The WRF4G Python Framework for regional climate simulations with WRF model

Antonio. S. Cofiño*
antonio.cofino@unican.es
@cofinoa

Santander Meteorology Group
Department of Applied Mathematics and Computational Sciences
Universidad de Cantabria, Santander, Spain

*Developers: C. Blanco, E. Cimadevilla, J. Fernández, V. Fernández-Quiruelas, L. Fita, M. García-Díez, A. Minondo

Aknowledgments: This work is partially funded by the Spanish R+D program through MINECO project INSIGNIA (CGL2016-79210-R) and ERDF
WRF4G is a software framework developed in Python (75%) by the Santander Meteorology Group, which provides:

- Flexible WRF experiment management, execution and monitoring, and ...
- ... run these experiments on hybrid distributed computing infrastructures (HDCI) concurrently in a coherent way.

Crédit : Laurent Fairhead (LMD/CNRS)
Reanalysis/Reforecasts/Hindcast

- High number (~$10^4$) of independent simulations
- High volume of output-data (>TB)
- Requires scalability

Regional climate simulation

- Long, continuous simulations; weeks of walltime
- High volume of output data (>TB)
- Recovering system for simulation restart

Weather Forecasting

- QoS and optimal resources: deadline for delivery

Sensitivity/ensemble studies

- Physical schemes, initial conditions and boundary conditions: uncertainty sampling
- Resource demanding experiments composed of many independent simulations
Reanalysis/Reforecasts/Hindcast

- SEAWIND project
- 21 years of daily reforecasts (36h each)
- 7,665 independent simulations

Regional climate simulation

- ESCENA & CORDEX projects: EUR, AFR and SAM
- 50 years (continuous run, 28-day restarts)
- 650 dependent simulations

Sensitivity/ensemble studies

- CORWES project
- Physics sensitivity study for CORDEX-Africa
- 8-member ensemble of 5-year continuous simulations
- 8 independent groups of 65 dependent simulations
SEAWIND experiment set: past reforecasts of an improved wind field over Europe for off-shore wind farms.

- Characteristics of each experiment
  - 21 years of daily reforecasts (36h each)
  - 7,665 independent simulations
- Computation cost of each experiment
  - Working Node Architecture
    - CPU: Dual 8 Cores CPU
    - RAM Memory: 16 GB
  - Result
    - WALLTIME (MPI job) = 21 x 365 x 70' ~ 2 years
    - Output = 21 x 365 x 17 GB ~ 130 TB
Desktop/Laptop (UI)
- Limited computational power and storage
- **User interface** to other computer resources

Workstation
- Multi-core, shared memory, moderate storage
- Typically **ssh access**

Local group/institutional cluster
- Multi-node, distributed memory, large storage
- ssh access, **batch system** (PBS, SGE, ...) to submit jobs

Mainframe/HPC site
- **Different architectures** and memory configurations
- ssh or kerberos and token based security

Cloud/Grid infrastructures
- Geographically and temporal distributed and on-demand scalability.
- **Huge amount of potential computational power** and storage, which is not trivial to take advantage of it for weather & climate applications
- Distributed and federated authentication and authorization infrastructures
A multidisciplinary approach to weather & climate

Meteorology Group
A multidisciplinary approach for weather & climate

http://www.meteo.unican.es

Statement of the problem
Statement of the problem

CARLOS
Blizzard

30 JOBS

PC

SSH

10 JOBS
Workstation Oceano
UC

10 JOBS
Supercomputer Altamira
RES Infrastructure

10 JOBS
Cloud/Grid
EGI (VO esr)

marcream01.in2p3.fr
ce130.cern.ch
gridgate.cs.tcd.ie
Statement of the problem

A multidisciplinary approach to weather & climate

Meteorology Group
A multidisciplinary approach for weather & climate

http://www.meteo.unican.es

PC
Blizzard

CARLOS

20,000 JOBS

Workstation Oceano
UC

Supercomputer Altamira
RES Infrastructure

.marcream01.in2p3.fr
ce130.cern.ch
gridgate.cs.tcd.ie

? JOBS

Cloud/Grid
EGI (VO esr)

RES Infrastructure

? JOBS
Statement of the problem

- Workstations Oceano, Sea, ...
- Supercomputer Altamira, MN3, ...
- RES Infrastructure

20,000 JOBS

- CARLOS
- Blizzard
- PC

- marcream01.in2p3.fr
- ce130.cern.ch
- gridgate.cs.tcd.ie

- EGI (VO esr), GISELA (VO prod.vo.eu-eela)
Requirements

- Uniform access to available resources
- Simple interface
- Robust and scalable
- Keep track of jobs
- Easy configuration
- Easy deployment and *batteries included*
WRF4G 2.0 Key features

- Written in **python**. Some bash scripts
- **Meta-scheduler** (DRM4G & GridWay)
- Advanced **CLI** with subcommands and **python API**:
  
  ```
  $ wrf4g exp test submit
  ```

- Dynamic management of **Hybrid-DCIs** (Grid, HPC,...)
- **Scalable** (~100,000 jobs)
- **Ready-to-run** (Linux). **WRF binaries** included. Some tools like nco, cdo OpenMPI stack.
- Simplification of **configuration files**. Easy experiment definition
- ORM abstraction layer for experiment’s metadata management
- **Workflow execution** (other climate models such as CAM)
- **Identity management**  (private/public keys and X509 tokens)
WRF4G splits a regular WRF simulation experiment into:

- **realizations**
  - A realization is an independent WRF simulation

- **chunks**
  - For convenience, a WRF realization are split into chunks. By definition, a chunk is a dependent partial simulation and requires the previous chunk to start.
  - Chunks depend on computing resource limitations: WALLTIME, RESOURCE DISK QUOTA, ... 
  - Chunks allow to customize the size of the input files (boundary and initial conditions).
WRF4G Benefits

WRF4G helps to manage, execute and monitor of complex experiments using WRF simulations in HDCI providing a coherent access to computing and storage resources.

- It allows efficient use of HDCI resources without increasing complexity to the researchers.
- Its ability to add resources on-demand makes it ideal for solving work peaks or SLA with Cloud resources.
- The meta-scheduling training and proper management of replicas are key to optimize the use of resources.


Large-scale climate simulations harnessing clusters, grid and cloud infrastructures

V. Fernández-Quiruelas*, C. Blanco, A.S. Cofiño, J. Fernández

Grupo de Meteorología, Dpto. Matemática Aplicada y C. Computación, Universidad de Cantabria, Santander, Spain
WRF4G helps to manage, execute and monitor of complex experiments using WRF simulations in HDCI providing a coherent access to computing and storage resources.

- It allows efficient use of HDCI resources without increasing complexity to the researchers.
- Its ability to add resources on-demand makes it ideal for solving work peaks or SLA with Cloud resources.
- The meta-scheduling training and proper management of replicas are key to optimize the use of resources.

---

**Diagram:**

- **Experiment 3.1**: Ended at 13 hours 37 minutes.
- **Experiment 3.2**: Ended at 33 hours 10 minutes.
WRF4G Benefits

WRF4G helps to manage, execute and monitor of complex experiments using WRF simulations in HDCI providing a coherent access to computing and storage resources.

- It allows efficient use of HDCI resources without increasing complexity to the researchers.
- Its ability to add resources on-demand makes it ideal for solving work peaks or SLA with Cloud resources
- The meta-scheduling training and proper management of replicas are key to optimize the use of resources
Meta-scheduler: DRM4G

- **CLI** offers users a command to submit, cancel, and monitor WRF simulations and configure resources.
- **GridWay core** is in charge of job execution and resource brokering.
- **Sched** is responsible for scheduling jobs.
- Middleware Access Driver (**MAD**)
  - Infrastructure Managers
    - ROCCI, BOTO, APACHE CLOUD, ...
  - Resource Managers
    - FORK, SGE, PBS SLURM, CREAM, GLOBUS, ...
  - Communicators Managers
    - LOCAL, SSH, GSISSH, OPENID, ...
Scalability experiment on a laptop (Intel Core i5 at 1.9 Ghz and 8GB of RAM)

10K job experiment:

<table>
<thead>
<tr>
<th>Meta-scheduler</th>
<th>Job assimilation</th>
<th>CPU</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GridWay</td>
<td>4 minutes (blocking system)</td>
<td>~13%</td>
<td>~20% (400 MB)</td>
</tr>
<tr>
<td>DRM4G</td>
<td>75 seconds</td>
<td>~5%</td>
<td>~5% (100 MB)</td>
</tr>
</tbody>
</table>

100K job experiment:

Memory and CPU usage by DRM4G.
$ wget https://meteo.unican.es/work/WRF4G/install.sh
$ bash ./install.sh

WRF4G installation script

--> Downloading wrf4g_x86_64_versions from ...

This script will install WRF4G version: 2.2.1

--> Downloading wrf4g-2.2.1-x86_64.tar.gz from ...
--> Unpacking wrf4g-2.2.1-x86_64.tar.gz in directory /home/user ...

Installation of WRF4G 2.2.1 is done!

In order to work with WRF4G you have to enable its environment with the command:

   . /home/user/wrf4g/bin/wrf4g_init.sh

You need to run the above command on every new shell you open before using WRF4G, but just once per session.
$ wget https://meteo.unican.es/work/WRF4G/install.sh
$ bash ./install.sh

---

Latest version: https://meteo.unican.es/trac/wiki/WRF4G2.0

This script will install WRF4G version: 2.2.1

--> Downloading wrf4g-2.2.1-x86_64.tar.gz from ...
--> Unpacking wrf4g-2.2.1-x86_64.tar.gz in directory /home/user ...

===================================
Installation of WRF4G 2.2.1 is done!
===================================

In order to work with WRF4G you have to enable its environment with the command:

    . /home/user/wrf4g/bin/wrf4g_init.sh

You need to run the above command on every new shell you open before using WRF4G, but just once per session.
$ wget https://meteo.unican.es/work/WRF4G/install.sh
$ bash ./install.sh

WRF4G installation script

--> Downloading wrf4g_x86_64_versions from ...

This script will install WRF4G version: 2.2.1

--> Downloading wrf4g-2.2.1-x86_64.tar.gz from ...
--> Unpacking wrf4g-2.2.1-x86_64.tar.gz in directory /home/user ...

Installation of WRF4G 2.2.1 is done!

In order to work with WRF4G you have to enable its environment with the command:

   . /home/user/wrf4g/bin/wrf4g_init.sh

You need to run the above command on every new shell you open before using WRF4G, but just once per session.
$ wget https://meteo.unican.es/work/WRF4G/install.sh
$ bash ./install.sh

WRF4G installation script

--> Downloading wrf4g_x86_64_versions from ...

This script will install WRF4G version: 2.2.1

--> Downloading wrf4g-2.2.1-x86_64.tar.gz from ...
--> Unpacking wrf4g-2.2.1-x86_64.tar.gz in directory /home/user ...

Installation of WRF4G 2.2.1 is done!

In order to work with WRF4G you have to enable its environment with the command:

```
. /home/user/wrf4g/bin/wrf4g_init.sh
```

You need to run the above command on every new shell you open before using WRF4G, but just once per session.
$ source ./wrf4g/bin/wrf4g_init.sh

$ wrf4g start
Starting DRM4G ...
  OK
Starting WRF4G_DB (MySQL) ...
  OK

$ wrf4g status
DRM4G is running
WRF4G_DB is running

$ wrf4g resource edit
Resource Configuration I

```
[altamira]
communicator = ssh
username = uc15003
frontend = altamiral1.ifca.es
private_key = ~/.ssh/id_rsa
lrms = slurm_res
max_jobs_running = 40
max_jobs_in_queue = 50

[marenostrum]
communicator = ssh
username = ecm86048
frontend = mn1.bsc.es
private_key = ~/.ssh/id_dsa
lrms = lsf
max_jobs_running = 50
max_jobs_in_queue = 60
```
## resources.conf

```
[egi_esr]
communicator = ssh
username = carlos
frontend = ui.macc.unican.es
private_key = ~/.ssh/id_rsa
grid_cert = ~/cert.p12
lrms = cream
vo = esr
```
resources.conf

[CESNET_Metacloud]
enable = true
communicator = ssh
vm_communicator= op_ssh
private_key = ~/.ssh/id_rsa
username = user
vm_user = drm4g_admin
frontend = ui.meteo.unican.es
lrms = rocci
cloud_provider = EGI FedCloud - CESNET-METACLOUD
myproxy_server = myproxy1.egee.cesnet.cz
flavour = Medium
virtual_image = Ubuntu-14.04
instances = 1
volume = 10
max_jobs_running = 5
$ wrf4g identity altamira conf

--> Configuring private and public keys ...
   Enter passphrase for key '/uc15/uc15003/.ssh/id_rsa':
   Identity added: /uc15/uc15003/.ssh/id_rsa (/uc15/uc15003/.ssh/id_rsa)
   Adding 'id_rsa.pub' to 'authorized_keys' on 'altamir1.ifca.es'
   Lifetime set to 604800 seconds
$ wrf4g identity altamira conf

--> Configuring private and public keys ...
Enter passphrase for key '/uc15/uc15003/.ssh/id_rsa':
Identity added: /uc15/uc15003/.ssh/id_rsa (/uc15/uc15003/.ssh/id_rsa)
Adding 'id_rsa.pub' to 'authorized_keys' on 'altamira1.ifca.es'
Lifetime set to 604800 seconds

$ wrf4g exp test start --template-exp=single

$ ls test
  experiment.wrf4g wrf4g_files

$ wrf4g exp test edit
experiment.wrf4g

[DEFAULT]
# Experiment configuration
experiment_name  = test
# Simulation domain
max_dom          = 1
# Experiment time-specification
start_date       = 2011-08-28_12:00:00
end_date         = 2011-08-30_00:00:00
calendar         = standard
chunk_size_h     = 12
# Namelist
namelist_version = 3.3.1
# Running options
np               = 1
requirements     = ARCH = "x86_64"
[resource:altamira]
AL_HOME = /uc15/uc15003 #auxiliar variable

# Input data
domain_path = %(AL_HOME)s/domains/Santander_50km
extdata_vtable = GFS
extdata_path = %(AL_HOME)s/input/NCEP/GFS
extdata_interval = 21600
extdata_preprocessor = default

# Output
output_path = %(AL_HOME)s/output
postprocessor = SFC

# apps
app_bundles = netcdf | %(AL_HOME)s/netcdf/netcdf-4.1.1.tar.gz
nco | %(AL_HOME)s/nco/nco-4.0.9.tar.gz
cdo | %(AL_HOME)s/cdo/cdo-1.3.0.tar.gz
wrf | %(AL_HOME)s/WRF/WRFbin-3.3.1.tar.gz
mpi | %(AL_HOME)s/openmpi/openmpi-1.4.tar.gz
$ wrf4g exp test create
Preparing namelist...
   ---> Single params run
   ---> Continuous run
      ---> cycle_chunks: test 2011-08-28_12:00:00 2011-08-30_00:00:00
      ---> chunks 1: test 2011-08-28_12:00:00 2011-08-29_00:00:00
      ---> chunks 2: test 2011-08-29_00:00:00 2011-08-29_12:00:00
      ---> chunks 3: test 2011-08-29_12:00:00 2011-08-30_00:00:00

$ wrf4g exp test submit
   ---> Submitting realization: test
      ---> Submitting chunk 1: 2011-08-28_12:00:00 2011-08-29_00:00:00
      ---> Submitting chunk 2: 2011-08-29_00:00:00 2011-08-29_12:00:00
      ---> Submitting chunk 3: 2011-08-29_12:00:00 2011-08-30_00:00:00
Monitoring

```
$ wrf4g exp test status
Realization Stat Chunks Host        Run.Sta       JID ext %
test        W  1/3      altamira Waiting  0 -  0.00

$ wrf4g exp test status
Realization Stat Chunks Host        Run.Sta       JID ext %
test        R  1/3      altamira Running  0 -  0.00

$ wrf4g exp test status
Realization Stat Chunks Host        Run.Sta       JID ext %
test        S  3/3      altamira Submitted 1 - 33.33

$ wrf4g exp test status
Realization Stat Chunks Host        Run.Sta       JID ext %
test        R  3/3      altamira Running  2 - 66.67

$ wrf4g exp test status
Realization Stat Chunks Host        Run.Sta       JID ext %
test        D  3/3      altamira Finished 2 0 100.00
```
Projects

INSIGNIA: Contribution to CORDEX Flagship Pilot Studies: regional climate downscaling and data publishing

CORDEX4CDS: Facilitate access to and manipulation of output of regional climate projections over Europe and boundary conditions from GCM simulations needed for future regional projections.

WRF4G: Adaptation of WRF Model to Grid Infrastructures and user-case for wind hindcast over Europe

The SCI-BUS project aims to ease the life of e-Scientists by creating a new science gateway customisation methodology based on the generic-purpose gUSE/WS-PGRADE portal family (WRF4SG)

Coordinated regional climate downscaling experiment using WRF: a contribution to the CORDEX initiative by the Spanish WRF community

CORDEX - Coordinated Regional Climate Downscaling Experiment: a WCRP-sponsored program to produce regional climate change scenarios globally


• The **WRF user community** usually:
  
  • designs **experiments** where many or **huge (!!)** simulations are required
  
  • has **access** to **hybrid distributed computer infrastructures** for running simulations

• **WRF4G** focus on **simplify** the design, execution and monitoring of WRF on several computing resources as an coherent meta-computer.

• **WRF4G** is available under **EUPL-1.1**
Thank you!

Contact: antonio.cofino@unican.es
@cofinoa

More info: “wrf4g” → I'm Feeling Lucky

Wiki: https://meteo.unican.es/trac/wiki/WRF4G2.0

Code: https://github.com/SantanderMetGroup/WRF4G

Job position: 2 years contract for a Python engineer in ESS software frameworks.
Let me know if you’re interested!!