HWRF Internals

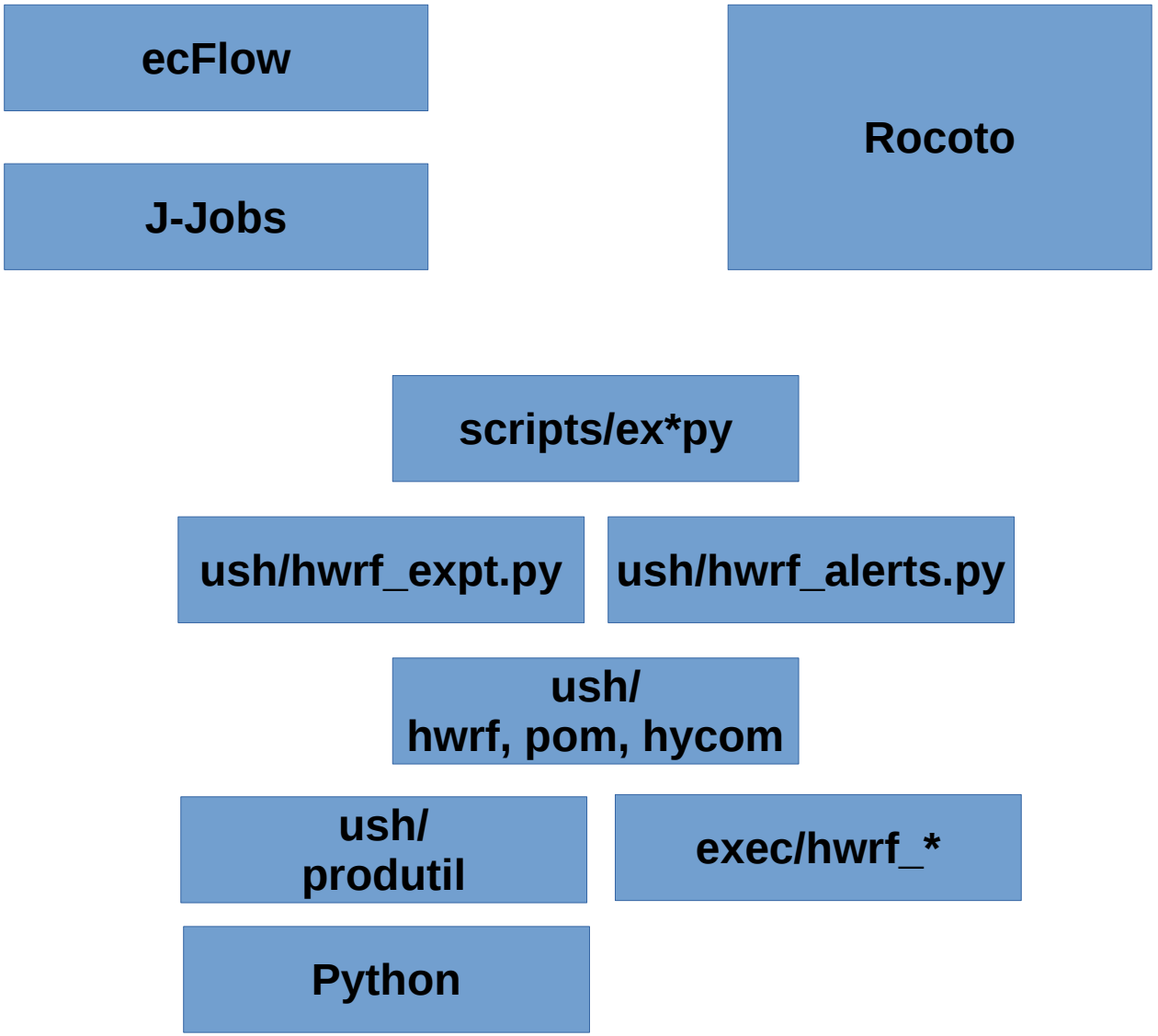
Sam Trahan

January 2016

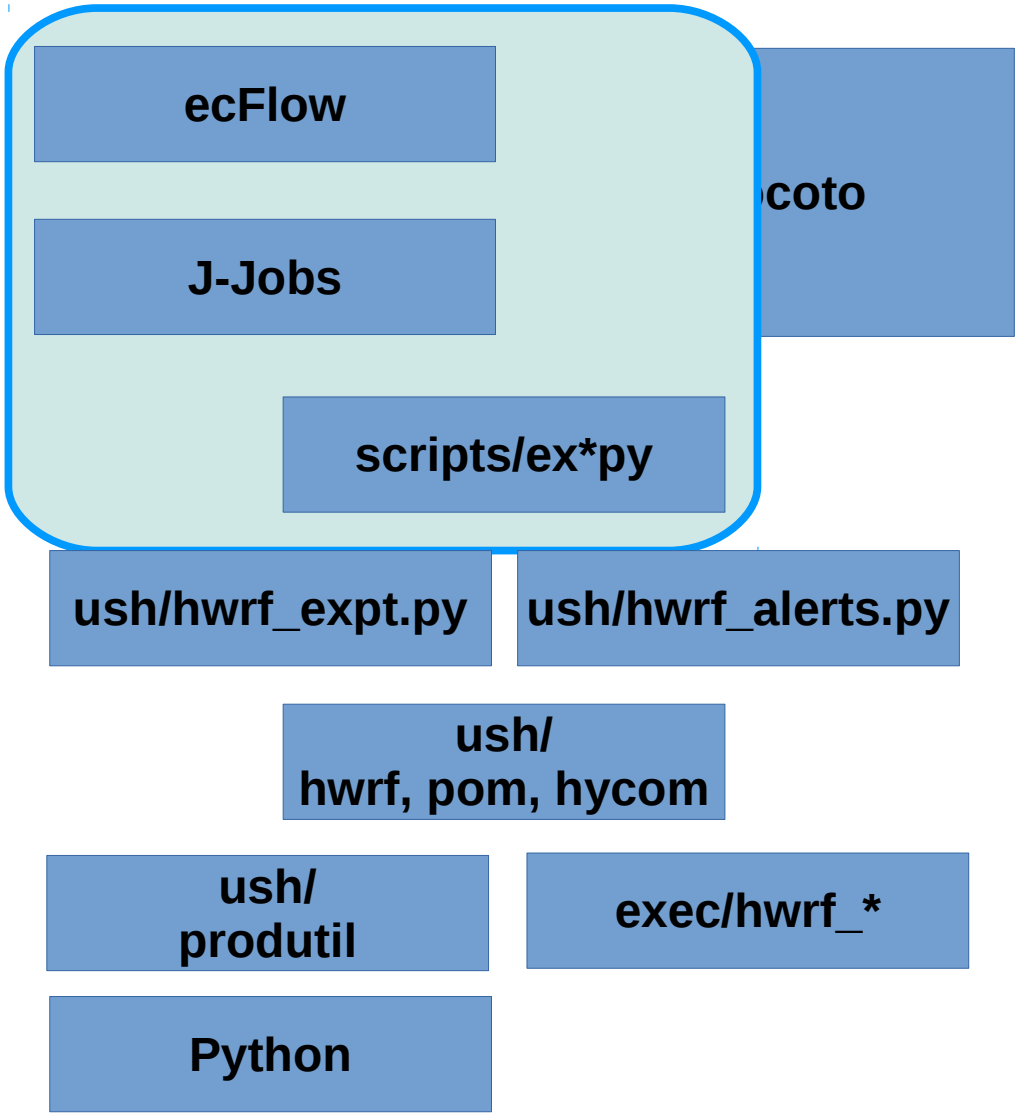
Overview

- Interaction between layers
- Configuration
- Tasks, Products and the Database

Interaction Between Layers

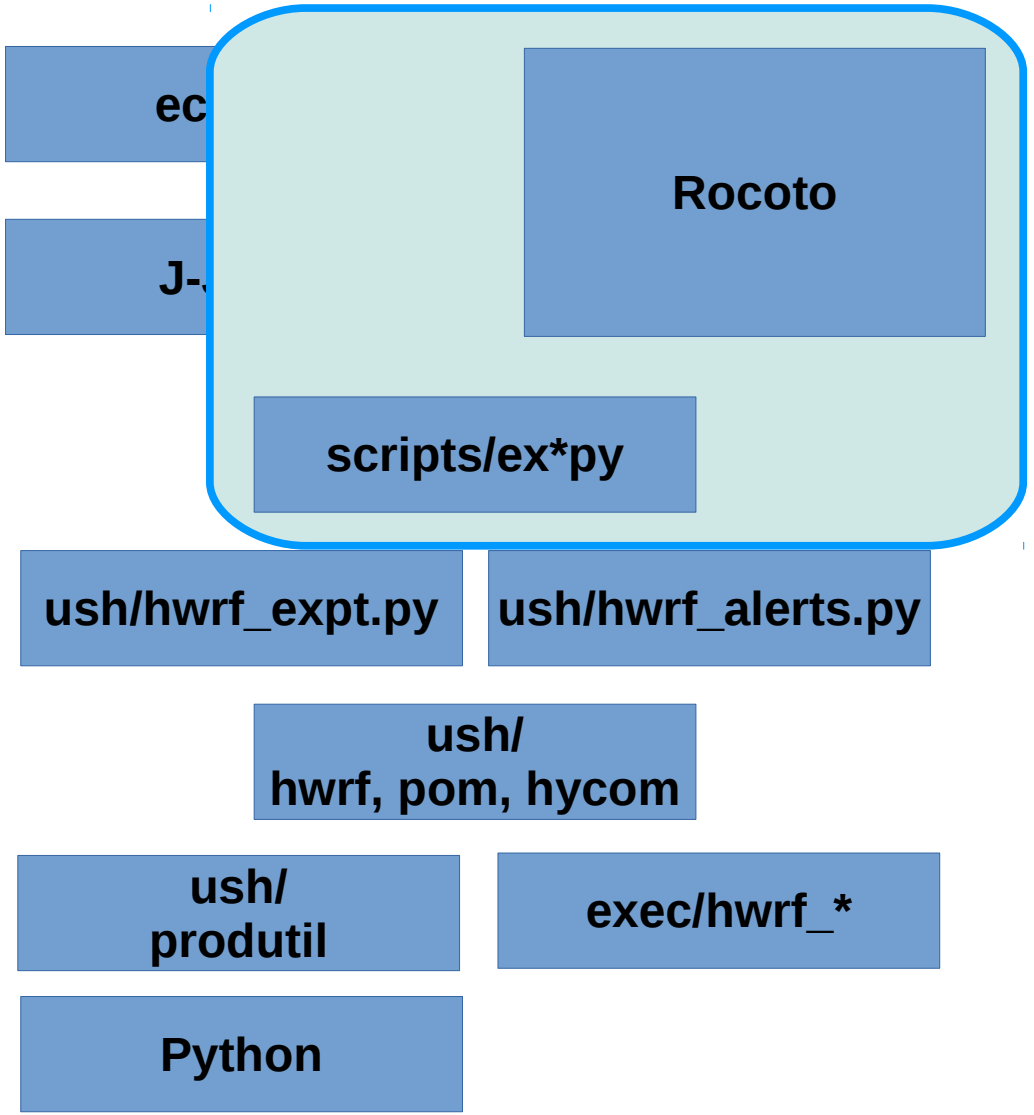


Interaction Between Layers



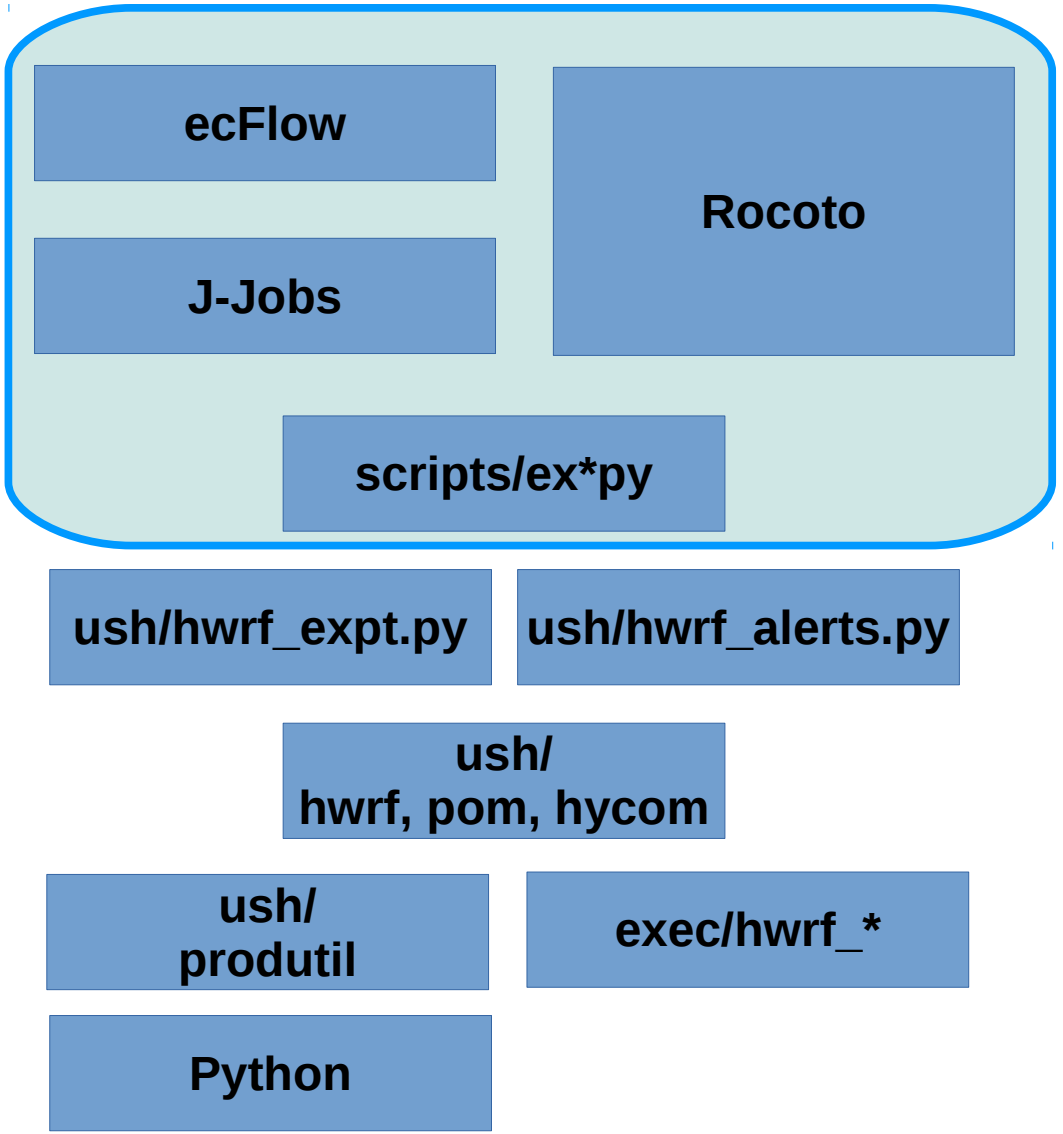
- ecFlow server
- Runs *.ecf scripts
- which call J-jobs
- which find python
- and call the script
  - may need to set env. vars or pass arguments

Interaction Between Layers



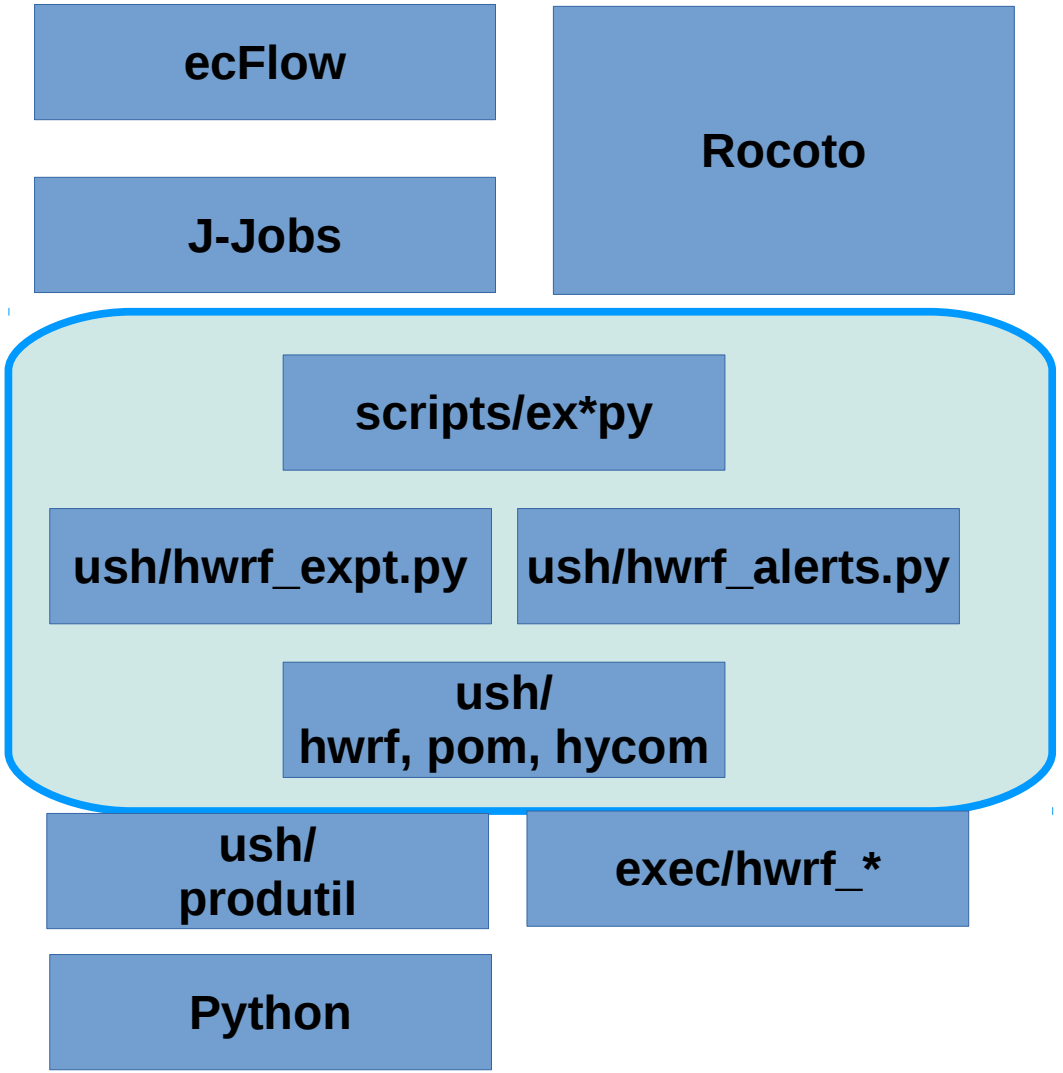
- rocoto/run_hwrf.py
  - makes XML file
  - calls rocotorun
  - which submits batch jobs
  - that run scripts
    - Same scripts, arguments, dependencies, env vars as ecFlow.

Interaction Between Layers



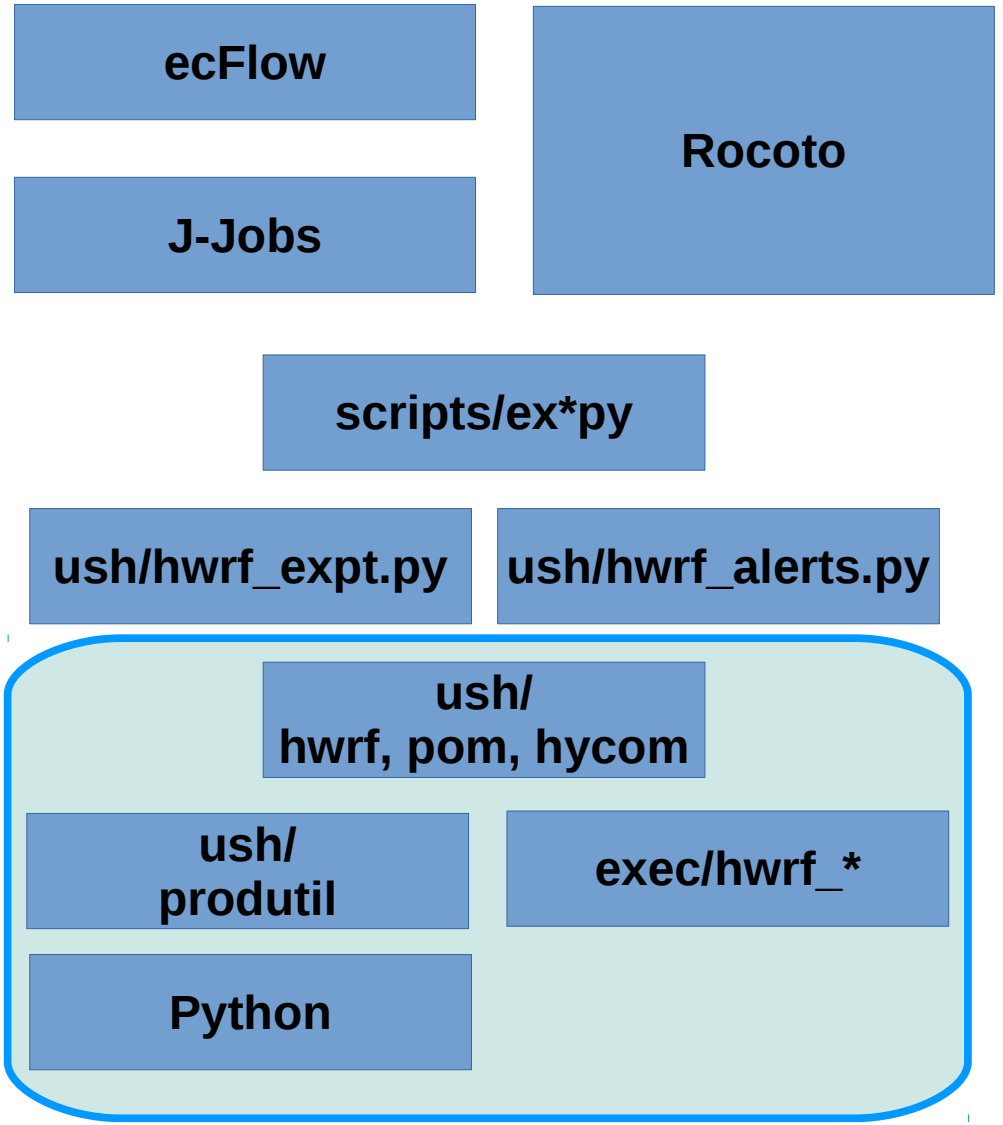
- Everything from scripts on down is identical regardless of workflow system.

Interaction Between Layers



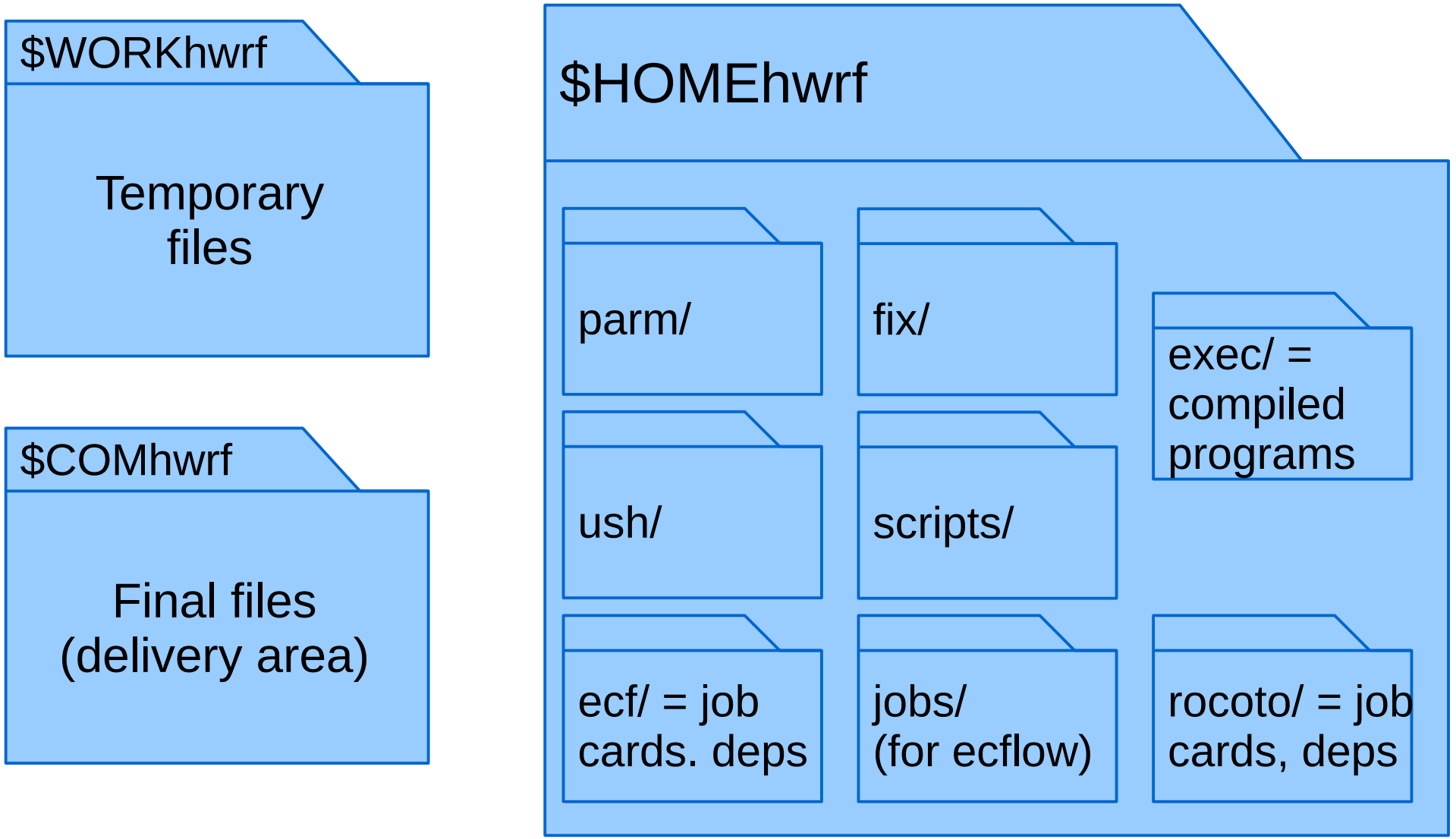
- scripts/ex*py
  - Load and initialize:
    - hwrf_expt
    - hwrf_alerts
  - hwrf_expt makes objects using hwrf, pom and hycom modules
  - Scripts run some class methods in those objects.

Interaction Between Layers



- ush/{hwrf,pom}/*py
  - Python classes that know how to run the HWRF system.
  - Built on top of produtil and the HWRF executables.
- ush/produtil
  - Python functions and classes that perform basic functionality
- Python core library underlies produtil.

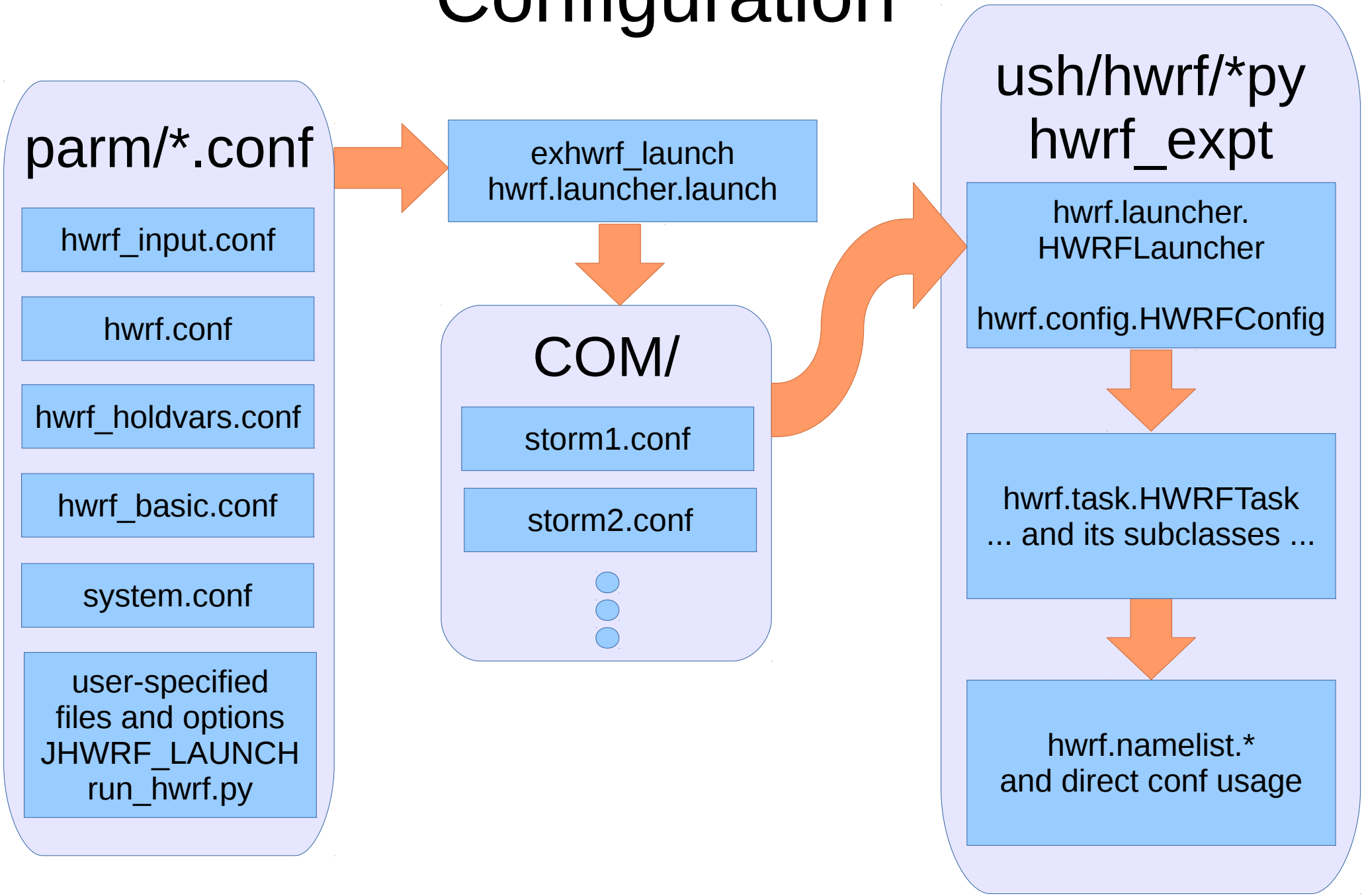
Directories



fix, parm
Data and Configuration

- fix - large binary files
- parm
  - Files used directly.
  - Files passed through configuration system to generate namelists
  - Python “conf” files
- HWRF Configuation System
  - UNIX Conf files
  - parm/*.conf
  - Config data can be substituted automatically to strings, namelists, etc.

Configuration

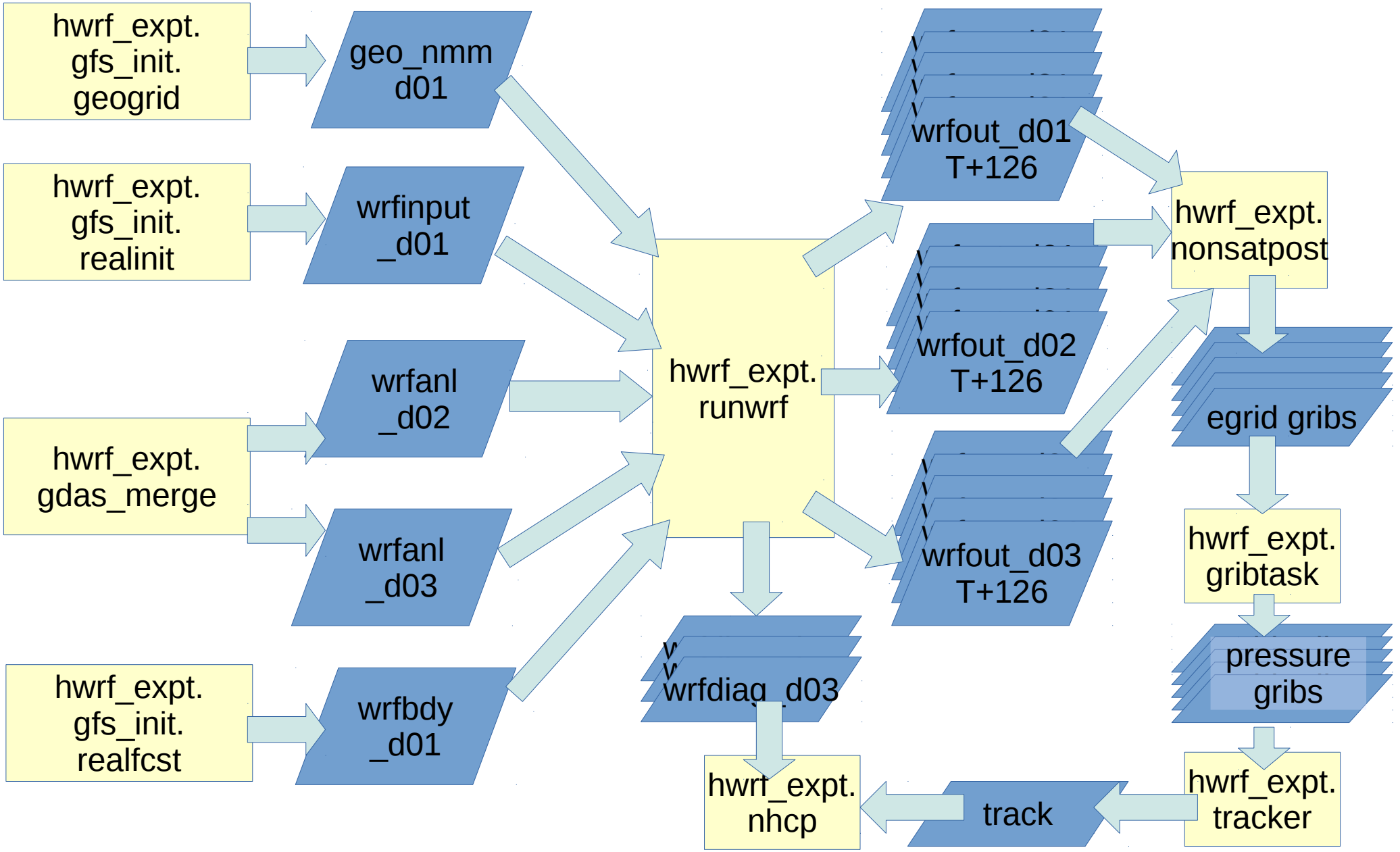


Tasks and Products

- Part of underlying object structure.
- A product is a deliverable.
  - Usually a file, with metadata
  - Maybe multiple files.
- A task is a mechanism that consumes and produces products, with a well-defined way of executing the mechanism (task.run()).

Task/Product Example

How do we track this many files!?



How do we track this many files!?

- Typical approaches are flawed:
  - Run “stat” or “ls -l” many times.
    - Waste of metadata, hard on filesystem.
  - Generate flag files.
    - Waste of metadata, hard on filesystem.
  - Rerun same operation multiple times as needed.
    - Waste of CPU and I/O.
  - Mix post and regridding in same job.
    - Huge waste during serial processing.
- **These limited, shell-based approaches are why the NCEP suite is so expensive.**

Database

Communicate Products and Tasks Between Jobs

- $WORKhwrf/hwrf_state.sqlite3
  - Communicate paths and data availability instantly between jobs.
  - Allows post and products to run in parallel in two jobs.
    - Separate serial and parallel pieces. Eliminate stat, ls -l, flag files.

Table “products”

id	available	location	type
geogrid::geo_nmm_nest	1	/new/location	Product
task::geogrid	10	/path/to/work/dir	Task

Table “metadata”

id	key	value
geogrid::geo_nmm_nest	minsize	100000000

Primary, Backup Data Sources

- Some tasks need inputs from other tasks.
  - runwrf.add_wrfinput(gdas_merge)
    - Get input from gdas_merge
  - runwrf.add_wrfinput(gfs_init.rstage3)
    - If GDAS merge fails, try getting wrfinput from GFS analysis vortex relocation step.
  - runwrf.add_wrfinput(gfs_init.realinit)
    - If relocation also failed, get it from GFS analysis
    - (This one is disabled; we would rather the workflow fail.)
- Generates a list of objects, each of which are queried for input, until one is found that has data
  - Intentionally fails unless [config] allow_fallbacks=yes

Fallbacks

- Many jobs have fallback options:
  - Run uncoupled.
  - Get wrfinput from GFS analysis relocation
  - etc.
- Some are enabled automatically via:
  - [config] allow_fallbacks=yes
  - (Turned on by default in operations.)
- Others can be done manually via editing $COMhwrf/storm*.conf, and resubmitting jobs.

More Information - Later Talks

- Object-Oriented Scripting
- Python “produtil” Package
- HWRF Logging Overview
- Troubleshooting HWRF
- Configuring HWRF
- Rocoto for HWRF
- HWRF Database
- Debugging HWRF Scripts
- Demo Session: Add a new component to HWRF