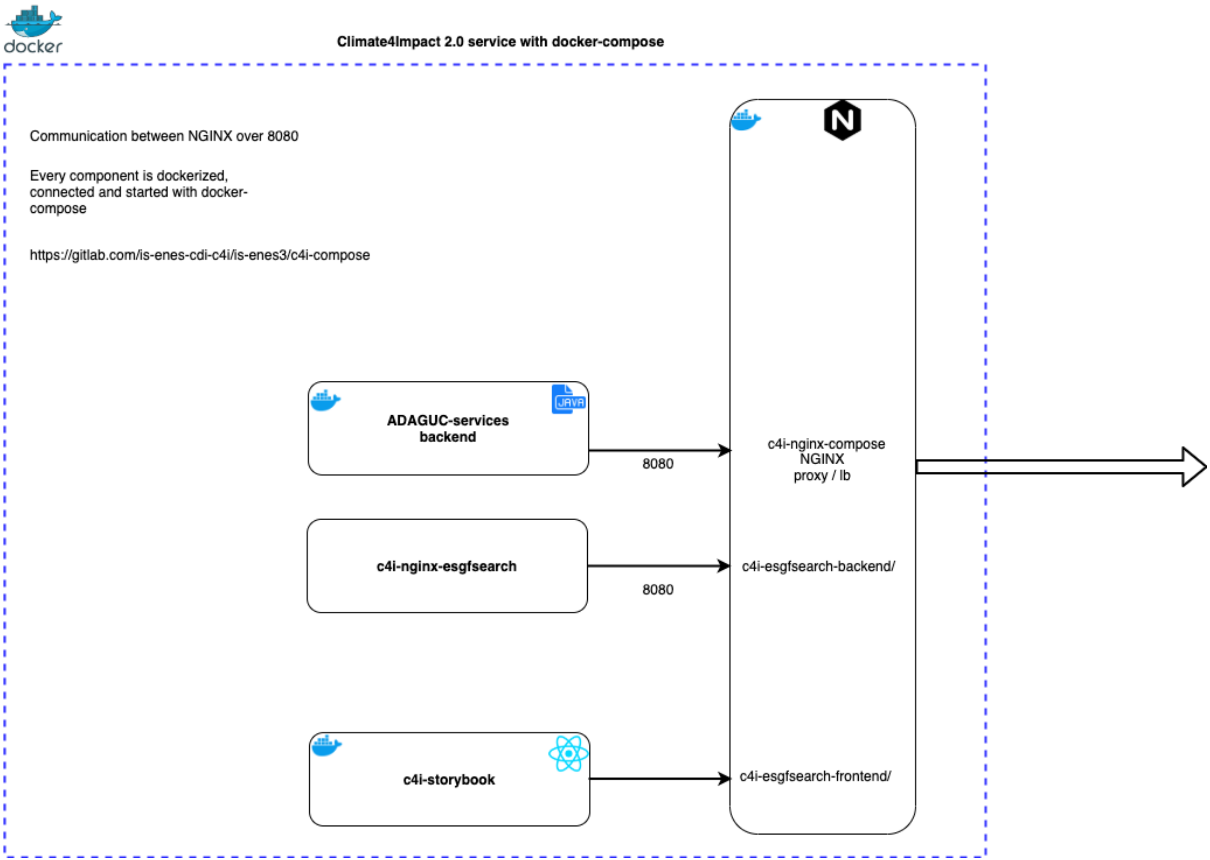

Fig. 37: Climate4Impact Docker-compose organization

5.10. ES-DOC

The ES-DOC (Earth System Documentation) project requires a software ecosystem that facilitates both the provision and the consumption of documentation of the CMIP6 workflow and, where possible, automating, the various and often complex stages involved.
The back-end software stack comprises utility libraries to automate [NFR#4] creation and publication of standardised documents and to store and systemise semantics for applicable controlled vocabularies, models, and ontologies; content storage through archive repositories; and a shell-script library to facilitate development using these tools on the system of multiple separately version-controlled repositories. The front-end stack consists of web services to manage documentation and errata databases, for generating and publishing the model documentation, for rewriting various URLs; and web applications that support the viewing, searching, and comparing of the published documentation, as well as serving and displaying other relevant content.
Fig. 38 shows the key steps involved under the ES-DOC architecture in getting the content from modellers, the documentation providers, to users (i.e. consumers of CMIP data), via core storage archives. The stages of composition and setup for the intermediate tooling chain and the life cycle of model documents are also included:

a) depicts the incorporation of standards into ES-DOC software tools, which may provide for other tools, resulting in some endpoint tools playing one or more direct role(s) in the archive population and access;

b) shows the population into the ES-DOC archive of content originating from the modelling community;

c) illustrates how the data stored there can be consumed by downstream users by three distinct routes.

In each case, relevant (groups of) people or 'actors' [torso symbols] or informational standards [purple boxes] act as inputs and interact with each other, with data content in NetCDF [dataset symbols] and template files [blue hexagons], and with various elements of the ES-DOC software stack [beige ovals] including web-based user interface (UI) components [green rounded boxes] and attributes [white round box] in-browser.

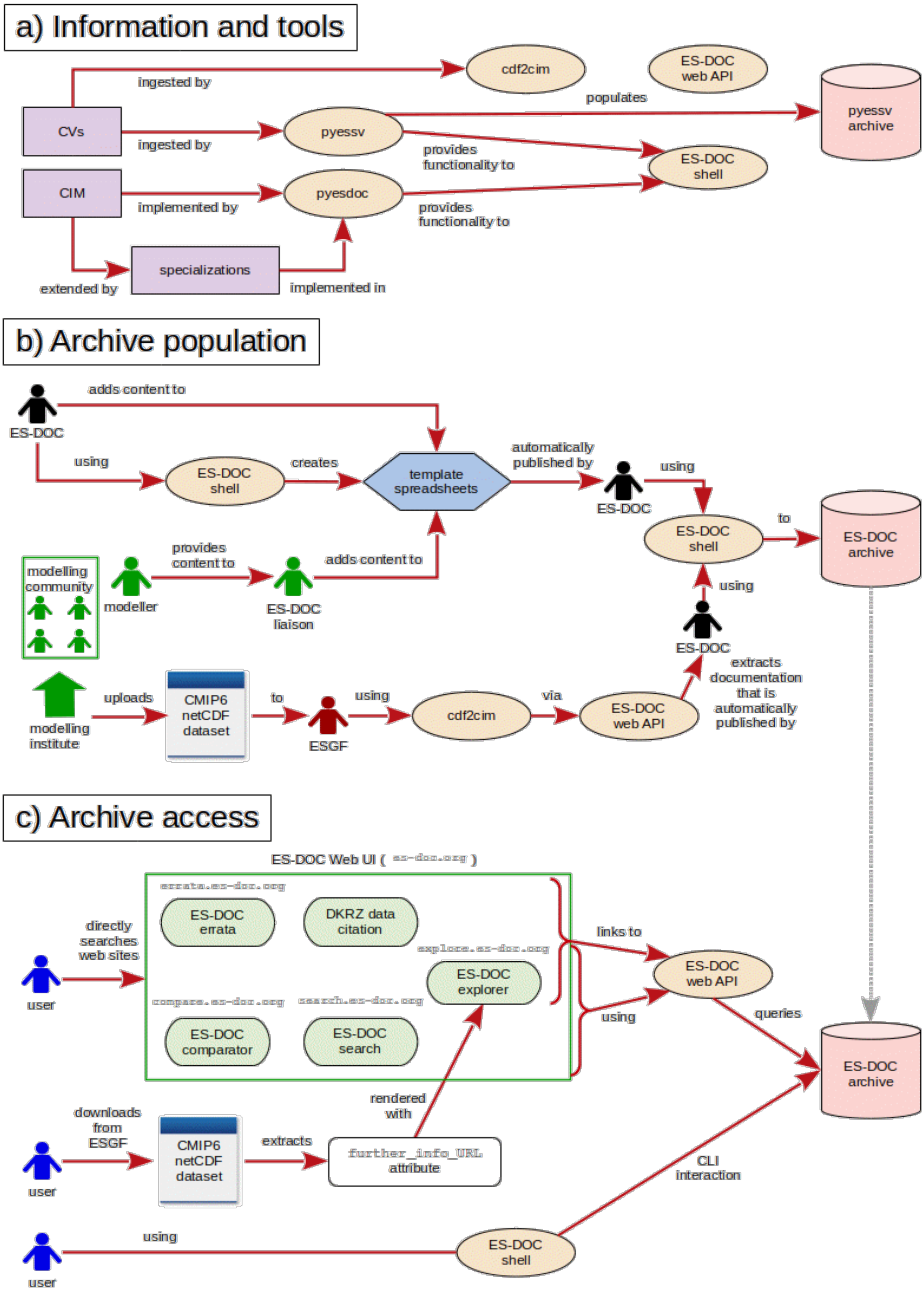

Fig. 38: ES-DOC architecture workflow

5.10.1. The tooling chain, storage, and hosting

Fig. 38 a) illustrates how various semantic standards are taken in by ES-DOC software libraries leading to tools that are utilised directly by actors in the archive population and consumption workflows (covered in subsequent sections), as well as the collection of controlled vocabularies into the pyessv-archive[56] (see section "The back-end software stack).

5.10.2. Collecting standards as a starting point

The effective start point for the entire workflow for taking documentation from producers to consumers, are semantic standards which define a rich set of artefacts to describe all aspects of numerical experiments and their requirements in a consistent and rigorous manner. These are the Common Information Model (CIM2), the model specialisations which constrain the generic CIM2 for application CMIP6 model descriptions, and other controlled vocabularies (henceforth CVs) which define other constraints on the properties outlined in the CIM2.

The CIM2 forms the basic conceptual data model, but to fully document a configured model composed of "realms" (e.g. atmosphere, ocean, land surface, etc.) the defined properties need to be customised on a realm-by-realm basis. This is encapsulated in "specialisations", for each of the eight CMIP6 realms plus the top-level set[57] which define metadata that extends the properties of CIM2. The specialisation repositories also contain code to render their content from the front-end web service.

5.10.3. The back-end software stack

The pyessv library[58] takes the WCRP CMIP6 CVs from their repository as well as INI files from ESGF and (amongst other roles) validates and serialises them as JSON, and populates the pyessv-archive. It is also used downstream in the cdf2cim library[59], which is used to map NetCDF files to CIM documents via the creation and manipulation of intermediate JSON structures holding metadata from the NetCDF headers of all published CMIP6 datasets. Note that cdf2cim requires its own archive to store that intermediate JSON (see the 'Storage' section below).

The client at the heart of the ES-DOC toolchain is referred to as the pyesdoc library[60]. It implements the supplemented CIM2 schema to conduct a number of core tasks relating to CIM2 documents, for example validating them against the schema, archiving them, and publishing them to the ES-DOC web service. A supplemental tool that is similar in function is pyosl[61], which defines a metamodel for the manipulation of a general ontology and provides methods and tools to do so.

56 https://github.com/ES-DOC/pyessv-archive
57 https://github.com/ES-DOC/cmip6-specializations-toplevel
58 https://github.com/ES-DOC/pyessv
59 https://github.com/ES-DOC/esdoc-cdf2cim
60 https://github.com/ES-DOC/esdoc-py-client
61 https://github.com/ES-DOC/pyosl

Together, pyessv and pyesdoc provide functionality that is coordinated by the ES-DOC shell that is ultimately what is utilised directly by the ES-DOC team for management of the ecosystem.
Finally, the ES-DOC Web API[62] [ESDOCFR#3] is a self-contained library that exposes the various endpoints of the web applications delivered by ES-DOC, as covered further in the content consumption section.

5.10.4. Servers and hosted components

The infrastructure for ES-DOC requires deployments with third-parties for exchange of data with users, web services and documentation.
The ES-DOC software libraries and institutional raw content live in repositories under the 'ES-DOC' and 'ES-DOC-INSTITUTIONAL' organisations, respectively, on GitHub. The web assets and databases are hosted on servers, from which the home page for the ES-DOC project[63] is delivered.

5.10.5. Storage

There are several artefacts which necessitate permanent storage and they are stored and updated as required either in dedicated version control repositories or in the databases on hosted servers. Data stored in this manner are: all standard vocabularies as converted to pyessv notation; all published cdf2cim[64] intermediate content encoded in the JSON format; all final model documentation as published CIM2 documents[65]; and finally, the institutional content, in per-institute repositories (50 at time of writing) under 'ES-DOC-INSTITUTIONAL'[66].

5.10.6. The role of ES-DOC in content provision

Fig. 38 b) conveys the means by which raw metadata content provided by modellers is converted to CIM2 documents that are stored in the ES-DOC archive, ready for consumption [ESDOCFR#1]. There are two approaches that each of the various institutes making up the modelling community can choose from to upload their content as a starting point.
The first approach centres around spreadsheets as the format for input and submission of documentation. The starting point in this case is the ES-DOC team, who design numerous template spreadsheets to present the information required, as delineated in CIM2 and the specialisations, to document distinct aspects of the modelling workflow, namely the citations, ensembles, experiments, institutes, models, responsible parties and (still in development, but soon) the conformances, machines and associated performances.

62 https://github.com/ES-DOC/esdoc-api
63 https://es-doc.org
64 https://github.com/ES-DOC/esdoc-cdf2cim-archive
65 https://github.com/ES-DOC/esdoc-archive
66 https://github.com/ES-DOC-INSTITUTIONAL

For a given template, the ES-DOC team creates a skeleton spreadsheet ready to hold the relevant text and then use the ES-DOC shell utility to populate it with the documentation artefacts from the relevant CIM2 classes, such that each item includes data entry fields, along with guidance and per-item descriptions, so all required items may be entered systematically. Code is created for each template spreadsheet so that its contents can be extracted and serialised, and archived. The ES-DOC shell is used to make and commit copies of it to every institutional repository in the 'ES-DOC-INSTITUTIONAL', including the option to pre-populate some items from other, already published sources.

The modelling institutions are free to fill them in and submit them via committing back to their repository. The timing, frequency, order, and grouping of submission are, however, up to them. The key actors in the completion and submission of spreadsheets are the ES-DOC liaisons for each institute. These are a small number of people (usually just one) per institute who are appointed to be responsible for organising the submission of the CMIP6 documentation for their group. Overall, they provide a small and well-defined interface between the modelling community and the ES-DOC team.

Once institutes have, under the instructions of their liaison, committed filled-out spreadsheets to their GitHub repository, the ES-DOC shell checks their configuration files to see if a 'publish' field has been set to 'on', which indicates that the group would like to publish the subset of content managed by that configuration file (e.g. for the citations or conformances spreadsheets separately, or only for particular model realms) they have in their repository. In the case publication has been activated, the ES-DOC back-end system has been configured to automatically update the published CIM2 documents with corresponding changes, and ingest these into the ES-DOC archive.

The main alternative pathway to content provision and document creation is bespoke to the documentation of ensembles of CMIP6 simulations, and is an automated process that derives the documentation from NetCDF files during the ESGF publication phase. This is the lowermost path shown in (1b). This functionality uses cdf2cim and the ES-DOC web API and is installed as part of the standard ESG publisher stack[67]. On the ESGF node, ES-DOC's cdf2cim is used to scan sets of all NetCDF datasets from the local file system as they are published, extracting the metadata from these and serialising it all into a single JSON structure (referred as a JSON "blob") per applicable simulation. These JSON blobs are pushed to the ES-DOC server. Successful server upload in this manner requires passing validation against the CIM2 schema. Once on the server-side, the ES-DOC Web API will, in an automatic process configured and controlled using the ES-DOC shell, take the JSON blobs and convert them into CIM2 documents which it publishes to the ES-DOC archive.

There is a third approach that may be used instead of the two described, but for CMIP6 this is only being used by the UK Met Office for model documentation. For this method, the modelling institute stores their own, bespoke database of documentation and uses pyesdoc to map its contents to CIM2 documents. These documents can then be published by the modelling

[67] https://github.com/ESGF/esg-publisher

institute itself. Once the latency of developing an in-house database system has been overcome, this method can be used to create and publish documentation with minimal effort.

5.10.7. The role of ES-DOC in content consumption

See Fig. 38 c) for an outline of the steps by which end-users can access any of the documentation which gets produced via the documentation provision workflow covered in the previous section. Note there are three distinct (main) pathways which can be taken. Each route may be more suitable in a given context and/or by preference of the consumer.

5.10.8. Web User Interface (UI) and its connection with related web services

The first and most visible inlet for documentation consumption is ES-DOC's browser-based user interface or 'Web UI' [ESDOCFR#2], which has a landing page domain of es-doc.org. Under this root domain, there are various web services that provide applications which are mapped to specific domains.

The core ES-DOC web assets accessible from there are applications that enable end-users to view and compare documentation across all models. The latter application, the comparator[68], only provides comparisons for selected aspects of CMIP models (only CMIP5 models at the time of writing - extensions to provide functionality for CMIP6 models are still under development). The view application allows users to search, select and view any available documentation[69].

There are two further major web resources which ES-DOC interfaces to, namely services for management and dissemination of errata and of data citation.

The Errata System, which is coordinated with ESGF, aims to track known issues and amendments in data that has been published in a timely manner. The core system is documented elsewhere for this deliverable so will not be discussed further other than to say that ES-DOC makes the errata from this system accessible and searchable from within its interfaces.

The (CMIP6 Data) Citation Service is managed by DKRZ (hence also recorded elsewhere for this deliverable) and hosted outside of the ES-DOC web UI. However, it is intrinsically connected to the ES-DOC front-end ecosystem via the "further info URL" landing pages (see next section).

5.10.9. The Further Info URL landing page and attribute

The second main route for accessing published metadata is a bottom-up approach via the "further info URL" [ESDOCFR#4], which is a landing page for the documentation of any CMIP6 simulation containing all documentation registered for a given ensemble of simulations, as well as links to the documentation for corresponding model and experiment.

---

68 https://compare.es-doc.org

69 https://search.es-doc.org, https://view.es-doc.org and https://explore.es-doc.org

The URL itself in each case is formed systematically from components of the CMIP6 Data Reference Syntax[70]. The domain formed from the further info URL is mapped to the explorer application which renders the landing page.
The route in this case begins with an end-user inspecting the contents of a dataset in which they are interested. All CMIP6 NetCDF datasets are required to have a global attribute called “further_info_URL” which gives the URL discussed above for the simulation the dataset belongs to. So, by extracting the value of the “further_info_URL” from a dataset, a user can simply copy it into the browser to access the landing page and from that, either directly or via following the links contained within, find all existing metadata relating to their dataset.

5.10.10. ES-DOC shell as an advanced access route

The third and final core route, shown from the lowermost user in Fig. 38 c), is for an end-user to utilise ES-DOC libraries via the ES-DOC shell in the same manner that the ES-DOC team does for operations.

5.10.11. Architectural design

In conclusion, the diagram presented in Fig. 39 shows the main blocks of the ES-DOC service, how they are distributed among the different layers of the ENES CDI architecture and the connections between them and with external components.

70 https://www.earthsystemcog.org/site_media/projects/wip/CMIP6_global_attributes_filenames_CVs_v6.2.6.pdf

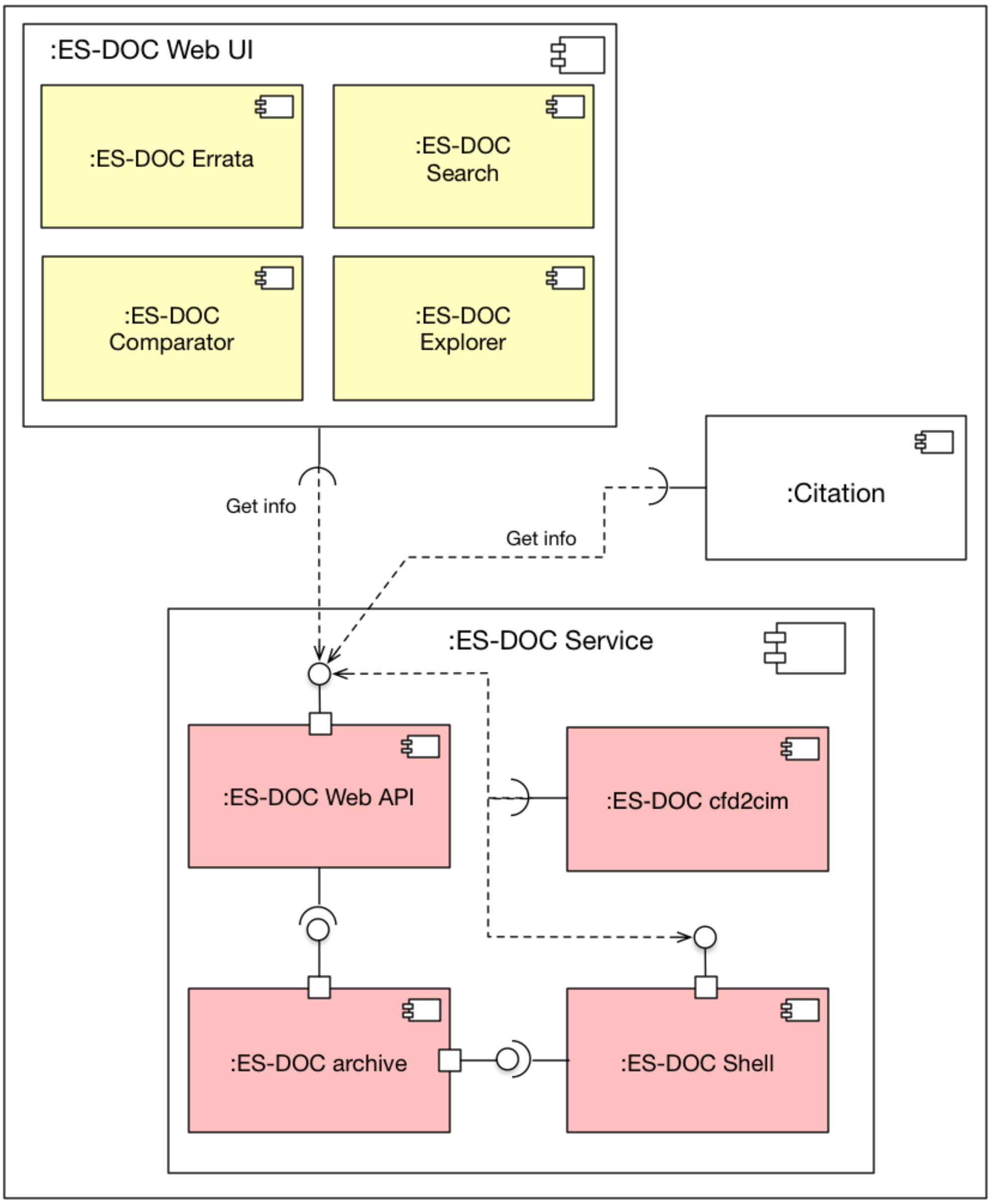

Fig. 39: Component diagram of the ES-DOC service

5.11. Climate and Forecast Conventions

The conventions for CF (Climate and Forecast) metadata are designed to promote the processing and sharing of files created with the NetCDF API. The CF conventions are increasingly gaining acceptance and have been adopted by a number of projects and groups as a primary standard. The conventions define metadata that provide a definitive description of what the data in each variable represents, and the spatial and temporal properties of the data. This enables users of data from different sources to decide which quantities are comparable, and facilitates building applications with powerful extraction, regridding, and display capabilities.

These conventions provide the foundation for exchange and analysis of all the model data distributed by the IS-ENES research infrastructure [NFR#11], so every file published and every file downloaded represents a transaction which is dependent on this standard as well as other service components.

The CF Conventions are developed and maintained through an international collaboration including partners in the US and European partners who are outside the IS-ENES consortium (for example, EUMETSAT is supporting the development of CF in order to exploit the standard for dissemination of satellite data products). The contributions of IS-ENES to the development and maintenance of the CF Conventions are in two areas: i) support for the CF Data Model and ii) support for CF Standard Names.

5.11.1. CF Data Model

The CF Data Model[71] identifies the fundamental elements ("constructs") of the CF conventions and shows how they relate to each other, independently of the NetCDF encoding.

IS-ENES3 funds the maintenance of the *cfdm* package, which implements the CF data model through its internal data structures and so is able to process any CF-compliant dataset. It is not strict about CF-compliance, however, so that partially conformant datasets may be ingested from existing datasets and written to new datasets. This is so that datasets which are partially conformant may nonetheless be modified in memory.

The cfdm[72] package plays a key role in developing and demonstrating the power of the CF Data Model.

5.11.2. CF Standard Names

A fundamental requirement for the exchange of scientific data is the ability to describe precisely the physical quantities being represented. Within the CF Conventions, this requirement is met by the standard name table, which contains a register of agreed terms established through a transparent community discussion [CFFR#1]. IS-ENES3 supports the moderation of the discussion of standard names and the publication of new tables.

The key elements are:

- user guidance on the conventions web pages[73];
- discussion of new proposals and issues in a github repository[74];
- preparation of new terms for publication in a Django application[75] [CFFR#2];
- publication of new terms in the conventions web pages and in a vocabulary service[76] [CFFR#2].

---

71 https://doi.org/10.5194/gmd-10-4619-2017
72 https://pypi.org/project/cfdm/
73 http://cfconventions.org/Data/cf-standard-names/docs/guidelines.html
74 https://github.com/cf-convention/discuss/issues
75 http://cfeditor.ceda.ac.uk/proposals/1
76 http://cfconventions.org/standard-names.html and http://vocab.nerc.ac.uk/collection/P07/current/

5.12. CMIP Data Request

The Data Request of CMIP6 defines all the quantities from simulations that should be archived. This includes both quantities of general interest needed from most of the CMIP6-endorsed model intercomparison projects (MIPs) and quantities that are more specialized and only of interest to a single endorsed MIP. The complexity of the Data Request has increased from the early days of model intercomparisons, as has the data volume. In contrast with CMIP5, CMIP6 requires distinct sets of highly tailored variables to be saved from each of the more than 200 experiments. This places new demands on the data request information base and leads to a new requirement for development of software that facilitates automated interrogation of the request and retrieval of its technical specifications [NFR#1].

5.12.1 Data Request Schema

The Data Request schema sets out the structure of the database and defines the technical content [DREQFR#9]. The database (see 5.12.3 below) is replacing an Excel workbook[77] used to compile the equivalent information for CMIP5 and the schema can be thought of as replacing the inline comments in the Excel workbook which explained links between sheets and rows. The schema is described in the "Data Request XML format" document contained in the package[78], and expressed as an XSD schema file[79].

5.12.2 Data Request Pages at WIP

The Data Request is compiled through a community discussion process. Organisational information is presented within the pages of the WGCM Infrastructure Panel[80].

5.12.3 Data Request Database

The core of the Data Request is an XML document[81] conforming to the Data Request Schema. The tables of the data request detail metadata for thousands of variables, and also requirements for output associated with hundreds of experiments [DREQFR#4].

5.12.4 Data Request Python Package and Repository

The CMIP data request is supported by a python API through the *dreqPy* python package. This package, bundled with the database and the documentation, is provided through the python Python Package Index.

---

[77] Provided by PCDMI: https://pcmdi.llnl.gov/mips/cmip5/datadescription.html
[78] http://proj.badc.rl.ac.uk/svn/exarch/CMIP6dreq/tags/latest/dreqPy/docs/dreqML.pdf
[79] http://proj.badc.rl.ac.uk/svn/exarch/CMIP6dreq/tags/latest/dreqPy/docs/dreq2Schema.xsd
[80] https://www.earthsystemcog.org/projects/wip/CMIP6DataRequest
[81] http://proj.badc.rl.ac.uk/svn/exarch/CMIP6dreq/tags/latest/dreqPy/docs/dreq.xml

The software package provides both a library which can be imported as a python module and a command line interface. It is published through the Python Package Index[82] and source code (including the database) is available from an SVN repository[83]. More information about the package is available at the Data Request Web Pages described in 5.12.2 above.

5.12.5 Data Request Web Site

For users who wish to browse the database contents without the complexity of installing software or reading through an XML database, there is a website[84] providing a browsable interface, including search options [DREQFR#6].

5.13. Metadata for Climate Indices

Climate indices play an important role in the practical use of climate and weather data. Their application spans a wide range of topics, from impact assessment in agriculture and urban planning, over indispensable advice in the energy sector, to important evaluation in the climate science community. Several widely used standard sets of indices exist through long-standing efforts of WMO[85] and WCRP Expert Teams (ETCCDI[86] and ET-SCI[87]), as well as European initiatives (ECA&D[88]) and more recently EU Horizon 2020 projects and Copernicus C3S activities. They, however, focus on the data themselves, leaving much of the metadata to the individual user. Moreover, these core sets of indices lack a coherent metadata framework that would allow for the consistent inclusion of new indices that continue to be considered every day. The Data Request for Climate Indices aims to fill this gap by actively working with the larger community to establish a climate index metadata standard that builds on the Climate and Forecast Conventions.

5.12.3 Overarching structure of the metadata standard repository

The proposed metadata standard is available in a public repository[89]. As the standard is in active development, the structure has not yet settled. However, the following overarching organisation is likely to remain.

The main component is the spreadsheet file that contains several tables as separate sheets. This spreadsheet is organised to facilitate interaction and discussions to further develop the metadata

[82] https://pypi.org/project/dreqPy
[83] http://proj.badc.rl.ac.uk/svn/exarch/CMIP6dreq/tags/latest
[84] w3id.org/cmip6dr/browse.html
[85] https://public.wmo.int/en
[86] http://etccdi.pacificclimate.org/
[87] http://www.wmo.int/pages/prog/wcp/ccl/ccl17/focusarea/fa3/CCl-17FA3ET-SCIWMO.php
[88] https://www.ecad.eu/indicesextremes/index.php
[89] https://bitbucket.org/cf-index-meta/cf-index-meta/src/master/

standard. The repository also contains a small python package that can convert the spreadsheet into a set of equivalent yaml files that are better suited for automatic processing. The pipeline functionality of the repository is set up to use this python package to automatically translate the spreadsheet information into yaml files whenever the spreadsheet is updated [NFR#7] [DREQFR#1]. While the spreadsheet focuses on facilitating human interaction, the yaml files simplify programmatic access to the metadata standard thus supporting development of software tools that draw on the common standard[90] [91] [NFR#11] [DREQFR#2].

5.12.3 Current content of the spreadsheet file

As the repository is in active development, the specific details are likely to change. The metadata is contained in a spreadsheet file named *master_table.xls* that contain the following sheets:

- *README* --- explains contents and formatting of the spreadsheet file itself.
- *index_definitions* --- the main table holding the metadata for the individual indices (see Table 8). Most of the indices developed by the ETCCDI and ET-SCI are included, as are the ones produced by ECA&D. However, some of the more complex indices remain to be included.
- *variables* --- specification of input variables (following CMIP5/6 and CORDEX naming rules). This sheet also gives common aliases for the variable names.
- *index_functions* --- contains details about the calculation methods used for the indices. This is referred to in the index_definitions sheet.
- *ECA&D* --- list of indices produced by ECA&D. Many of these are already covered by existing entries in the index_table sheet.

At the moment, only the *index_definitions* and *variables* sheets of the spreadsheet file are transformed into the corresponding *index_definitions.yml* and *variables.yml* files. The *README* sheet, insofar it gives information beyond the present document, only applies to the spreadsheet. The *ECA&D* sheet mainly exists to track the open indices of that collection until they are fully integrated into the *index_definitions* table. Consequently, those two sheets are not likely to be converted in the future. The *index_functions* sheet will be added to the yaml files in the future.

Table 8: Brief description of the columns in the *index_definition* table.

Column header	*Description*
VarName	Variable name in the output file
ready	Indicator whether the metadata is complete or not (1-ready, 0- not ready)
Conventions	Preliminary

[90] https://meetingorganizer.copernicus.org/EGU2020/EGU2020-22155.html
[91] https://presentations.copernicus.org/EGU2020/EGU2020-22155_presentation.pdf

input	Required input variables
OUTPUT_standard_name	CF standard name of the output index variable (where possible)
OUTPUT_cell_methods	CF cell methods describing the index (where possible)
OUTPUT_units	UDUNITS2 units that follow established usage of the index
OUTPUT_proposed_standard_name	Possible new CF standard name for consideration
OUTPUT_proposed_cell_methods	Possible new CF cell method for consideration
OUTPUT_proposed_units	Established “units” that may not follow or meet UDUNITS2 requirements
OUTPUT_long_name	Long name describing the index
OUTPUT_reference	Source organisation for the index
allowed_freq	Preliminary
default_freq	Preliminary
index_function	Index function (external) implementing the index calculation
N_parameters	Number of parameters (arguments) required by the index_function
parameter_name	“Name” of the parameters
PARAMETER_definition	Parameters (argument) definition required by the index_function
ET_short_name	Preliminary information derived from ETCCDI and ET-SCI documentation (likely not complete or authoritative)
ET_long_name	
ET_definition	
ET_comment	

5.13.1. Architectural design

The component diagram in Fig. 40 represents the physical distribution of the main elements of the Data Request service, as described both in Section 5.12 and in Section 5.13.

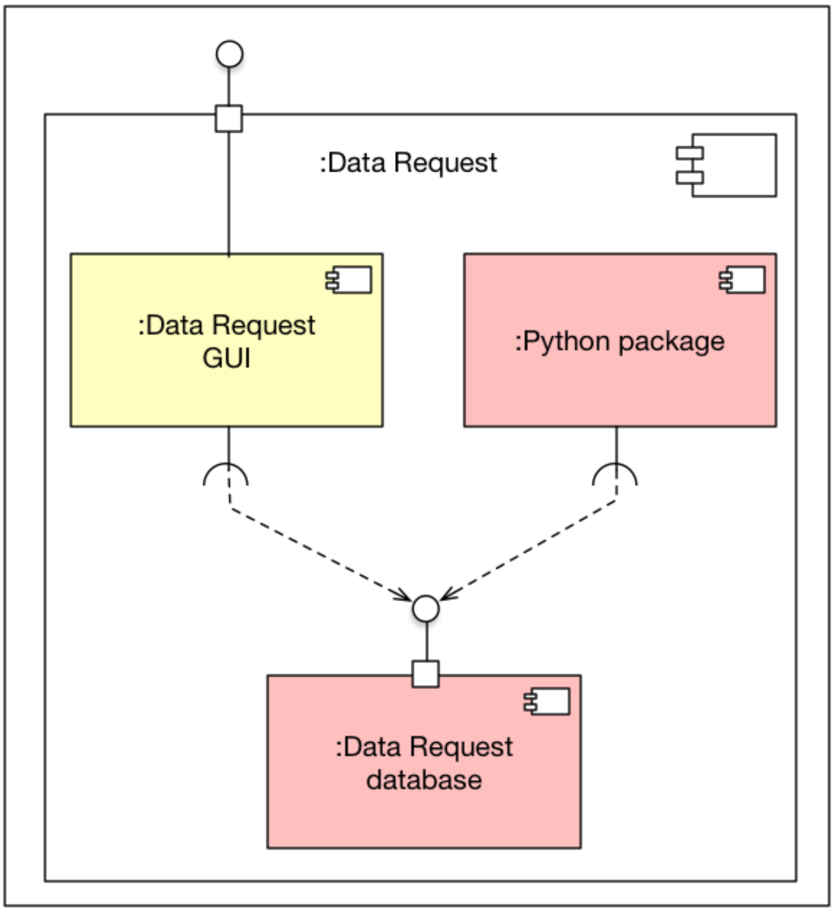

Fig. 40: Data Request component diagram

5.14. Identity Management and Access Entitlement

As part of the ESGF Future Architecture initiative, the system for identity management and access entitlement has been redesigned and updated taking into account advances in technologies and standards in this field and experiences using and operating ESGF over the ten years since it was first designed. The major changes are

- The replacement of the legacy OpenID 2.0 standard for single sign-on with OpenID Connect
- Support for user delegation using OAuth 2.0 in concert with OpenID Connect
- Token-based authentication in place of SSL client-based authentication with X.509 short-lived user certificates.

We consider these changes within the framework of the functional requirements listed in Table 9.

Table 9: Brief description of the columns in the *index_definition* table.

[IDFR#1]	Delegation
[IDFR#2]	Delegation with service chaining
[IDFR#3]	Role based access control
[IDFR#4]	Tokens
[IDFR#5]	IdP trust

5.14.1. OpenID Connect Overview

OpenID Connect provides the functionality for single sign-on in the revised architecture replacing the implementation based on the legacy OpenID 2.0. OpenID Connect builds on the OAuth 2.0 standard. Fig. 41 below shows the flow for sign-in from a browser. The user accesses a resource at a site which requires user authentication. This site acts as an OpenID Relying Party (RP) i.e. it relies on authentication assertion from a third party site, an OpenID Provider (or Identity Provider). The RP prompts the user to select an OpenID Provider (OP) to sign in with. This could be their home institution or for example a commercial provider such as Google, Microsoft or Facebook.

Once the user has selected an OP, the RP issues a HTTP redirect response requesting the user's browser redirect to the OP site (1). The user signs in and approves the request from the Relying Party (1.1 - 1.5). The OP redirects the browser back to the RP. With the special authorisation code issued by the OP, the RP can then request a token (1.8). The RP must also include a client id and secret in this request. Each RP that accesses a given OP must register with it a priori. In this registration process the RP is issued with these credentials in order to identify itself as a valid registered RP in user sign-in process.

The OP returns an access token but the OpenID Connect standard augments the OAuth 2.0 standard, so that the OP also returns a special ID Token (1.9). Tokens are digitally signed and contain metadata about the user. The ID Token is the OP's assertion that the user successfully authenticated with the OP. The RP may request additional user metadata by querying the UserInfo endpoint. The RP must present the valid access token in order to be authorised by the OP to retrieve this information.

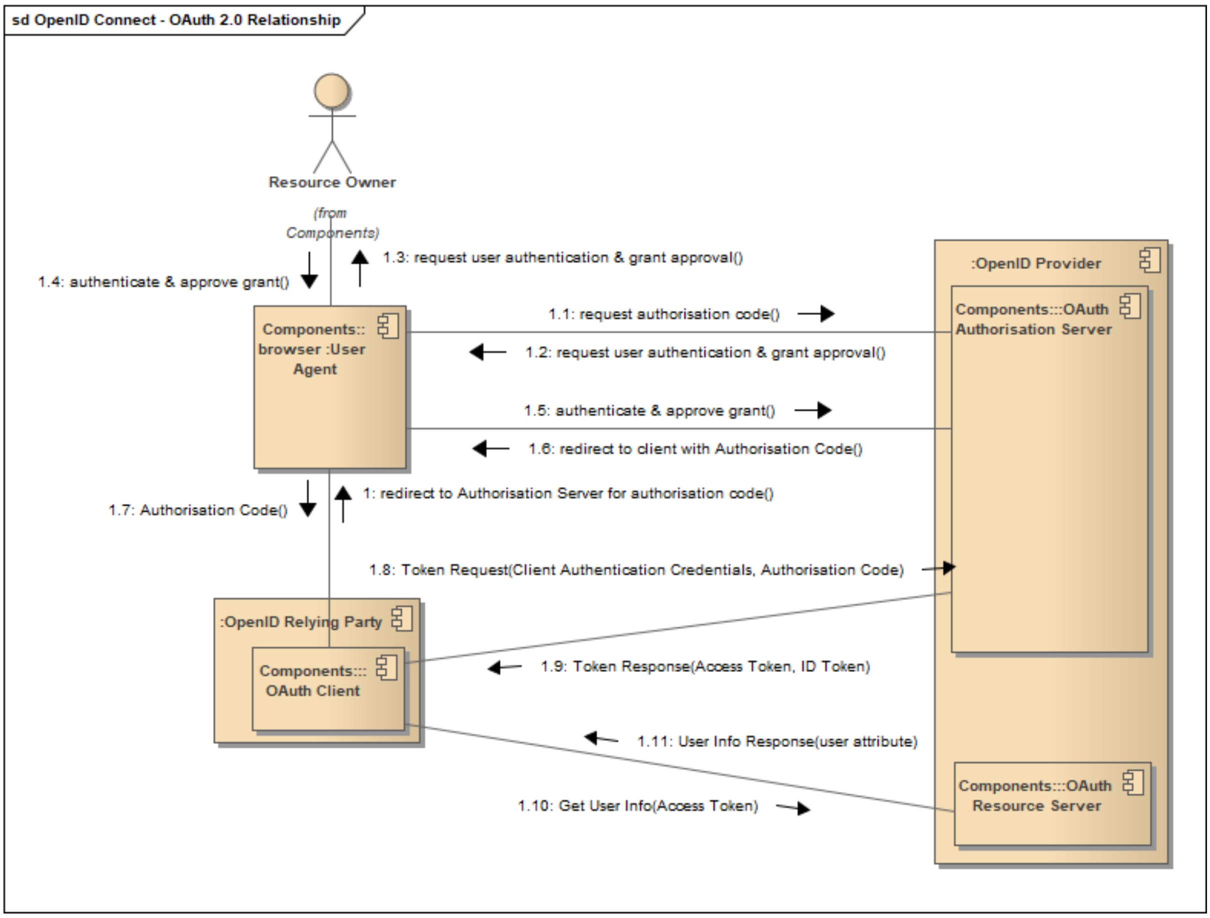

Fig. 41: Browser sign-in workflow

5.14.2. IdP Proxy concept [IDFR#5]

The new architecture for ESGF IdEA introduces the concept of an Identity Provider Proxy. This has been adopted directly from the recommendations of the AARC project architectural blueprint[92]. The rationale can be explained in the context of the legacy ESGF system: ESGF currently has many IdPs and many Relying Parties - each Data Node and Index Node instance is effectively an RP. There is an effective many-to-many trust relationship: each IdP trusts all the RPs in the federation and each RP trusts every IdP. OAuth 2.0 and OpenID Connect explicitly secure these relationships: every RP must register with every OpenID Provider (i.e an IdP) in the federation and obtain credentials to enable them to authenticate with a given OP. Registration of an RPs with OPs must be completed as a prerequisite to an RP joining the federation. Given the many-to-many relationship, this would make a complex process that would be difficult to bootstrap and maintain into the future (Fig. 42).

[92] https://aarc-project.eu/architecture/

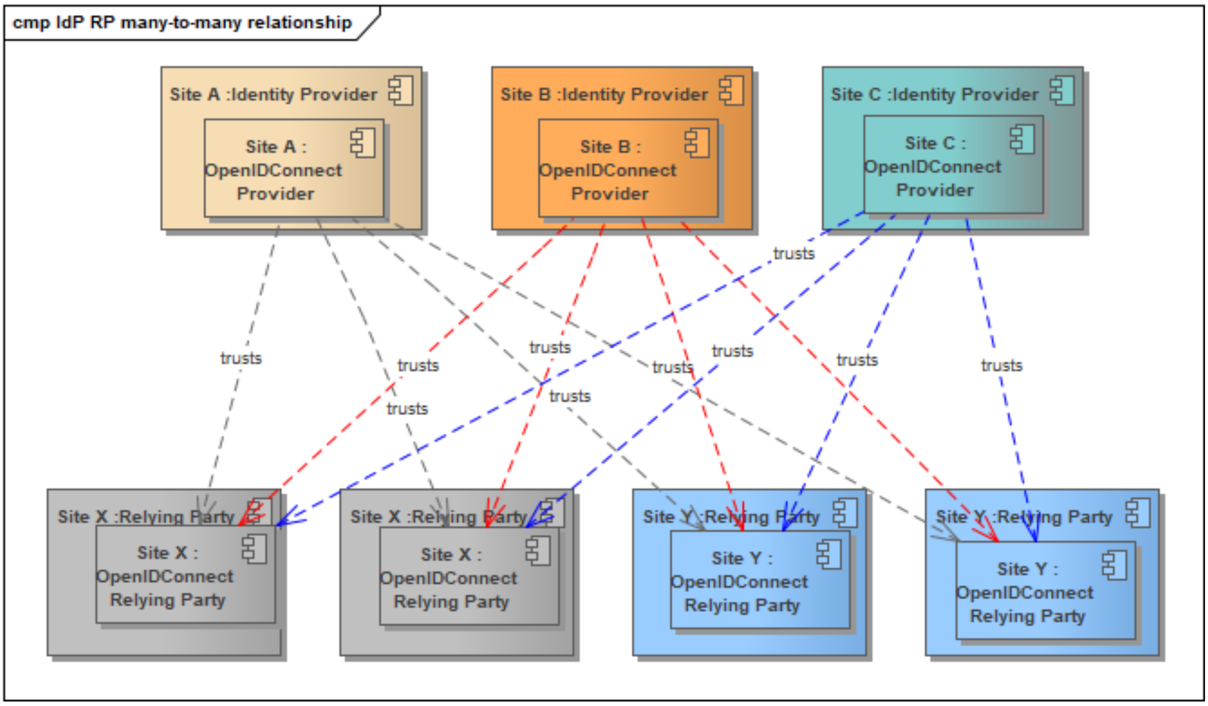

Fig. 42: Model with each Relying Party in the federation registered with every IdP. The IdP Proxy model simplifies this complex relationship by mediating trust through a central IdP intermediary

The IdP Proxy concept (Fig. 43) circumvents this many-to-many problem by placing a single federation-wide IdP as an interface between RPs and OPs. Given the adoption of OpenID Connect, the central IdP is itself implemented as an OP. Instead of every RP needing to register with every OP, RPs register only with the central OP. The OP itself acts as an RP for all the IdPs that the federation wishes to support. These include sites in the federation which wish to operate these but also opens the possibility of integrating third-party identity providers such as other research federations or commercial providers such as, for example, Google, Microsoft and Facebook.

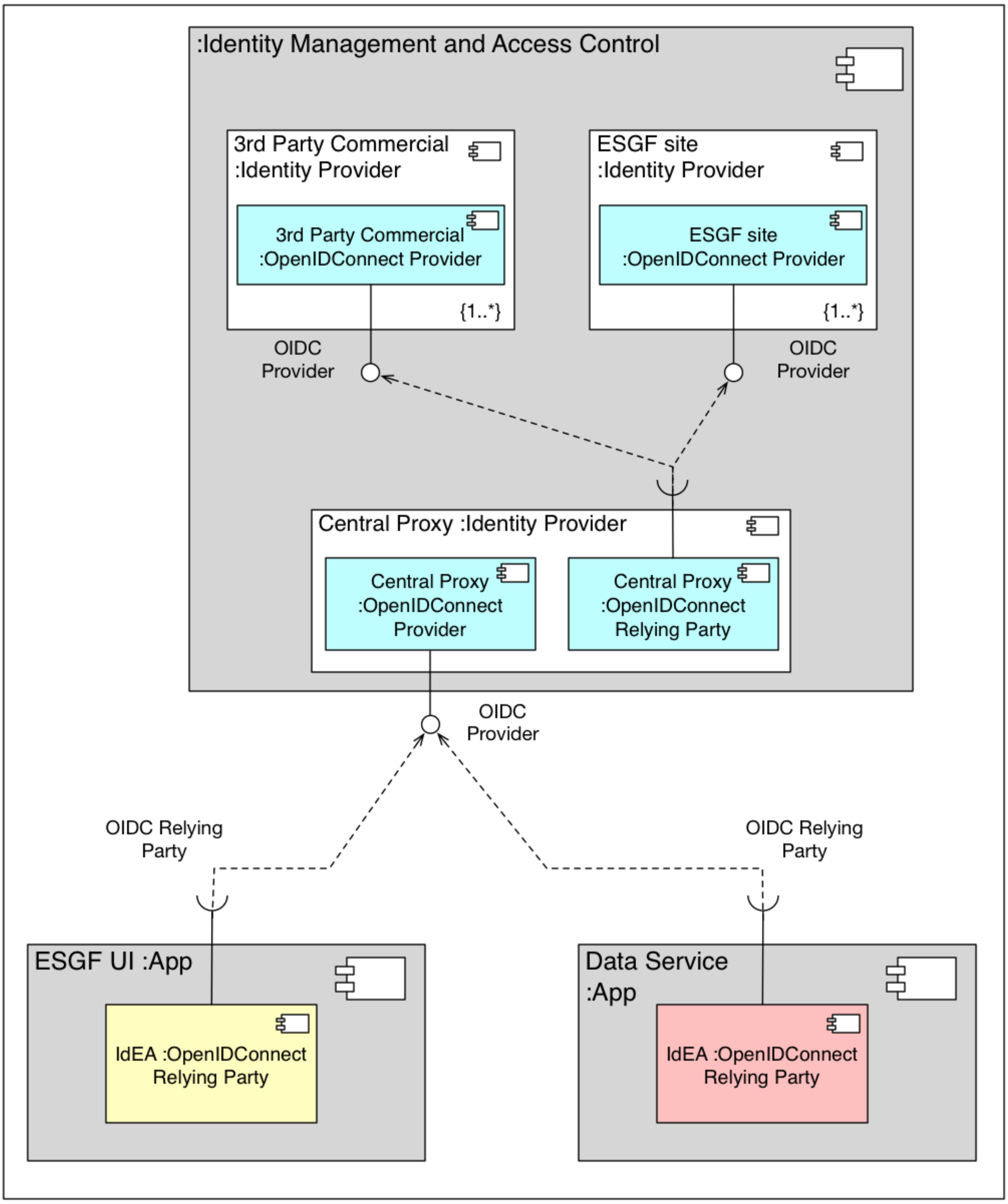

Fig. 43: IdP Proxy Architecture

5.14.3. ESGF Relying Party Registration with IdP Proxy

As described in the previous section, a set of trust relationships need to be established between IdPs and Relying Parties. The IdP Proxy model effectively splits and simplifies this model:

- Each Relying Party registers only with a single IdP i.e. the federation-wide IdP Proxy

- The IdP Proxy itself behaves as a Relying Party in order to link up with institution-specific IdPs and commercial ones such as for example, Google and GitHub.

5.14.3.1. Registration of site Relying Parties with Central IdP Proxy

This process is an important one for the deployment of ESGF nodes at a given host site. In order to complete successfully and for the site to join the federation, any services which require authentication (i.e. Relying Parties) will need to register with the IdP Proxy. The flow is shown below in Fig. 44:

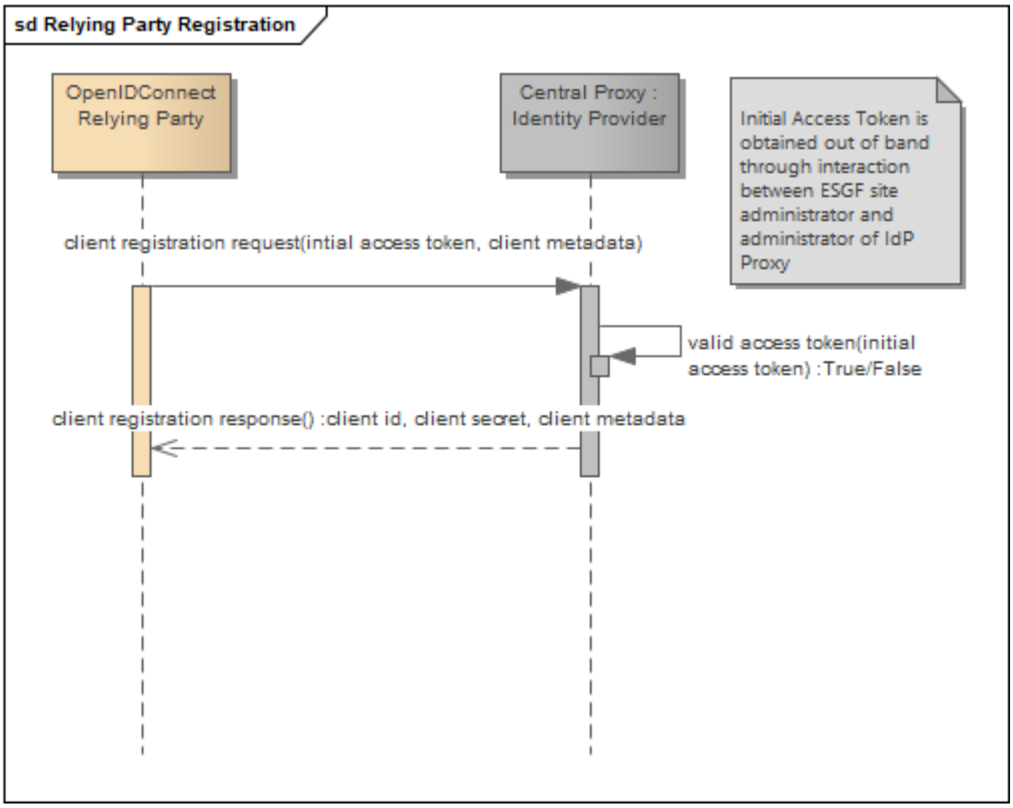

Fig. 44: OpenID Connect Dynamic Client Registration for ESGF site Relying Party with central IdP Proxy

The Relying Party makes a dynamic client registration web service request to the central proxy providing metadata about itself in order to be registered. This metadata includes for example, redirect URIs that the IdP should use during the normal OpenID Connect authentication flow for a user. The Relying Party authenticates with the Central Proxy by providing an initial access token. This is obtained out of band of this flow and is a necessary prerequisite for registration. Keycloak[93], an open source implementation of an IdP supports the ability for a user registered with it to obtain such a token. Therefore the flow to obtain a token would entail:

- ESGF site admin registers for an account with Central Proxy

[93] https://www.keycloak.org/

- (Central proxy admin grants ESGF site admin necessary privileges to obtain an initial access token)
- ESGF site admin runs deployment script to make a web service call to obtain an initial access token. The ESGF site admin enters their credentials with the Central Proxy in order for this call to be invoked
- Script then invokes dynamic client registration using the initial access token obtained

5.14.3.2. Registration of Central IdP Proxy with ESGF IdPs and Commercial IdPs

Similarly, a process is needed to register the Central IdP Proxy with institutional IdPs in the federation and Commercial IdPs that the federation wishes to link accounts with. The latter can be done at the time of deployment of the Central IdP Proxy since the commercial IdPs are already available to link with. For ESGF institutional IdPs, site administrators of these services will need to coordinate with the administrator of the central proxy so that the latter can register the central proxy with the given institutional IdP. A means of automating this process could be investigated so that it happens as part of the institutional IdP deployment process.

5.14.4. IdP Proxy Sign-in flow

Having registered an ESGF site's Relying Parties with the central proxy, users can now sign in using it. The flow is shown in Fig. 45.

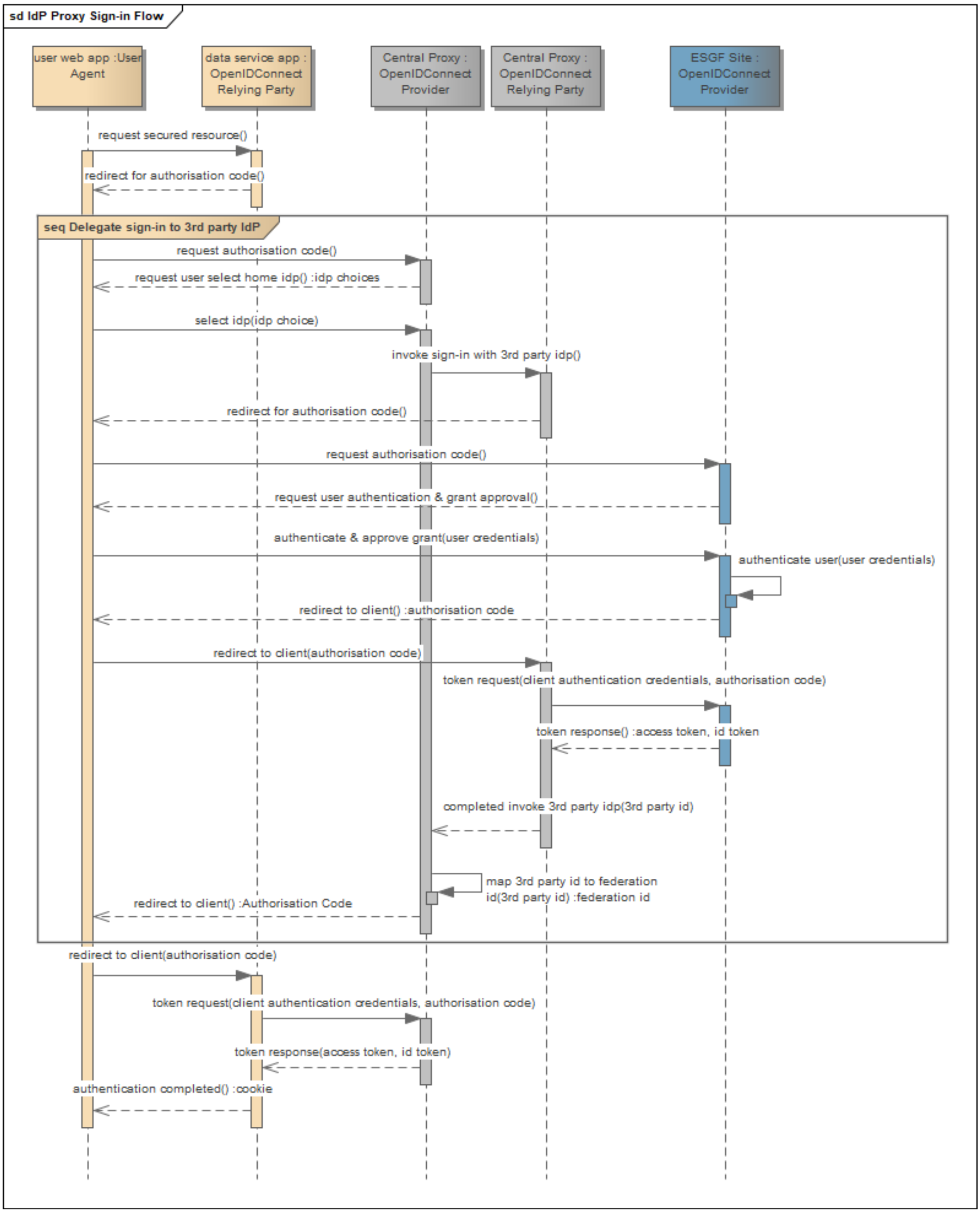

Fig. 45: sign-in flow from ESGF service (Relying Party) to ESGF Institutional IdP (shown in blue) via Central IdP Proxy

Items to note in the flow:

- OpenID Connect Relying Parties can be effectively hard-wired to redirect users to the central IdP proxy since this is the single point through which all authentication is mediated

- When the user reaches the Central IdP Proxy user interface they are presented with options as to which IdP to sign-in with - an ESGF institutional IdP e.g. CEDA, or a commercial one e.g. Google.
- On the first occasion the user signs in, the Central Proxy will need to disambiguate the user identifier it receives back from third party IdP against user identifiers already known to the proxy. If there is a name collision with an existing user identifier in use, the Central Proxy will need to handle this gracefully e.g. prompting the user to select an alternative ID. **This is an essential step.**

5.14.5. Federated Authorisation

Besides managing user authentication between different identity providers, the system must manage federated authorisation [IDFR#3]: the mediation of access control decisions between distributed entities and security domains. The existing ESGF system uses a system of Attribute Services and Virtual Organizations (VOs) to manage this federated access. Data nodes from multiple sites can enforce the same access rights by referencing a common attribute service from a VO-level authority responsible for registration of users for a given access role. This is best explained by an example. In the original system, CMIP5 data access was restricted and access was under the authority of LLNL. Users registered for the CMIP5 access roles at LLNL's site. Having registered, when they request CMIP5 data from any node in the federation, the node's authorisation service calls out to LLNL's attribute service to verify the user's entitlement to the CMIP5 access role.

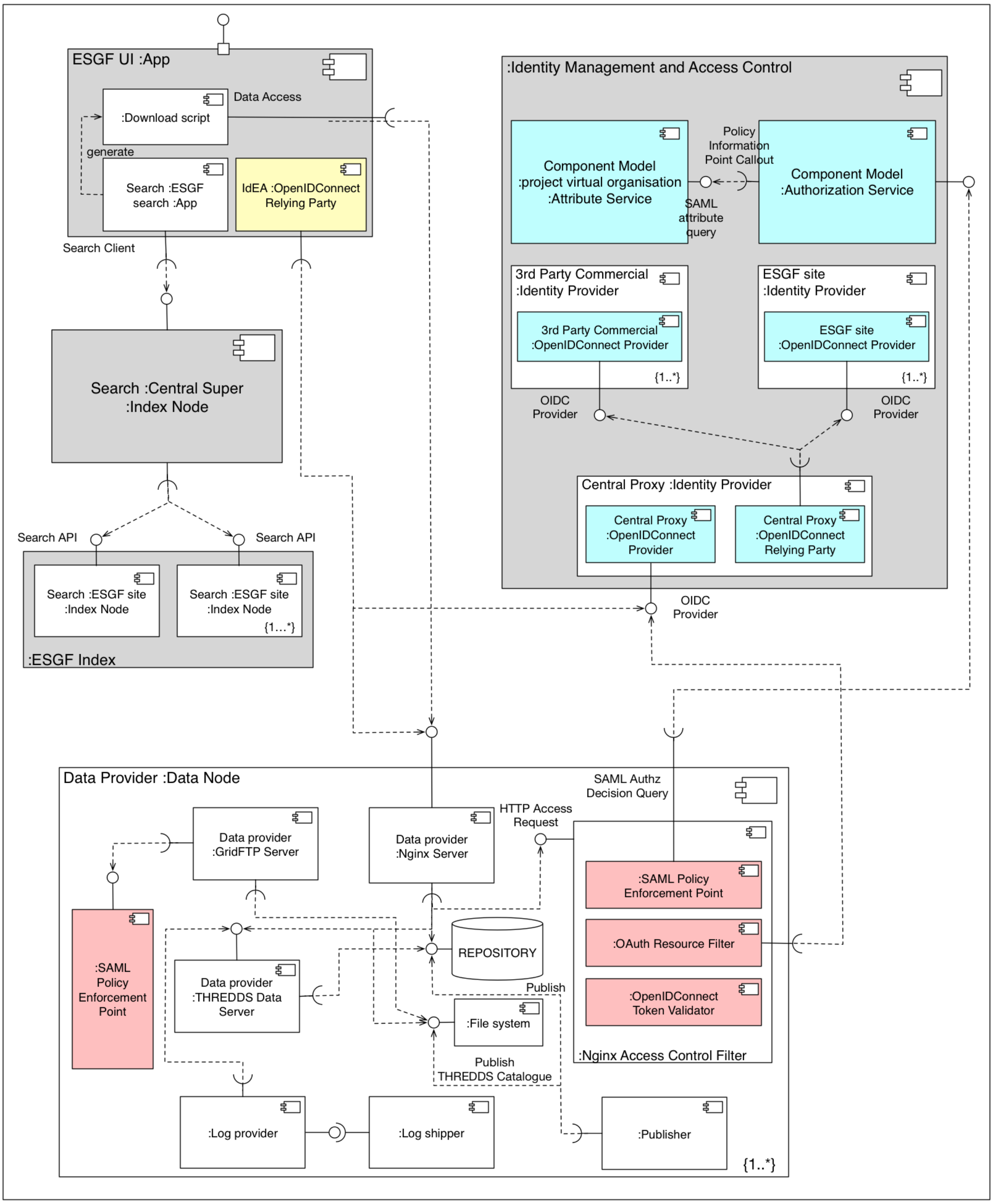

Fig. 46: Component diagram of the secured Data Node linking with IdP Proxy and Authorisation services

Fig. 46 above, shows the arrangement for a secured Data Node. The Data Node acts as a Relying Party, interacting with the Central IdP Proxy to mediate user authentication. Having authenticated, the Node checks the access policy for the resource the user has requested and if necessary, calls out to the registered Attribute Service responsible for returning information about whether the user is registered for the required role.

5.14.6. Delegation with Service Chaining

Delegation [IDFR#1] [IDFR#2] [IDFR#4] with service chaining represents the most technically demanding use case to implement. This is based explored via an existing scenario in ENES whereby the Climate4Impact Portal wishes to invoke a third-party processing service on behalf of a user. This might be for example a request to regrid some data that the portal wishes to display for the user. The third party service in this example is a WPS instance hosted at DKRZ. As a starting point, we assume that authentication with the WPS is required. This is likely to be a valid assumption since the user is accessing processing resources on computers under DKRZ's authority, something which will need to be monitored and regulated.

The first step is for the Climate4Impact Portal to invoke an OAuth 2.0 delegation in order to obtain an access token for authentication with the WPS. This could be achieved using different OAuth flows. The most common flow is an Authorisation Code Grant involving a browser redirect to the Central IdP Proxy, authentication of the user, and user granting permission for the Climate4Impact portal to act on its behalf. These steps are similar to the flow shown in Fig. 44 for OpenID Connect sign-in.

Fig. 47 below shows the subsequent steps. The Climate4Impact Portal obtains an access token from the Central IdP Proxy and then uses this to invoke the WPS with the required authorization. The flow at the WPS is broken down into these steps:

- Action as an OAuth Resource filter checking the validity and scope of the access token
- Checking the authorization of the user to invoke the given process

However, the use case is further complicated by the fact that the WPS itself needs delegated authorisation from the user to access some secured data at a third-party Data Node at Site A. In order to do so the WPS itself needs to obtain an access token. In order to achieve this the commonly used Authorization Code Grant flow is not possible because there is no direct browser interaction between the user and the WPS. A solution therefore in this case is to use the Client Credentials flow since no direct interaction is required between the user and the WPS (acting as an OAuth Client). In this case, the WPS presents its OAuth client id and secret to the OAuth Server (Central IdP Proxy) and obtains an access token in response. With this token it can re-try its request to the Data Node. The Data Node validates the token, performs the necessary authorisation checks for the data and if successful returns the data to the WPS.

On successful completion of the WPS process, it returns a response to the Climate4Impact Portal.

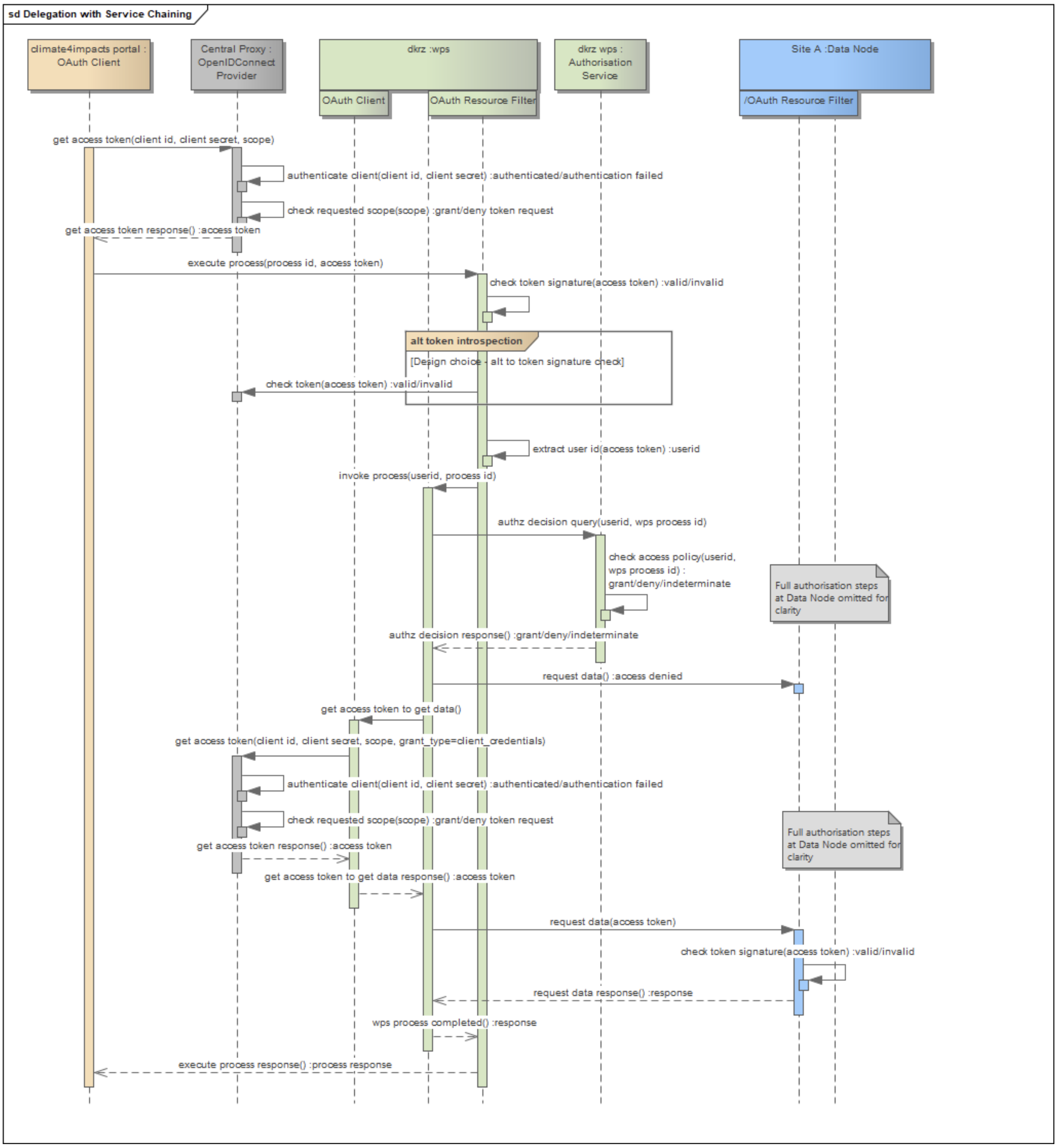

Fig. 47: User delegation with service chaining - the user invokes the Climate4Impact portal which invokes a WPS which itself requests secured data from a Data Node.

6. PLANNED RELEASES

Starting from the design of the ENES CDI software stack proposed in this document, during the IS-ENES3 lifetime a series of subsequent releases is planned, as shown in Table 10.

Table 10. ENES-CDI planned releases and related deliverables

Deliverable	Description	Deadline
D10.2	First release of the ENES CDI software stack	December 2020 (month 24)
D10.3	Second release of the ENES CDI software stack	December 2021 (month 36)
D10.5	Final release of the ENES CDI software stack	December 2022 (month 48)

A first release is scheduled for December 2020 (month 24), along with the deliverable D10.2, with the aim to have a first implementation of the CDI components related to core data distribution/management, compute and analytics, vocabulary management, documentation and impact studies. A review of the technical requirements collected by the milestone M10.1 and a subsequent refinement of the software architecture will be also provided, to be in line with emerging user needs and improve the degree of fulfillment of the requirements.

Afterwards, the deliverables D10.3 in December 2021 (month 36) and D10.5 in December 2022 (month 48) will drive the second and the third releases, respectively. The latter will be the final implementation of the ENES CDI software stack released in the IS-ENES3 project.

7. CONCLUSIONS

This document addresses the deliverable D10.1 "*Architectural document of the ENES CDI software stack*" of the IS-ENES3 project, within the WP10/JRA3 "*ENES Climate Data Infrastructure software stack developments*", and provides a detailed design of the ENES Climate Data Infrastructure software stack according to the requirements collected in the milestone M10.1 "*Technical requirements on the software stack*".

After a brief introduction to the ENES CDI (Section 2), a summary of the main functional and non-functional requirements has been presented in Section 3. Sections 4 and 5 have provided, respectively, the overall proposed architecture, the design of each software stack component as well as some preliminary implementation activities. Finally, in Section 6 a plan for future releases has been presented.

This report will drive the implementation of the ENES CDI during the IS-ENES3 project lifetime and it will periodically be revised to adapt to changes with respect to existing needs as well as to include emerging needs which were not captured before. The diagrams used give a clear picture of the whole system and how it will be shaped to fulfill the established requirements. The ENES CDI software stack releases will constitute the European contribution (developed during the IS-ENES3 project lifetime) to the long-term view provided by the international ESGF Future Architecture[94]. The different releases of the ENES CDI in D10.2, D10.3 and D10.5 reflect this development. Technical constraints and side effects are discussed in this deliverable, the ESGF Future Architecture Report as well as M5.1 "Draft architecture design" and M5.2 "ESGF CMIP6 summary" from NA4/WP5; cultural and resourcing constraints are part of the sustainability process carried out in NA1/WP2. Indeed, the transition to a new, sustainable ENES-CDI is a longer process throughout the project's runtime and it also requires international coordination and consideration of non-technical side effects.

[94] http://doi.org/10.5281/zenodo.3928223

REFERENCES

[1] Is-ENES3 Milestone 10.1 "Technical requirements on the ENES Climate Data Infrastructure software stack" - https://zenodo.org/record/3731018#.Xp3Pn-tS9yB

[2] L. Cinquini, D. Crichton, C. Mattmann, J. Harney, G. Shipman, F. Wang, R. Ananthakrishnan, N. Miller, S. Denvil, M. Morgan, Z. Pobre, G. M. Bell, C. Doutriaux, R. Drach, D. Williams, P. Kershaw, S. Pascoe, E. Gonzalez, S. Fiore, R. Schweitzer, "The Earth System Grid Federation: An open infrastructure for access to distributed geospatial data", Future Generation Computer Systems, Volume 36, 2014, Pages 400-417, ISSN 0167-739X, http://dx.doi.org/10.1016/j.future.2013.07.002.

[3] Key words for use in RFCs to Indicate Requirement Levels
https://www.ietf.org/rfc/rfc2119.txt

[4] Lin, D., Crabtree, J., Dillo, I. *et al.* The TRUST Principles for digital repositories. *Sci Data* **7,** 144 (2020). https://doi.org/10.1038/s41597-020-0486-7

[5] S. Fiore, P. Nassisi, A. Nuzzo, M. Mirto, L. Cinquini, D. Williams, G. Aloisio, "A Climate Change Community Gateway for Data Usage & Data Archive Metrics across the Earth System Grid Federation", 11th International Workshop on Science Gateways (IWSG 2019), 12-14 June 2019, Ljubljana, Slovenia

GLOSSARY

Table 11. Glossary definitions

Acronym	Explanation
AAI	Authentication and Authorization Infrastructure
ADAGUC	Atmospheric Data Access for the Geospatial User Community
APEX	Oracle Application Express
API	Application Programming Interface
C4I	Climate4Impact
CDO	Climate Data Operators
CERA	Climate and Environmental data Retrieval and Archiving
CF	Climate Forecast
CIM	Common Information Model
CLF	Common Log Format
CLI	Command Line Interface
CMIP	Coupled Model Intercomparison Project
COPDESS	Coalition for Publishing Data in the Earth and Space Sciences
Copernicus C3S	Copernicus Climate Change Service
CORDEX	Coordinated Regional climate Downscaling Experiment
CSV	Comma-Separated Values
CV	Controlled Vocabulary
DDC	Data Distribution Centre
DFM	Dimensional Fact Model
DOI	Digital Object Identifier
DRS	Data Reference Syntax
ECAS	ENES Climate Analytics Service
ECA&D	European Climate Assessment & Dataset

EGI	European Grid Infrastructure
ENES	European Network for Earth System Modelling
ENES CDI	ENES Climate Data Infrastructure
EOSC	European Open Science Cloud
ES-DOC	Earth System Documentation
ESGF	Earth System Grid Federation
ESIP	Earth Science Information Partners
ESMValTool	Earth System Model Evaluation Tool
ETCCDI	Expert Team on Climate Change Detection and Indices
ET-SCI	Expert Team on Sector-specific Climate Indices
EU	European
EUDAT	European Association of Databases for Education and Training
EUMETSAT	European Organisation for the Exploitation of Meteorological Satellites
F2F	Face to Face
FORCE11	The Future of Research Communications and e-scholarship
FR	Functional Requirement
GridFTP	File Transfer Protocol (FTP) for Grid Computing
GUI	Graphic User Interface
HPC	High Performance Computing
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure-as-a-Service
ICCLIM	Indice Calculation CLIMate
ICT	Information and Communication Technologies
IdP	Identity Provider

Input4MIPs	Input Datasets for Model Intercomparison Projects
IPCC	Intergovernmental Panel on Climate Change
IPCC AR6	IPCC Sixth Assessment Report
IPCC WGI	IPCC Working Group I
IPCC TG-Data	IPCC Task Group on Data Support for Climate Change Assessments
IPCC WGI TSU	IPCC WGI Technical Support Unit
IS-ENES	InfraStructure for the ENES modelling
IT	Information Technologies
JRA	Joint Research Activity
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
KPI	Key Performance Indicator
MIPs	Model Intercomparison Projects
NA	Networking Activity
NetCDF	Network Common Data Form
NFR	Non-Functional Requirement
OAI/PMH	Open Archives Initiative Protocol for Metadata Harvesting
Obs4MIPs	Observations for Model Intercomparisons Project
OGC	Open Geospatial Consortium
OP	OpenID Provider
OpenID RP	OpenID Relying Party
OS	Operating System
PaaS	Platform-as-a-Service
PCMDI	Program for Climate Model Diagnosis & Intercomparison
PID	Persistent Identifier
PO	Project Officer

pyessv	Python Earth Science Standard Vocabularies
REST	Representational State Transfer
RI	Research Infrastructure
SaaS	Software-as-a-Service
Scholix	A Framework for Scholarly Link eXchange
SSL	Secure Sockets Layer
Sys-admin	System administrator
TNA	Trans-National Access
TXT	Text
UI	User Interface
URL	Uniform Resource Locator
VA	Virtual Access
VO	Virtual Organization
WCRP	World Climate Research Programme
WCS	Web Coverage Service
WDCC	World Data Center for Climate
WG	Working Group
WGCM	Working Group on Coupled Modeling
WIP	WGCM Infrastructure Panel
WMO	World Meteorological Organization
WP	Work Package
WPS	Web Processing Service
XML	eXtensible Markup Language