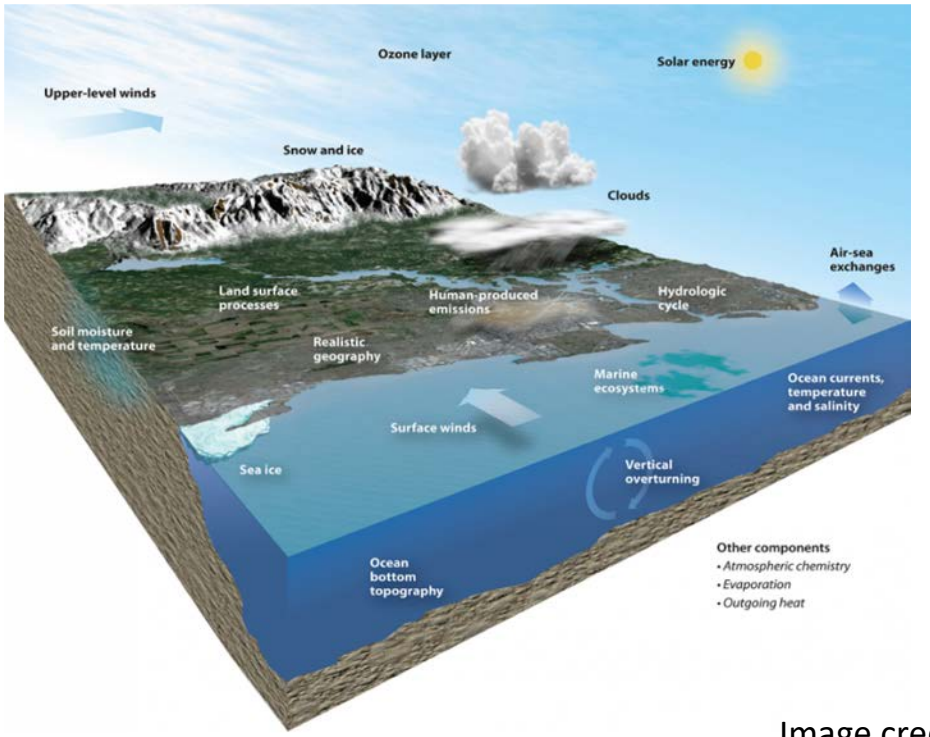
A new workflow for CESM™ to address CMIP6 challenges

Sheri Mickelson
NCAR

3rd ENES Workshop on Workflows
13 September 2018

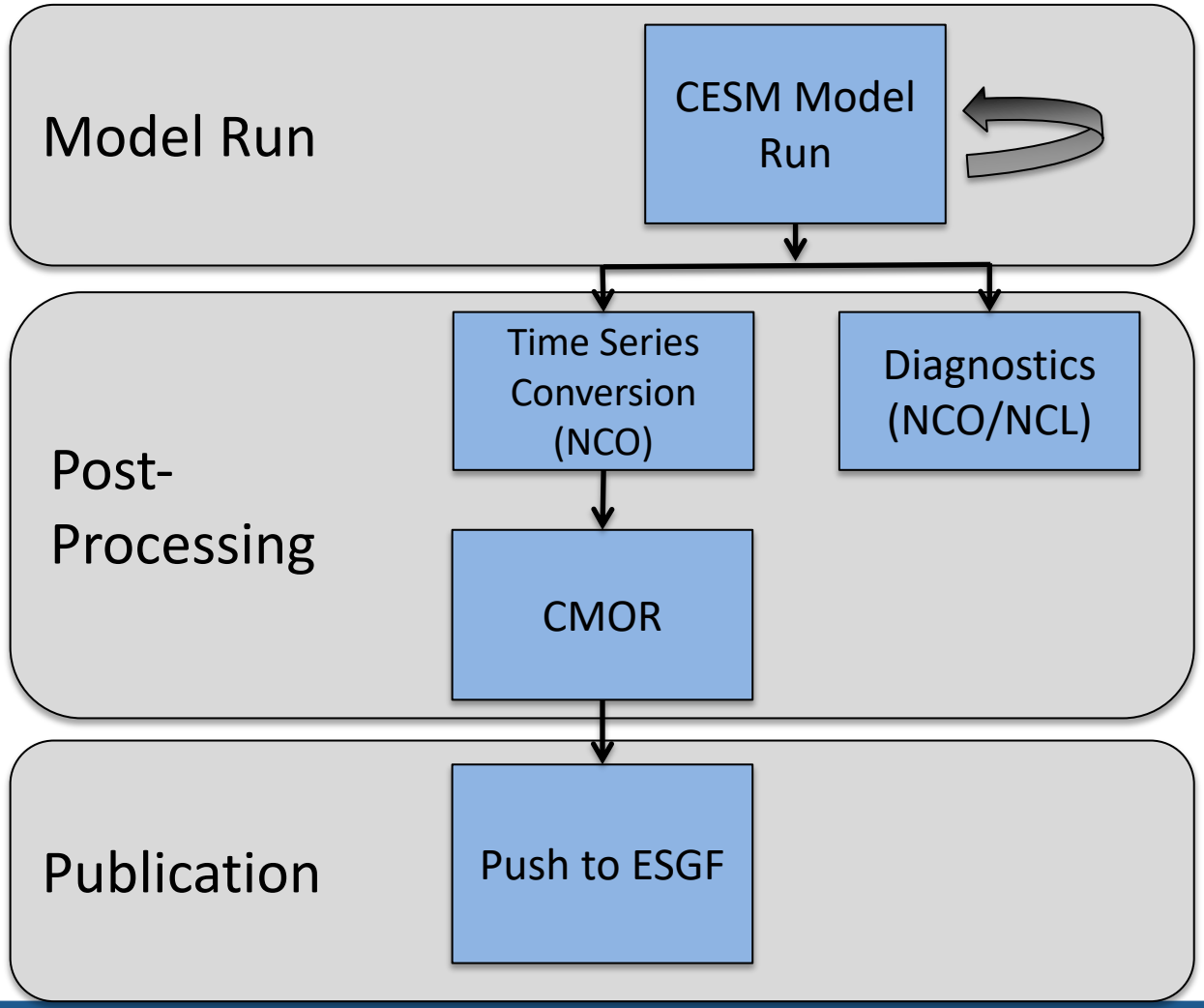
The Community Earth System Model (CESM)



- CESM is a fully-coupled climate model
- CESM is sponsored by the National Science Foundation and the U.S. Department of Energy, with contributions from the University community

Image credit: https://www2.cisl.ucar.edu/software/community-models

CESM's CMIP5 Workflow



Lessons We Learned From CMIP5

- CESM was the first model to complete their simulations, but the last to complete publication. Why?
  - All of the post-processing was serial and it took a long time to run
  - Workflow was error prone and was time consuming to debug
  - Too much human intervention was needed between post-processing steps and time was wasted
  - There was only one person who knew the status of all of the experiments

NCAR's CMIP6 Plans

(DECK and Tier I Experiments)

- Currently participating in about 23 MIPS
  - Just over 100 different experiments total
- Over all experiments, we will simulate roughly 23,287 years of climate
- The total cost will be roughly 230M core hours

Background image: Eyring, Veronika & Bony, Sandrine & Meehl, Gerald & A. Senior, Catherine & Stevens, Bjorn & Ronald, Stouffer & E. Taylor, Karl. (2016). Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization. Geoscientific Model Development. 9. 1937-1958. 10.5194/gmd-9-1937-2016.

Complexity Comparison

CMIP5

- 25 Experiments
- Timeline: 3 years
- Output size: 800TB
- Published size: 200TB

CMIP6

- 102 Experiments
- Timeline: 1 year
- Output size: 8PB (estimate)
- Published size: 2PB (estimate)

Complexity Comparison

CMIP5

- 25 Experiments
- Timeline: 3 years
- Output size: 800TB
- Published size: 200TB

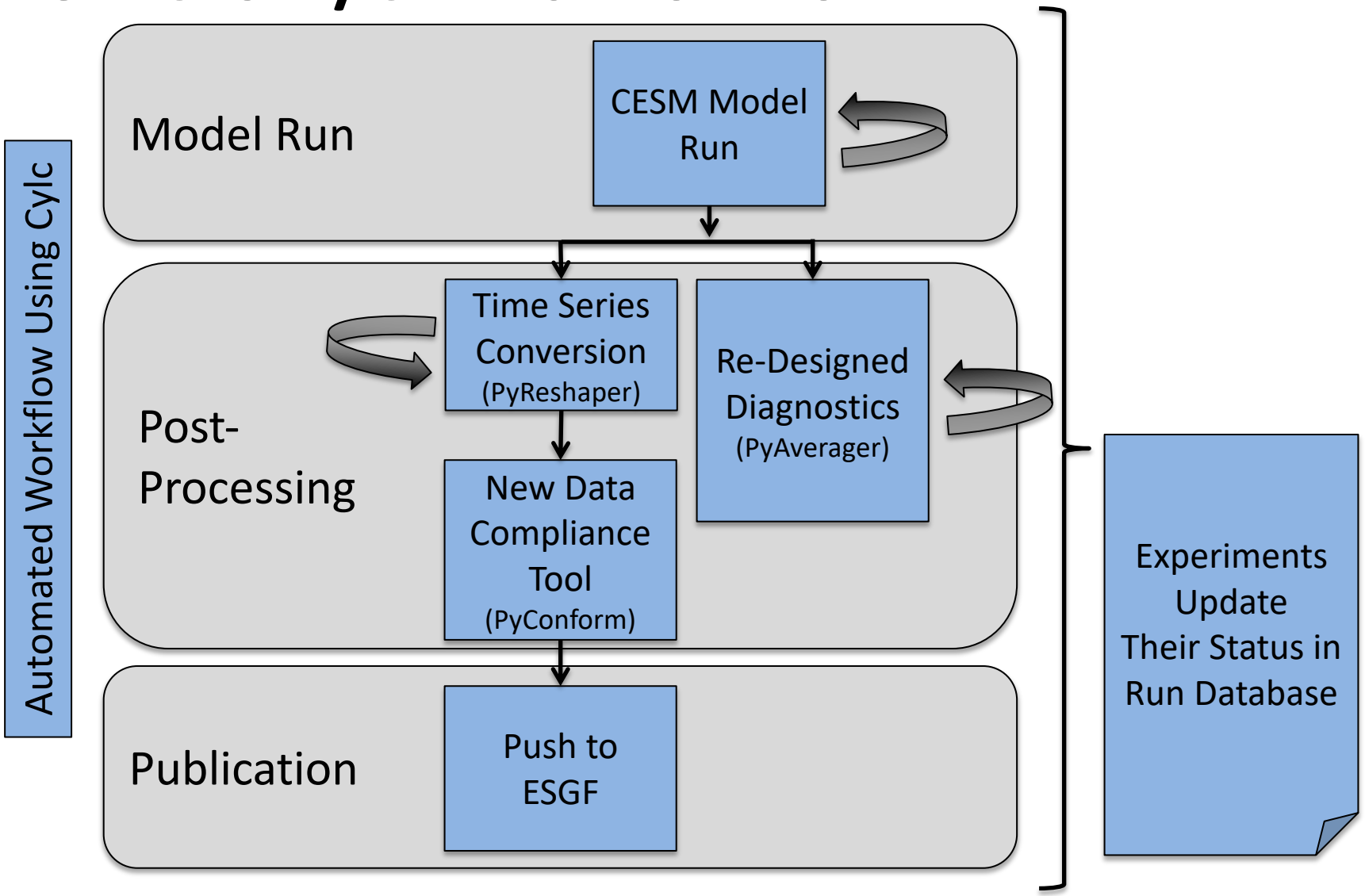
CMIP6

- 102 Experiments
- Timeline: 1 year
- Output size: 8PB (estimate)
- Published size: 2PB (estimate)

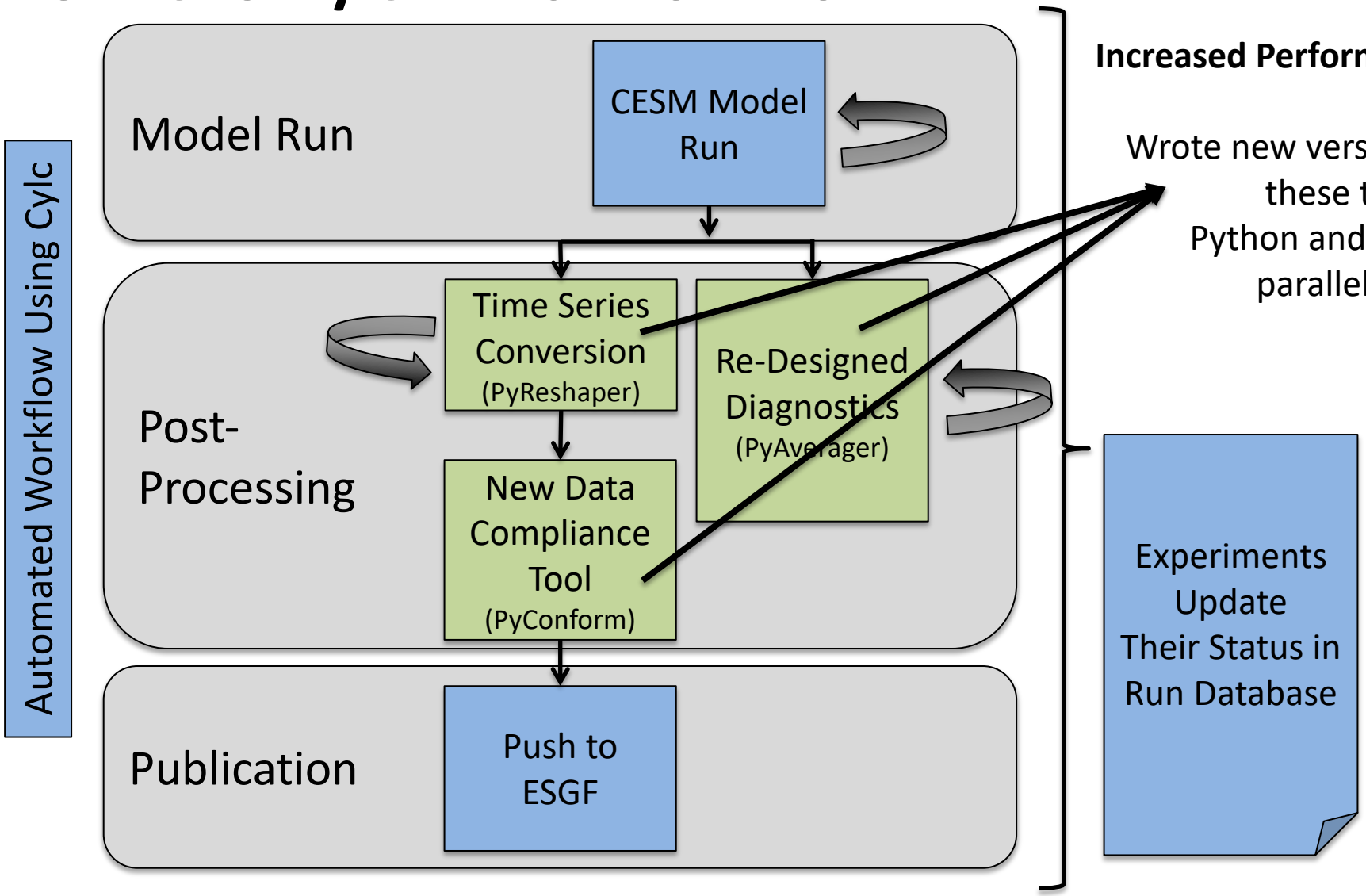
We needed better methods!

http://www.bbc.com/earth/story/20170510-terrifying-20m-tall-rogue-waves-are-actually-real

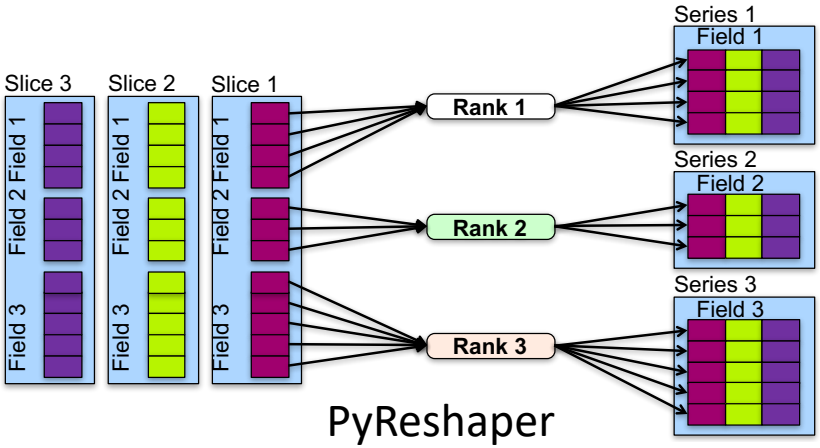
New CESM/CMIP6 Workflow



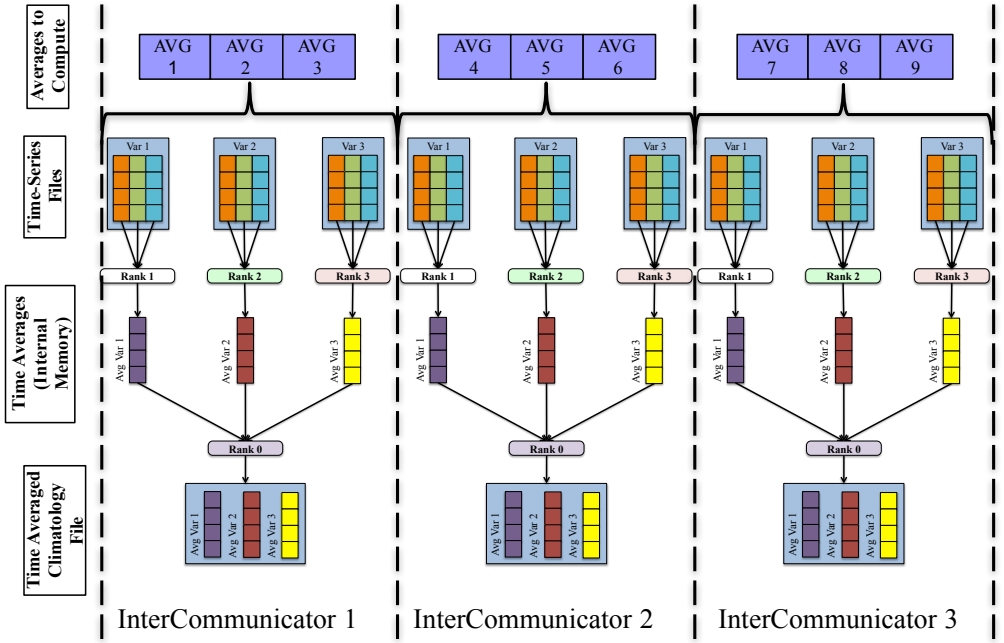
New CESM/CMIP6 Workflow



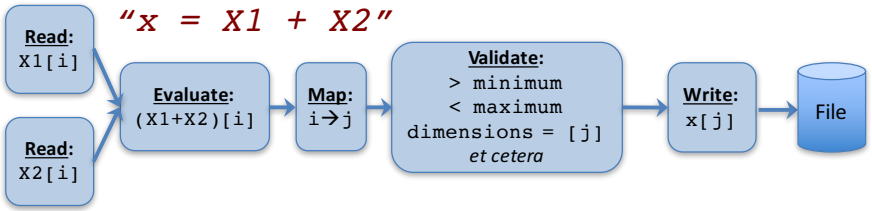
Parallelization Methods

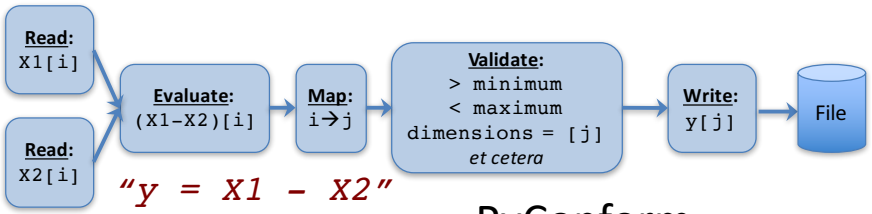


PyReshaper



PyAverager

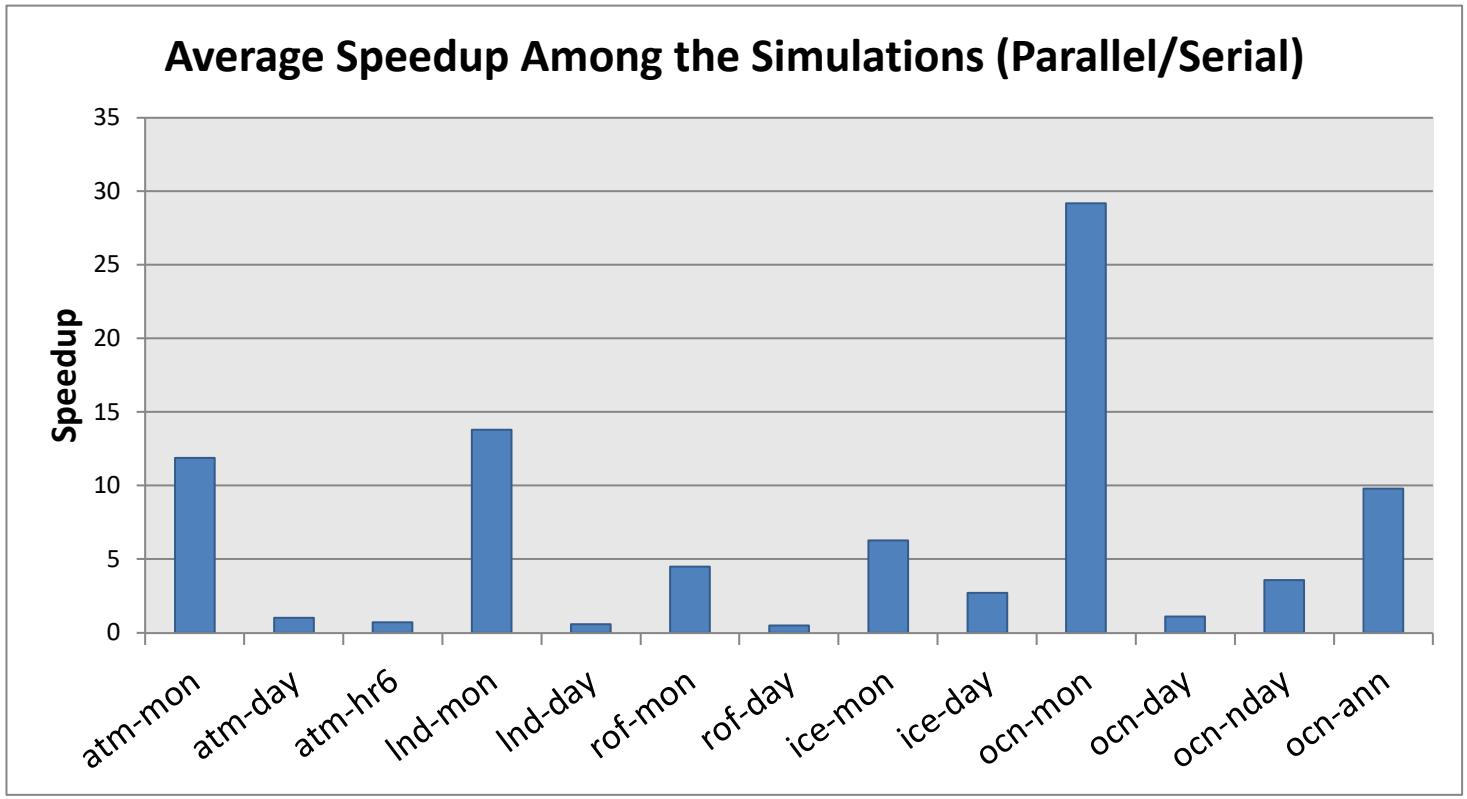




PyConform

PyReshaper

Converts files that have all variables and one time step to files that have one variable and multiple time steps



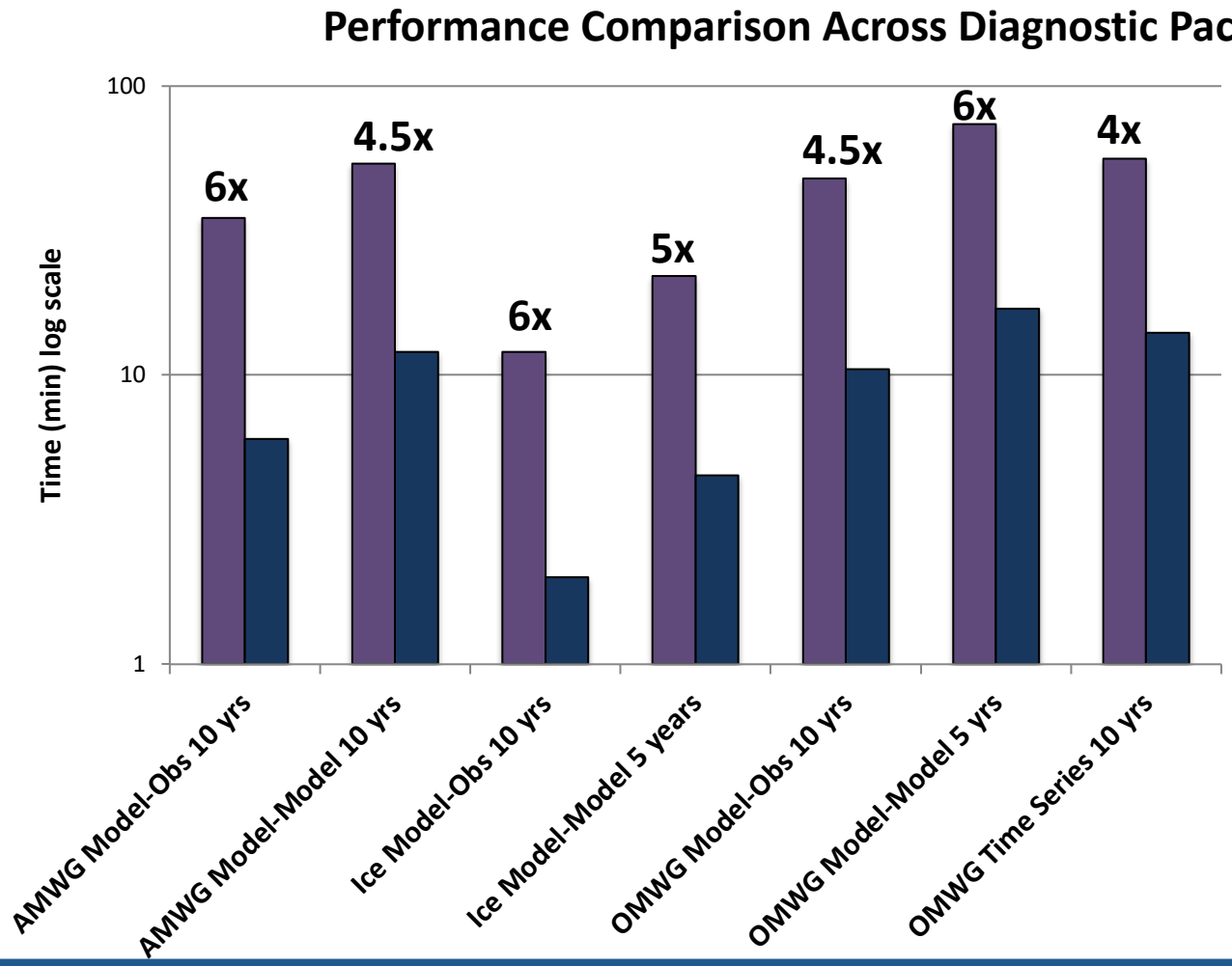
Timing results credit: Gary Strand

Maximum throughput in one day was 25TB

There was a 13x speedup for monthly output

There was an overall speedup of 6.5x across all output streams

Diagnostic Package Performance Improvements



The climatology files are calculated in parallel and the NCL plotting scripts are ran in parallel

PyConform - 1st Step

(Python Climate Output Formatter)

- Users need to create a text file with a “definition” that describe input variable(s) to output variable
  - Examples:
    - `cfc11global=f11vmr`
    - `cfc12global=f12vmr`
    - `ch4=vinth2p(CH4, hyam, hybm, plev19, PS, P0)`
    - `mc=CMFMC+CMFMCDZM`
- Then users run an input generator script that matches the “definitions” to its variable information within the CMIP6 Data Request
  - The Data Request lists variables requirements:
    - Units
    - Dimensions
    - Descriptions
    - And a lot more …

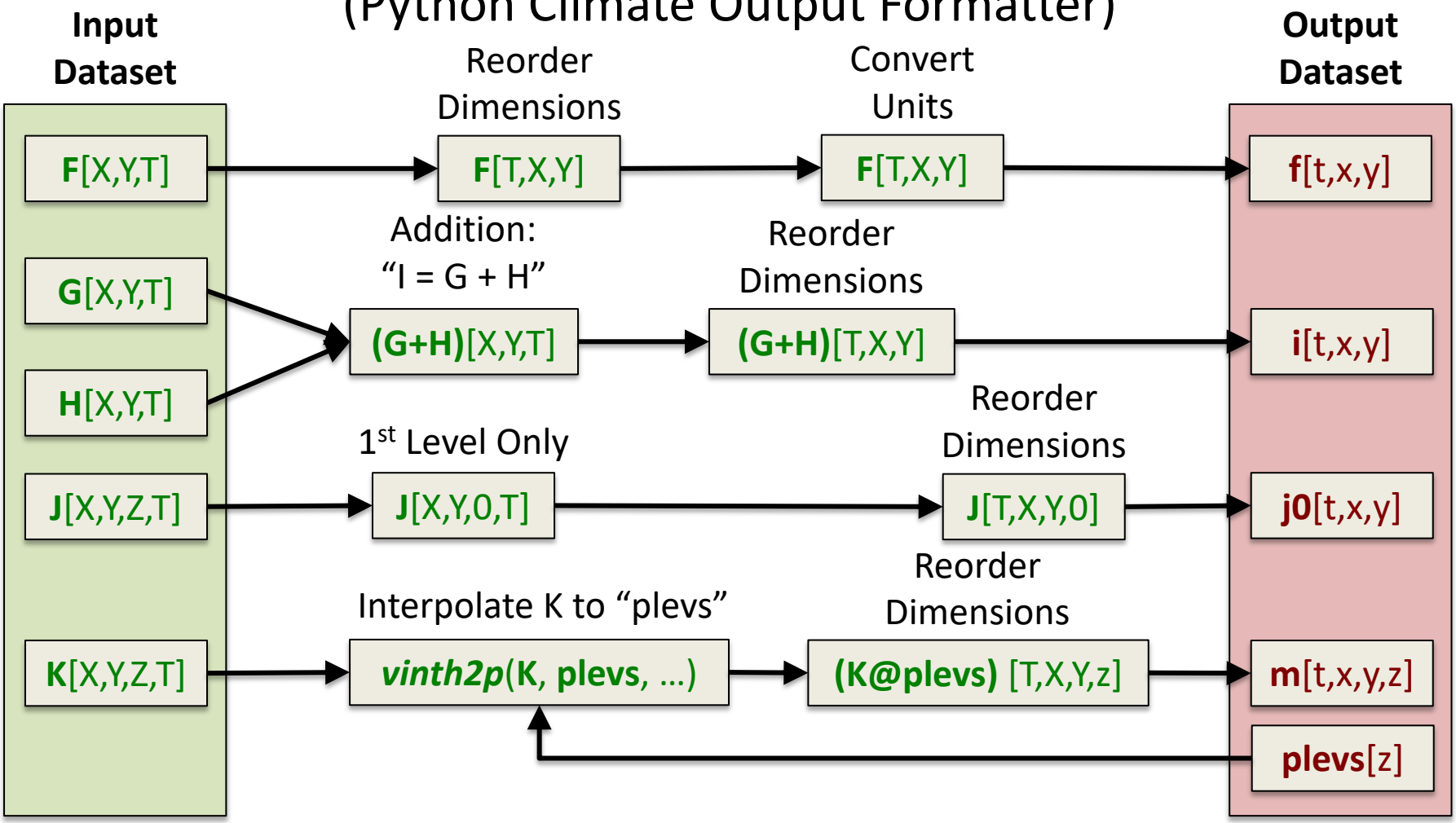
Example Input File (json format)


```
"ua": {
  "attributes": {
    "_FillValue": "1e+20",
    "cell_measures": "area: areacella",
    "cell_methods": "time: mean",
    "comment": "\"Eastward\" indicates a vector component which is positive when directed eastward (negative westward). Wind is defined as a two-dimensional (horizontal) air velocity vector, with no vertical component. (Vertical motion in the atmosphere has the standard name upward_air_velocity.)",
    "description": "\"Eastward\" indicates a vector component which is positive when directed eastward (negative westward). Wind is defined as a two-dimensional (horizontal) air velocity vector, with no vertical component. (Vertical motion in the atmosphere has the standard name upward_air_velocity.)",
    "frequency": "mon",
    "id": "ua",
    "long_name": "Eastward Wind",
    "mipTable": "Amon",
    "out_name": "ua",
    "prov": "Amon ((isd.003))",
    "realm": "atmos",
    "standard_name": "eastward_wind",
    "time": "time",
    "time_label": "time-mean",
    "time_title": "Temporal mean",
    "title": "Eastward Wind",
    "type": "real",
    "units": "m s-1",
    "variable_id": "ua"
  },
  "datatype": "real",
  "definition": "vinth2p(U,hyam, hybm, plev19, PS, P0)",
```


This is just a sample of one of the variable sections. There are other parts of the file that list other variables and then global attributes to be added to the output file.

PyConform - 2nd Step

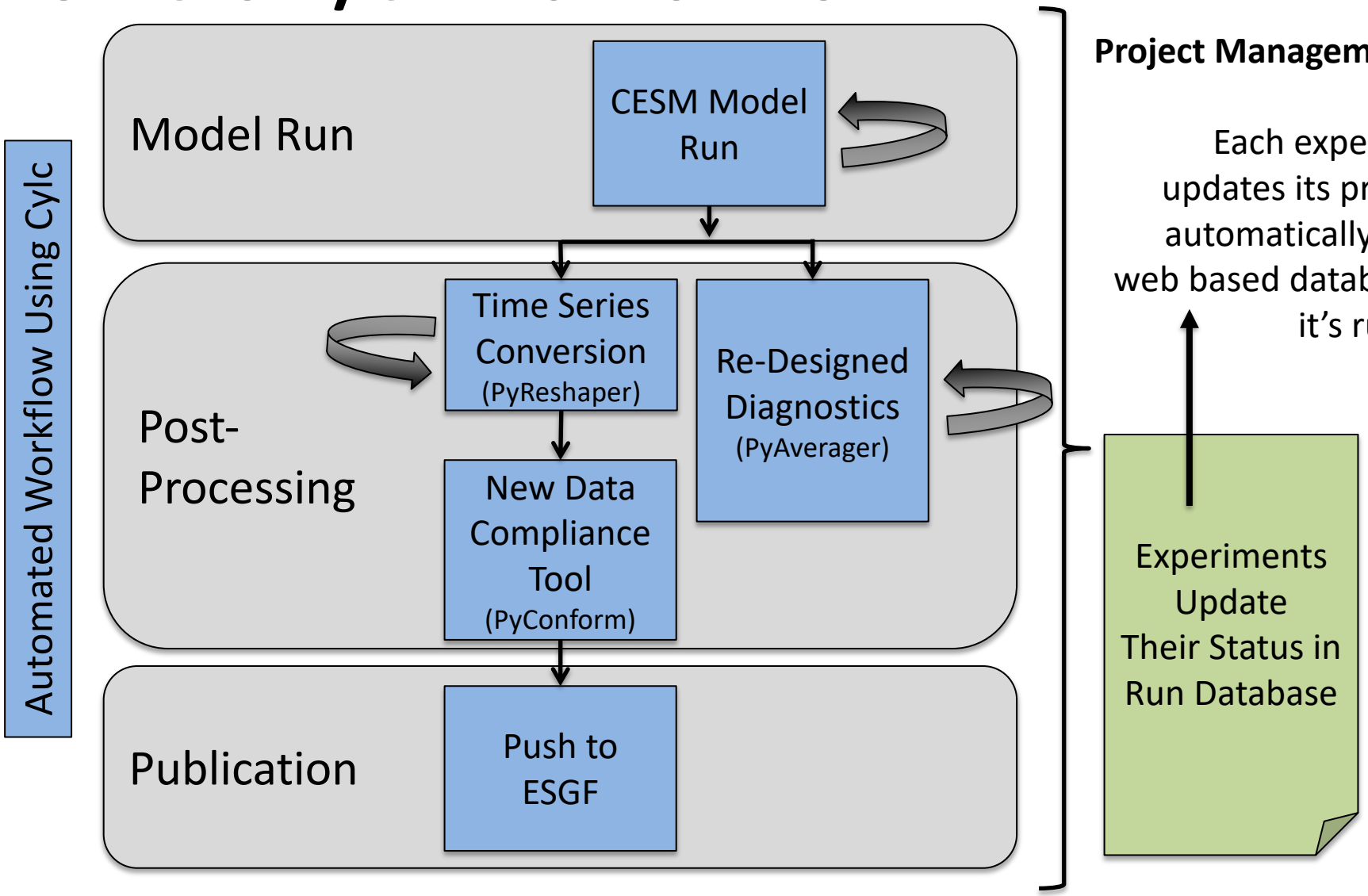
(Python Climate Output Formatter)



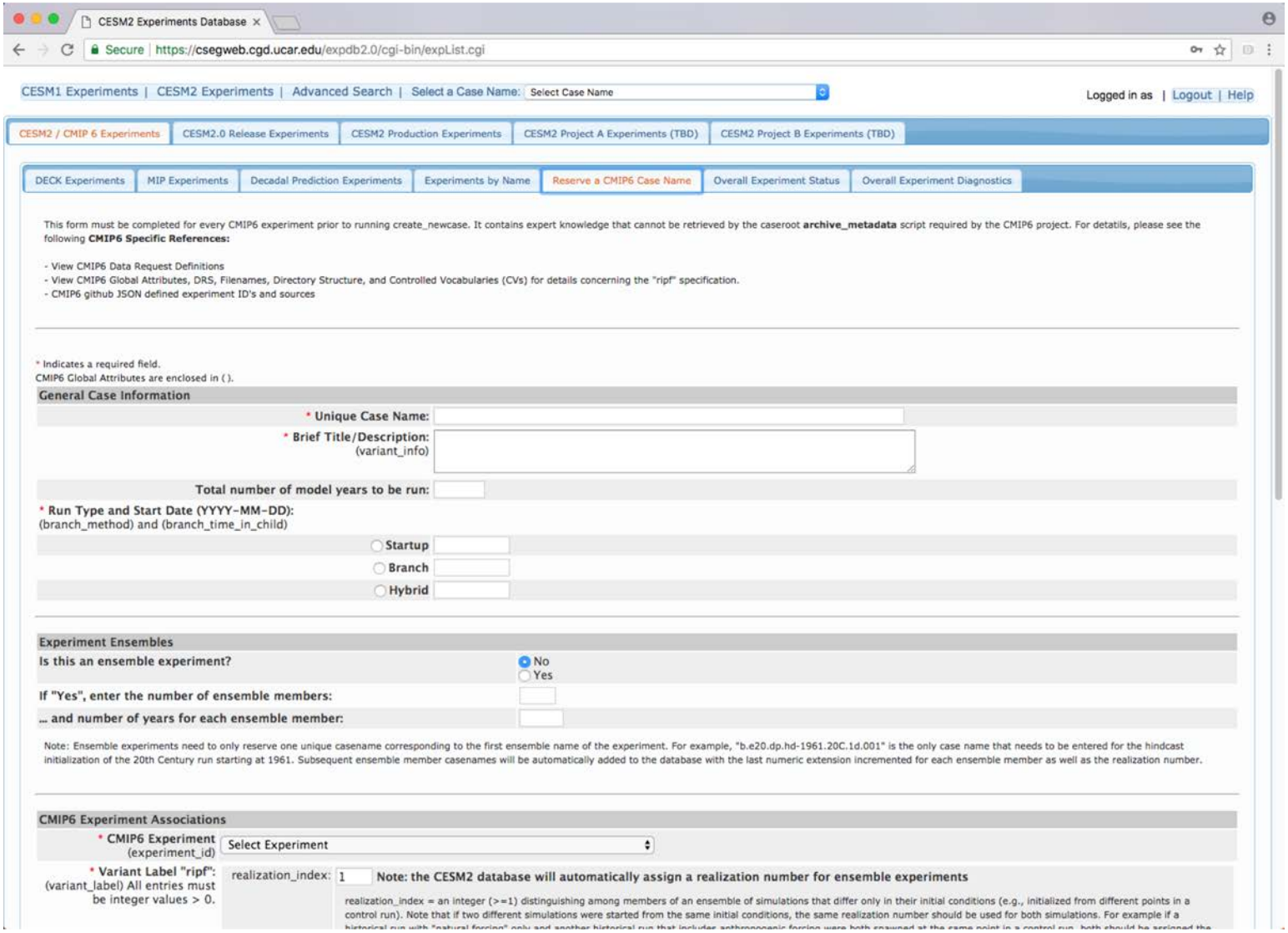
Credit: Kevin Paul

16x to 38x speedup over our old Fortran code and CMOR

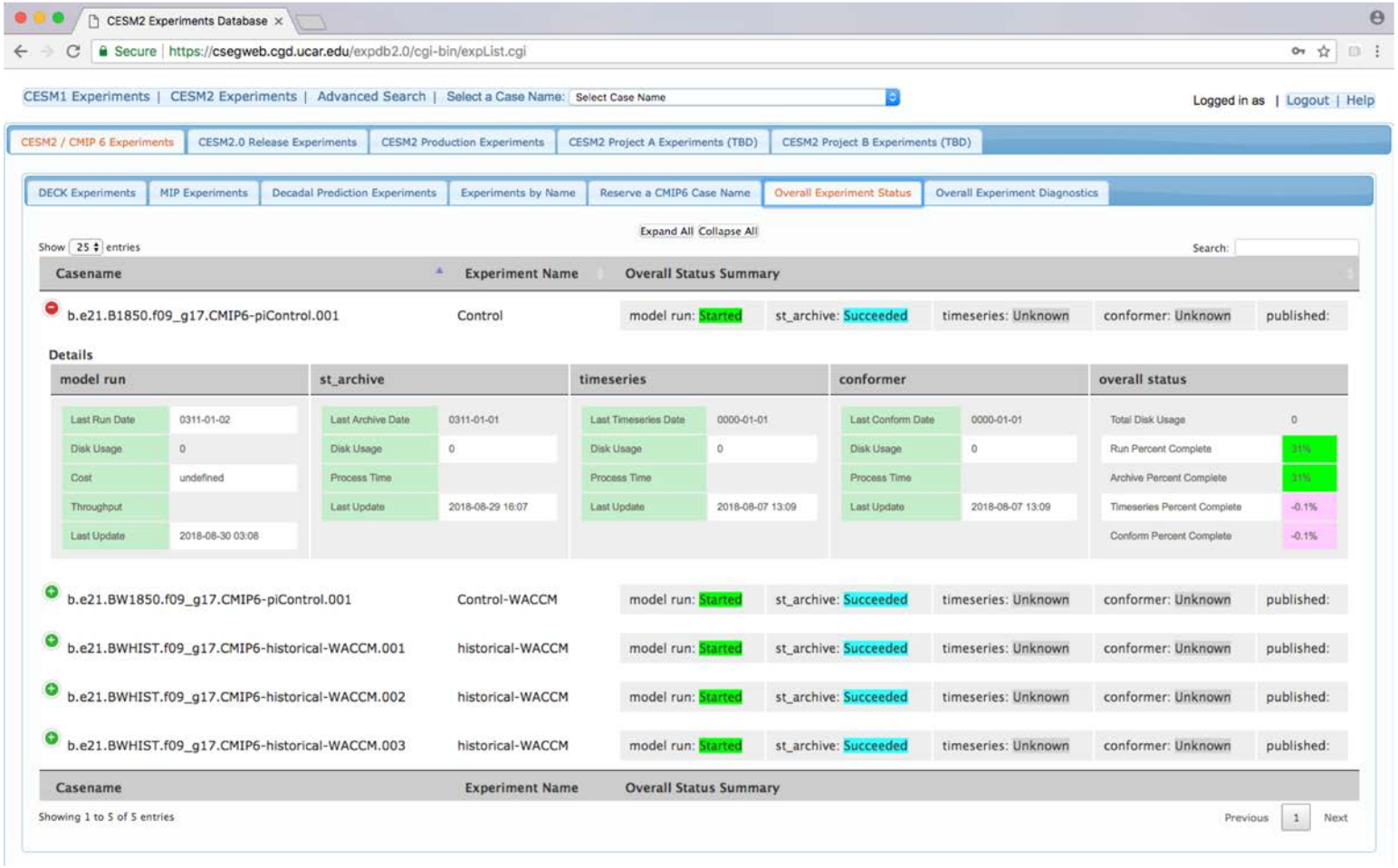
New CESM/CMIP6 Workflow



CESM's Experiment Database

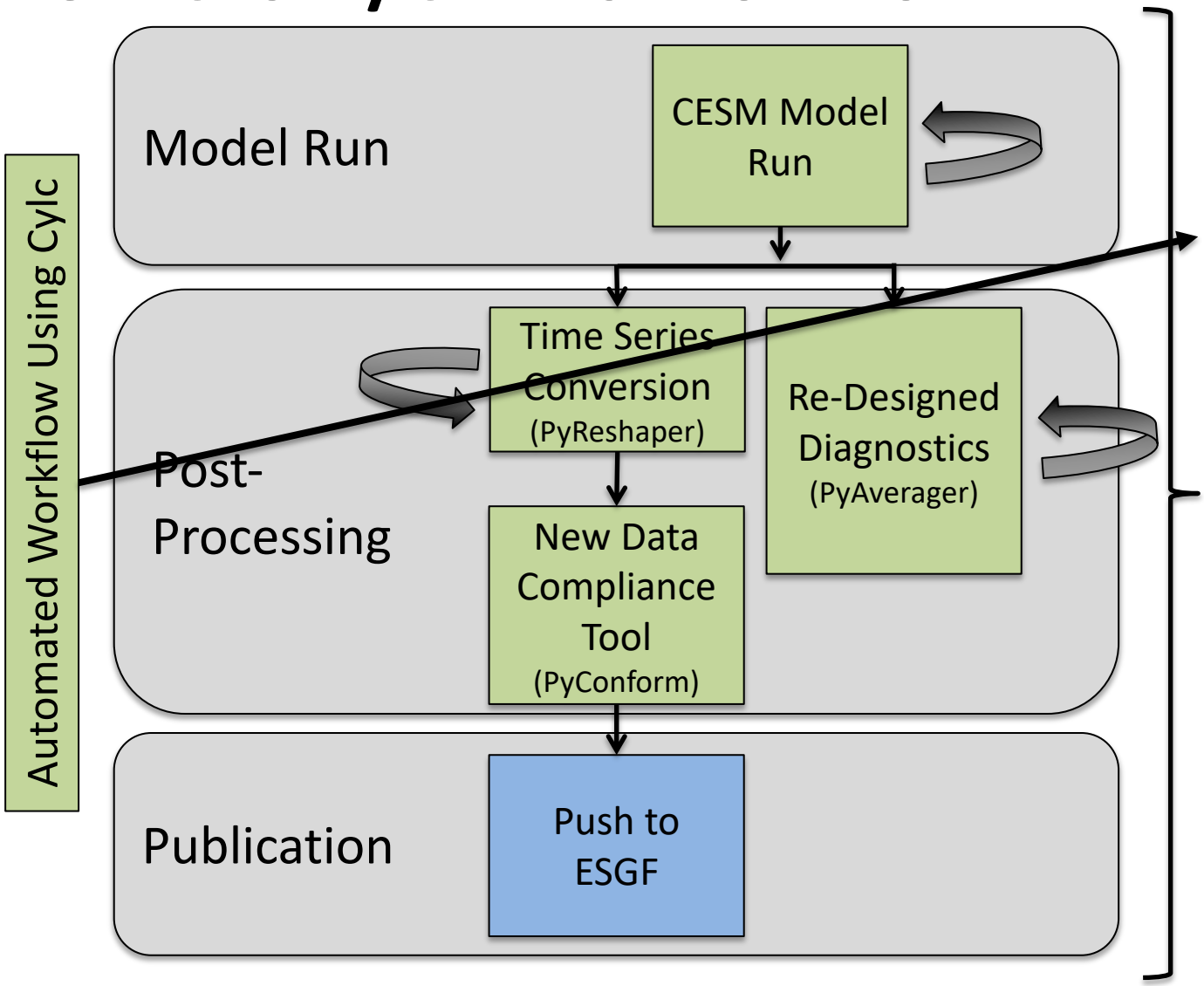


CESM's Experiment Database



This web based database has been very helpful for managers to check simulation progress and to look at results all in one place.

New CESM/CMIP6 Workflow



Automatic Suite Generation

Why? Because our users are new to Cylc, we wanted to make the transition as easy as possible to help with positive adoption

How? This is possible because CESM and our post-processing tools allow us to query the experiment to get the needed information to set up specific tasks , their dependencies, and how to submit each task to the queue

Automatic Suite Generation

User

Set in the CESM Experiment XML Files:

```
Run Length = 100 yrs
Restart = Every 10 yrs
Run diagnostics = Every 10 yrs
Run timeseries = True
Conform data = True
```

The user then runs a script to create the Cylc suite

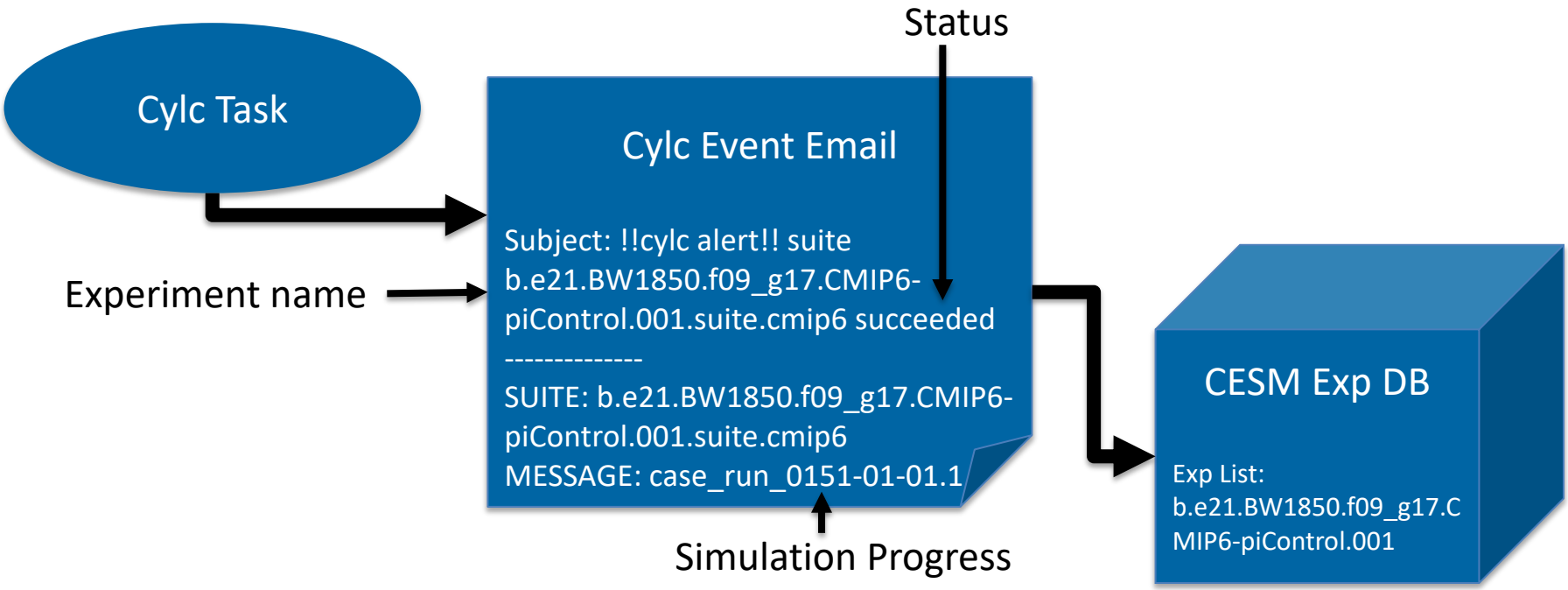
1. Look at the XML settings in the CESM env
2. Construct a dependency graph based on what the user wants to run and when
3. Look at CESM env to find out how to run each task
4. Create a suite.rc for the user based on this information
5. Register the suite for the user

```
#!Jinja2
[cylc]
[scheduling]
   [[dependencies]]
      graph = """
      case_run_0011-01-01 => case_st_archive_0011-01-01
      case_st_archive_0011-01-01 => case_run_0021-01-01
...
[runtime]
   [[root]]
      [[[environment]]]
      {% for i in range(0,dates_case_run|length) %}
      [[case_run_{{dates_case_run[i]}} ]]
      [[[job]]]
           method = pbs
           execution time limit = PT12H
           execution retry delays = PT30S, PT120S, PT600S
      [[[directives]]]
           -q = regular
           -N = b.e21.B1850.f09_g17.CMIP6-piControl.001.run
           -r = n
           -j = oe
           -S = /bin/bash
           -l = select=120:ncpus=36:mpiprocs=12:ompthreads=3

...
```

Communication Between Cylc and the Experiment Database

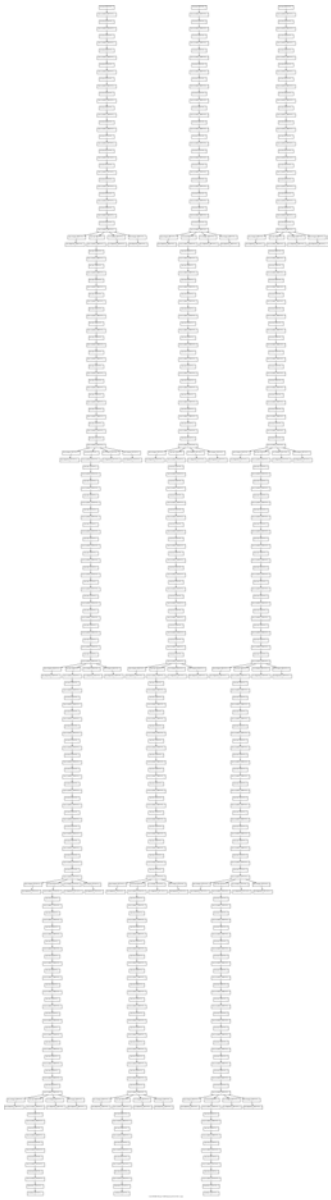
We used a naïve approach of just having Cylc email the database with progress updates and we parse the emails to update the correct database entries



Simple Workflows

Example of a single member simulation for our piControl CMIP6 experiment

- Simulates 1,000 years of climate under 1850 conditions, with each CESM run task simulating 10 years of climate
- Runs the model and archiving step about 100 times each
- Runs each of the diagnostic packages 10 different times during the simulation, every 100 years
- Creates the timeseries files
- Conforms data to meet CMIP6 standards



Ensemble Workflows

Example of a 3 member ensemble of our high top historical experiment

Each member:

- Simulates the climate from year 1850 through 2014, each CESM run task simulates 2 years of climate
- Runs the model and archiving step about 82 times each
- Runs each of the diagnostic packages 5 different times during the simulation
- Creates the timeseries files
- Conforms data to meet CMIP6 standards

Other CESM Experiments That Used Cylc (non CMIP6)

- Used Cylc to complete 1,240 out of 1,860 total runs and postprocessed ~750 TB timeslice output in about 1 month
- Used Cylc to run and postprocess part of a 30 member ensemble in a couple of months
- Used Cylc to build and run over 20,000 forecast ensembles in a couple of months

Questions?

- PyReshaper
  - https://github.com/NCAR/pyreshaper
- PyAverager
  - https://github.com/NCAR/pyAverager
- PyConform
  - https://github.com/NCAR/PyConform
- CESM/Cylc WF
  - https://github.com/NCAR/CESM-WF

Contact Info mickelso .at. ucar.edu