Large-scale EXecution for Industry & Society

Deliverable D7.8

Final Deployment of Test-bed Infrastructure Components with Full Interoperable Model Layers

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	D7.8 | Final Deployment of Test-bed Infrastructure Components with Full Interoperable Model Layers
RESPONSIBLE AUTHOR	Fabien Brocheton (NUM)
WORKPACKAGE ID | TITLE	WP7 | Weather and Climate Large-scale Pilot
WORKPACKAGE LEADER	CIMA
DATE OF DELIVERY (CONTRACTUAL)	30/06/2021 (M30)
DATE OF DELIVERY (SUBMITTED)	29/07/2021 (M31)
VERSION | STATUS	V1.0 | Final
TYPE OF DELIVERABLE	R (Report)
DISSEMINATION LEVEL	PU (Public)
AUTHORS (PARTNER)	Fabien Brocheton (NUM); Emanuele Danovaro (ECMWF); Paola Mazzoglio (ITHACA); Paolo Pasquali (ITHACA); Antonio Parodi (CIMA); Andrea Parodi (CIMA); Mirko d'Andrea (CIMA); Marco Ferretti (TESEO); Paolo Ferretti (TESEO); Laurent Ganne (Atos); Alberto Scionti (LINKS); Mohamad Hayek (LRZ); Martin Golasowski (IT4I)
INTERNAL REVIEW	Tomáš Martinovič (IT4I); Vytautas Jancauskas (LRZ); Stephan Hachinger (LRZ)

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	Table of Content	15/05/2021	Fabien Brocheton (NUM)
0.2	First complete draft of the document	30/06/2021	Various contributions from all Authors
0.3	Update after internal review 1	20/07/2021	Fabien Brocheton (NUM), Antonio Parodi (CIMA)
1.0	Final check of the deliverable	29/07/2021	Katerina Slaninova (IT4I)

GLOSSARY

ACRONYM	DESCRIPTION
3DVAR	Three-dimensionnal variationnal assimilation technique
4DVAR	Four-dimensionnal variationnal assimilation technique
AAI	Authentication and Authorization Infrastructure
ADMS	Atmospheric Dispersion Modelling System
AFWA	U.S. Air Force Weather Agency
API	Application Programming Interface
ARW	Advanced Research Weather Research and Forecasting
ASCII	American Standard Code for Information Interchange
BUFR	Binary Universal Form for the Representation of meteorological data
CGM	Conjugate Gradient Method
DAM	[LEXIS] Dynamic Allocation Module
DDI	[LEXIS] Distributed Data Infrastructure
EOF	Empirical Orthogonal Function
ERDS	Extreme Rainfall Detection System
ETKF	ensemble transform Kalman filter
GDAL	Geospatial Data Abstraction Library
GFS	NCEP operational Global Forecast System
GPM	Global Precipitation Measurement
HDF5	Hierarchical Data Format version 5
HEAppE	High-End Application Execution Middleware

HPC	High Performance Computing
ICPD	Italian Civil Protection Department
IFS	Integrated Forecasting System of ECMWF
IMERG	Integrated Multi-satellitE Retrievals for GPM IR Infrared
IoT	Internet Of Things
iRODS	Integrated Rule-Oriented Data System
JAXA	Japan Aerospace Exploration Agency
KPI	Key Performance Indicator
LITTLE_R format	an ASCII-based observation file format used in WRF
MARS	Meteorological Archival and Retrieval System of ECMWF
NASA	National Aeronautics and Space Administration
NCAR	National Center for Atmospheric Research
NCEP	National Centers for Environmental Prediction
NETCDF	Network Common Data Form NWP Numerical Weather Prediction
NMC method	National Meteorological Center method
NOAA	National Oceanic and Atmospheric Administration
NWP	Numerical Weather Prediction
PREPBUFR	Processing performed at NCEP on BUFR meteorological data
QNM	Quasi-Newton Method
QOS	Quality of Service
RISICO	RISchio Incendi e COordinamento
TOSCA	Topology and Orchestration Specification for Cloud Applications
WCDA	Weather and Climate Data API
WRF	Weather Research and Forecasting
WRFDA	WRF variational data assimilation
WMS	Web Map service
WPS	WRF Preprocessing System
Yorc	Ystia orchestrator
Ystia	ATOS Ystia Orchestrator, based on TOSCA and Alien4Cloud technologies

TABLE OF PARTNERS

ACRONYM	PARTNER
Avio Aero	GE AVIO SRL
Atos	BULL SAS
AWI	ALFRED WEGENER INSTITUT HELMHOLTZ ZENTRUM FUR POLAR UND MEERESFORSCHUNG
BLABS	BAYNCORE LABS LIMITED
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
EURAXENT	MARC DERQUENNES
GFZ	HELMHOLTZ ZENTRUM POTSDAM DEUTSCHESGEOFORSCHUNGSZENTRUM GFZ
ICHEC	NATIONAL UNIVERSITY OF IRELAND GALWAY / Irish Centre for High-End Computing
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

EXECUTIVE SUMMARY **7**

1 INTRODUCTION **9**

2 MAPPING OF WP7 USE CASE TO LEXIS **10**

2.1 GENERAL VIEW 10
2.2 UPDATE ON DATA EXCHANGE 11
2.2.1 DDI (LRZ) *11*
2.2.2 WCDA API and provision of IFS global forecasts (ECMWF) *11*
2.3 UPDATE ON ORCHESTRATION AND SCHEDULING MECHANISM 13

3 STATUS OF DEPLOYMENT OF SPECIFIC WP7 COMPONENTS **13**

3.1 SMART GATEWAY FOR IOT DATA 13
3.2 PROVISION OF MOJI MOBILE PHONE DATA 13

4 EXECUTION OF WRF REGIONAL FORECAST OVER ITALY AND FRANCE **14**

4.1 ASSIMILATION 14
4.1.1 WRFDA structure and observational capability *14*
4.1.2 WRFDA variational data assimilation *16*
4.1.3 3DVAR implementation *17*
4.2 VIRTUALIZATION/ DOCKER / MAPPING ON HPC 18
4.2.1 WPS docker *18*
4.2.2 LEXIS downloader docker *18*
4.2.3 Download-ECMWF-data *18*
4.2.4 WRFDA HEAppE template *19*

5 DEPLOYMENT OF WP7 APPLICATIONS **19**

5.1 CONTINUUM HYDROLOGICAL MODELLING 19
5.1.1 Workflow and updated description *19*
5.1.2 Orchestration and Docker for virtualization *20*
5.2 RISICO RISK FIRE MODELLING 21
5.2.1 Workflow and updated description *21*
5.2.2 Orchestration and Docker for virtualization *21*
5.3 URBAN AND INDUSTRIAL AIRQUALITY FORECASTING 22
5.3.1 Workflow and updated description *22*
5.3.2 Orchestration and Docker for virtualization *23*
5.4 AGRICULTURAL DECISION MAKING 25
5.4.1 Workflow and updated description *25*
5.4.2 Orchestration and Docker for virtualization *25*
5.5 EXTREME RAINFALL DETECTION SYSTEM 25
5.5.1 Workflow and updated description *25*
5.5.2 Orchestration and Docker for virtualization *26*

6 SUMMARY **28**

A DETAILS ON BACKGROUND ERROR CALCULATION FOR ASSIMILATION WITH WRF **31**

B DETAILS ON WPS DOCKER **33**

C DETAILS ON LEXIS DOWNLOADER DOCKER **34**

D DETAILS ON FLOOD-PROOFS DOCKER **35**

E DETAILS ON RISICO DOCKER **39**

LIST OF TABLES

TABLE 1 THE WRFDA OBSERVATION CATALOGUE 15
TABLE 3 ADMS EXECUTION SCRIPT PARAMETERS 24
TABLE 2 BACKGROUND ERROR COVARIANCES MATRIX OPTIONS 32

LIST OF FIGURES

FIGURE 1 POSITION OF WP7 IN THE LEXIS PROJECT 7
FIGURE 2 LIST OF THE DIFFERENT WP7 APPLICATIONS 9
FIGURE 3 WP7 WORKFLOWS MAPPING ON LEXIS PLATFORM 10
FIGURE 4 WRFDA IN THE WRF MODELLING SYSTEM. 15
FIGURE 5 OUTER LOOP SCHEMATIZATION 17
FIGURE 6 LEXIS HYDROLOGICAL PREDICTION WORKFLOW DESCRIPTION 20
FIGURE 7 LEXIS FOREST FIRE PREDICTION WORKFLOW DESCRIPTION 21
FIGURE 8 LEXIS AIR QUALITY PREDICTION WORKFLOW DESCRIPTION 22
FIGURE 9 EXCERPT FROM THE ADMS POWERSHELL SCRIPT 25
FIGURE 10 ERDS PREDICTION WORKFLOW DESCRIPTION 26

EXECUTIVE SUMMARY

The goal of the LEXIS (Large-scale Execution for Industry & Society) project is to design and implement a platform for executing complex workflows in which HPC, Big Data and Cloud domains will converge. Such a platform will take advantage of the large-scale, geographically distributed resources exposed by HPC centres through their respective infrastructures. To this end, a co-design activity was proposed and is aimed at ensuring the integration of all needed technological elements. Application workflows that require HPC and Cloud resources, as well as need to process large amount of data (Big Data) will be effectively executed on the LEXIS platform. It integrates orchestration, data and security management components and a dedicated portal to expose its federated resources. Among the business cases WP7 will deliver a system for prediction of water-air-quality-food-energy nexus phenomena and their associated socio-economic impacts. WP7 involves multiple, interconnected, model layers:

- Global weather and climate models,
- Regional weather models,
- Domain-specific application models (such as hydrological, drought and fire forecasts),
- Impact models providing information for key decision and policy makers.

Position of the deliverable in the whole project context

Deliverable D7.8 is a product of the WP7 (Weather and Climate Large-scale Pilot), and is related to Task 7.3 entitled "Regional Weather and Climate: Assimilation of Local in-situ Unstructured Observations in High-Resolution Downscaling of Global Forecast", Task 7.4 entitled "Cloud-Based Domain Specific Application Modelling, Forced by Regional Forecasts and Environmental Observations" and Task 7.5 entitled "Cloud-Based Socio-Economic Impact Modelling Based on Exposure Information and Environmental Forecasts".

This document is an update and final version of the Deliverable D7.4 [1] and Deliverable D7.6 [2].

The KPIs and performance achieve from the deployment of each WP7 applications on the LEXIS platform will be presented in the Deliverable D7.9 [3], as well as the specific **Chyba! Odkaz není platný.**rely on the results obtained from the other WP7's applications.

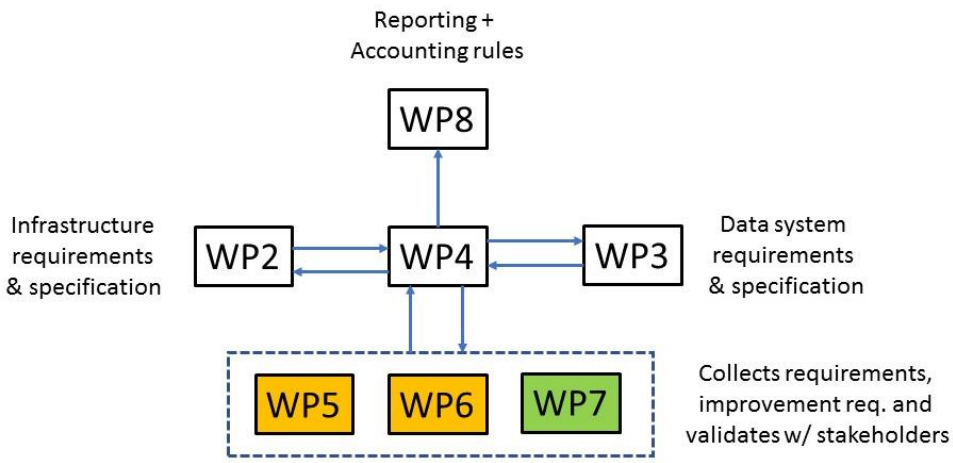

Figure 1 Position of WP7 in the LEXIS project

As depicted in Figure 1, WP7 with its weather and climate models will be one of the primary sources, together with WP5 and WP6, for setting the foundations and testing capabilities of the LEXIS platform, and specifically of the orchestrator. Conversely WP2 will provide the main inputs concerning requirements and specifications, as well as ensuring that developments done in WP7 will be aligned with the general requirements for the LEXIS platform. WP3 is in charge of developing the LEXIS storage solution on which workflows and the orchestrator rely on to ensure capability of storing data. WP4 lays the foundations for the orchestration of application workflows during their

whole lifetime (from the definition to the execution and completion), security aspects and monitoring of used resources. WP8 is focused on creation of the portal and analysing monitoring data for billing purposes.

Description of the deliverable

This deliverable presents an update of the description of each WP7's application use cases and their deployment on the LEXIS platform. After an introduction, Section 2 presents general mapping of WP7 applications on the LEXIS platform and provides updates on LEXIS tools (data exchange, orchestration, scheduling) in link to WP7 needs. Section 3 presents an update of the deployment of specific WP7 components. Section 4 is devoted to the deployment of WRF simulations since it is the backbone of all WP7 applications which are presented in Section 5.

Contributors for the deliverable content are:

- CIMA as the leader of WP7,
- CIMA as the responsible of the activities concerning WRF and the applications with Continuum and RISICO models,
- ITHACA as the responsible of the applications with ERDS model,
- NUM as the responsible of the applications with ADMS and ADMS Urban models, and responsible for the preparation of this document,
- ECMWF as the responsible of the activities concerning WCDA API,
- TESEO as the responsible of the activities concerning the deployment of a smart gateway for collecting, pre-processing and transmitting in-situ observations
- Atos, with the support of IT4I, as the responsible of WP7 workflows mapping on LEXIS platform.

1 INTRODUCTION

In the WP7 "Weather and Climate Large-scale Pilot ", various use cases in terms of application, domain size, etc. will be tested in the LEXIS infrastructure.

Figure 2 presents these applications. We can notice two elements from this figure: i) the simulated domain goes from global to local and ii) use cases are not independent and interact with each other: e.g. IFS weather forecast of ECMWF will be used as an input of CIMA WRF regional forecast which will be used as an input of NUM urban air-quality forecast.

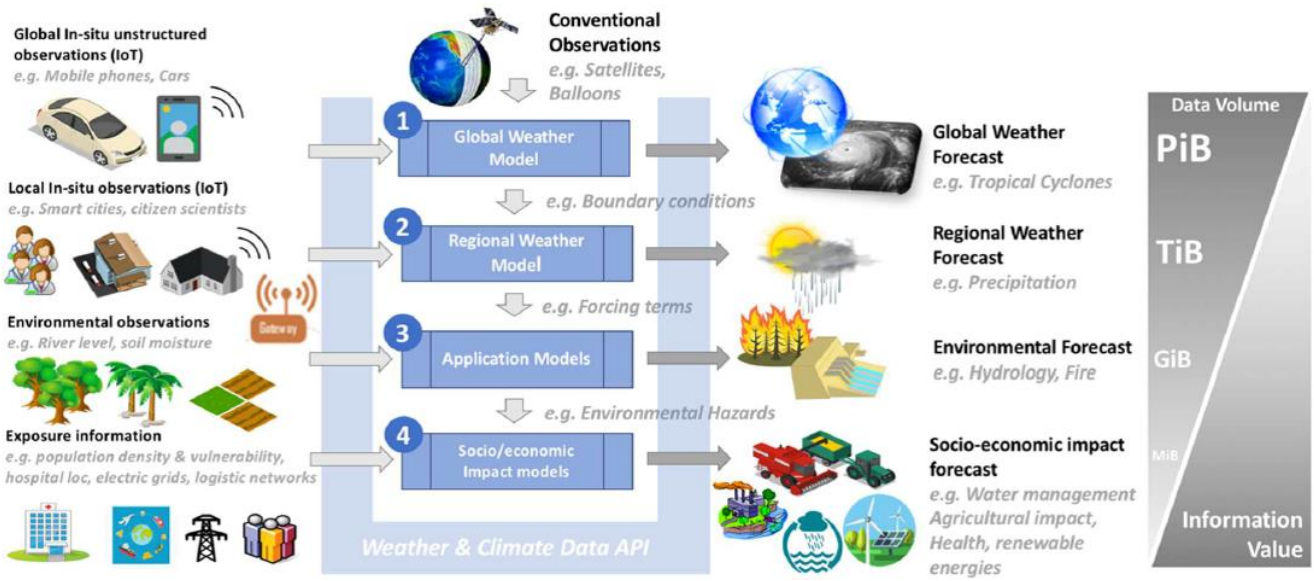

Figure 2 List of the different WP7 applications

One of the key points of WP7 is the exchange and handling of datasets from one application to another. The deployment of each application in the LEXIS infrastructure is not only the porting of each model on HPC/Cloud servers, but also the interaction between applications, as well as the visibility of the final outputs on the LEXIS portal. As a consequence, the deployment of WP7 applications rely on generic LEXIS components and on specific WP7 components.

Section 2 presents general mapping of WP7 applications on the LEXIS platform and provides updates on LEXIS tools (data exchange, orchestration, scheduling) in link to WP7 needs. Section 3 presents an update of the deployment of specific WP7 components. Section 4 is devoted to the deployment of WRF simulations since it is the backbone of all WP7 applications which are presented in Section 5. Note that the main description of each application is available in Deliverable D7.4 [1].

2 MAPPING OF WP7 USE CASE TO LEXIS

2.1 GENERAL VIEW

The WP7 workflows components are mapped as follows on the LEXIS platform (see Figure 3):

- WRF model is executed on HPC systems (LRZ and IT4I),
- The other WP7 models (Continuum, RISICO, ADMS, ADMS Urban and ERDS) are executed on cloud computing (LRZ and IT4I) using virtualization,
- Full workflows are executed by the LEXIS Orchestration System,
- Data exchange between the WP7 models are supported by DDI and WCDA API is limited to the access of IFS data.

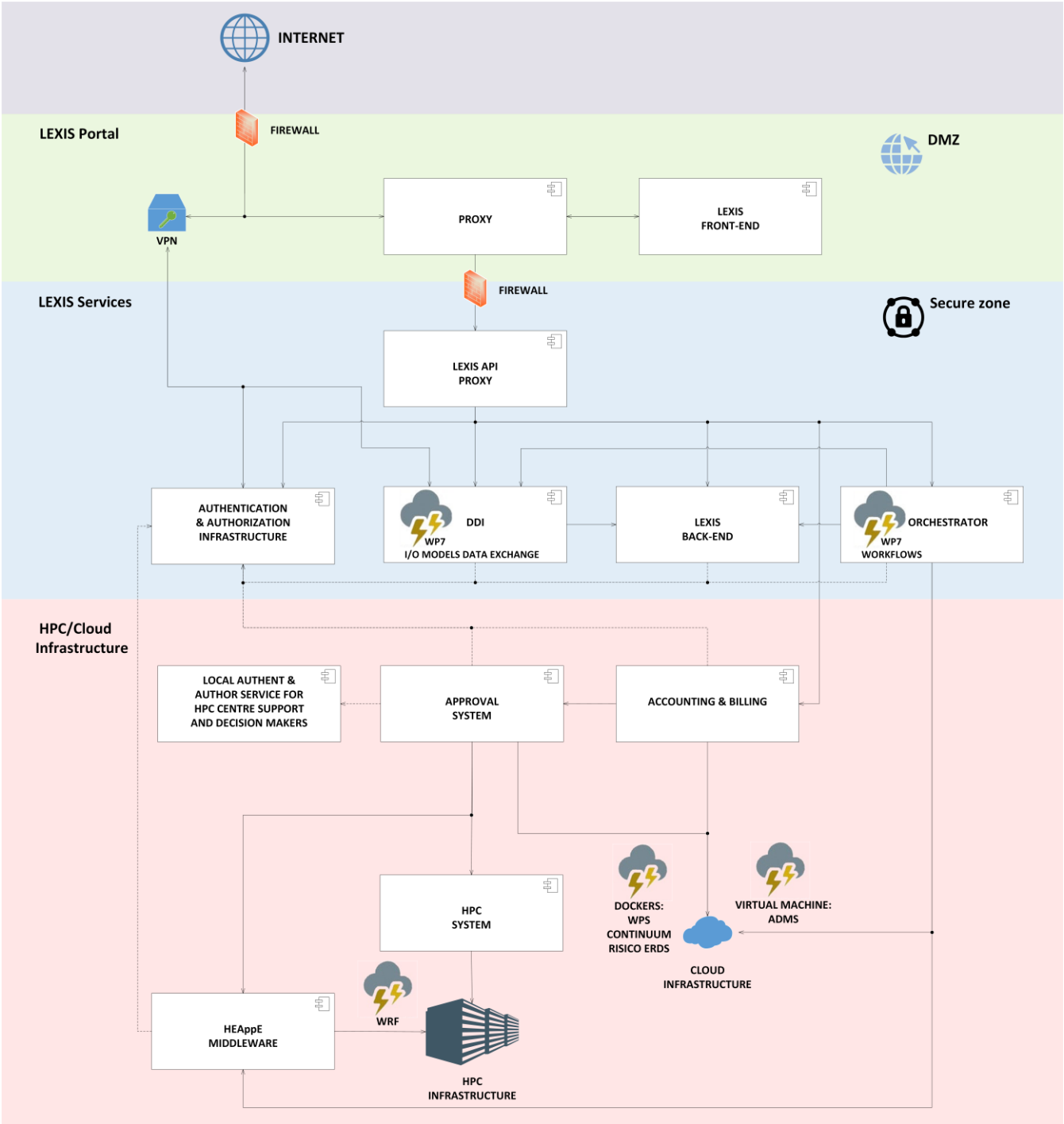

Figure 3 WP7 workflows mapping on LEXIS platform

These workflows are executed on LEXIS Cloud and HPC infrastructures by the Orchestration System provided in Deliverable D4.6 [4].

This Orchestration System provides an application deployment platform for Hybrid HPC/Cloud applications. The applications to be deployed are modelled using TOSCA [5] (Topology and Orchestration Specification for Cloud Applications), an OASIS consortium standard language to describe an application made of components, with their relationships, requirements, capabilities, operations. Associated workflows - sequences of operations on application components - can then be executed and monitored by the Orchestration System.

These TOSCA applications and associated workflows are developed incrementally, integrating new features as they become available, like for example the use of LEXIS Weather and Climate Data API (WCDA) to manage datasets. They are available under the LEXIS-project GitHub organization in repository application-templates [6].

Virtualization is understood in computing science as the creation of a virtual (rather than actual) version of something, such as virtual computer hardware platforms, storage devices, computer network resources and visualization services. Virtualization concept was developed around 1960s, to logically divide and assign the system resources provided by mainframe computers between different applications. Since its infancy, the meaning of the term has evolved and broadened. However currently two main approaches can be identified, namely: virtual machines and containers.

A virtual machine is a software usually identified as an image that is created within a computing environment called the host. A virtual machine performs tasks such as running applications acting as a separate computer thus ideal for testing other operating systems like beta releases and running software and applications.

Container is meant as a lightweight, stand-alone, executable package of a software designed to contain all the libraries, configuration files, dependencies, and other necessary parts (e.g. static data) to execute the application. Docker is a software development tool and a virtualization technology that makes it easy to develop, deploy, and manage applications by using containers. Docker enables to separate applications from infrastructure so that the software can be deployed quickly. With a docker approach, the infrastructure can be managed in the same ways you manage your applications.

Within the LEXIS WP7 activities, all components of each use case are available as docker containers, with a specific case for the ADMS air quality model which adopt a virtual machine approach based on windows.

These model virtualizations are described in Section 4 for WRF and Section 5 for the other WP7 models.

2.2 UPDATE ON DATA EXCHANGE

2.2.1 DDI (LRZ)

The DDI at LRZ and IT4I host the required datasets needed for the WP7 activities. This includes input datasets needed to run the workflows and the output datasets. To visualize the result data available on the DDI, data have to be moved to CIMA and transfer to the Dewetra platform of the Italian Civil Protection. To automatically have the data at CIMA, a client deployment of the DDI is needed. The client deployment is based on deploying an iRODS client on CIMA premises and connecting it to the iRODS servers at LRZ and IT4I. Furthermore, datasets downloaded from the Italian Civil Protection department will be hosted on the DDI and only moved to HPC resources when a workflow is triggered.

2.2.2 WCDA API and provision of IFS global forecasts (ECMWF)

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

section complements Deliverable D7.5 [7] and provides an update on the WCDA deployment and exploitation within WP7 workflows.

Main changes introduced with the current version (v0.7.2) are:

- Added support for https. The endpoints are:
  - ECMWF - `https://polytope.ecmwf.int/api/v1`
  - LRZ - `https://wcda.lrz.lexis.tech/api/v1`
  - IT4I - `https://wcda.it4i.lexis.tech/api/v1`

  Support for http will be phased out shortly.

- Introduced QoS policies and support to parallel requests. WCDA users can submit up to 6 simultaneous requests. In case of additional requests, WCDA will properly queue them. Test users are not allowed to submit simultaneous requests.
- Authentication can now be based on Keycloak tokens, providing full integration with LEXIS AAI. LEXIS users are mapped with ECMWF users and thus their activity through WCDA is recorded and properly reported as part of their MARS activities.

The following example shows how to use LEXIS Keycloak credential to access WCDA.

This example requires WCDA client version v0.6.1 or newer that can be obtained with the following command:

```
python3 -m pip install --upgrade git+https://git.ecmwf.int/scm/lex/polytope-client.git@master
```

Once the client has been upgraded, you should either remove the `$HOME/.polytope-client` directory, or wipe the client configuration as follows:

```
polytope unset config all
```

After that, you can set your Keycloak credentials via environment variables and start using the client as usual.

```
export POLYTOPE_USERNAME=my_keycloak_account@email.com
export POLYTOPE_PASSWORD=my_keycloak_password
python3

from polytope.api import Client
c = Client()
c.list_collections()

request = {
    'stream': 'oper',
    'levtype': 'sfc',
    'param': '165.128/166.128/167.128',
    'step': '0',
    'time': '00/12',
    'date': -1,
    'type': 'an',
    'class': 'od',
    'expver': '0001',
    'domain': 'g'
}
# Retrieve data
c.retrieve('lexis-mars', request, 'output.grib')

# List the active requests
ids = c.list_requests()
```

2.3 UPDATE ON ORCHESTRATION AND SCHEDULING MECHANISM

Following the requirements from WP7 use cases, the LEXIS Orchestration Service has to provide the capabilities for flexibly submitting jobs on the geographically distributed cluster resources, based on their availability. To this end, the LEXIS Orchestration Service will embed a Dynamic Allocation Module (DAM) which is responsible to determine, at the time of the request submission, the most suitable set of resources depending on their availability, load and the dataset's location. The combination of the Ystia orchestration toolset and the DAM also allows one to select more than one location where to launch the execution of a given job. DAM ranks all the execution locations according to different evaluation metrics and collected monitoring information and reports to Ystia a subset of them (i.e., one or more depending on the number of requested locations). This feature will be of great benefit in the management of the WP7 workflows since it allows to submit the WRF jobs on LRZ and IT4I locations. The DAM is currently under final development and in test. Return of experience on the DAM will be discussed in Deliverable D7.9 [3].

3 STATUS OF DEPLOYMENT OF SPECIFIC WP7 COMPONENTS

3.1 SMART GATEWAY FOR IOT DATA

Deliverable D7.7 [8] is specifically dedicated to the development and deployment of the TESEO Smart Gateway. It furthermore describes the integration of a weather station with this system / IoT-to-HPC architecture such that HPC simulations can be fed with its data after appropriate pre-processing.

Deliverable D7.7 [8] reports how the outcomes of Deliverable D7.3 [9] were translated into an easy to manufacture and deploy unit. For this purpose, the IoT distinctive functional blocks of the Smart Gateway and of the Weather Sensors Node were combined together to achieve an efficient layout, allowing rugged construction for a long-lasting operation. Then it is illustrated in Deliverable D7.7 [8] how the Smart Gateway was engineered for both purposes mentioned above: a front-door to the HPC for the meteorological observations on one side, and on the other for the "connected things" eventually involved in the use-cases proposed by the LEXIS Open Call participants. Deliverable D7.7 [8] details the electronic design of the Smart Gateway and of the Weather Sensors Node. All electronic cards composing the system are extensively described and motivation for the key engineering and integration choices is given. Deliverable D7.7 [8] focuses also on the firmware implementation of the on-board processors: details are given on the operating system, hardware management libraries and main application. To conclude, Extensive description is given in Deliverable D7.7 [8] for all sensors and devices, how data is collected, validated, pre-processed and fed to the LEXIS infrastructure.

To this date, two weather demonstrator's stations are installed at the TESEO and at the CIMA Foundation premises and validation tests were performed.

3.2 PROVISION OF MOJI MOBILE PHONE DATA

Moji is a Chinese Company based in Beijing that designs and operates online weather and advertising application throughout China.

In the context of LEXIS, ECMWF made an agreement with Moji about the collection, analysis and exploitation of meteorological data (e.g. temperature, pressure) acquired by tens of thousands of mobile phones.

Due to Chinese regulations, direct data transfer from Beijing to Europe was not possible, so Moji initial plan was to set up a server in Hong Kong for data collection and egress.

Unfortunately, in November 2019, Moji legal office suggested that an activity involving data exit, data authorisation and related matters was too risky and Moji decided to cancel the cooperation activity. This decision may be due to

the imposition of Chinese national security laws on Hong Kong, which generated the protests that started in March 2019.

The last communication from Moji dates back to 21 Nov 2019, any other attempt to liaise with them, unfortunately, failed.

4 EXECUTION OF WRF REGIONAL FORECAST OVER ITALY AND FRANCE

4.1 ASSIMILATION

4.1.1 WRFDA structure and observational capability

The data assimilation is a technique that combines a Numerical Weather Prediction (NWP) output (first guess or background forecast) with observations and their respective error statistics providing as product an improved state of the atmosphere (analysis). The improvement of the forecast abilities using data assimilation techniques became an increasingly important topic of research in the last few years. WRFDA is a freely available data assimilation software allowing the NWP community to perform both research and operational implementation in the same framework [10].

The WRF model had a significant data assimilation component from the beginning of its development. After an initial discussion between major partners (NCAR, National Oceanic and Atmospheric Administration (NOAA), the U.S. Air Force Weather Agency (AFWA), Oklahoma University, and the U.S. Naval Research Laboratory) in 1999-2000 the basic requirements of the data assimilation system were defined to satisfy both operational and research communities. The key idea at the base of the system is that WRFDA has to be robust and efficient, accurate, computationally efficient, portable, well documented and easy to use. WRFDA features three-dimensional and four-dimensional data assimilation (3DVAR, 4DVAR) as well as a hybrid variational ensemble technique (ensemble transform Kalman filter (ETKF) – 3DVAR). The first version of this data assimilation system was distributed in 2003 with only 3DVAR implemented (named WRF 3DVAR), and subsequently updated in two steps: WRF 2.0 and WRF 3.0 [11, 12]. Furthermore in 2004 the 4DVAR has been released [13] and the system's name was changed in WRF-VAR. Finally, in 2008 the release of the ETKF-3DVAR hybrid variational/ensemble system led to the final name WRFDA.

All the techniques can assimilate a wide range of observations ranging from the more traditional ones (surface, rawinsonde, aircraft, wind profiler, and atmospheric motion vectors) to the newer ones (radar radial velocity and reflectivity, satellite-based observations such as radiance, GPS radio occultation measurements etc.) as reported in Table 1 [14].

CONVENTIONNAL SURFACE AND UPPER-AIR, RADAR, AND PRECIPITATION DATA	SATELITTE AND RELATED GROUND-BASED RETRIEVALS	SATELLITE RADIANCES
SYNOP, METAR, SHIP, BUOY, mesonet (surface pressure, 2-m temperature and specific humidity q, 10-m wind speed U/wind direction V)	Atmospheric motion vectors	HIRS, AMSU-A, AMSU-B, MHS from NOAA and MetOp series
Radiosonde (U,V,T,q)	SATEM thicknesses	SSMIS from DMSP series
Pibal (upper-air U, V)	Scatterometer oceanic surface winds	ATMS from SNPP
AIREP/ACARS/AMDAR, TAMDAR (U, V, T, q)	Wind profiler (U, V)	AIRS from Aqua

Radar derived: reflectivity, radial velocity, retrieved rainwater, retrieved humidity within cloud	Ground-based GPS total precipitable water and zenith total dely	IASI from MetOp series
Stage IV precipitation (for 4DVAR only)	GPS refractivity	SEVIRI from Meteosat series
	AIRS-retrieved T and q profiles	MWTS/MWHS from FY-3 series
		AMSR-2 from GCOM-WI

Table 1 The WRFDA Observation Catalogue

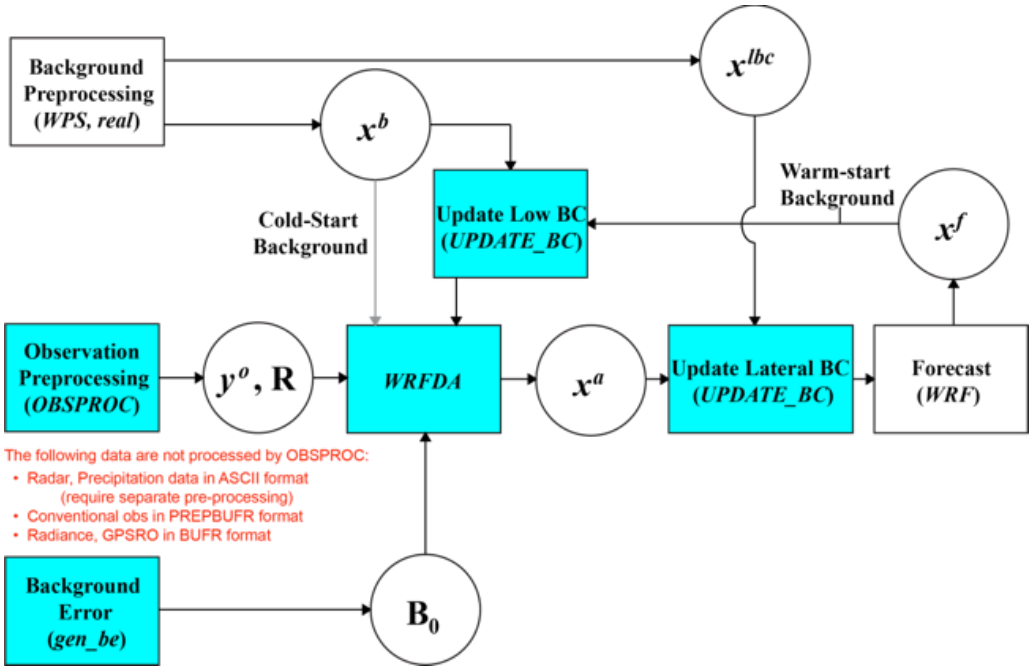

Figure 4 WRFDA in the WRF modelling system.

In blue the WRFDA components are reported with the relation of the rest of the WRF system (white boxes). Where: x^b: first guess, either from a previous WRF forecast or from WPS/real.exe output. x^{lbc}: lateral boundary from WPS/real.exe output. xa: analysis from the WRFDA data assimilation system. x^f: WRF forecast output. y^o: observations processed by OBSPROC (note: PREPBUFR input, radar, radiance, and rainfall data do not go through OBSPROC). B^0: background error statistics from generic BE data (CV3) or `gen_be` utility. R: observational and representative error statistics.

One of the main advantages of WRFDA is that it is built directly into the WRF modelling system (see Figure 4) providing a direct interface with the other WRF components.

The inputs for WRFDA are (Figure 4):

- The background forecast (x^b): in cold-start mode it is a forecast or analysis from another model interpolated to ARW grid through the use of WPS and Real programs of WRF system, in warm-start mode (cycling mode) it is the output of a short-range forecast (typically 1-6 hours) run with WRF-ARW system.
- Observations (y^0): they can be provided in PREBUFR or LITTLE_R format, and an observation pre-processor (OBSPROC) is designed to reform and quality check observations that after can be directly read in WRFDA.
- Background error covariances matrix (B): necessary to define the spatial and multivariance response of the analysis to observations. It is calculated off-line through the use of `gen_be` utility.

When the data assimilation of all observations ends, the analysis state (x^a) must be merged with the existing boundary conditions (x^{lbc}) using the UPDATE_BC utility.

Summarizing, together with WRFDA there are three main utilities helpful for efficient data assimilation: OBSPROC for observation preprocessing, `gen_be` for background error estimation and UPDATE_BC to merge the analysis product of data assimilation with the existing boundary condition file.

Observation Preprocessing (OBSPROC)

OBSPROC provides observations (y^0) for WRFDA. The program reads LITTLE_R format (a text-based format) and its functions are:

- Read the observations from LITTLE_R file and the run time and space coverage from a namelist file.
- Remove observations that are outside the temporal window and the spatial domain.
- Through the hydrostatic pressure assumption retrieve pressure or height based on observed information.
- Check multi-level observations for vertical consistency and superadiabatic conditions.
- Estimate the error for each observation from a pre-specified error file.
- Write an output file in ASCII or BUFR format to be direct ingested in WRFDA.

Background Error Calculation (`gen_be`)

The background error covariance statistics are necessary in WRFDA cost function minimization to weight errors in features of background forecast field. The WRFDA's `gen_be` utility estimates domain-specific climatological background error covariance matrix based on input training data that could be time series of forecast differences (the so called "NMC method" [15] or perturbations from an ensemble prediction system [12].

Details on the method are given in Appendix A.

Update boundary conditions (`update_bc`)

Using the analysis state (output of WRFDA) as initial conditions, it is necessary to first update the lateral boundary conditions to reflect differences between background and analysis state. Only boundary conditions of domain 1 must be updated because boundary conditions for nests domains are calculated from domain 1. To pursue this aim, the script `update_bc.csh` is provided and it works with the following input:

- The analysis state (initial conditions after data assimilation),
- Original lateral boundary conditions of 1 domain (wrfbdy_d01),
- A namelist called `parame.in`.

After running the executable `update_bc.exe`, the old boundary conditions will be overwritten with the updated one.

4.1.2 WRFDA variational data assimilation

In general, variational systems can be classified as those techniques that provide an analysis state of the atmosphere through the minimization of a cost function given two sources of data: a background forecast (first guess) and observations. The main advantages of the variational data assimilation technique are the possibility to assimilate a wide range of observations related nontrivially to standard atmospheric variables (e.g. radiance) and the imposition of a dynamic balance that can be implicit in the forecast model itself (in case of 4DVAR) or explicit using some balance equations (in case of 3DVAR). In WRFDA both 3DVAR and 4DVAR are available from WRF 3.0 [12] release. Prior to WRF 3.0 only 3DVAR was available and the cost function minimization used a modified version of limited memory Quasi-Newton Method (QNM), starting from WRF 3.0 the alternative Conjugate Gradient Method (CGM) is implemented. This update requires the complete linearisation of WRFVAR's inner loop that was a limitation overcame through the introduction of multiple outer loops with the purpose of iterate over non-linear solutions (like observation operators, balance constraints in 3DVAR and the forecast itself in 4DVAR) through the use of the WRFDA's analysis state from the previous iteration as new first guess (Figure 5). The use of multiple outer loops can be also very useful in variational quality control because usually observations are rejected if the magnitude of the observation minus the background forecast differences are larger than a given threshold (typically

a multiple of the observation error standard deviation). This choice implicitly assumes that the first guess is accurate, but in case of areas with large forecast errors, there was the danger that good observations could be rejected. The outer loops allow observations rejected in previous iteration to be accepted in the following iteration if the observation minus the updated first guess (the analysis from the previous outer loop) became smaller than the specified threshold.

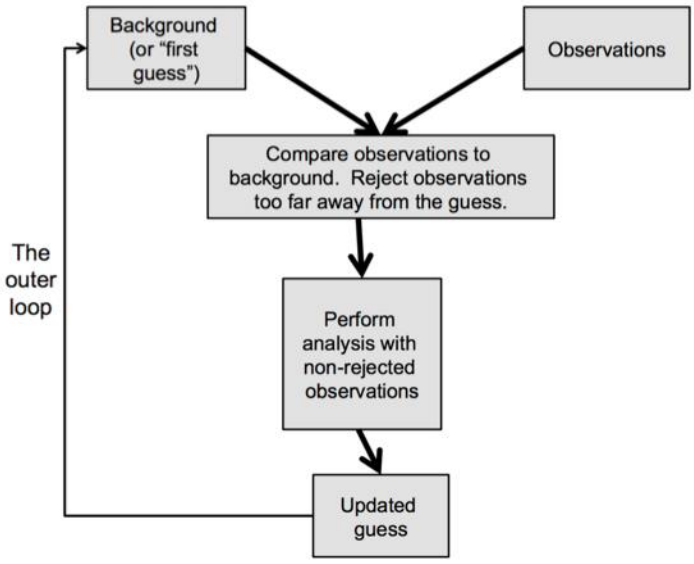

Figure 5 Outer loop schematization

4.1.3 3DVAR implementation

The 3DVAR was the first data assimilation technique released in WRF model, the basic goal of this technique is to provide an optimal estimate of the true state of the atmosphere through the minimization of the cost function reported in Equation 1 [16].

$$J(x) = J^b + J^0 = \frac{1}{2}(\mathbf{x}-\mathbf{x}^b)^{\mathrm{T}}\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}^b) + \frac{1}{2}(\mathbf{y}-\mathbf{y}^0)^{\mathrm{T}}(\mathbf{E}+\mathbf{F})^{-1}(\mathbf{y}-\mathbf{y}^0) \qquad (1)$$

In summary, the 3DVAR problem is the iterative solution of Equation 1 to find the analysis state ($\mathbf{x}$) that minimizes the cost function $J(x)$ which represents the a posteriori maximum likelihood estimate of the true state of the atmosphere, combining the two sources of data: observations ($\mathbf{y}^0$) and background ($\mathbf{x}^0$). $\mathbf{B}, \mathbf{E}$ and $\mathbf{F}$ are the background, observation (instrumental) and representitivity error covariance matrices respectively used as weight to fit the two sources of data. The representitivity error is introduced from the use of an observation operator (H) to transform the gridded analysis to the observation space: $y = H(x)$y.

Equation 1 assumes that the error covariances are described by a Gaussian PDF with 0 mean error, but can be also used alternative costs functions that relax this constraint [17]. Furthermore, Equation 1 neglects correlations between observation and background errors, as it is usually done in variational approach [15, 18, 19].

The multidimensional cost function is efficiently solved through the use of adjoint operations (like multidimensional application of the chain-rule for partial differentiation). Considering a model state x with n degrees of freedom (grid points number times independent variables number), the calculation of the J^b term of Equation 1 requires $\sim O(n^2)$ calculations. In a typical NWP problem $n^2 \sim 10^{12}$-10^{14}, then a direct solution is not feasible in the time window necessary for data assimilation in operational issues. A solution to reduce the computational cost of J^b calculation is to calculate it in terms of control variables defined as $x' = U_v$, where x' is the analysis increment ($x - x^b$) and U transform is designed to nondimensionalize the problem and to allow the use of efficient filtering techniques that approximate the full background error covariance matrix. With a U transform well designed, the

condition numbers will be small and the product UU^T will quite match the B matrix. In terms of analysis increments, the Equation 1 can be rewritten as:

$$J(v) = J^b + J^0 = \frac{1}{2}\mathbf{v}^{\mathrm{T}}\mathbf{v} + \frac{1}{2}\left(\mathbf{y}^{0'} - \mathbf{HUv}\right)^{\mathrm{T}}(\mathbf{E}+\mathbf{F})^{-1}(\mathbf{y}^{0'} - \mathbf{HUv}) \quad (2)$$

Where $\mathbf{y}^{0'} = \mathbf{y}^0 - \mathbf{H}(\mathbf{x}^b)$ is the innovation vector and $\mathbf{H}$ is the linearisation of H.

4.2 VIRTUALIZATION/ DOCKER / MAPPING ON HPC

WRFDA is the core of the WRF simulation, it takes as input static geodata previously stored in LEXIS DDI and GFS (or IFS) global forecast for Italy domain (or France domain). WRFDA simulation improves resolution of the GFS model over the Italy (France) region, meanwhile assimilating Italian (French) radar data and Wunderground weather stations observation.

4.2.1 WPS docker

As said, WRFDA needs a set of files containing initial and boundary conditions extracted from a global forecast dataset.

The responsibility of the WPS docker is to transform data coming from GFS/IFS into such initial and boundary conditions. In order to do so, it also uses a dataset containing static geographical data, previously stored on DDI.

Pre-compiled container images are available on docker hub: `cimafoundation/wps-da.gfs` for GFS or `cimafoundation/wps-da.ifs` for IFS.

Sources are available on GitHub: `meteocima/wps-da.gfs` for GFS or `meteocima/wps-da.ifs` for IFS.

The container expects three different volumes to be mounted to read or write input and output data. Moreover, it expected a set of argument to be provided as environment variables.

Details on WPS docker are given in Appendix B.

4.2.2 LEXIS downloader docker

The responsibility of the LEXIS downloader docker is to download data stored in CIMA Webdrops platform that are necessary to run the simulations.

Precompiled container images are available on docker hub: `cimafoundation/lexis-download-docker`.

Sources are available on GitHub: `meteocima/lexis-download-docker`

The container expects three different volumes to be mounted to read or write input and output data. Moreover, it expected a set of arguments to be provided as environment variables.

The container expects a volume to be mounted to write output data. Moreover, it expected a set of arguments to be provided as environment variables.

Details on LEXIS downloader docker are given in Appendix C.

4.2.3 Download-ECMWF-data

For running WRF simulations, the users have the choice to download the initial and boundary conditions provided by ECMWF Integrated Forecasting System (IFS) using the python client, called **polytope**.

For this purpose, we can run the download-ECMWF-data command. It will download the input files needed for your simulations.

Source code and further information are available on GitHub: `meteocima/download-ECMWF-data`

4.2.4 WRFDA HEAppE template

The main part of the WRF simulation runs on HPC infrastructure. It's wrapped in a HEAppE plugin[1] in order to simplify orchestration via York.

The process requires results from WPS container and LEXIS downloader container and is orchestrated by the same Golang command used in WPS docker, whose source code could be found on GitHub: `meteocima/wrfda-runner`.

The orchestrator alternates run of `wrf.exe` and `da_wrfvar.exe` executable, in order to assimilate data from radars and from weather stations.

Template arguments

The template receives arguments by means of an `arguments.txt` file, previously produced by the WPS container. It should be saved under inputs directory. This is a simple text file that contains the configuration file to use and all the dates to simulate, one for each line.

Internals

The template is made by following parts:

- `wrfda-runner` binary executable of orchestrator
- `france-config.ifs.cfg` configuration file used for simulations over France domain
- `italy-config.ifs.cfg` configuration file used for simulations over Italy domain
- `namelists.france nameslists` to configure WRF and WRFDA
- `namelists.italy` nameslists to configure WRF and WRFDA
- `covar-matrices-it` error covariance matrices over italy domain
- `covar-matrices-fr` error covariance matrices over france domain
- `inputs` this directory should contains input and boundary conditions created by WPS docker
- `observations` this directory should contains radars and weather stations data as created by LEXIS download docker.

5 DEPLOYMENT OF WP7 APPLICATIONS

5.1 CONTINUUM HYDROLOGICAL MODELLING

5.1.1 Workflow and updated description

Concerning the **hydrological prediction**, the complete workflow involves different LEXIS tasks such as the meteorological model WRF, including a WRFDA data assimilation system and the fully distributed hydrological model Continuum (Figure 6). The WRF model is executed as a task on high performance computing facilities (using HPC computational projects at LRZ and IT4I) after the preparation of initial and boundary conditions provided by the WRF Preprocessing System WPS. The WPS task is executed on cloud computing facilities (LRZ and IT4I) and it allows processing ECMWF's IFS and the NCEP's GFS global circulation model data to generate input fields for the WRF model itself. The WRFDA task is a flexible, state-of-the-art atmospheric data assimilation system that is portable and efficient on available parallel computing platforms: WRFDA is a task executed on cloud computing facilities (using HPC computational projects at LRZ and IT4I). The Continuum hydrological model is developed by CIMA and it can work both in the pre-event analysis and forecast phase. Continuum is meant to be a model with a reduced number of parameters able to take advantage of all the information available via remote sensing

[1] HEAppE plugin: https://github.com/lexis-project/yorc-heappe-plugin

techniques. Continuum model is executed as sequential task on cloud computing facilities (using HPC computational projects at LRZ and IT4I). The results of the hydrological prediction workflow will be published on the MyDewetra platform, a web-based real-time system for hydro-meteorological forecasting and monitoring developed by CIMA on behalf of the Italian Civil Protection Department.

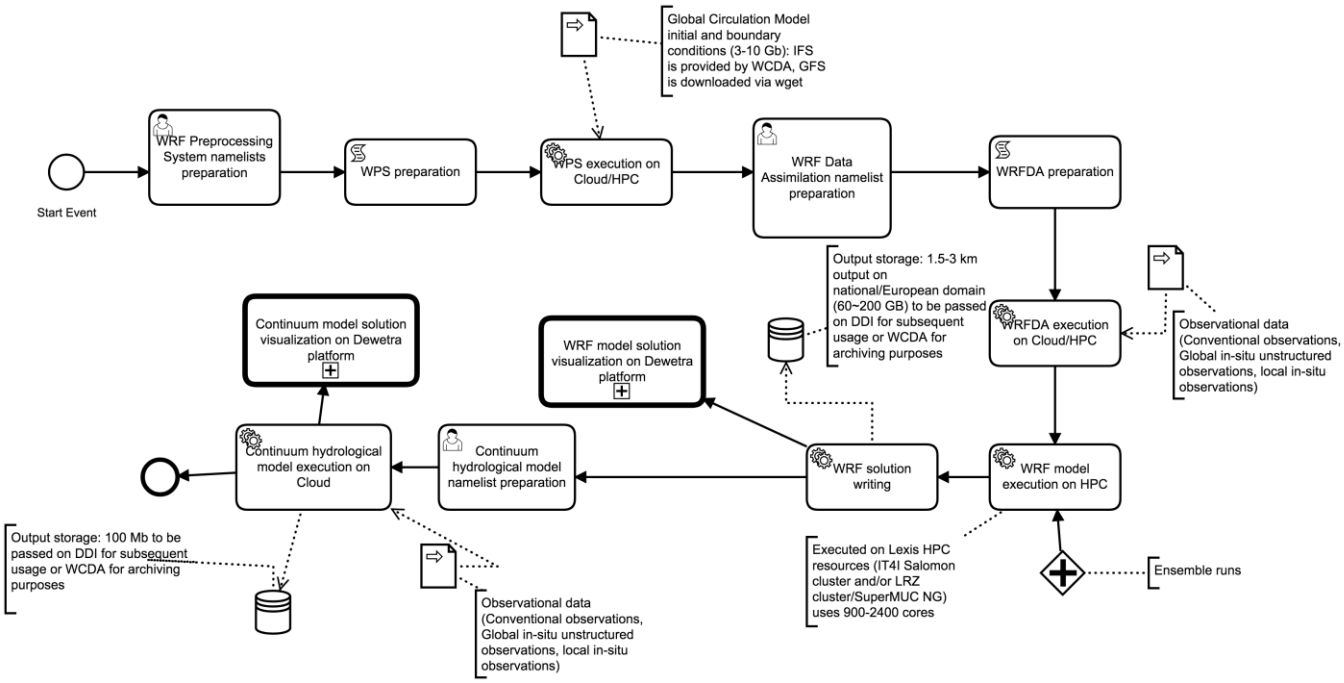

Figure 6 LEXIS hydrological prediction workflow description

5.1.2 Orchestration and Docker for virtualization

In practice, the workflow is then:

- Downloading the required input data (static geographical data and Global Forecast System data) for a date specified by the user,
- Pre-processing these data on a Cloud Compute instance using a pre-processing container provided by CIMA[2],
- transferring this data to a WRF HEAppE job on a HPC cluster,
- post-processing WRF on a Cloud Compute instance using two post-processing containers provided by CIMA[3],
- Preparing the datasets for the Hydrological model for a given region in a container (in the use case, 17 instances of this container are run in parallel for 17 regions of Italy),
- Running the Continuum hydrological model,
- Transferring these results to LEXIS DDI and optionally on a SFTP server when specified by the user.

A detailed description of the workflow and its TOSCA implementation is available in the LEXIS project GitHub[4].

The responsibility of the Flood-PROOFS dockers are to adapt the data needed for running the Continuum model, run the Continuum model itself, and create the output datasets in netcdf and ascii formats.

Further information and more explanations about the Flood-PROOFS containers are available in the related GitHub repository[5] and in Appendix D.

[2] CIMA pre-processing container: https://github.com/meteocima/wps-da.gfs
[3] CIMA post-processing containers: https://github.com/c-hydro/fp-docker
[4] LEXIS project GitHub: https://github.com/lexis-project/application-templates/tree/master/weather-climate/applications/continuum
[5] Floos-PROOFS containers in GitHub: cima-lexis/fp-docker

5.2 RISICO RISK FIRE MODELLING

5.2.1 Workflow and updated description

Concerning the **forest fire risk prediction**, the complete workflow involves the aforementioned meteorological task WRF, including the WRFDA data assimilation system, and the fire risk model RISICO (see Figure 7), in place of the Continuum model in the hydrological workflow. RISICO is a fire danger rating system integrating meteorological observations and forecasts with vegetation cover and topography data: its modules describe dead fine fuel moisture conditions, the potential rate of spread, and the potential fire line intensity. The RISICO model is executed a sequential task on cloud computing facilities (LRZ and IT4I).

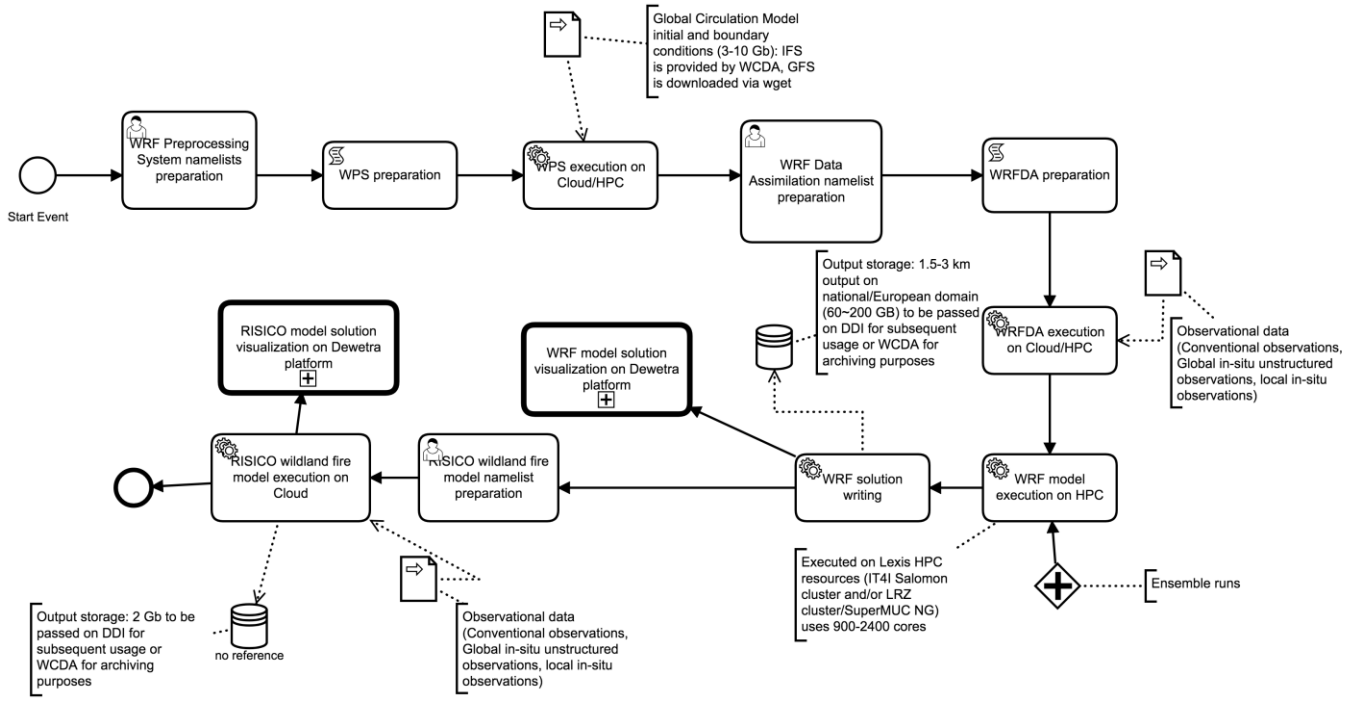

Figure 7 LEXIS forest fire prediction workflow description

5.2.2 Orchestration and Docker for virtualization

In practice the workflow is then:

- Downloading the required input data (static geographical data and Global Forecast System data) for a date specified by the user,
- Pre-processing these data on a Cloud Compute instance using a pre-processing container provided by CIMA[6],
- Transferring this data to a WRF HEAppE job on a HPC cluster,
- Post-processing WRF on a Cloud Compute instance using a post-processing RISICO container provided by CIMA[7],
- Transferring RISICO results to LEXIS DDI.

A detailed description of the workflow and its TOSCA implementation is available in the LEXIS project GitHub[8] and in Appendix E.

[6] CIMA pre-processing container: https://github.com/meteocima/wps-da.gfs
[7] Post-processing RISICO container: https://github.com/cima-lexis/risico-docker
[8] LEXIS forest fire workflow and TOSCA implementation in GitHub: https://github.com/lexis-project/application-templates/tree/master/weather-climate/applications/risico

5.3 URBAN AND INDUSTRIAL AIRQUALITY FORECASTING

5.3.1 Workflow and updated description

For the **air quality prediction**, the complete workflow involves the aforementioned meteorological task WRF, including the WRFDA data assimilation component, and the ADMS (Atmospheric Dispersion Modelling System) model (see Figure 8). ADMS (for industrial use case) and ADMS Urban (for urban use case) models are distributed as a Windows desktop application, it uses binary (NetCDF) and text files as input/output data format. The ADMS and ADMS urban models are executed as a task on cloud computing facilities (IT4I and LRZ).

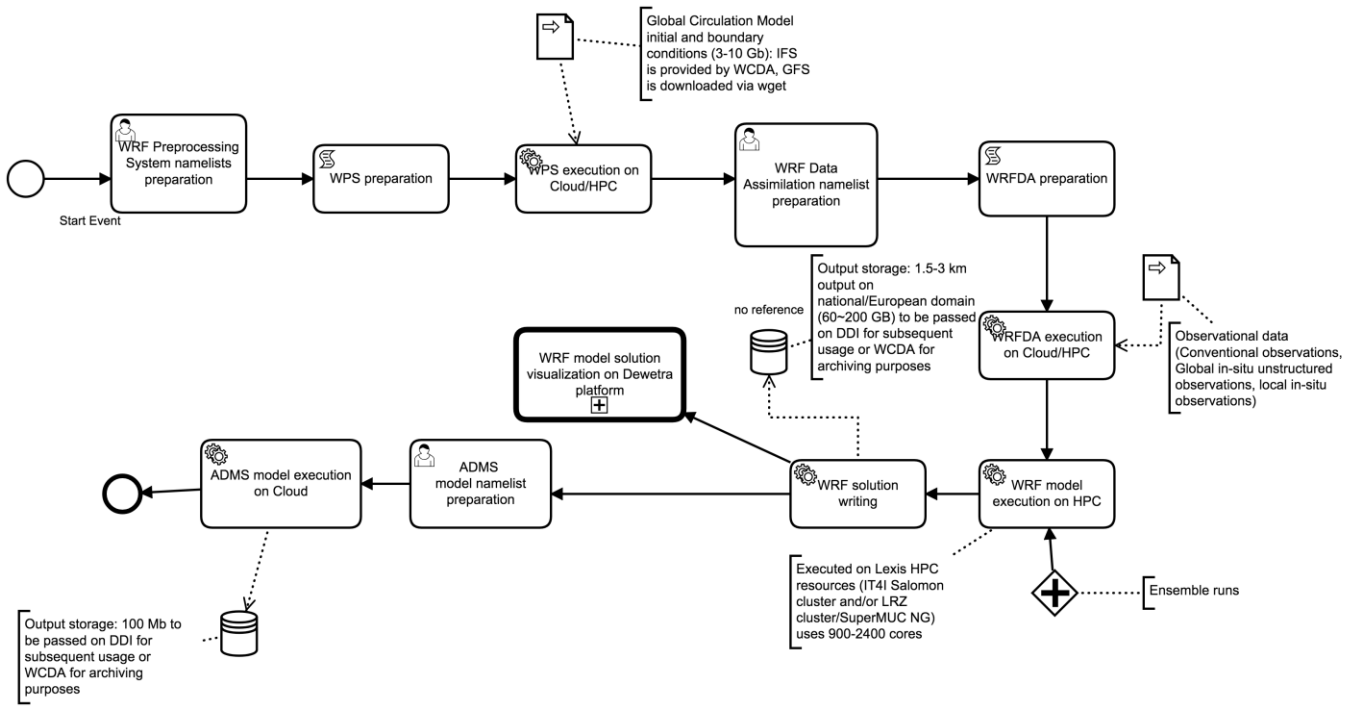

Figure 8 LEXIS air quality prediction workflow description

The industrial use case concerns the forecast of SO_2 peaks from the atmospheric release of an industrial site. This site is a refinery of the TOTAL company for which NUM provides operational forecasts since 2005. The urban case concerns the forecast of air quality over the city of Paris.

The objectives of NUM to use the LEXIS platform is to:

1. Improve the performance of air quality forecasts based on expected improvement of weather forecasts in order to reduce the number of forecasted miss and/or false air quality peaks. To do that the deployment of WRF on LEXIS platform allows to use a better spatial resolution than those performed by NUM (3km) and a better initial start of the WRF simulation with assimilation of surface and radar observation data. Moreover, LEXIS is a way to test easily via the development of the WCDA API the initialisation of WRF forecast by the global IFS simulation performed by ECMWF.
   In the two uses cases (industrial and urban), the performance will be then evaluated by comparing LEXIS simulations to NUM operational simulations. This evaluation will be performed on past episodes. The consequence is the adaptation (as described Section 4) of the initial WRF workflow (and configuration of WRF) in order to download and use (i) global IFS forecast (ii) from the past to initiate the WRF simulations.

2. Demonstrate that some limitations on air quality urban modelling can be removed easily via the access to a platform such as LEXIS.
   This will be illustrated in the Deliverable D7.9 [3], but we can already mention that the Paris urban case has been largely improved in terms of:
   - Simulated area with a domain which covers now the full metropolitan region of Paris (772 km^2 with 950,000 output points at surface) and not only the "center" of Paris (106 km^2 with 80,000 output points at surface).
   - Emission sources since no limitations were introduced on the simulated roads which are now of 76,000 compared to 3,500 previously, and the other emission sources (considering also the increase of domain) are now 4,000 compared to 800.

   The practical consequence is that the urban case requires now to manage 160 elements (see next section on virtualization) in regards to 13 previously.

5.3.2 Orchestration and Docker for virtualization

The workflow for the ADMS use case is:

- Downloading the required input data (static geographical data and ECMWF data) for a date specified by the user,
- Pre-processing these data on a Cloud Compute instance using a pre-processing container provided by CIMA[9],
- Transferring this data to a WRF HEAppE job on a HPC cluster,
- Post-processing WRF on a Cloud Compute instance using a container executing a script provided by NUM retrieved from LEXIS DDI,
- Running ADMS on a Cloud Compute Instance using a specific Windows images available on the OpenStack Cloud locations provided by LEXIS. There is an option to choose between urban or industrial execution. A windows compute instance is created the selected location. The orchestrator executes then a PowerShell script on this Windows instance to generate ADMS results.
- Transferring these results to LEXIS DDI and optionally on a SFTP server when specified by the user.

For convenience, a separate ADMS post-processing workflow is also implemented. This workflow takes in input a WRF results dataset that was stored in DDI by a previous ADMS template workflow execution and runs only the post-processing described above without having to execute the pre-processing and HPC computation phases of the ADMS workflow.

A detailed description of these workflows and their TOSCA implementations is available in the LEXIS project GitHub[10].

Concerning the execution of ADMS, ADMS Urban and Industrial are both native Windows applications which can be automatized using command line parameters. We have explored several options for its automatization in OpenStack Cloud, like creating a dedicated image which would run a pre-defined script on boot or using completely blank image where orchestrator would deal with installation and running of the ADMS.

The solution we have implemented in the end uses a MS Windows image with armed cloud-init script[11] which runs a set of tasks right after the image is booted and takes care for example of setting up a hostname, starting services and so on based on parameters passed by the orchestrator via the OpenStack metadata. One of the features of cloud-init is launching a script supplied in the metadata by the user. We are using this mechanism to pass the PowerShell script which handles the ADMS execution. This approach is very similar to what Ystia is doing when running commands on Linux VMs via SSH. In this case, the commands are passed as instance parameters via

[9] CIMA pre-processing container: https://github.com/meteocima/wps-da.ifs
[10] ADMS workflows and TOSCA implementation in GitHub: https://github.com/lexis-project/application-templates/tree/master/weather-climate/applications/adms
[11] Cloudbase Cloud-init https://cloudbase.it/cloudbase-init/

standard OpenStack API. Ystia handles passing of the script to the OpenStack API, therefore risks connected to execution of arbitrary commands in VMs are tied to integrity of a particular Ystia instance.

Based on the previous evaluation of the software, IT4I and Atos created a Windows Power Shell script which implements the following steps:

- Create local directory in the instance.
- Retrieve LEXIS DDI token.
- Use LEXIS DDI API to download datasets with ADMS application, static and variable data passed as parameter of the workflow.
- Extract the ADMS app and data to the temporary directory and install license files.
- Run the ADMS application.
- Upload the results and application output back to the LEXIS DDI.

The script gets its execution parameters from the OpenStack metadata described in Table 2. The script itself is passed as `user_data` field in the instance metadata.

METADATA KEY	DATA TYPE	DESCRIPTION
adms_type	Either one of "urban" or "industrial"	Used to select the type of ADMS application to be run
adms_urban_app_dataset	Path to LEXIS DDI dataset	Dataset with ADMS Urban app
adms_urban_static_dataset	Path to LEXIS DDI dataset	Dataset with ADMS Urban static data
adms_industrial_app_dataset	Path to LEXIS DDI dataset	Dataset with ADMS Industrial app
adms_industrial_static_dataset	Path to LEXIS DDI dataset	Dataset with ADMS Industrial static data
aai_lexis_user_uuid	Subject ID of the LEXIS AAI User	Identity which is used to connect to LEXIS DDI
aai_broker_token	base64 encoded string	Token for iRODS auth_microservice
aai_broker_endpoint	URL	Endpoint for iRODS auth_microservice
ddi_endpoint	URL	Endpoint for LEXIS DDI iRODS API
meteo_dataset_name	string	Name (metadata) of the dataset with meteodata
adms_result_dataset_name	string	Name (metadata) of the output dataset

Table 2 ADMS Execution script parameters

The script uses the LEXIS DDI iRODS API which provides functionality to download and upload data to LEXIS DDI using simple HTTP requests. The amount of data handled by the ADMS is in the range of hundreds of MBs. This allows to handle all data locally in the instance and no additional staging is needed. This allowed us to integrate the

data handling directly in the script using the LEXIS DDI APIs. Short excerpt from the script which handles the actual execution of the ADMS app is in Figure 9.

```
# Copy *.met files to the static dataset dir
echo "Copying .met files to static data dir"
$met_files | Foreach-Object {Copy-Item -Force -Path $_.FullName -Destination $static_data_path}

# Run ADMS for each project file based on suffix
echo "Running ADMS"
Set-Location $static_data_path
Get-ChildItem $static_data_path -Filter $app_suffix |
Foreach-Object {
echo "Running ADMS for:" $_.BaseName
Start-Process -Wait -FilePath "$app_path/$app_bin" -ArgumentList "/E2",$_.FullName
}
```

Figure 9 Excerpt from the ADMS PowerShell script

5.4 AGRICULTURAL DECISION MAKING

5.4.1 Workflow and updated description

This use case concerns the use of weather forecast in order to run some agronomic models to predict some parameters such as production yields.

These agronomic models are run by LIMAGRAIN company on their own servers, in consequence the workflow to be executed on LEXIS infrastructure corresponds to only execution of WRF weather model such as described in Section 4.

5.4.2 Orchestration and Docker for virtualization

The workflow to run a WRF simulation is:

- Downloading the required input data (static geographical data and Global Forecast System data) for a date specified by the user,
- Pre-processing these data on a Cloud Compute instance using a pre-processing container provided by CIMA[12]
- Transferring this data to a WRF HEAppE job on a HPC cluster,
- Transferring these results to LEXIS DDI (Distributed Data Infrastructure) and optionally on a SFTP server when specified by the user.

A detailed description of the workflow and its TOSCA implementation is available in the LEXIS project GitHub[13].

5.5 EXTREME RAINFALL DETECTION SYSTEM

5.5.1 Workflow and updated description

The Extreme Rainfall Detection System (ERDS) is a service for the monitoring and forecasting of rainfall events with a global geographic coverage developed by ITHACA. Available capabilities include the analysis of both near real-time rainfall measurements and rainfall forecasts, for the delivering of extreme rainfall alerts.

[12] CIMA pre-processing container: https://github.com/meteocima/wps-da.ifs
[13] Agriculture workflows and TOSCA implementation in GitHub: https://github.com/lexis-project/application-templates/tree/master/weather-climate/applications/agriculture

NASA/JAXA Global Precipitation Measurement (GPM) IMERG (Integrated Multi-SatellitE Retrieval for GPM) Early run data is downloaded as a near real-time source of rainfall measurements. GPM IMERG Early run data are characterized by a temporal resolution of 30 minutes, a 0.1° x 0.1° spatial resolution, a spatial coverage between 90° N and 90° S and a 4 hours latency. The data is downloaded every hour to provide maps of accumulated rainfall over the past 12, 24, 48, 72 and 96 hours.

This system is also able to provide rainfall forecasts using Global Forecast System (GFS) models data, characterized by a 0.25° x 0.25° spatial resolution and worldwide coverage. GFS model runs 4 times a day and data obtained from the 00 and 12 UTC model executions is used to provide forecasts for the upcoming 12, 24, 48, 72 and 96 hours.

The main novelty of this system is the inclusion of Weather Research and Forecasting Model (WRF) data at a 7.5 km resolution provided by CIMA, executed daily within the LEXIS WP7. Forecasts related to the upcoming 24 and 48 hours are used to provide more detailed information over Europe.

Both near real-time rainfall measurements and rainfall forecasts are then processed to detect extreme rainfall. The extreme rainfall detection methodology is based on rainfall thresholds [20]: if, for a selected aggregation interval, the accumulated precipitation exceeds the threshold, an alert is provided. Threshold values were evaluated, for the different time intervals, using an empirical approach: rainfall events that occurred in the last years were analysed to determine the threshold values that allow to maximize the number of identified events, minimizing the number of false alarms. These values vary from place to place and allow to obtain alerts at the same resolution of the input data.

Maps of accumulated rainfall and related rainfall alerts are then published through a WebGIS application, developed in an open source environment.

The entire workflow is summarized in Figure 10.

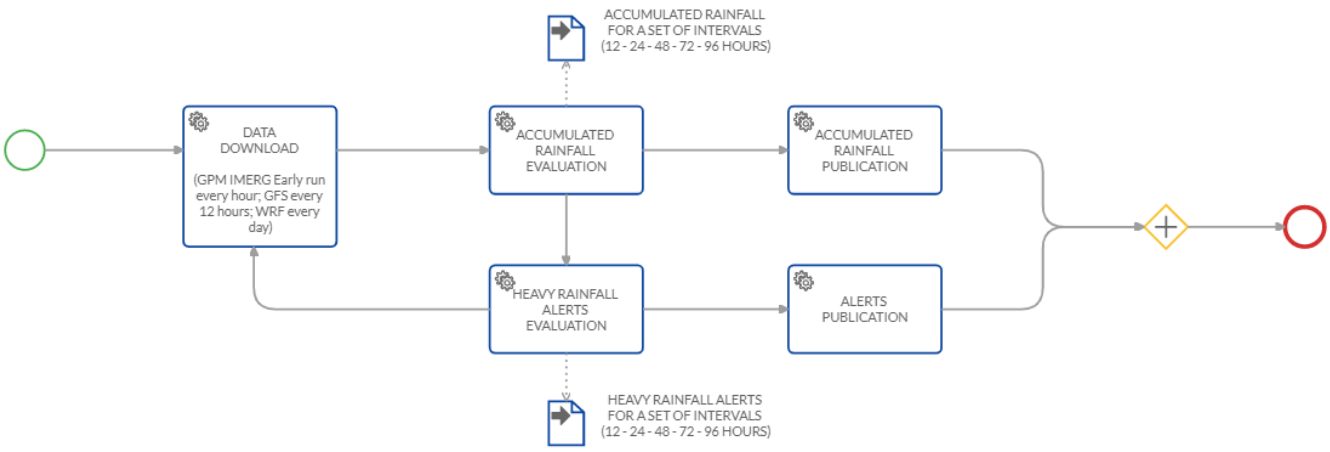

Figure 10 ERDS prediction workflow description

5.5.2 Orchestration and Docker for virtualization

For the rainfall data management two essential and mandatory containers are used.

One container is responsible for the GPM and GFS input data download and processing. Basically, it manages the following operations[14][15]:

- input data download from the official repositories (GPM IMERG and GFS data),
- conversion into numpy arrays,

[14] GPM: https://github.com/ITHACA-org/gpm-accumul
[15] GFS: https://github.com/ITHACA-org/gfs-accumul

- calculation of the accumulated rainfall depths,
- comparison with the thresholds for every accumulation interval and every location,
- extraction of potential alerts,
- writing the output .tiff files with the accumulated precipitation and with the potential alerts.

To allow for performing these operations, this container was prepared with a version of Python3, the GDAL software with its Python libraries, and a series of additional Python libraries such as h5py (for reading HDF5 files), numpy (for managing multidimensional arrays), and scipy (for statistical analysis). Moreover, the software wgrib2[16] was installed as an interface to access GFS data. This container must have access to the internet to gather the necessary input data.

The other container is responsible for WRF data management. The procedure[17] is similar to the first container described above:

- input data download from CIMA repository (WRF data),
- conversion into numpy arrays,
- calculation of the accumulated rainfall depths,
- comparison with the thresholds for every accumulation interval and every location,
- extraction of potential alerts,
- writing the output .tiff files with the accumulated precipitation and with the potential alerts.

Finally, a working instance of the GeoServer application[18] is responsible for the publication of the accumulated precipitation data as well as the alerts through the WMS standard. As a standalone servlet the GeoServer package has been installed for use with a Tomcat servlet container[19].

Port 80 has been open to the public to allow for a proper publication through the http protocol. Moreover, a shared folder with the containers has been set up to read the data produced by the containers and publish them.

A web application is used to visualize the data on a map. ERDS outputs (accumulated rainfall data and alerts) are available[20], as well as on the original ERDS website[21].

[16] wgrib2: https://www.cpc.ncep.noaa.gov/products/wesley/wgrib2
[17] WRF data management: https://github.com/ITHACA-org/aux-accumul
[18] Geoserver: http://geoserver.org/
[19] Tomcat servlet container: https://tomcat.apache.org
[20] ERDS outputs (accumulated rainfall data and alerts): https://erds.lexis.tech/
[21] ERDS website: http://erds.ithacaweb.org/

6 SUMMARY

In this document we describe the final deployment of each WP7 use case inside the LEXIS infrastructure.

LEXIS orchestration and data transfer management allows one to run WRF (the main part of WP7 use case) in any HPC infrastructure of the LEXIS project and execute the different WP7 applications (Continuum, RISICO, ADMS, ERDS) on any cloud infrastructure of LEXIS, or even to communicate with partner's servers.

WRF workflow allows to manage different global weather initialization (GFS or IFS), assimilation or not (of surface observations, radars observations), on operational basis (daily forecast) or simulate past episodes.

All others application's workflows are operational.

Metrics of execution in the LEXIS infrastructure, performance of the system and KPI will be properly discussed in the LEXIS Deliverable D7.9 [3].

REFERENCES

[1] LEXIS Deliverable, *D7.4 Deployment of the First Test-bed Infrastucture Components in HPC/Cloud.*

[2] LEXIS Deliverable, *D7.6 Deployment of Test-bed Infrastructure Components and Adoption of Weather and Climate Data Interchange for Model Layer Interoperability.*

[3] LEXIS Deliverable, *D7.9 Final Report (KPI Included) on Demonstration and Validation of the Weather & Climate Test-bed Applied to Selected Cases.*

[4] LEXIS Deliverable, *D4.6 Design and Implementation of the HPC-Federated Orchestration System – Final.*

[5] „OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA),“ 2020. [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca.

[6] “LEXIS project application templates repository,” [Online]. Available: https://github.com/lexis-project/application-templates.

[7] LEXIS Deliverable, *D7.5 First Release and Test-bed Deployment of Weather and Climate Data Interchange for both In-situ Unstructured Observations and Model Data Output.*

[8] LEXIS Deliverable, *D7.7 Field Deployment of a Smart Gateway for Collecting, Pre-processing, and Transmitting In-situ Obeservations.*

[9] LEXIS Deliverable, *D7.3 Design of a Smart Gateway for Collecting, Pre-processing, and Transmitting In-situ Observations.*

[10] D. Barker, X. Y. Huang, Z. Liu and et al., “The weather research and forecasting model's community variational/ensemble data assimilation system: WRFDA,” *Bulletin of the American Meteorological Society,* vol. 6, no. 93, pp. 831-843, 2012.

[11] D. M. Barker, W. Huang, Y. R. Guo and Q. N. Xiao, “A Three-Dimensional (3DVAR) Data Assimilation System For Use With MM5: Implementation and Initial Re-sults,” *Mon. Wea. Rev.,* no. 132, pp. 897-914, 2004.

[12] W. C. Skamarock and et al., “A description of the Advanced Research WRF version 3. NCAR Tech Note NCAR/TN-475+STR,” 2008. [Online]. Available: www.mmm.ucar.edu/wrf/users/docs/arw_v3.pdf.

[13] X. Y. Huang, Q. Xiao, D. M. Barker and et al., “Four-Dimensional Variational Data Assimilation for WRF: Formulation and Preliminary Results,” *Mon. Wea. Rev.,* no. 137, pp. 299-314, 2009.

[14] J. G. Powers, J. B. Klemp, W. C. Skamarock and et al., “The weather research and forecasting (WRF) model: overview, system efforts, and future directions,” *Bulletin of the American Meteorological Society,* 2017.

[15] D. F. Parrish and J. C. Derber , “The National Meteorological Center’s Spectral Statistical Interpolation analysis system,” *Mon. Wea. Rev.,* no. 120, p. 1747–1763, 1992.

[16] W.-S. Wu, R. J. Purser and D. F. Parrish, “Three-Dimensional Variational Analysis with Spatially Inhomogeneous Covariances,” *Mon. Wea. Rev.,* no. 130, p. 2905–2916, 2002.

[17] I. Dharssi, A. C. Lorenc and N. B. Ingleby, “Treatment of gross errors using maximum probability theory,” *Quart. J. Roy. Meteor. Soc.,* no. 118, p. 1017–1036, 1992.

[18] X. Zou, F. Vandenberghe, M. Pondeca and Y.-H. Ku, “Introduction to adjoint techniques and the MM5 adjoint modeling system,” *NCAR Tech. Note NCAR/TN-435 1 STR,* p. 110, 1997.

[19] A. C. Lorenc, “Modelling of error covariances by 4D-Var data assimilation,” *Quart. J. Roy. Meteor. Soc.,* no. 129, p. 3167–3182, 2003.

[20] P. Mazzoglio, F. Laio, S. Balbo, P. Boccardo and F. Disabato, “Improving an extreme rainfall detection system with GPM IMERG data,” *Remote Sensing,* vol. 11, no. 6, p. 677, 2019.

[21] C. M. Hayden and R. J. Purser, “Recursive Filter Objective Analysis of Meteorological Fields: Applications to NESDIS Operational Processing,” *Journal of Applied Meteorology,* vol. 34, no. 1, p. 3, 1995.

[22] R. J. Purser, W.-S. Wu, D. Parrish and N. Roberts, “Numerical Aspects of the Application of Recursive Filters to Variational Statistical Analysis. Part I: Spatially Homogeneous and Isotropic Gaussian Covariances,” *Monthly Weather Review,* vol. 131, p. 1524, 2003.

A DETAILS ON BACKGROUND ERROR CALCULATION FOR ASSIMILATION WITH WRF

The NMC-method estimates climatological background error covariances using a process that assume background errors to be well approximated by averaged forecast difference statistics (see Equation 2):

$$\mathbf{B} = \overline{(\mathbf{x}'^{b} - \mathbf{x}'^{t})(\mathbf{x}'^{b} - \mathbf{x}'^{t})^{\mathrm{T}}} = \overline{\varepsilon_b \varepsilon_b{}^{\mathrm{T}}} \approx \overline{(\mathbf{x}'^{\mathrm{T}+24} - \mathbf{x}'^{\mathrm{T}+12})(\mathbf{x}'^{\mathrm{T}+24} - \mathbf{x}'^{\mathrm{T}+12})^{\mathrm{T}}} \quad (2)$$

Where $\mathbf{x}'^{t}$ is the true state of the atmosphere and ε_b is the background error. The overbar means an average over time and/or space.

However, the background error covariance matrix is computed not in the model space $\mathbf{x}'(u, v, T, q, ps)$ but in a control variable space ($\mathbf{v}$) related to the model space through the control variable transform ($\mathbf{U}$):

$$\mathbf{x}' = \mathbf{U}\mathbf{v} = \mathbf{U}_p \mathbf{U}_v \mathbf{U}_h \mathbf{v} \quad (3)$$

The expansion $\mathbf{U} = \mathbf{U}_p \mathbf{U}_v \mathbf{U}_h$ in Equation 3 represents a series of operations [19] implemented in different stages of covariance modelling: horizontal correlations ($\mathbf{U}_h$), vertical covariances ($\mathbf{U}_v$) and multivariate covariances ($\mathbf{U}_p$). The v components are chosen so that their error cross-correlation are negligible, so the $\mathbf{B}$ matrix will be block-diagonalized.

Gen_be calculates the background error covariance matrix through 4 stages:

- Stage 0: stage 0 converters transform model-specific data (alternative models use different grids, variables, data format, etc.) to standard perturbation fields and metadata and output them in a standard binary format. The standard fields are:
  - Perturbations: stream function $\Psi'(i, j, k)$, velocity potential $\chi'(i, j, k)$, temperature $T'(i, j, k)$, relative humidity $RH'(i, j, k)$, surface pressure $p's(i, j)$.
  - Full-fields: height $z(i, j, k)$, latitude $\kappa(i, j)$ (required to produce error statistics stored in terms of physics variables rather than tied to a grid).
- Stage 1: remove the time-mean in order to calculate covariance between fields.
- Stage 2: In this stage is provided a statistics for the unbalanced fields (χ_u, T_u, P_{su}) used as control variables in WRFDA. The unbalanced control variables are the difference between the full and balanced components of the field. In Stage 2 the balanced component of particular fields is modelled through a regression analysis of the fields using specific predictor fields [16]. The regression coefficients obtained are output to be used in WRFDA Up transform. The regression analysis results in three sets of regression coefficients:
  - Velocity potential – streamfunction regression:
    $$\chi_u(k) = c(k)\Psi(k)$$
  - Temperature – streamfunction regression:
    $$T_b(k) = \sum_{k,l} G(k1, k)\Psi(k1)$$
  - Surface pressure – streamfunction regression:
    $$P_{sb} = \sum_{k,l} W(k1)\Psi(k1)$$

  Note: perturbation notation dropped for clarity.
  The sum over $k1$ relates to integral relationship between mass fields and the wind field. The regression coefficient c, G and W do not vary horizontally by default. Once computed the regression coefficients, the unbalanced components of fields will be:

  $$\chi_u(k) = \chi(k) - c(k)\Psi(k)$$

  $$T_u(k) = T(k) - \sum_{k,l} G(k1, k)\Psi(k1)$$

  $$p_{su} = p_s - \sum_{k.l} W(k1)\Psi(k1)$$

  The fields calculated in this way will be the input for the next stage (spatial covariances).

- Stage 3: calculates statistics necessary for the vertical component of the control variable transform. The vertical transform $\mathbf{U}_v$ is applied using an empirical orthogonal function (EOF) decomposition of background error covariances [10]. The vertical component of $\mathbf{B}(\mathbf{B}_v)$ is calculated for each 3D control variable and eigenvector decomposition is calculated on model levels k. $\mathbf{B}_v$ considering K model levels will be a $K \times K$ positive-definite, symmetric matrix and given an estimate of $\mathbf{B}_v$ (through NMC method), the eigendecomposition $\mathbf{B}_v = \mathbf{E}\Lambda\mathbf{E}^{\mathbf{T}}$ of the matrix is performed to obtain eigenvectors $\mathbf{E}$ and eigenvalues Λ. After this calculation, the entire sequence of 3D control variables is projected into EOF space (vertical transform $\mathbf{U}_v$):

  $$\mathbf{v}_p = \mathbf{U}_v\mathbf{v}_v = \mathbf{E}\Lambda^{1/2}\mathbf{v}_v$$

- Stage 4: computes the last requirement of background error covariance, the horizontal error correlations, through the use of recursive filters [21, 22] in case of regional applications and power spectra for global mode. In regional applications the horizontal correlations are computed between each 2D field grid points, binned as a function of distance. A Gaussian curve is fitted to the data to provide the length scales of correlation that is used in the recursive filter algorithm (refer to Barker et al. [11] for further details). In contrast, in global applications power spectra are computed for each vertical modes (K) control variables Ψ, χ_u, T_u, RH_u and p_{su}.

From a more practical point of view, in `gen_be` utility the users have four choices to define the B matrix, called CV3, CV5, CV6 and CV7. Each of them has different properties and considers different control variables as it is reported in Table 3.

CV OPTION	CONTROL VARIABLES
CV3	$\Psi, \chi_u, T_u, q, p_{su}$
CV5	$\Psi, \chi_u, T_u, RH_s, p_{su}$
CV6	$\Psi, \chi_u, T_u, RH_{su}, p_{su}$
CV7	u, v, T, RH_s, p_s

Table 3 Background error covariances matrix options

where: Ψ is the streamfunction, χ_u is the unbalanced velocity potential, T_u is the unbalanced temperature, p_{su} is the unbalanced surface pressure, RH_s is the pseudo relative humidity, RH_{su} is the unbalanced pseudo relative humidity and p_s is the surface pressure.

The CV3 option is provided in WRFDA and the control variables are in physical space while CV5, CV6 and CV7 are obtained through the use of `gen_be` and the control variables are in eigenvector space. The main difference between CV3 and others is linked to the vertical covariance: CV3 uses the vertical recursive filter, while others use EOF to model the vertical covariance. Also, the recursive filters to model horizontal covariance are different. In general, CV3 is a global $\mathbf{B}$ matrix that could be used in any regional domain, while CV5, CV6 and CV7 are domain-dependent $\mathbf{B}$ matrices and should be generated on forecasts or ensemble data from the same domain that will be used for data assimilation. Furthermore, the difference between CV5, CV6 and CV7 is only linked to the control variables they use.

B DETAILS ON WPS DOCKER

Input and output volumes

This container expects three volumes to be mounted under `/output`, `/input` and `/geogrid`.

- GFS files should be saved under `/input`,
- Static geo data should be saved under `/geogrid`,
- `/output` will contains resulting file at the end of the container run.

Environment variables

- `WPS_START_DATE` - first date of the range you want to prepare
- `WPS_END_DATE` - last date of the range you want to prepare.
- `WPS_MODE` - type of simulation you want to prepare. It can assume following values: '`WARMUPDA`', or '`WRFDA`'
- `WRFDA` mode - preprocess the data needed to run a WRFDA simulation with data assimilation. It includes in the preprocessed data instant WPS_START_DATE-3HOUR and WPS_START_DATE-6HOUR, that are used during data assimilation. It will be used by Continuum and ADMS simulations when data assimilation will be used.
- `WARMUPDA` mode - preprocess the data needed to run three WRF simulation, with data assimilation. Two of this WPS execution preprocess the data to run two WRF simulation (with no assimilation) for days `WPS_START_DATE-1` and `WPS_START_DATE-2`. These two sets of data are the "warmup" data. The other WPS execution prepares the normal WRF simulation (with assimilation) from `$WPS_START_DATE` to `$WPS_END_DATE`. Warmup data is actually used by RISICO simulation.

The container will produce in output a set of `netcdf` files and an `arguments.txt` file that contains information that will be consumed by the WRFDA HEAppE template. This file will contain the name of configuration file that will be used by the HPC part, and a list of starting dates to run. The HPC part needs to know which configuration file to use in order to tell Italian and France configuration. The `WARMUPDA` mode always produces three dates of output, and should be used only to start a RISICO chain. The `WRFDA` mode always produces a single date, and should be used for all other Italy and France chains.

Container internals

The container uses the version WPS v4.1 and WRF v4.1.5 models. The models could be compiled using a slightly customized CIMA version from these forks:

- `https://github.com/meteocima/WPS.git branch v4.1-smoothpasses-cima`
- `https://github.com/meteocima/WRF.git branch v4.1.5-cima`

Both models are compiled using Intel Parallel Studio XE Composer Edition 2020 in order to improve performance.

C DETAILS ON LEXIS DOWNLOADER DOCKER

Input and output volumes

This container expects a volume to be mounted under `/output`. It will contain resulting files at the end of the container run.

Environment variables

The container expects the following arguments, as environment variables:

- `START_DATE` - date of start of simulation,
- `DOWNLOAD_TYPE` - Type of data to download. Can be either "`RISICO`" | "`CONTINUUM`" | "`ADMS`" | "`LIMAGRAIN`",
- `WEBDROPS_USER` - User to authenticate with webdrops service,
- `WEBDROPS_PWD` - Password to authenticate with webdrops service.

`WEBDROPS_USER` and `WEBDROPS_PWD` arguments are used to authenticate on CIMA webdrops service.

`DOWNLOAD_TYPE` arguments is used to tell the kind of data to download. Value corresponds to the name of the last simulation in each available chain. The meaning of flag values is explained below in more details.

Kind of downloads

- `RISICO` is used to download data needed by the RISICO chain. Downloaded data consists of:
  - A spatial interpolated map of data coming from Wunderground weather stations. Interpolation is performed by CIMA Webdrops engine.
  - Raw data of Wunderground weather stations, converted in a text format understood by WRF.
  - Raw data of Italian radar data, converted in a text format understood by WRF.
  - Since RISICO requires 2 days of data for warming up its calculation, points 1,2,3 are repeated 3 times, for the date requested and for 1 and 2 days before.
- `CONTINUUM` is used to download data needed by the Continuum chain. Downloaded data consists of:
  - A time interpolated map of data coming from Wunderground weather stations. Interpolation is performed by CIMA Webdrops engine.
  - Raw data of Wunderground weather stations, converted in a text format understood by WRF.
  - Raw data of Italian radar data, converted in a text format understood by WRF.
- `LIMAGRAIN` is used to download data needed by the LIMAGRAIN chain. Downloaded data consists of raw data of Wunderground weather stations, converted in a text format understood by WRF.
- `ADMS` is used to download data needed by the ADMS chain. Downloaded data consists of raw data of Wunderground weather stations, converted in a text format understood by WRF.

Note that for France simulation, the France radar data are not stored in CIMA WebDrops server, so they must be downloaded directly by LEXIS DDI.

Container internals

The container runs the `lexisdn` command that takes care of downloading data. Sources are available on GitHub: `cima-lexis/lexisdn`.

The command also has two dependencies, used to convert radar and Wunderground data in a text format understood by WRF. Source for these two libraries are also available on GitHub: `meteocima/dewetra2wrf and meteocima/radar2wrf`

D DETAILS ON FLOOD-PROOFS DOCKER

Environment variables

The Docker environment variable file (.env) is crucial when you're creating complex container deployments. This file allows you to declare environment variables for your containers. This comes in quite handy, as the .env file can be reused for other containers or quickly edited. For the Flood-PROOFS virtualization an example of .env file is reported below.

Docker static argument(s)

```
image_version='latest'
image_base_name='c-hydro'
image_app_name='fp_framework'
image_app_file='fp-docker_file'
image_repository=''
container_workdir='/home/fp/fp_entrypoint/'
container_name='lexis'
image_app_entrypoint_main='fp_docker_entrypoint_app_main.py'
image_app_entrypoint_configuration='fp_docker_entrypoint_app_configuration.
json'
```

Docker dynamic argument(s) `[SOURCE::local, TARGET::container]`

```
SOURCE_DATA_STATIC='$HOME/docker-ws/data/data_static/'
SOURCE_DATA_DYNAMIC_SOURCE='$HOME/docker-ws/data/dynamic_data/source/'
SOURCE_DATA_DYNAMIC_ARCHIVE='$HOME/docker-ws/data/dynamic_data/archive/'
TARGET_DATA_STATIC='/home/fp/fp_mount/data/data_static/'
TARGET_DATA_DYNAMIC_SOURCE='/home/fp/fp_mount/data/data_dynamic/source/'
TARGET_DATA_DYNAMIC_ARCHIVE='/home/fp/fp_mount/data/data_dynamic/archive/'
RUN_TIME_NOW=2019-05-12 11:00
RUN_TIME_STEPS_OBS=4
RUN_TIME_STEPS_FOR=0
RUN_NAME='fp_run'
RUN_ENS=false
```

All the information is used in building and running parts. To achieve a correct settings of Flood-PROOFS Dockers, all the fields must be filled. In the following lines, an example of fields of docker invariable file is presented.

Generally, the first part is for the building section:

- `image_version`: version of the image `[string]`,
- `image_base_name`: base name of the image `[string]`,
- `image_app_name`: application name of the image `[string]`,
- `image_app_file`: filename of the image Dockerfile `[string]`,
- `image_repository`: source root of the image (e.g. in GitHub) `[string]`,
- `container_workdir`: absolute path of the working directory referred to the container `[string]`,
- `container_name`: name of the container `[string]`,
- `image_app_entrypoint_main`: application filename for configuring entrypoint activities `[string]`,
- `image_app_entrypoint_configuration`: configuring filename of the entrypoint activities `[string]`.

In the second part, users can find the running section:

- `SOURCE Folders`: absolute paths referred to the host folders `[string]`:
  - `SOURCE_DATA_STATIC`: folders for static datasets,
  - `SOURCE_DATA_DYNAMIC`: folders for dynamic datasets,
- `TARGET folders`: absolute paths referred to the container folders `[string]`:

  - `TARGET_DATA_STATIC`: folders for static datasets,
  - `TARGET_DATA_DYNAMIC`: folders for dynamic datasets,
- `RUN_TIME_NOW`: reference time of the simulation (e.g. time reference for a test case or for a real-time simulation) `[yyyy-mm-dd HH:MM]`,
- `RUN_TIME_STEPS_OBS`: time steps in the simulation observed part `[integer]`,
- `RUN_TIME_STEPS_FOR`: time steps in the simulation forecasting part `[integer]`,
- `RUN_NAME`: name of the simulation `[string]`,
- `RUN_ENS`: simulation mode (deterministic/probabilistic) `[boolean: false/true]`.

Other environment variables can be listed for different dockers and procedures.

Run applications

For running each component of the operational chain, the users have to set, for example, the case based on an example of NWP deterministic run. This case is based on two different procedures:

- **Hyde Docker**: to collect the observed datasets and the nwp wrf outcome; finally, to prepare datasets for the hydrological model,
- or, **HMC Docker**: to run the Continuum hydrological model using the data previously collected.

The dockers have to be prepared following the steps below:

- Download or collect from repositories the input datasets,
- Create and update the `fp-docker_variables_hmc.env` and `fp-docker_variables_hyde.env` files according with the host, the containers and the simulation features.

The **Hyde Docker** has to be prepared and run following the steps below:

1. Configuring the folders and the parameters in the `fp-docker_variables_hyde.env` file:

```
SOURCE_DATA_STATIC='/datasets/data/static_data/land/'
SOURCE_DATA_DYNAMIC_RAW='/datasets/dynamic_data/'
SOURCE_DATA_DYNAMIC_PROCESSED='/datasets/processed_data/'
RUN_TIME_NOW=2020-06-17 00:00
RUN_TIME_STEPS_OBS=10
RUN_TIME_STEPS_FOR=24
RUN_NAME='fp_hyde_wrf'
RUN_DOMAIN='marche'
RUN_ENS=false
```

2. Building an image:

```
>> ./fp-docker_builder.sh -f fp-docker_variables_hyde.env
```

3. Put the observed datasets and the nwp wrf datasets in the `SOURCE_DATA_DYNAMIC_RAW` folder; the structure of datasets will be as follows:

```
docker datasets
├── dynamic_data
│   ├── anag-ANEMOMETRO.json
│   ├── anag-IGROMETRO.json
│   ├── anag-PLUVIOMETRO.json
│   ├── anag-RADIOMETRO.json
│   ├── anag-TERMOMETRO.json
│   ├── ANEMOMETRO.json
│   ├── auxhist23_d03_2020-06-17_00:00:00
│   ├── auxhist23_d03_2020-06-17_01:00:00
│   ├── ...
│   ├── ...
│   ├── auxhist23_d03_2020-06-18_00:00:00
```

```
│   ├── IGROMETRO.json
│   ├── PLUVIOMETRO.json
│   ├── RADIOMETRO.json
│   └── TERMOMETRO.json
├── processed_data
└── static_data
    └── land
    ├── marche.aspect.txt
    ├── marche.dem.txt
    ├── marche.hillshade.txt
    ├── marche.roughness.txt
    └── marche.slope.txt
```

4. Running the container in executable mode:

```
>> ./fp-docker_runner_hyde.sh -f fp-docker_variables_hyde.env
```

5. Collecting the datasets in the SOURCE_DATA_DYNAMIC_PROCESSED folder of the Hyde docker and copy it to the source folder SOURCE_DATA_DYNAMIC_SOURCE of the HMC docker:

```
>> cp
/docker_testcase/datasets_hmc/dynamic_data/wrf_{date_outcome_string}_{do
main_string}.nc.gz /docker_testcase/datasets_hmc/dynamic_data/'
```

If the users need to open the Hyde Docker in interactive mode, they can use the follow command-line:

```
>>  docker run --workdir /home/fp/fp_entrypoint/\ -e
APP_MAIN=fp_docker_entrypoint_app_main_hyde.py -e
APP_CONFIG=fp_docker_entrypoint_app_configuration_hyde.json\ --rm  --
env-file fp-docker_variables_hyde.env\
   -it --entrypoint bash  --mount
type=bind,source=${SOURCE_DATA_STATIC},target=${TARGET_DATA_STATIC}\
   --mount
type=bind,source=${SOURCE_DATA_DYNAMIC_RAW},target=${TARGET_DATA_DYNAMIC
_RAW}\
   --mount
type=bind,source=${SOURCE_DATA_DYNAMIC_PROCESSED},target=${TARGET_DATA_D
YNAMIC_PROCESSED}\
c-hydro/fp_framework_hyde
```

The **HMC Docker** has to be prepared and run following the steps below:

1. Configuring the folders and the parameters in the `fp-docker_variables_hmc.env` file:

```
SOURCE_DATA_STATIC_LAND='/datasets/static_data/land/'
SOURCE_DATA_STATIC_POINT='/datasets/static_data/point/'
SOURCE_DATA_DYNAMIC_SOURCE='/datasets/dynamic_data/'
SOURCE_DATA_DYNAMIC_RESTART='/datasets/dynamic_restart/'
SOURCE_DATA_DYNAMIC_ARCHIVE='/datasets/docker_data/'
RUN_TIME_NOW=2020-03-20 00:00
RUN_TIME_STEPS_OBS=10
RUN_TIME_STEPS_FOR=15
RUN_NAME='fp_run'
RUN_ENS=false
```

The users have to be check that the dynamic datasets are in the `SOURCE_DATA_DYNAMIC_SOURCE`. Particularly:

- The wrf datasets elaborated by the Hyde Docker,
- The observed datasets for the previous 24 hours available on the docker_testcase folders hosted by github.

2. Building the image:

```
>> ./fp-docker_builder.sh -f fp-docker_variables_hmc.env
```

3. Running the container in executable mode:

```
>> ./fp-docker_runner_hmc.sh -f fp-docker_variables_hmc.env
```

4. Collecting the results of the Continuum hydrological model in the `SOURCE_DATA_DYNAMIC_ARCHIVE`.

If the users need to open the HMC Docker in interactive mode, they can use the follow command-line:

```
>>  docker run --workdir /home/fp/fp_entrypoint/\
   -e          APP_MAIN=fp_docker_entrypoint_app_main_hmc.py          -e
APP_CONFIG=fp_docker_entrypoint_app_configuration_hmc.json\
   --rm  --env-file fp-docker_variables_hmc.env\
   -it --entrypoint bash\
   --mount
type=bind,source=${SOURCE_DATA_STATIC_LAND},target=${TARGET_DATA_STATIC
_LAND}\
   --mount
type=bind,source=${SOURCE_DATA_STATIC_POINT},target=${TARGET_DATA_STATI
C_POINT}\
   --mount
type=bind,source=${SOURCE_DATA_DYNAMIC_RESTART},target=${TARGET_DATA_DY
NAMIC_RESTART}\
   --mount
type=bind,source=${SOURCE_DATA_DYNAMIC_SOURCE},target=${TARGET_DATA_DYN
AMIC_SOURCE}\
   --mount
type=bind,source=${SOURCE_DATA_DYNAMIC_ARCHIVE},target=${TARGET_DATA_DY
NAMIC_ARCHIVE}\
   c-hydro/fp_framework_hmc
```

E DETAILS ON RISICO DOCKER

The responsibility of the RISICO docker is to adapt the data needed for running the RISICO model on the RISICO own binary format, run the RISICO model itself, and create the output netcdf.

Sources are available on GitHub: `cima-lexis/risico-docker`

The container expects a volume to be mounted to read and write input and output data, and a single environmental variable.

Input and output volume

This container expects a volume to be mounted under `/home/risico/data`. The mounted volume should contain a subfolder named `wrf`, containing the input files for the RISICO model. An output folder will be created in the mounted volume, containing the output data from the RISICO simulation.

Environment variables

The container expects the following arguments, as environment variables:

- `RISICO_RUN_DATE` - run date for the model, format is `YYYYMMDD0000`