Large-scale EXecution for Industry & Society

Deliverable D7.5

First release and test-bed deployment of Weather and Climate Data API for both in-situ unstructured observations and model data output

Co-funded by the Horizon 2020 Framework Programme of the European Union

Grant Agreement Number 825532

ICT-11-2018-2019 (IA - Innovation Action)

DELIVERABLE ID | TITLE	D7.5 | First release and test-bed deployment of Weather and Climate Data API for both in-situ unstructured observations and model data output
RESPONSIBLE AUTHOR	Emanuele Danovaro (ECMWF)
WORKPACKAGE ID | TITLE	WP7 | Weather and Climate Large-scale Pilot
WORKPACKAGE LEADER	ECMWF
DATE OF DELIVERY (CONTRACTUAL)	31/03/2020 (M15)
DATE OF DELIVERY (SUBMITTED)	27/03/2020 (M15)
VERSION | STATUS	V1.0 | Final
TYPE OF DELIVERABLE	R (Report)
DISSEMINATION LEVEL	PU (Public)
AUTHORS (PARTNER)	Emanuele Danovaro (ECMWF)
INTERNAL REVIEW	Massimo Sardo (CIMA), Klodiana Goga (LINKS)

Project Coordinator: Dr. Jan Martinovič – IT4Innovations, VSB – Technical University of Ostrava
E-mail: jan.martinovic@vsb.cz, **Phone:** +420 597 329 598, **Web:** https://lexis-project.eu

DOCUMENT VERSION

VERSION	MODIFICATION(S)	DATE	AUTHOR(S)
0.1	First Draft	15/01/2020	Emanuele Danovaro (ECMWF)
0.2	First Internal Review	26/02/2020	Massimo Sardo (CIMA)
0.3	Second Internal Review	10/03/2020	Klodiana Goga (LINKS)
1.0	First public version	25/03/2020	Emanuele Danovaro (ECMWF)

GLOSSARY

ACRONYM	DESCRIPTION
API	Application Program Interface
BUFR	Binary Universal Form for the Representation of meteorological data
FDB	Field DataBase
GRIB	Gridded Binary or General Regularly-distributed Information in Binary form
HDF5	Hierarchical Data Format version 5
HPCF	High-Performance Computing Facility
JSON	JavaScript Object Notation
MARS	Meteorological Archival and Retrieval System
NetCDF	Network Common Data Form
NVRAM	Non-Volatile Random-Access Memory
NWP	Numerical Weather Prediction
ODB	A data format for observations
POSIX	Portable Operating System Interface for Unix
PWS	Personal Weather Station
REST	Representational State Transfer
URL	Uniform Resource Locator
WCDA	Weather and Climate Data API
WMO	World Meteorological Organization
WRF-ARW	Weather Research and Forecasting Model - Advanced Research WRF

TABLE OF PARTNERS

ACRONYM	PARTNER
Avio Aero	GE AVIO SRL
AWI	ALFRED WEGENER INSTITUT HELMHOLTZ ZENTRUM FUR POLAR UND MEERESFORSCHUNG
BLABS	BAYNCORE LABS LIMITED
Bull/Atos	BULL SAS
CEA	COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES
CIMA	Centro Internazionale in Monitoraggio Ambientale - Fondazione CIMA
CYC	CYCLOPS LABS GMBH
ECMWF	EUROPEAN CENTRE FOR MEDIUM-RANGE WEATHER FORECASTS
GFZ	HELMHOLTZ ZENTRUM POTSDAM DEUTSCHESGEOFORSCHUNGSZENTRUM GFZ
IT4I	VYSOKA SKOLA BANSKA - TECHNICKA UNIVERZITA OSTRAVA / IT4Innovations National Supercomputing Centre
ITHACA	ASSOCIAZIONE ITHACA
LINKS	FONDAZIONE LINKS / ISTITUTO SUPERIORE MARIO BOELLA ISMB
LRZ	BAYERISCHE AKADEMIE DER WISSENSCHAFTEN / Leibniz Rechenzentrum der BAdW
NUM	NUMTECH
O24	OUTPOST 24 FRANCE
TESEO	TESEO SPA TECNOLOGIE E SISTEMI ELETTRONICI ED OTTICI

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

The Weather and Climate Data API (WCDA) is the data management layer for curated weather data in LEXIS. It is responsible for storing and organizing weather observations from a variety of sources (including in-situ unstructured observations), as well as numerical weather prediction outputs and intermediate weather data. This report focuses on WCDA architecture and provides detailed information for WCDA exploitation by WP7 workflow designers. The first release of WCDA focuses on management of global-scale NWP GRIB model output. Extensions to meso-scale NWP NetCDF model output and unstructured observations are foreseen and described as part of this report.

Position of the deliverable in the whole project context

This deliverable is part of Task 7.1 entitled “Develop Weather and Climate Data API”, which is the first task in WP7 (Weather and Climate Large-scale Pilot). Observational datasets and model outputs are crucial in weather & climate workflows. WCDA plays a pivotal role in data annotation, storage and distribution. An analysis of relevant datasets has been provided in deliverables D7.1: Architectural requirements and system design for interchange of weather & climate model output between HPC and Cloud environments [1] (M3) and D7.2: Architectural requirements and system design for interchange of in-situ unstructured weather & environmental observations [2] (M3). Most WP7 models and the YSTIA workflow orchestrator [2] will benefit from the outcome of this deliverable.

Description of the deliverable

The deliverable begins with an introduction to the WCDA architecture and supported datasets. Section 2 provides all details for WCDA user access, by describing the RESTful endpoints and the WCDA python client (polytope) exposing a simple CLI interface and a python native API. Available instances of the WCDA are listed in Section 3, while Section 4 describes the further developments to fulfil all the requirements described in deliverables D7.1 and D7.2.

1 WCDA ARCHITECTURE

ECMWF has developed, and is actively evolving, an optimized Weather and Climate Data API (WCDA) to enable the exchange of weather and climate data among models composing a Lexis workflow. The WCDA is implemented as a RESTful API and follows industry norms in terms of GET/POST/DELETE operations. Due to the sizes of data involved, requests to the WCDA may take a considerable time to fulfil (ranging from minutes to hours). As a result, downloading requests to the WCDA will be asynchronous, returning a poll-able URL where the client may check the progress and finally retrieve their data.

Due to its RESTful API design the WCDA will enable pilot applications to directly submit requests and retrieve the required weather and climate data as long as they can see the WCDA end-points; this may be useful during model development and tuning.

Figure 1 represents the current technical design for the WCDA. ECMWF has a flexible object storage solution which is well optimized for weather & climate data, called FDB (Fields Database). The FDB is an internally provided service, used as part of ECMWF's weather forecasting software stack. It operates as a domain-specific object store, designed to store, index and serve meteorological fields produced by ECMWF's forecast model. The FDB serves as a 'hot-object' cache inside ECMWF's high-performance computing facility (HPCF) for the Meteorological Archival and Retrieval System (MARS). MARS makes many decades of meteorological observations and forecasts available to end users. Around 80% of MARS requests are served from the FDB directly, typically for very recently produced data. A subset of this data is later re-aggregated and archived into the permanent archive for long-term availability. Usage of the FDB will allow the WCDA to meet the requirements of data sizes (MBs to TBs) and will perform well even with heterogeneous data sizes.

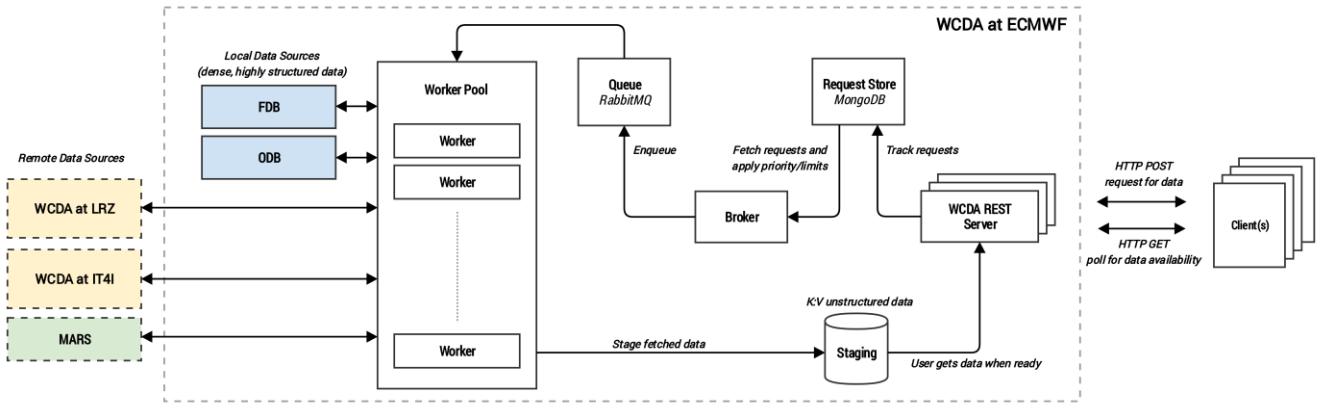

Figure 1 - Technical design for Weather and Climate Rest API

1.1 DATA MODEL & AVAILABLE ARCHIVES

The current release of WCDA is designed to efficiently handle global NWP model outputs, encoded in GRIB file format. GRIB (General Regularly-distributed Information in Binary form) is a concise data format, used in meteorology to store historical and forecast weather data. It is standardized by the World Meteorological Organization's Commission for Basic Systems. GRIB files are a collection of self-contained records of 2D data (GRIB messages), and the individual records stand alone as meaningful data, with no references to other records or to an overall schema. Each GRIB record has two components - the part that describes the record (the header), and the actual binary data itself. The data in GRIB-1 are typically converted to integers using scale and offset, and then bit-packed. GRIB-2 also has the possibility of compression.

Further WCDA releases will add support to mesoscale NWP model outputs encoded in NetCDF format (see Section 4.1) and weather observations coming from the LEXIS smart gateway, authoritative weather stations and personal

weather stations (PWS), as described in Deliverable D7.2 [2]. Observational data will be encoded in ODB or BUFR format.

1.2 INTEGRATION WITH LEXIS AAI

WCDA has been developed with an abstraction layer to adapt easily to existing authentication infrastructure. Full integration with Lexis AAI is not yet available but will be provided in the next release of WCDA.

Request Authorization is performed at two levels: a coarse-level authorization will be offered by Lexis AAI and will identify users authorized to access weather & climate data (mainly WP6 and WP7 users), and a second, fine-grained authorization level, will be integrated with ECMWF licencing and authorization system to finely check all user requests. As an example, a WCDA user may be authorized to access archive data, but not allowed to fetch real-time operational data.

2 USER DATA ACCESS

2.1 REST API

WCDA can serve different collections of data (e.g. forecast data, climate data), that are represented as separate end-points. HTTP requests to each end-point will include authorization metadata in the header and data-specific metadata in the message content. Data will be indexed using multiple scientifically meaningful keys. Endpoints are provided to query collection contents and their data schemas. Table 1 provides a preliminary list of WCDA end-points.

URL	Method	Result
.../api/v1/auth/users	POST	200: confirmation of successful registration
	DELETE	200: confirmation of removal of user
.../api/v1/auth/tokens	POST	200: token
...api/v1/user	GET	200: user information such as request limits and number of requests submitted
.../api/v1/requests	GET	200: a list of all user's live requests (json)
.../api/v1/requests/{collection}	POST	202: request accepted, polling URL returned for user to poll request status 303: data immediately returned
	GET	200: a list of all user's live requests (json)
.../api/v1/requests/{request_id}	GET	200: request corresponding to request_id returned (json)
	DELETE	200: request corresponding to request_id removed, confirmation returned
.../api/v1/downloads/{download_hash}	GET	200: data returned (in GRIB or json format)
.../api/v1/collections	GET	200: a list of available collections (json)
.../api/v1/collections/{collection}	GET	200: a request schema of the specified collection (json)

Table 1 WCDA RESTful end points

2.2 WCDA PYTHON CLIENT

To simplify adoption and integration of WCDA in Lexis workflows, we have developed a small python client, called polytope. It can be executed as CLI application, or as a simplified Python 3 API interface.

2.2.1 WCDA CLI commands

An authorized user can query WCDA from a shell by issuing the following commands:

```
polytope list config
polytope set config username <username>
polytope login
polytope list credentials
```

```
polytope describe user
```

Query parameters are specified using MARS language and can be embedded in-line, as in the following example:

```
polytope retrieve -e "collection = fc, \
    stream = oper,
    levtype = sfc,
    param = 165.128/166.128/167.128,
    dataset = interim,
    step = 0,
    grid = 0.75/0.75,
    time = 00/06/12/18,
    date = 2011-03-23/to/2011-03-24,
    type = an,
    class = ei,
    expver = 0001,
    target = " output_cli.grib
```

As an alternative, query parameters can be specified in a separate file, as in the following example:

```
cat > request.yaml <<EOF
{
  'collection' : 'fc',
  'stream' : 'oper',
  'levtype' : 'sfc',
  'param' : '165.128/166.128/167.128',
  'dataset' : 'interim',
  'step' : '0',
  'grid' : '0.75/0.75',
  'time' : '00/06/12/18',
  'date' : '2011-03-23/to/2011-03-24',
  'type' : 'an',
  'class' : 'ei',
  'expver' : '0001',
  'target' :
}
EOF

polytope retrieve request.yaml output_cli.grib
```

Users can check the status of active requests and revoke a running one with the following commands:

```
polytope list requests
polytope revoke request-id
```

2.2.2 API example

Simple Python 3 application using polytope client:

```
from polytope_client import api
c = api.Client()

# insert your own <username>
c.set_config('username', 'your_username')

request = """
    collection = fc,
    stream = oper,
    levtype = sfc,
    param = 165.128/166.128/167.128,
    dataset = interim,
    step = 0,
    grid = 0.75/0.75,
    time = 00/06/12/18,
    date = 2015-03-23/to/2015-03-24,
    type = an,
    class = ei,
    expver = 0001,
    target = """

c.retrieve(request, 'output_api.grib', inline_request = True)
```

3 TEST-BED DEPLOYMENT

We decided to deploy three instances of WCDA, one for each Lexis computing center: ECMWF, LRZ and IT4I. In the following we will list WCDA endpoints and capabilities for each site.

3.1 ECMWF

WCDA rev 0.1 has been deployed on ECMWF cloud infrastructure since December 2019.

URL/IP: polytope.ecmwf.it

Network restrictions: none

Data resources: ECMWF MARS archive, ECMWF operational IFS model output

3.2 LRZ

WCDA rev 1.0 has been deployed on LRZ experimental DataStorage machine since March 2020

URL/IP: 10.156.247.241

Network restrictions: LRZ internal access or VPN

Data resources: Attached FDB storage

For integration with the LEXIS orchestrator, it will not be possible for pilot applications running on HPC compute nodes to directly interact with the WCDA due to a lack of internet access on these nodes in LRZ. Therefore, WCDA instance running on LRZ-LEXIS experimental storage will retrieve local and/or remote (ECMWF) data and insert this data into a staging area temporarily; the orchestrator will then manage moving this data to the relevant compute node where the pilot application is running.

3.3 IT4I

WCDA will be installed on IT4I cloud resources and will exploit Ceph data storage [3]. Endpoint and availability are still to be confirmed.

4 FURTHER DEVELOPMENTS

4.1 NETCDF INTEGRATION

To guarantee fast and efficient data access, WCDA adopts a field-based data source called FDB for storing NWP model output. FDB inherits WMO GRIB field naming convention and supports semantic field retrieval based on GRIB field names and/or IDs.

In the requirement elicitation phase, it emerged the need to extend WCDA / FDB to support NetCDF file generated by LEXIS workflows.

NetCDF (Network Common Data Form) is a self-describing, machine-independent data format that supports the creation, access, and sharing of array-oriented scientific data. Version 4.0 (released in 2008) allows the use of the Hierarchical Data Format version 5 (HDF5). As described in D7.1 [1], HDF5 is a very generic, filesystem-like data format with user-defined metadata stored in the form named attributes.

Full support to general NetCDF files may strongly affect some key features of WCDA and FDB such as efficient field encoding and semantic queries, and is not required by LEXIS workflows, so we restricted our support to a widely accepted subset of NetCDF called NetCDF Climate and Forecast Metadata Conventions (NetCDF-CF).

In LEXIS workflows, most NetCDF files are generated by WRF-ARW mesoscale NWP model, thus we have defined a mapping between relevant WRF model outputs, encoded in NetCDF-CF, and GRIB. The field mapping is reported in Table 2.

NetCDF-CF name	WRF name	GRIB name	GRIB ID	Description
surface_specific_humidity	Q2	2sh	174096	2-meter specific humidity
surface_temperature	T2	2t	167	2-meter temperature
surface_air_pressure	PSFC	sp	134	Surface pressure
snowfall_amount	ACSNOW	sdwe	260056	ACCUMULATED SNOW - Water equivalent of accumulated snow depth
lwe_thickness_of_ stratiform_precipitation_ amount	RAINNC	lsp	142	ACCUMULATED TOTAL GRID SCALE PRECIPITATION
eastward_wind	U_PL	u	131	Pressure level data, U wind
northward_wind	V_PL	v	132	Pressure level data, V wind
air_temperature	T_PL	t	130	Pressure level data, Temperature
relative_humidity	RH_PL	r	157	Pressure level data, Relative humidity
geopotential_height	GHT_PL	gh	156	Pressure level data, Geopotential Height
dew_point_temperature	TD_PL	dpt	3017	Pressure level data, Dew point temperature
specific_humidity	Q_PL	q	133	Pressure level data, Mixing ratio

Table 2 GRIB - NetCDF-CF - WRF field mapping

WCDA will be extended to support NetCDF-CF data, by sharing a NetCDF file in the supported data fields, efficiently stored in FDB as GRIB messages. NetCDF data files can be re-constructed by re-composing FDB fields in NetCDF

format. Data conversion may exploit Data harmonization services provided by Task 3.3 and be efficiently implemented on LEXIS DDI Burst Buffers, available at LRZ and IT4I computing premises.

4.2 CEPH BACK-END @IT4I

The FDB is able to scale efficiently to arbitrary numbers of writer processes whilst retaining a continuously consistent state from the perspective of reading processes, irrespective of any potential failures. To guarantee such level of consistency, each FDB writer, writes its own data and indexes information without reference to other processes, with the data only added to the global namespace once the writing process finalises its execution and the data has been flushed. A single reference is added to the end of a global Table Of Contents, with this step being a rare event and the only one that requires synchronisation. This can be achieved using only the guarantees of the append operation on a POSIX filesystem, obviating the need for MPI or any explicit synchronisation or locking mechanism. This also permits the writers to be truly independent of each other. By deferring writing the indexing structure to disk until finalisation of the index, it ensures that subsequent readers are accessing a read-only index structure that won't be updated again, making the efficient loading and navigation of the metadata in-memory a trivial task.

To future-proof the design of the FDB for coping with upcoming changes in the scientific and technological landscape, an active choice has been made to clearly delineate a boundary between the metadata handling and control in the front-end and backends which perform the storage. To implement the immediately required functionality, only one backend is required (operating on a POSIX- compliant filesystem, currently Lustre [4]). However, the system has been designed taking into account the possibilities of upcoming technology, such as NVRAM (Non-Volatile Random-Access Memory), and distributed storage systems providing strong guarantee on data consistence. Ceph can provide a reasonable support to data consistency, thus we are able to extend FDB to exploit Ceph storage via low level librados. And it will be adopted on the WCDA instance deployed on IT4I computing resources.

5 CONCLUSION

The Weather and Climate Data API has the goal to provide a clean and effective interface for interchange of in-situ unstructured weather & environmental observations and weather products. This document focuses on WCDA architecture, usage, and deployment on Lexis infrastructure. It also describes further planned developments, which will extend WCDA capabilities.

REFERENCES

[1] LEXIS Deliverable, *D7.1 Architectural Requirements and System Design for Interchange of Weather & Climate Model Output between HPC and Cloud Environments.*

[2] LEXIS Deliverable, *D7.2 Architectural requirements and system design for interchange of in-situ unstructured weather & environmental observations.*

[3] S. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long and C. Maltzahn, "Ceph: A Scalable, High-Performance Distributed File System," in *Proceedings of the 7th Conference on Operating Systems Design and Implementation (OSDI '06)*, November 2006.

[4] P. Schwan, "Lustre: Building a file system for 1000-node clusters.," in *Proceedings of the 2003 Linux Symposium*, 2003.