ESGF: Compute Working Team
Python End User API

ESGF F2F Workshop, Washington, DC, December 2016

Charles Doutriaux, Jason Boutte, ESGF Compute Working Team

December 7th, 2016
Why?

- Server API is done and language agnostic

- What does this mean for the end-user?
  - Long URLs: http://aims2.llnl.gov/wps?
    service=WPS&version=1.0.0&request=Execute&datainputs=[domain=[];variable=[{"id":"tas|v0","uri":"http://esg.cnrm-game-meteo.fr/thredds/dodsC/CMIP5_CNRM/output1/CNRM-CERFACS/CNRM-CM5/historical/mon/atmos/Amon/r1i1p1/v20110901/tas_Amon_CNRM-CM5_historical_r1i1p1_185001-189912.nc"},{"id":"tas|v1","uri":"http://esg.cnrm-game-meteo.fr/thredds/dodsC/CMIP5_CNRM/output1/CNRM-CERFACS/CNRM-CM5/historical/mon/atmos/Amon/r2i1p1/v20110901/tas_Amon_CNRM-CM5_historical_r2i1p1_185001-189912.nc"},{"id":"tas|v2","uri":"https://esgf.nccs.nasa.gov/thredds/dodsC/CMIP5/NASA/GISS/historical/E2-H_historical_r1i1p1/tas_Amon_GISS-E2-H_historical_r1i1p1_185001-190012.nc"},{"id":"tas|v3","uri":"https://esgf.nccs.nasa.gov/thredds/dodsC/CMIP5/NASA/GISS/historical/E2-H_historical_r2i1p1/tas_Amon_GISS-E2-H_historical_r2i1p1_185001-190012.nc"}];operation=[{"input":["v0","v1"],"name":"ensemble.cdat","result":"cnrm"},{"input":["v2","v3"],"name":"ensemble.cdat","result":"giss"},{"input":["cnrm","giss"],"name":"multi.mv"}];]
Solutions?

- Do nothing: esgf users are smart! They’ll figure it out

- Develop a UI:
  - Time consuming
  - Not necessarily falling into the skills set of a “compute” team
  - User will reject it (uggly, not enough functionalities, etc...)

- Develop an API to unlock the potential for all.
We went for the API

- We picked Python
  - Widely adopted
  - Lots of tools for web things already
  - ESGF eco-system is filled with it

- Mainly we had to implement our Web API specific keywords
## Implemented

- **WPS:** Provides details of a WPS server (Identification & Provider), access to all processes supported by a server.

- **Dimension:** Provides information about a spatial or temporal dimension; the start, stop and stepping.

- **Domain:** Provides information for sub-setting input data along spatial and temporal axes.

- **Variable:** Provides information about an input file; URI, variable name, mask, domain

- **Gridder:** Provides details used to re-grid a Variable

- **Mask:** Provides a way to define the areas of interest by applying comparisons between Variable data and/or pre-defined functions (sin, cos, etc)

- **(NamedParameter:** Provides the ability to pass additional parameters to a process (API can auto generate these))

- **Operation:** Provides access to a WPS process, allows execution on a remote server and access to the processes output
# First we need to import a few things
from esgf import WPS
from esgf import Variable
# Now let's connect to our server
wps = WPS('http://0.0.0.0:8000/wps/', username="wps_test", password="Abc123!!")
# let's use a variable in the file we are serving locally via the server
T = Variable('/ServedData/famipc5_ne30_v0.3_00003.cam.h0.1979-01_regrid.nc', 'T')
# Now let's select the process we want to use
process = wps.get_process('cdat.averager')
# Let's ask the server to average T over lat/lon
process.execute(inputs=[T], axes='longitude|latitude')
# Ok we could use this further in wps or we can use it locally
# let's load the file locally
import cdms2
f=cdms2.open(process.output.uri)
T_avg = f[process.output.var_name]
print T_avg.shape
Future

- Keep Compatibility with ESGF Compute API changes
- Support workflow creation
- Add server discovery tools
  - Federated server selection
  - Server capability querying
  - Server load monitoring
  - Server holdings
Further Reading/Getting Involved

- Code: [http://github.com/ESGF/esgf-compute-api](http://github.com/ESGF/esgf-compute-api)
- Anaconda: [https://anaconda.org/uvcdat/esgf-compute-api](https://anaconda.org/uvcdat/esgf-compute-api)
- Demo tomorrow: [http://github.com/doutriaux1/jupyter-notebook/cwt](http://github.com/doutriaux1/jupyter-notebook/cwt)
- Email: [esgf-cwt@llnl.gov](mailto:esgf-cwt@llnl.gov)