Large-scale EXecution for Industry & Society

Deliverable D7.6

Deployment of test-bed infrastructure components and adoption of Weather and Climate Data Interchange for model layer interoperability

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	D7.6 | Deployment of test-bed infrastructure components and adoption of Weather and Climate Data Interchange for model layer interoperability
RESPONSIBLE AUTHOR	Antonio Parodi (CIMA)
WORKPACKAGE ID | TITLE	WP7 | Weather and Climate Large-scale Pilot
WORKPACKAGE LEADER	CIMA
DATE OF DELIVERY (CONTRACTUAL)	31/03/2020 (M15)
DATE OF DELIVERY (SUBMITTED)	01/04/2020 (M16)
VERSION | STATUS	V1.2 | Final
TYPE OF DELIVERABLE	R (Report)
DISSEMINATION LEVEL	PU (Public)
AUTHORS (PARTNER)	Antonio Parodi (CIMA); Laurent Ganne (Bull/Atos); Andrea Parodi (CIMA); Paola Mazzoglio (ITHACA); Fabien Brocheton (NUM)
INTERNAL REVIEW	Thierry Goubier (CEA); Natalja Rakowsky (AWI)

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: sections 1-1.5	02/03/2020	Antonio Parodi (CIMA)
0.2	Second draft: section 2	02/03/2020	Laurent Ganne (Bull/Atos) Fabien Brocheton (NUM) Andrea Parodi (CIMA)
0.3	Third draft: revision of figures	04/03/2020	Antonio Parodi (CIMA)
0.4	Fourth draft: revision of section 1.5	06/03/2020	Paola Mazzoglio (ITHACA)
0.5	Fifth draft: inclusion of reviewer comments	13/03/2020	Antonio Parodi (CIMA)
1.0	First public version	26/03/2020	Antonio Parodi (CIMA)
1.1	Final check	26/03/2020	Kateřina Slaninová (IT4I)
1.2	Update of Figure 6	01/04/2020	Jan Křenek (IT4I) Jan Martinovič (IT4I)

GLOSSARY

ACRONYM	DESCRIPTION
ADMS	Atmospheric Dispersion Modelling System
ECMWF	European Centre for Medium-Range Weather Forecasts
ERDS	Extreme Rainfall Detection System
GPCC	Global Precipitation Climatology Centre
IFS	Integrated Forecasting System
IMERG	Integrated Multi-Satellite Retrievals for GPM
NASA	National Aeronautics and Space Administration
NCEP	National Centers for Environmental Prediction
NOAA	National Oceanic and Atmospheric Administration
NWP	Numerical Weather Prediction
RISICO	RISchio Incendi e COordinamento
WPS	WRF Preprocessing System
WRF	Weather and Research Forecasting

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

None

EXECUTIVE SUMMARY

The LEXIS (Large-scale Execution for Industry & Society) project will design and implement a platform for executing complex workflows, where the 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 is aimed at ensuring the integration of all needed technological elements. As such, 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. From this viewpoint, the LEXIS platform integrates orchestration, data and security management architectural components that make the platform properly exposing its federated resources through a dedicated portal. Among the business cases WP7 will deliver a system for prediction of water-food-energy nexus phenomena and their associated socio-economic impacts. WP7 will involve multiple model layers chained together:

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

Position of the deliverable in the whole project context

Deliverable D7.6 is a product of the WP7 (Weather and Climate Large-scale Pilot), and it is related to Task 7.4 and Task 7.5 activities. This document is to be delivered at the end of M15.

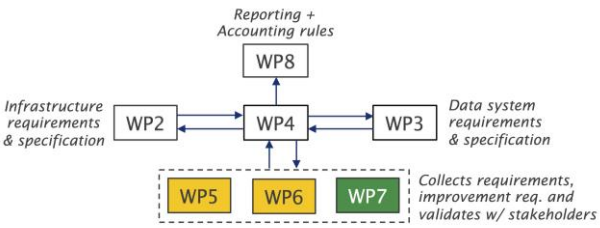

Figure 1 - Position of WP7 in the LEXIS project

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 LEXIS storage solution on which workflows and orchestrator rely on to ensure capability of storing data. WP4 sets 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 using monitoring data for billing purposes.

Description of the deliverable

This deliverable focuses on the first deployment of WP7 models on the LEXIS platform as follows:

- WPS (WRF Preprocessing System) docker description,
- Continuum hydrological model docker description,
- RISICO (RISchio Incendi e COordinamento) forest fire risk model docker description,
- ADMS (Atmospheric Dispersion Modelling System) virtual machine description,
- ERDS (Extreme Rainfall Detection System) docker description,
- Mapping of the virtual instances on the LEXIS services.

Contributors for the deliverable content are:

- CIMA as the leader of WP7, responsible for the preparation of this document,
- CIMA as the responsible of the activities concerning WPS, Continuum and RISICO dockers,
- ITHACA as the responsible of the ERDS model,
- NUM as the responsible of the ADMS model,
- Bull/Atos as the responsible of WP7 workflows mapping on LEXIS platform.

1 WEATHER AND CLIMATE WORKFLOWS

The LEXIS WP7 includes different numerical models ranging from weather prediction to hydrological prediction, forest fire risk forecast, air quality and industrial pollution forecasting, as well as extreme rainfall detection system.

Concerning the **hydrological prediction**, the complete workflow involves different LEXIS tasks such as the meteorological model WRF (Weather and Research Forecasting), including a WRFDA data assimilation system and the fully distributed hydrological model Continuum (Figure 2). 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 (European Centre for Medium-Range Weather Forecasts) IFS (Integrated Forecasting System) and NCEP (National Centers for Environmental Prediction) GFS (Global Forecast System) 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 [1] 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 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.

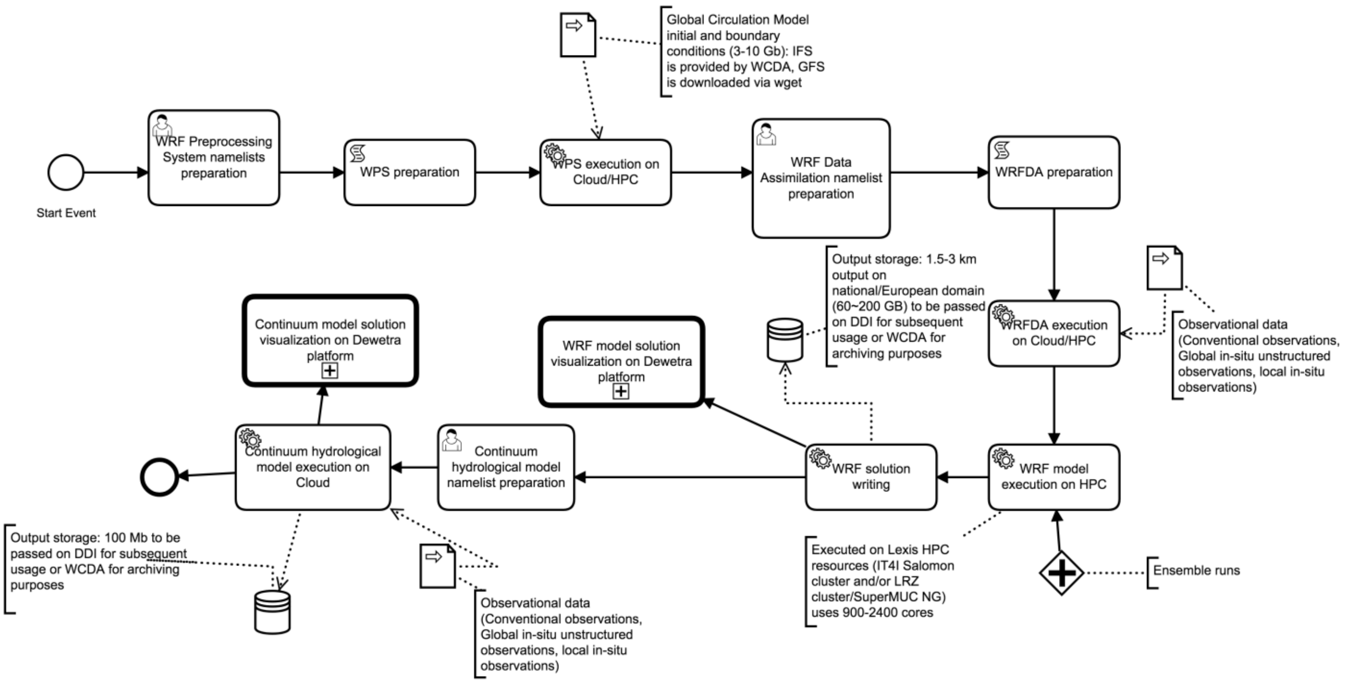

Figure 2 - LEXIS hydrological prediction workflow description

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 (Figure 3), in place of the Continuum model in the hydrological workflow. RISICO [2] 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).

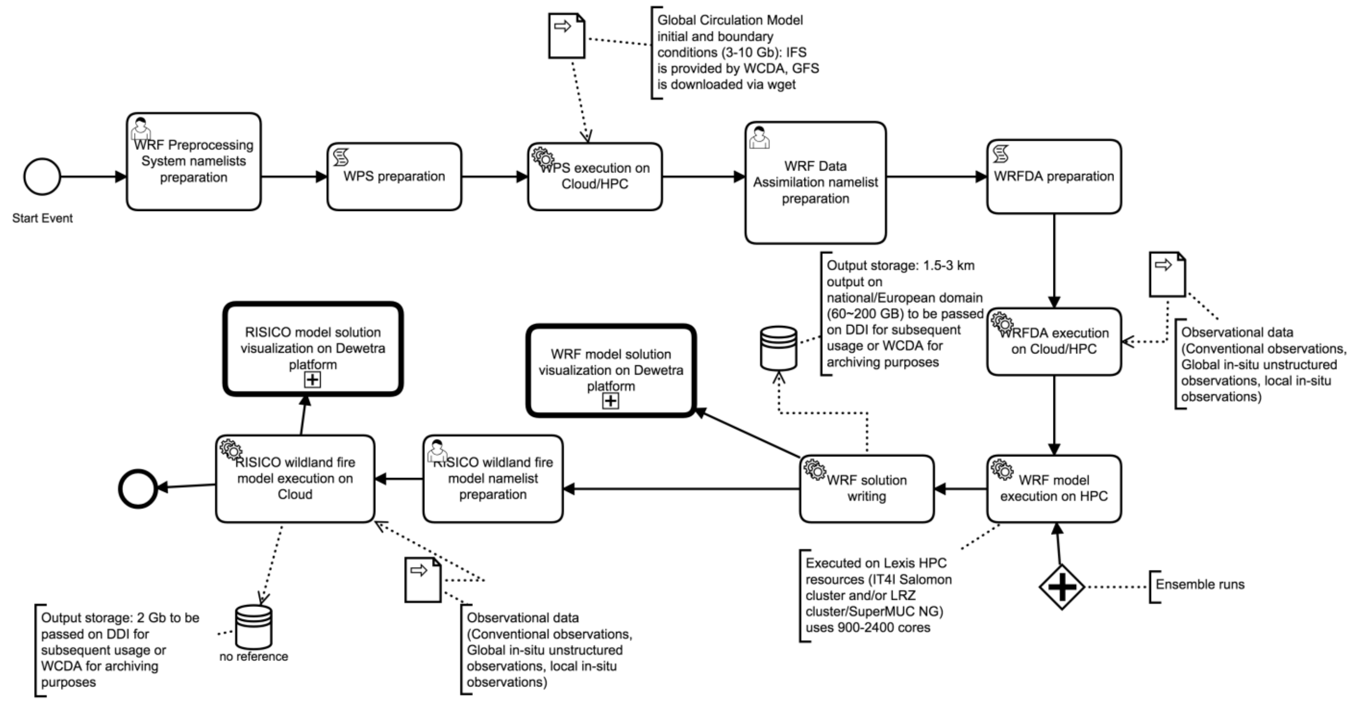

Figure 3 - LEXIS forest fire prediction workflow description

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 (Figure 4). ADMS [3] is distributed as a Windows desktop application, it uses binary (NetCDF) and text files as input/output data format. The ADMS model is executed a task on cloud computing facilities (IT4I and LRZ).

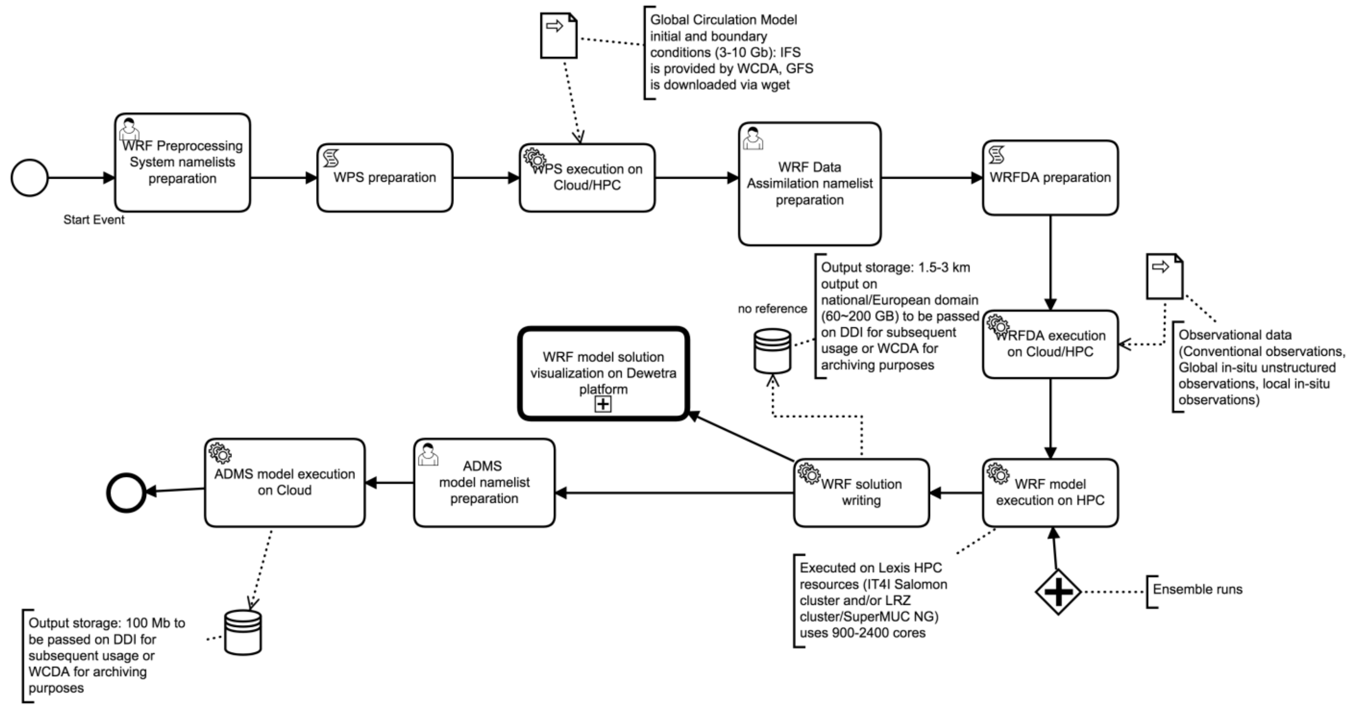

Figure 4 - LEXIS air quality prediction workflow description

The **Extreme Rainfall Detection System (ERDS)** [4] is a service for the monitoring and forecasting of exceptional rainfall events, with a nearly global geographic coverage. Available capabilities include the analysis of both the near real-time and the forecast rainfall amount for different lead times, with the aim to deliver extreme rainfall alerts. NASA Global Precipitation Measurement (GPM) data, in particular the IMERG (Integrated Multi-Satellite Retrieval for GPM) early run products, are downloaded as near real-time source of rainfall measurements. GPM IMERG data are characterized by a temporal resolution of 30 minutes, a 0.1° x 0.1° spatial resolution and a spatial coverage between 90° N and 90° S. The ERDS methodology is based on the concept of threshold (red ellipsoid in Figure 5). A threshold represents the amount of precipitation needed to trigger a flood event induced by extreme rainfall. Specifically, if for a selected aggregation interval the accumulated precipitation exceeds the threshold, an alert is provided. This set of thresholds has been calculated at a 1° x 1° spatial resolution for every aggregation interval on the basis of the mean annual precipitation that affects each place of Earth's surface. The mean annual precipitation was calculated using 10 years of GPCC monthly monitoring products. This system is also able to provide longer lead-time alerts (up to 4 days) for heavy rain, using forecast rainfall data coming from NOAA-GFS (Global Forecast System) deterministic weather prediction models, with a 0.25° x 0.25° spatial resolution and worldwide coverage, updated on a 12-hour basis.

Alerted areas can be exploited for the definition of specific Areas of Interest (i.e., areas particularly affected by heavy rainfall) that could be used as input for two different processes. The first one involves the execution of higher resolution forecasting models (not applicable over a global domain due to computational constraints). The second one is the initiation of a Satellite Emergency Mapping (SEM) mechanism. The timely identification of the most affected areas is, in fact, a valuable aspect for the pro-active request of satellite images that will be used in the emergency map production. After the reception of the satellite images, a more precise identification of the flooded areas will be performed by means of a computer-aided photointerpretation.

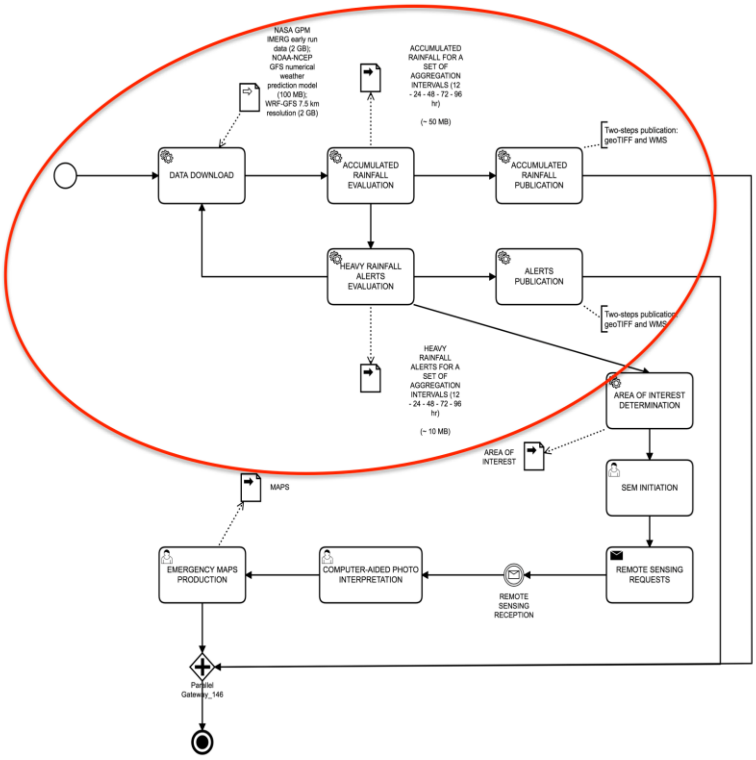

Figure 5 - ERDS prediction workflow description

1.1 VIRTUALIZATION APPROACH

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 [5] 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 resulting 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 [6] 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, the virtualization of the WPS, Continuum, RISICO, and ERDS modelling components is based on a docker framework.

Conversely the ADMS air quality model adopts a process (on windows) virtual machine approach. The virtualization of the WRFDA data assimilation system will be addressed in a subsequent phase of WP7.

1.1.1 WPS docker

This docker is meant to create and enable new build and deploy methods for complex NWP codes such as WPS and WRF in order to reduce time spent on different software compilations in various computing environments. The docker repository includes:

- GFS: empty directory that will contain the files of GFS model,
- IFS: empty directory that will contain the files of IFS model,
- namelist-prepare: go script to evaluate namelists templates,
- WPS_GEOG: empty directory that will contain the static GEO data,
- wps.compile: docker container to compile WPS, WRF and their relative dependencies,
- wps.run: base container to run WPS (inherited by wps.gfs and wps.ifs),
- wps.gfs: container to run WPS based on input from gfs model,
- wps.ifs: container to run WPS based on input from ifs model.

Build dependencies image

The following steps are required to build the image providing the dependencies:

```
cd wps.compile
docker build .
docker tag <resulting image id> wps.compile
./run.sh
```

These commands build all the dependencies of WPS and WRF, and put them in `../wps.run` where they will be added to that image.

Build base WPS image

These commands will build the `wps.run` image, that is the command parent of the two containers that will be run: `wps.ifs` and `wps.gfs`

```
cd wps.run

docker build .

docker tag <resulting image id> wps environment
```

Build IFS and GFS WPS image

The following steps are required:

```
cd wps.gfs
docker build .
docker tag <resulting image id> wps.gfs
cd ../wps.ifs
docker build .
docker tag <resulting image id> wps.ifs
```

Run IFS container

The following steps are required:

1. Make sure that ifs directory contains ifs files for the day/s you want to forecast.
2. Make sure that WPS_GEOG directory contains static geo data for the run.
3. Start and end arguments should be in YYYYMMDDHH format, and they must correspond with time the range of ifs files.

```
cd wps.ifs
./run 2019101000 2019101200          # start and date and time of the time
range to forecast.
```

Run GFS container

The following steps are required to run the GFS container:

1. Make sure that GFS directory contains gfs files for the day/s you want to forecast.
2. Make sure that WPS_GEOG directory contains static geo data for the run.
3. Start and end arguments should be in YYYYMMDDHH format, and they must correspond with time the range of ifs files.

```
cd wps.gfs
./run 2019101000 2019101200          # start and date and time of the time
range to forecast.
```

1.2 CONTINUUM HYDROLOGICAL MODEL DOCKER

The main goal of the Continuum model docker is to organize an operational chain using the computing resources available in a cloud framework. The docker includes:

- docker_configuration
    - fp-docker_builder.sh
    - fp-docker_file
    - fp-docker_runner.sh
    - fp-docker_variables.env
- docker_entrypoint
    - fp_docker_entrypoint_app_configuration.json
    - fp-docker_entrypoint_app_interface.sh
    - fp_docker_entrypoint_app_main.py
- AUTHORS.rst
- CHANGELOG.rst
- LICENSE.rst
- README.rst

The Continuum simulations that will be implemented are defined as follow:

fp_state_ws_observed

Simulation based on the weather station observations to generate the initial conditions of the Hydrological Model Continuum. The simulation covers the observed period.

```
simulation_length_obs: 10 days
simulation_length_for: 0 days
simulation_domain_n: NA
simulation_type: deterministic
simulation_n: 1/day * simulation_domain
```

fp_run_ws_observed

Simulation based on the weather station observations to compute the time-series datasets (e.g discharge, dams volume and level) and the spatial information (e.g. soil moisture, evapotranspiration and snow cover) using the Hydrological Model Continuum. The simulation covers the observed period.

```
simulation_length_obs: 2 days
simulation_length_for: 0 days
simulation_domain_n: NA
simulation_type: deterministic
simulation_n: 1/day * simulation_domain
```

fp_run_nwp_deterministic

Simulation based on the weather station observations and on the nwp datasets to compute the time-series datasets (e.g discharge, dams volume and level) and the spatial information (e.g. soil moisture, evapotranspiration and snow cover) using the Hydrological Model Continuum. The simulation covers both the observed and the forecasting periods.

```
simulation_length_obs: 2 days
simulation_length_for: 2 days
simulation_domain_n: NA
simulation_type: deterministic
simulation_n: 1/day * simulation_domain
```

fp_run_nwp_probabilistic

Simulation based on the weather station observations and on the perturbed and disaggregated nwp datasets to compute the time-series datasets (e.g discharge, dams volume and level) and the spatial information (e.g. soil moisture, evapotranspiration and snow cover) using the Hydrological Model Continuum. The simulation covers both the observed and the forecasting periods.

```
simulation_length_obs: 2 days
simulation_length_for: 2 days
simulation_domain_n: NA
simulation_type: probabilistic
simulation_n: 30/day * simulation_domain
```

fp_run_test

For testing each components of the operational chain, the users have to launch procedures following the steps below:

- Download the docker_testcase folders from the github repository;
- Create and update the fp-docker_variables.env file according with the host, the container and the simulation features;
- Organize data in SOURCE folders; particularly, folders have to be organized as follows:

```
SOURCE_DATA_STATIC='/docker_testcase/data/static_data/'
SOURCE_DATA_RESTART='/docker_testcase/data/restart_data/'
SOURCE_DATA_DYNAMIC_OBS='/docker_testcase/data/dynamic_data/observation/'
SOURCE_DATA_DYNAMIC_FOR='/docker_testcase/data/dynamic_data/observation/'
SOURCE_DATA_ARCHIVE='/docker_testcase/archive/'
```

- Building the image:

```
./fp-docker_builder.sh -f fp-docker_variables.env
```

- Running the container in executable mode:

```
./fp-docker_runner.sh -f fp-docker_variables.env
```

- Collect data in the `SOURCE_DATA_ARCHIVE` folders.

1.3 RISICO FOREST FIRE RISK MODEL DOCKER

This repository contains a Docker container allowing to run the RISICO model using WRF output. The docker contains:

- src: source directory for the containers
  - RISICO2015: RISICO source code
  - adapter: input and ouput python adapters for RISICO
  - risico: static data and configuration for the RISICO model
- build.sh: build script for the container
- run.sh: run script for the RISICO model inside the container

To build the image, after you clone the repository locally, it is necessary to run the build.sh shell script.

Conversely to run RISICO model:

- Create a data folder in the host machine,
- Copy the WRF output to the data/wrf,
- Run the model in your host, replacing /path/to/data/with the absolute path of the data directory and RUNDATE in YYYYMMDDHHMM format (HH and MM should be 0000).

```
docker run -it \ -v /path/to/data/:/home/risico/data \  risico \
$RUNDATE
```

1.4 ADMS MODEL

The virtual machine deployment is currently ongoing (M14). The ADMS execution is realised through a batch script, which iterates over the input elements and executes the application for each one in parallel. LEXIS tested this application on its experimental cloud infrastructure in order to gain knowledge necessary to run this type of application as part of the WP7 workflow. LEXIS explored two test cases: urban and industrial environments. ADMS has been tested in a virtual machine running on VMware ESXi 6.7 hypervisor, having 24 vCPUs (Intel Cascade Lake), 8 GB RAM, Windows 10 Desktop.

The Industrial test case consisted only of a single input element and the simulation took less than 1 minute to finish. The Urban test case had 20 input elements, processed in parallel, and took less than 17 minutes to finish. The application itself creates only a single computing thread, therefore the opportunity for parallel execution is only in the number or granularity of the input elements. Since this parallelization is trivial, the execution mode of this application can be defined as a tradeoff between the amount of allocated vCPUs in a Cloud and the execution time. LEXIS proposes two methods for Cloud deployment of this application. Since this application is meant to be executed on Windows Desktop, remote execution opportunities such as SSH are limited. Therefore, the application can be deployed in a VM which will mount a particular storage, on which input data will be put and a service will be deployed in the VM which would periodically check the storage for new input data and run the application if necessary. This option reduces overhead caused by spawning a Windows VM, however it allocates vCPUs, which are idle for large amount of time.

The second option is to create a Windows VM image which will have a start-up batch script which will access a pre-defined storage with input data, run the simulation, upload the results and switch the VM off. This approach is better for the Cloud use-case since the orchestrator will access the OpenStack API and can simply stage the input data in the pre-defined storage and spin up a particular VM. LEXIS prepared such a start-up script and successfully tested it in the experimental VMware deployment

1.5 ERDS MODEL

The ERDS docker deployment is currently ongoing (M14). Two essential and mandatory containers are foreseen, and two additional containers are optional. The details follow in the sections below.

The first container (*container 1*) is responsible for the input data download and processing. Basically, it manages the following operations:

1. Input data download from the official repositories (GPM IMERG and GFS APCP data);
2. Conversion into numpy arrays;
3. Calculation of the accumulated values;
4. Comparison with the thresholds for every accumulation interval and every location;
5. Extraction of potential alerts;
6. Writing the output .tiff files with the accumulated precipitation and with the potential alerts.

In order to allow for performing these operations, this container must be 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 h5 files, numpy, for managing multidimensional arrays, and scipy for statistical analysis. Moreover, the software wgrib2 must be installed as an interface to access GFS data.

Certainly, this container must have access to the internet in order to be able to gather the necessary input data.

The second container (*container 2*) is responsible for the publication of the accumulated precipitation data as well as of the alerts by means of the WMS standard.

In order to do so, it must be equipped with a working instance of the GeoServer application (http://geoserver.org/). Given that this software is Java-based and is taking advantage of the GeoTools library, these two dependencies are a requirement for the *container 2* as well.

A port must be open to the public in order to allow for a proper publication through http protocol. Moreover, a shared folder with the *container 1* must be set up in order to allow the *container 2* to read the data produced by the *container 1* and publish them.

A third container (*container 3, optional*) may be generated in case of need, and contains the client application that will be served for visualizing the data on a map. This is what happens with the site http://erds.ithacaweb.org/. In such case, a web server must be installed, and nginx is the preferred solution. The client application, which is based on the Leaflet Javascript library, must be loaded on the container as well. A port open to the public must be set for allowing communication with the potential clients.

A fourth container (*container 4, optional*) may be set up in order to store ancillary geospatial data that may serve as reference data in the visualizer app (web map). The storage will be done by means of a geospatially-enabled DBMS. PostgreSQL + PostGIS is the preferred solution. An internal network communication with the *container 2* and in particular with the GeoServer application must be set up in order to allow this one to read the data from the DBMS and display them in the WMS tiles.

2 WP7 WORKFLOW MAPPING ON LEXIS PLATFORM

The WP7 workflows components will be mapped as follows on the LEXIS platform (see Figure 6):

- WRF model will be executed on HPC systems (LRZ and IT4I),
- Dockers and virtual machines will be executed on cloud computing,
- Workflows will be executed by the LEXIS Orchestration System,
- DDI will support the data exchange between the WP7 models.

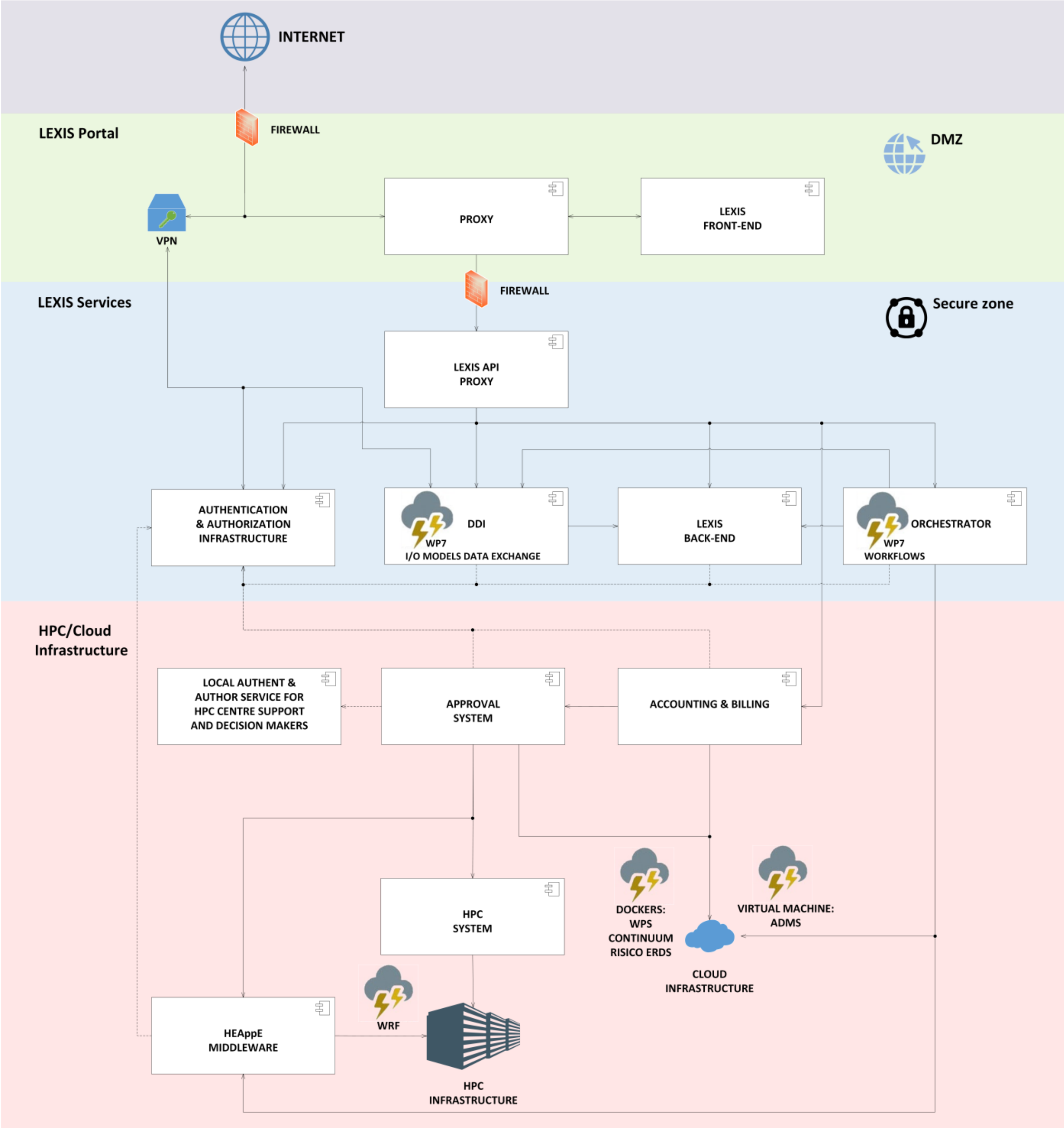

Figure 6 - WP7 workflows mapping on LEXIS platform

Workflows will be executed by the LEXIS Orchestration System described in deliverable D4.2 [7].

This Orchestration System provides an application deployment platform for Hybrid HPC/Cloud applications. The applications to be deployed are modelled using TOSCA [8] (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 will be 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 [9].

Hereafter the TOSCA implementations of RISICO and Continuum workflows are described.

2.1 RISICO WORKFLOW

The RISICO workflow is implemented at it follows:

- Creating a Virtual Machine on LEXIS cloud,
- Retrieving GFS model and Geographical data on this Virtual Machine,
- Running the WPS container,
- Submitting a WRF job on the HPC infrastructure,
- Running the RISICO container once the WRF job has produced its result files.

In practice, the graphical view of the RISICO application template (see Figure 7) shows the following components hosted on a Virtual Machine:

- Docker, to run containers,
- WFS_GFS, container performing the pre-processing (see CIMA WPS model[1]),
- RISICO, container performing the post-processing (see CIMA RISICO model[2]),
- CreateDirs, component creating directories expected by containers,
- GFSData, component downloading Global Forecast System files from a web site,
- GEOGData, component downloading geographical files from a web site,
- CopyToJob, component copying pre-processing results to a Job input directory,
- CopyFromJob, component copying Job computation results to the compute instance.

And a job: WRF, HEAppE Job performing a computation on the HPC infrastructure

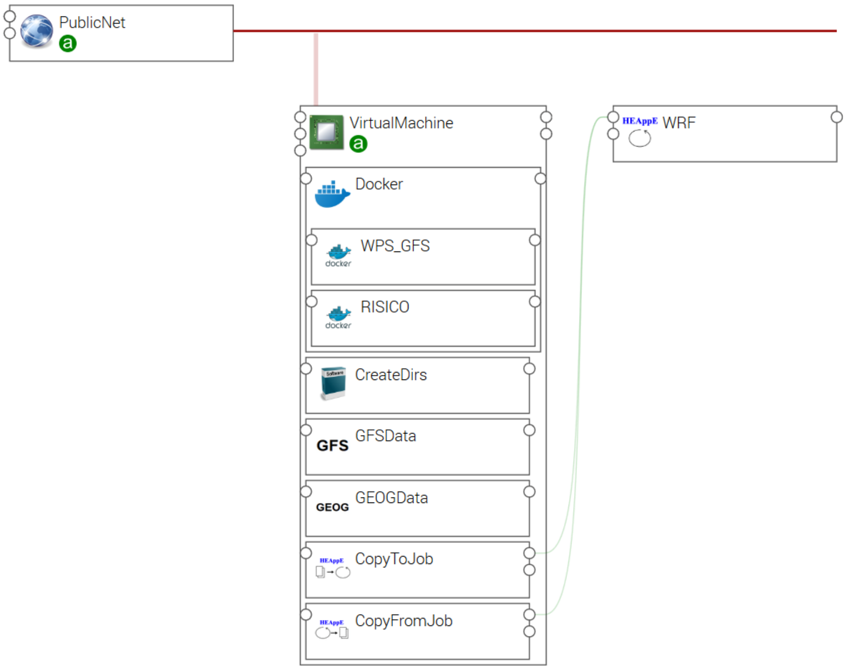

Figure 7 - Graphical view of the RISICO application template

The associated workflow (see Figure 8) creates first a Virtual Machine, then downloads in parallel GFS and geographical data files. Docker is then installed on the Virtual Machine and the WPS GFS pre-processing container is run.

[1] CIMA WPS model: https://github.com/cima-lexis/wps.docker
[2] CIMA RISICO model: https://github.com/cima-lexis/risico-docker

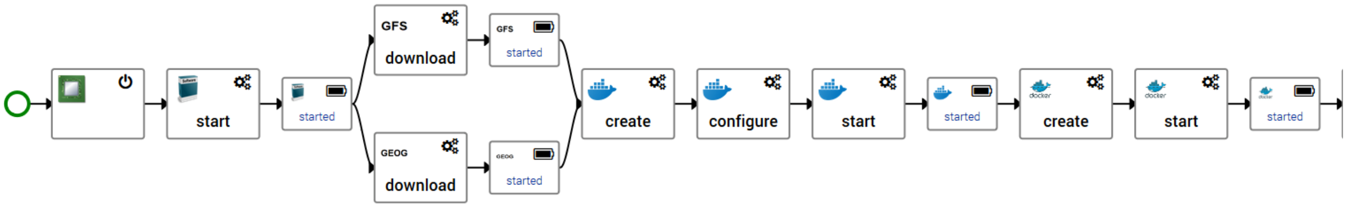

Figure 8 - RISICO workflow - part 1

Once done, the workflow goes on creating an HEAppE job on the HPC infrastructure (see Figure 9). The next step enables file transfers for this job. Then, the WPS GFS pre-processing results are copied to the job (component CopyToJob operation). The Job is then submitted, and the orchestrator waits until it ends. Once done, the Job results are copied to the Virtual Machine (component CopyFromJob operation).

Finally, in parallel:

- File transfer is disabled for the job and the job is deleted,
- RISICO post-processing container is run.

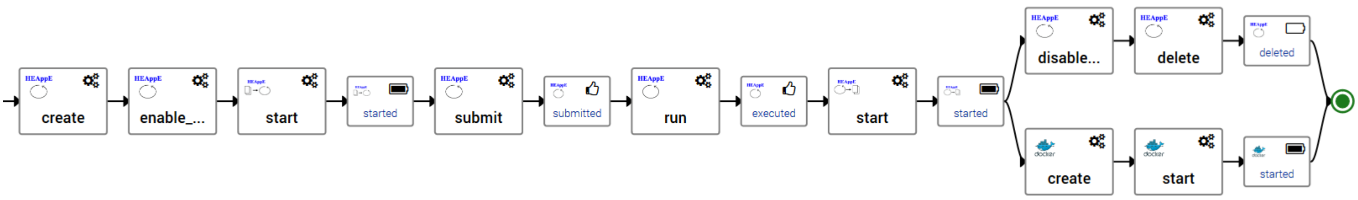

Figure 9 - RISICO workflow - part 2

2.2 CONTINUUM WORKFLOW

The Continuum workflow is implemented as follows:

- Creating a Virtual Machine on LEXIS cloud,
- Retrieving GFS model and Geographical data on this Virtual Machine,
- Running the WPS container,
- Submitting a WRF job on the HPC infrastructure,
- Running the Continuum container once the WRF job has produced its result files.

In practice, the graphical view of the Continuum (see Figure 10) application template shows the following components hosted on a Virtual Machine:

- Docker, to run containers,
- WFS_GFS, container performing the pre-processing (CIMA WPS model),
- Continuum, container performing the post-processing (see CIMA FloodProofs[3]),
- CreatePreProcessDirs, component creating directories expected by the pre-processing container,
- CreatePostProcessDirs, component creating directories expected by post-processing container,
- GFSData, component downloading Global Forecast System files from a web site,
- GEOGData, component downloading geographical files from a web site,
- CopyToJob, component copying pre-processing results to a Job input directory,
- CopyFromJob, component copying Job computation results to the compute instance,
- ContinuumTestData, component downloading tests data used by the post-processing container.

And a job: WRF, HEAppE Job performing a computation on the HPC infrastructure.

[3] FloodProofs: https://github.com/cima-lexis/

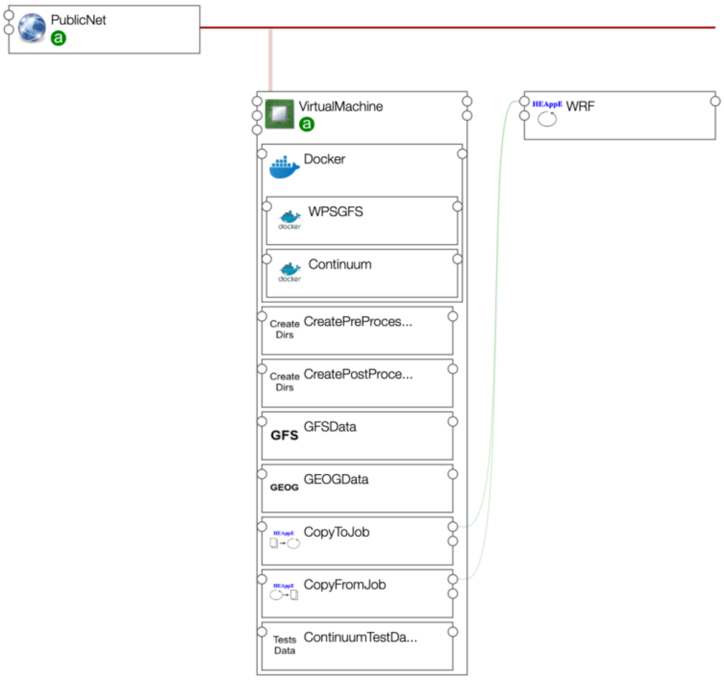

Figure 10 - Graphical view of the Continuum application template

A workflow named *RunContinuumWithTestData* (see Figure 11) provides a simplified version of the final Run workflow already implemented here, but that will require a new Continuum post-processing container able to manage HEAppE job WRF results.

This workflow *RunContinuumWithTestData* creates first a Virtual Machine. The next step creates directories that will be mounted in the post-processing container. The next step installs and start Docker on this Virtual Machine. The next step download Test Data that will be used as input data by the post-processing container. The next step creates and start the post-processing docker container.

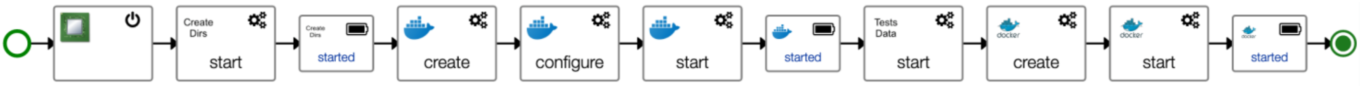

Figure 11 - Continuum workflow *RunContinuumWithTestData*

The *Run* workflow runs the full workflow, from pre-processing to post-processing. It creates first a Virtual Machine (see Figure 12). Then downloads in parallel GFS and geographical data files. Docker is then installed on the Virtual Machine and the WPS GFS pre-processing container is run.

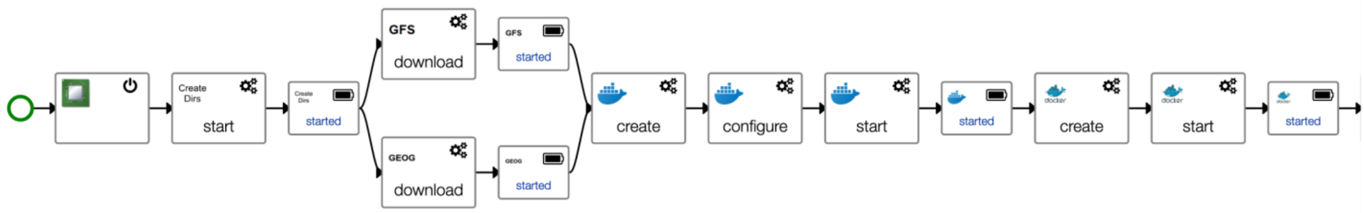

Figure 12 - Continuum workflow *Run* - part 1

Once done, the workflow goes on creating a HEAppE job on the HPC infrastructure (see Figure 13). The next step enables file transfers for this job. Then, WPS GFS pre-processing results are copied to the job (component CopyToJob operation). The Job is then submitted, and the orchestrator waits until it ends. Once done, Job results are copied to the Virtual Machine (component CopyFromJob operation). Finally, in parallel:

- File transfer is disabled for the job and the job is deleted,
- Test data are fetched and the Continuum post-processing container is run.

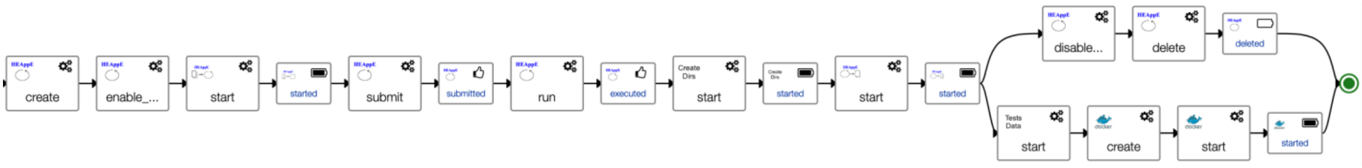

Figure 13 - Continuum workflow *Run* - part 2

3 CONCLUSION

We presented the deployment of the components of the WP7 workflows, based on docker and virtual machines. The present version of the workflows does not include the data assimilation of the meteorological data into the WRF model.

The next steps will be mainly focused on high availability/urgent computing scenarios, namely:

- WPS docker executed both on LRZ and IT4I (high-availability),
- WRF submitted on both sites (LRZ and IT4I) and executed on the first source providing the requested number of nodes,
- Indexing and transfer of data using WCDA as well as NetCDF/GRIB conversion on burst buffer nodes,
- ADMS execution on IT4I.

REFERENCES

[1] F. Silvestro, S. Gabellani, F. Delogu, R. Rudari and G. Boni, "Exploiting remote sensing land surface temperature in distributed hydrological modelling: the example of the Continuum model," *Hydrology and Earth System Sciences,* vol. 1, no. 17, p. 39, 2013.

[2] P. Fiorucci, F. Gaetani and R. Minciardi, "Development and application of a system for dynamic wildfire risk assessment in Italy," *Environmental Modelling & Software,* vol. 6, no. 23, pp. 690-702, 2008.

[3] C. A. McHugh, D. J. Carruthers and H. A. Edmunds, "ADMS–Urban: an air quality management system for traffic, domestic and industrial pollution," *International Journal of Environment and Pollution,* Vols. 3-6, no. 8, pp. 666-674, 1997.

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

[5] R. A. Meyer and L. H. Seawright, "A virtual machine time-sharing system," *IBM Systems Journal,* vol. 3, no. 9, pp. 199-218, 1970.

[6] C. Boettiger, "An introduction to Docker for reproducible research," *ACM SIGOPS Operating Systems Review,* vol. 1, no. 49, pp. 71-79, 2015.

[7] LEXIS Deliverable, *D4.2 Design and Implementation of the HPC-Federated Orchestration System - Intermediate.*

[8] "OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA)," [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca. [Accessed 15 3 2020].

[9] "LEXIS project application templates repository," [Online]. Available: https://github.com/lexis-project/application-templates. [Accessed 10 3 2020].