Toward a Robust Unified Workflow Toolbox and Framework

Christina Holt and Paul Madden

Emily Carpenter, Naureen Bharwani, Brian Weir, Fredrick Gabelmann

Improving Scientific Software Conference

April 15, 2024

The Unified Workflow Project: The Team

A UFS Agile Team operating under EPIC's SAFe welcoming contributors with various objectives to achieve a common end goal – ***unification of workflows across UFS***

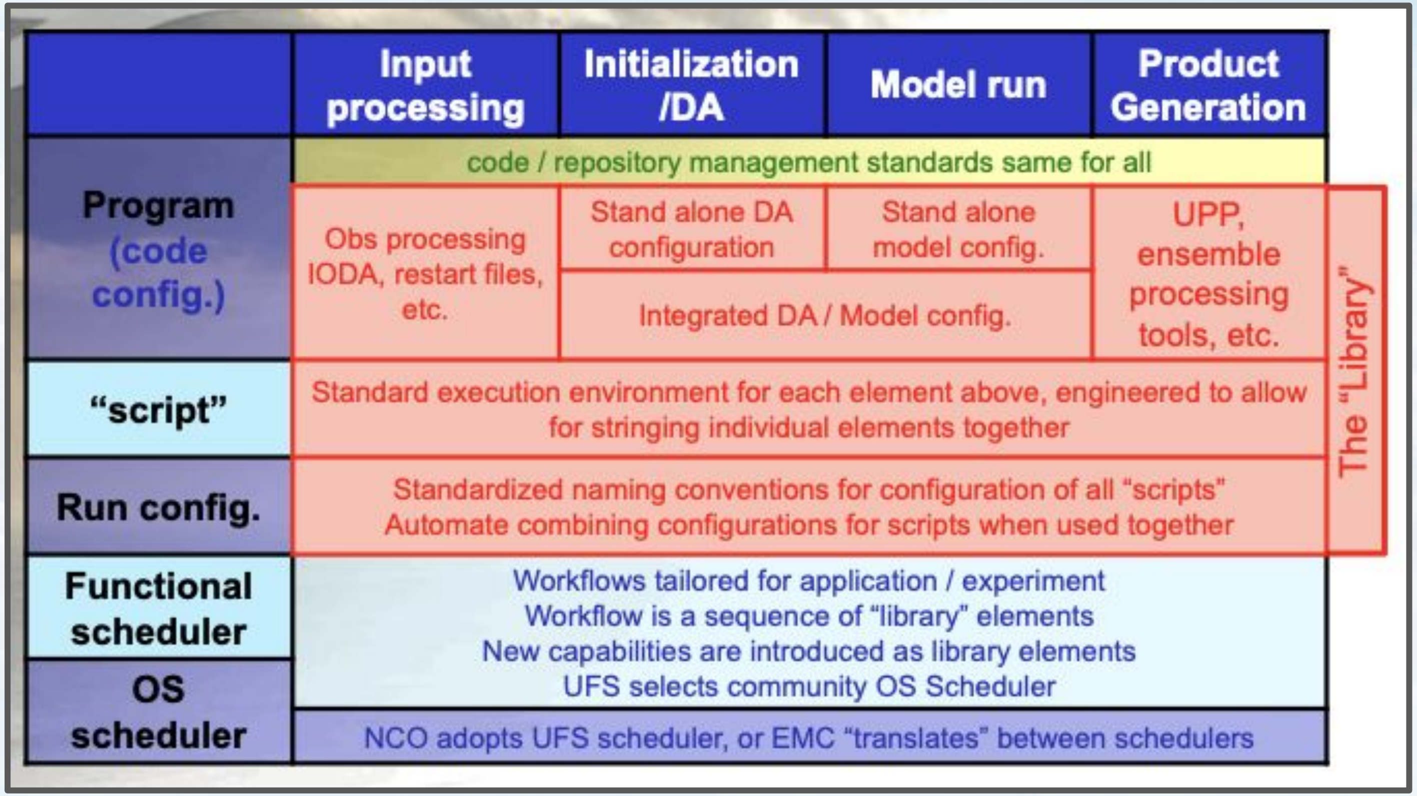
The current team comprises staff from CIRES / NOAA GSL with funding from the JTTI and SENA, and Raytheon/Element 84 Staff under the EPIC Contract

The Unified Workflow Project: The Goals

Unification:	Multiple apps using the ***same tools*** to perform the ***same tasks***
Ease of Use:	***Flattening the learning curve*** to get what you need out of a UFS App
Flexibility:	***Empowering scientists*** to realize experiments without being limited to what already exists
Facilitate R2O:	***Research and operations*** configuring and running the ***same components*** with the ***same languages and infrastructure***

Powered By GSL

UFS Workflow Workshop Ideas



	Input processing	Initialization /DA	Model run	Product Generation	
Program (code config.)	code / repository management standards same for all				
	Obs processing IODA, restart files, etc.	Stand alone DA configuration	Stand alone model config.	UPP, ensemble processing tools, etc.	The "Library"
		Integrated DA / Model config.			
"script"	Standard execution environment for each element above, engineered to allow for stringing individual elements together				
Run config.	Standardized naming conventions for configuration of all "scripts" Automate combining configurations for scripts when used together				
Functional scheduler	Workflows tailored for application / experiment Workflow is a sequence of "library" elements New capabilities are introduced as library elements UFS selects community OS Scheduler				
OS scheduler	NCO adopts UFS scheduler, or EMC "translates" between schedulers				

The Unified Workflow Project: The Software

A Python package that provides command line tools and a Python API to perform **common workflow tasks** and **drive UFS components** (forecast model, post processing, verification, etc.)

We publish each release to the ufs-community Anaconda channel for easy installation

We're planning to integrate the tools into multiple UFS Applications
Currently working with
Short Range Weather (a limited-area, static domain system)
Land DA (stand-alone JEDI-based land state cycling)

Powered By GSL

uwtools v2.1.1

Install it with conda:

```
$ conda install -c ufs-community uwtools=2.1.*
```

Use the command line tools:

```
$ uw [mode] [action] [-h]
```

Use the API:

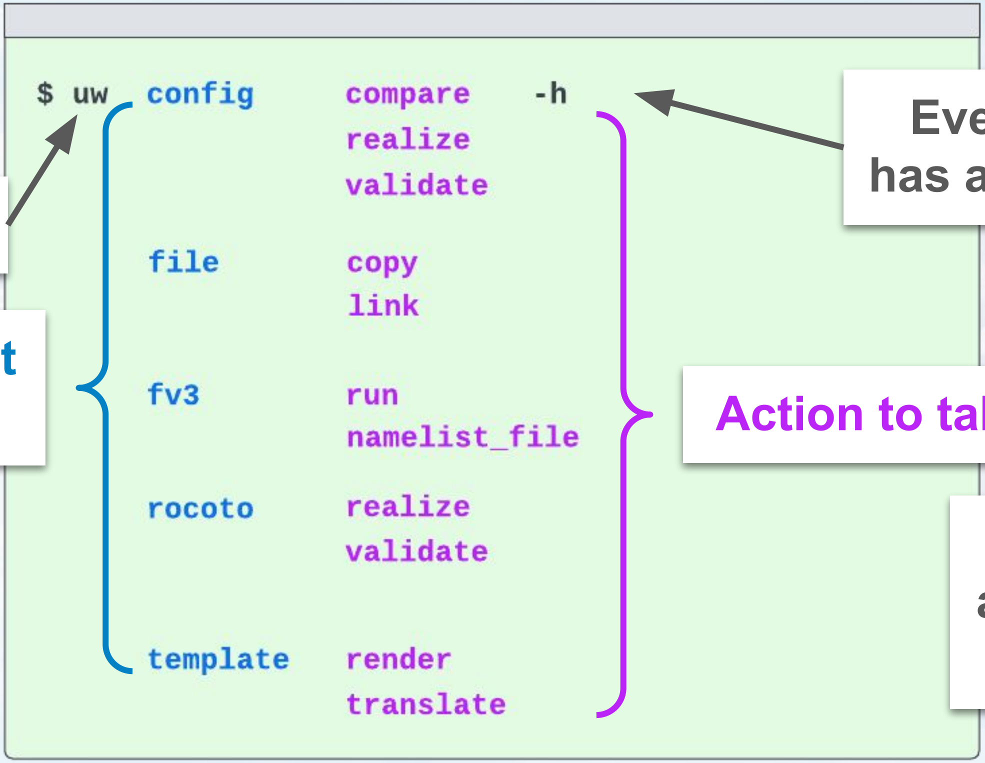
```
import uwtools.api as uwtools
```

On GitHub:

On Read the Docs:

Powered By GSL

Intro to uwtools CLI



```
$ uw  config      compare    -h
                  realize
                  validate

      file        copy
                  link

      fv3         run
                  namelist_file

      rocoto      realize
                  validate

      template    render
                  translate
```

One Interface: `uw`

Mode for what to act on

Action to take

Every level has a help flag

Each combo has an API equivalent not shown here

Powered By GSL

Generic Tools: Config

compare, transform, modify, and validate
key/value configurations
YAML, **Fortran namelists**, **INI**, **bash**

Powered By GSL

Generic Tools: Config Compare

```
(uwtools) $ uw config compare --file-1-path input1.nml --file-2-path input2.nml
[2024-01-25T17:52:27]     INFO - input1.nml
[2024-01-25T17:52:27]     INFO + input2.nml
[2024-01-25T17:52:27]     INFO ---------------------------------------------------------------------
[2024-01-25T17:52:27]     INFO atmos_model_nml:       blocksize:  - 32 + 40
[2024-01-25T17:52:27]     INFO atmos_model_nml:      ccpp_suite:  - FV3_GFS_v16 + FV3_HRRR
[2024-01-25T17:52:27]     INFO cires_ugwp_nml:    launch_level:  - 27 + 25
[2024-01-25T17:52:27]     INFO fv_core_nml:   agrid_vel_rst:  - False + None
[2024-01-25T17:52:27]     INFO fv_core_nml:        d2_bg_k2:  - 0.0 + 0.04
...
```

Powered By GSL

Generic Tools: Config Realize

input file

```
platform:
  NCORES_PER_NODE: 40

task_run_fcst:
  RUN_FCST_TN: "run_fcst"
  PE_MEMBER01: 221
  NNODES_RUN_FCST: !int '{{ (PE_MEMBER01 + PPN_RUN_FCST - 1) // PPN_RUN_FCST }}'
  PPN_RUN_FCST: !int '{{ platform.NCORES_PER_NODE // OMP_NUM_THREADS_RUN_FCST }}'
  WTIME_RUN_FCST: 04:30:00
  MAXTRIES_RUN_FCST: 1
  OMP_NUM_THREADS_RUN_FCST: 2
```

apply supplemental file

```
task_run_fcst:
  OMP_NUM_THREADS_RUN_FCST: 4
```

output file

```
platform:
  NCORES_PER_NODE: 40

task_run_fcst:
  RUN_FCST_TN: "run_fcst"
  PE_MEMBER01: 221
  NNODES_RUN_FCST: 23
  PPN_RUN_FCST: 10
  WTIME_RUN_FCST: 04:30:00
  MAXTRIES_RUN_FCST: 1
  OMP_NUM_THREADS_RUN_FCST: 4
```

Powered By GSL

Generic Tools: Template

Leverage **Jinja** for a Turing-complete templating language

https://jinja.palletsprojects.com/en/3.1.x/

Powered By GSL

Generic Tools: Template Render

Tools are designed to leverage Linux pipes and env vars

```
(uwtools) $ export cycle=2023092112
(uwtools) $ echo '{{ cycle[0:4] }}' | uw template render
2023
(uwtools) $ echo '{{ cycle[8:] }}' | uw template render
12
(uwtools) $ echo '{{ cycle[0:4] }}' | uw template render --values-needed
[2024-03-01T12:25:05]     INFO Value(s) needed to render this template are:
[2024-03-01T12:25:05]     INFO cycle
```

`--values-needed` flag to introspect templates and provide a list of required variables

Generic Tools: Template Render

Render templates from environment variables or values files

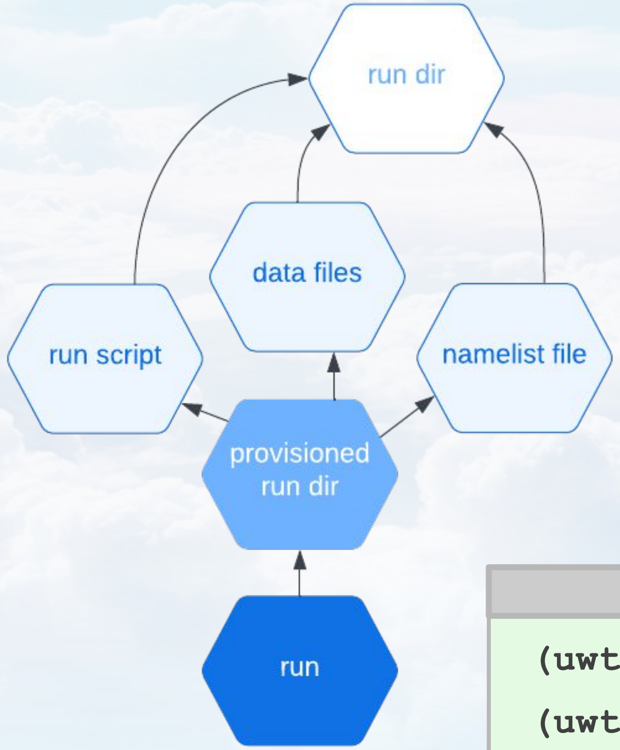
```
start_year:             {{ cycle[0:4] }}
start_month:            {{ cycle[4:6] }}
start_day:              {{ cycle[6:8] }}
start_hour:             {{ cycle[8:] }}
start_minute:           0
start_second:           0
nhours_fcst:            {% if cycle[8:] == 12 %}48{%else%}12{%endif%}
dt_atmos:               {{DTATMOS}}
...
```

Powered By GSL

Graph-Oriented Drivers

Powered By GSL

Graph-Oriented Drivers: Basics



A driver is responsible for configuring and executing an NWP component program.

Each task is represented by a node in a graph. Dependencies are represented by edges. The driver can request any node in the graph.

Directed Acyclic Graphs are the core principle behind "workflow managers" and provide a valuable set of task-execution properties.

```
(uwtools) $ uw ungrib run …
(uwtools) $ uw ungrib namelist_file …
```

Powered By GSL

Graph-Oriented Drivers: How?

Consider make's file- and timestamp-based model, which implies a DAG.

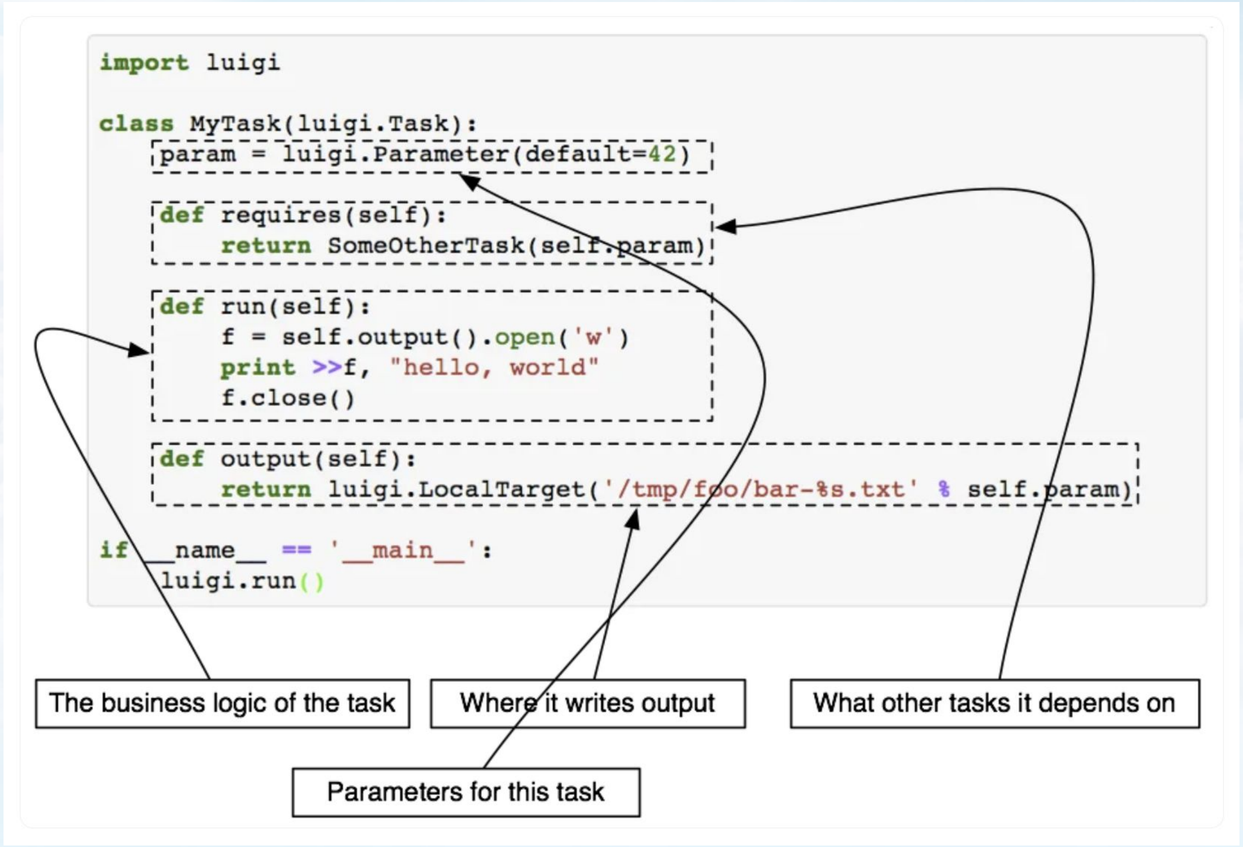
Targets have recipes to build them, and depend on certain prerequisites.

```
foo: foo.c bar.o baz.o
    cc -o foo foo.c bar.o baz.o

bar.o: bar.c
    cc -c bar.c

baz.o: baz.c
    cc -c baz.c
```

```
foo: foo.c bar.o baz.o
    cc -o $@ $^

%.o: %.c
    cc -c $<
```

Graph-Oriented Drivers: How?

Spotify's Luigi workflow manager generalizes make's model to deal with assets other than files, and criteria other than timestamps.



```
import luigi

class MyTask(luigi.Task):
    param = luigi.Parameter(default=42)

    def requires(self):
        return SomeOtherTask(self.param)

    def run(self):
        f = self.output().open('w')
        print >>f, "hello, world"
        f.close()

    def output(self):
        return luigi.LocalTarget('/tmp/foo/bar-%s.txt' % self.param)

if __name__ == '__main__':
    luigi.run()
```

The business logic of the task
Where it writes output
What other tasks it depends on
Parameters for this task

Powered By GSL

Graph-Oriented Drivers: How?

The technique used by uwtools involves decorated Python functions* that yield values that guide driver behavior.

```
@task
def runscript(self):
    d = rundir() # another @task
    path = ref(d) / "runscript"
  ❶ yield f"Runscript {path}"
   ②yield asset(path, path.is_file)
   ③yield d
   ④write_runscript(path)
```

❶ Yield the task name, for logging.

② Yield the asset – comprising a reference to the asset, usable by other tasks; and a function the framework can call to check readiness – that this task makes ready.

③ Yield any task(s) whose assets are required by this task.

④ Imperative logic to ready the asset(s).

* Note the lack of error handling, logging, etc., which is handled by the framework.

Graph-Oriented Drivers: Properties

- **Iteration:** A task graph can be executed multiple times, potentially making progress toward its final state with each execution.
- **Composition:** One task graph can be "wired up" to a larger one by declaring it a dependency.
- **Selective Execution:** Tasks (subgraphs) can be executed à la carte.
- **Toil Reduction:** Essential logging, error checking, etc., handled by framework.
- **Idempotence:** Work done once is not repeated, regardless of re-execution and without additional guard logic.
- **Self-Healing:** Deleted or damaged outputs/assets can be recreated by executing the graph again – possibly with corrected configuration.

Graph-Oriented Drivers: Demo – a Composite Task

```
@tasks
def provisioned_run_directory(self):
    yield "provisioned run directory"
    yield [
        self.gribfiles(),
        self.namelist_file(),
        self.runscript(),
        self.vtable(),
    ]
```

The **`@tasks`** decorator defines a task that simply "contains" other tasks and produces no output of its own. It is "ready" when all its contained tasks are.

A "provisioned run directory" contains everything needed to run a component, but stops just short of running it.

* A third decorator, `@external`, models assets that must exist a priori, like `.c` files for `make`.

Graph-Oriented Drivers: Demo – Partial Progress

```
% uw ungrib provisioned_run_directory --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib provisioned run directory: Initial state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Initial state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Checking requirements
[2024-04-08T19:08:20]  WARNING File /home/maddenp/ungrib/gfs.t06z.pgrb2.0p25.f000: State: Pending (EXTERNAL)
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Requirement(s) pending
[2024-04-08T19:08:20]  WARNING 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Final state: Pending
[2024-04-08T19:08:20]  WARNING 20240408 06Z ungrib GRIB files: Final state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Initial state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Checking requirements
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Requirement(s) ready
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Executing
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Final state: Ready
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib runscript.ungrib: Initial state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib runscript.ungrib: Checking requirements
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib runscript.ungrib: Requirement(s) ready
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib runscript.ungrib: Executing
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib runscript.ungrib: Final state: Ready
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Initial state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Checking requirements
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Requirement(s) ready
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Executing
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Final state: Ready
[2024-04-08T19:08:20]  WARNING 20240408 06Z ungrib provisioned run directory: Final state: Pending
```

Powered By GSL

Graph-Oriented Drivers: Demo – Partial Progress

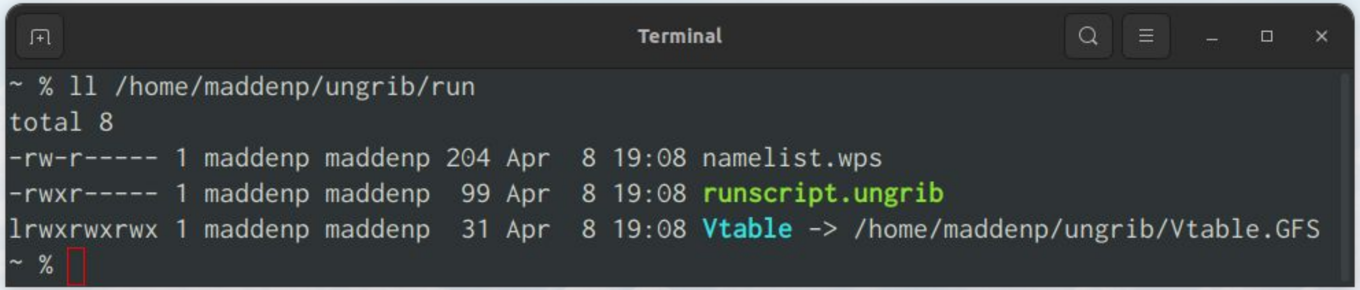
```
% uw ungrib provisioned_run_directory --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib provisioned run directory: Initial state: Pending
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-08T                                                                                   ding
[2024-04-08T                                                                                   nts
[2024-04-08T
[2024-04-08T                                                                                   ding
[2024-04-08T                                                                                   ng
[2024-04-08T
[2024-04-08T                                                                                   ding
[2024-04-08T                                                                                   nts
[2024-04-08T                                                                                   dy
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Executing
[2024-04-08T19:08:20]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/Vtable: Final state: Ready
[2024-04-08T19:08:20]  WARNING 20240408 06Z ungrib provisioned run directory: Final state: Pending
```

Properties demonstrated:

- **Iteration:** Progress, if incomplete, was made.
- **Composition:** `provisioned_run_directory` is composed of several other tasks
- **Selective Execution:** `provisioned_run_directory` is a sub-task of the larger `run` task, but we could execute it directly.
- **Toil Reduction:** Missing dependency handling, logging with timestamps, etc. all provided by framework with no code

Powered By GSL

Graph-Oriented Drivers: Demo – Task Completion



```
~ % ll /home/maddenp/ungrib/run
total 8
-rw-r----- 1 maddenp maddenp 204 Apr  8 19:08 namelist.wps
-rwxr----- 1 maddenp maddenp  99 Apr  8 19:08 runscript.ungrib
lrwxrwxrwx 1 maddenp maddenp  31 Apr  8 19:08 Vtable -> /home/maddenp/ungrib/Vtable.GFS
~ %
```

```
% uw ungrib provisioned_run_directory --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib provisioned run directory: Initial state: Pending
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Initial state: Pending
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Checking requirements
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Requirement(s) ready
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Executing
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Final state: Ready
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib provisioned run directory: Final state: Ready
```

Powered By GSL

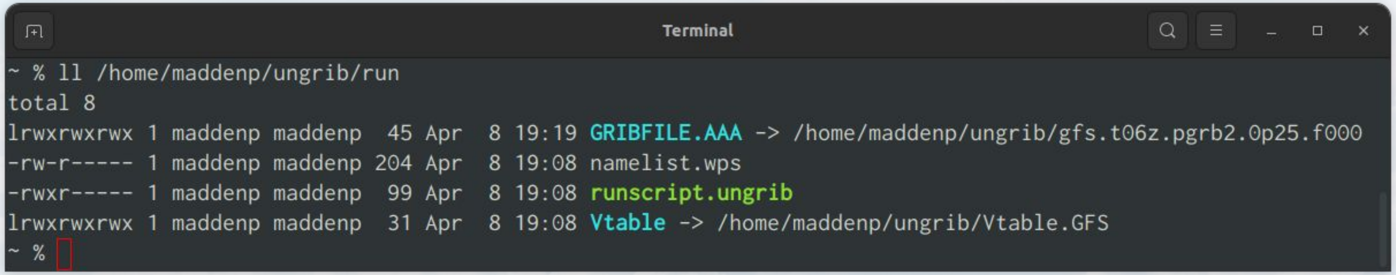
Graph-Oriented Drivers: Demo – Task Completion

Properties demonstrated:

- **Iteration:** Progress to task completion was made.
- **Idempotence:** Only work left incomplete by the previous invocation was performed.

```
% uw ungrib provisioned_run_directory --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib provisioned run directory: Initial state: Pending
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Initial state: Pending
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Checking requirements
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Requirement(s) ready
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Executing
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/GRIBFILE.AAA: Final state: Ready
[2024-04-08T19:19:29]     INFO 20240408 06Z ungrib provisioned run directory: Final state: Ready
```

Graph-Oriented Drivers: Demo – Steady State



```
~ % ll /home/maddenp/ungrib/run
total 8
lrwxrwxrwx 1 maddenp maddenp  45 Apr  8 19:19 GRIBFILE.AAA -> /home/maddenp/ungrib/gfs.t06z.pgrb2.0p25.f000
-rw-r----- 1 maddenp maddenp 204 Apr  8 19:08 namelist.wps
-rwxr----- 1 maddenp maddenp  99 Apr  8 19:08 runscript.ungrib
lrwxrwxrwx 1 maddenp maddenp  31 Apr  8 19:08 Vtable -> /home/maddenp/ungrib/Vtable.GFS
~ %
```

```
% uw ungrib provisioned_run_directory --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-08T19:26:54]     INFO 20240408 06Z ungrib provisioned run directory: Initial state: Pending
[2024-04-08T19:26:54]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-08T19:26:54]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-08T19:26:54]     INFO 20240408 06Z ungrib provisioned run directory: Final state: Ready
```

Powered By GSL

Graph-Oriented Drivers: Demo – Self-Healing

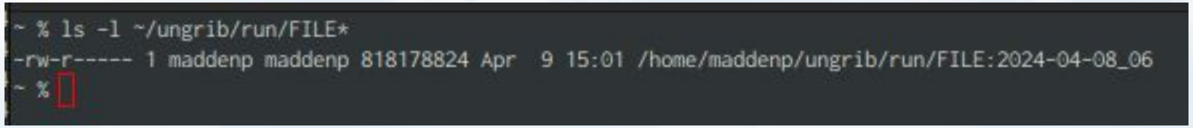
If an asset is removed, the driver can be re-run to create it again. This might include a change to the YAML config to obtain an updated asset (namelist file in this case).

```
% rm /home/maddenp/ungrib/run/namelist.wps
% uw ungrib provisioned_run_directory --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-08T20:52:12]     INFO Validating config against internal schema ungrib
[2024-04-08T20:52:12]     INFO 0 UW schema-validation errors found
[2024-04-08T20:52:12]     INFO Validating config against internal schema platform
[2024-04-08T20:52:12]     INFO 0 UW schema-validation errors found
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib provisioned run directory: Initial state: Pending
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Initial state: Pending
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Checking requirements
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Requirement(s) ready
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Executing
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib /home/maddenp/ungrib/run/namelist.wps: Final state: Ready
[2024-04-08T20:52:12]     INFO 20240408 06Z ungrib provisioned run directory: Final state: Ready
```

Powered By GSL

Graph-Oriented Drivers: Demo – Complete Run

```
% uw ungrib run --cycle 2024-04-08T06 --config-file config.yaml
[2024-04-09T15:01:20]     INFO Validating config against internal schema ungrib
[2024-04-09T15:01:20]     INFO 0 UW schema-validation errors found
[2024-04-09T15:01:20]     INFO Validating config against internal schema platform
[2024-04-09T15:01:20]     INFO 0 UW schema-validation errors found
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib run: Initial state: Pending
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib run: Checking requirements
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib run via local execution: Initial state: Pending
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib run via local execution: Checking requirements
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib provisioned run directory: Checking requirements
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib GRIB files: Checking requirements
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib run via local execution: Requirement(s) ready
[2024-04-09T15:01:20]     INFO 20240408 06Z ungrib run via local execution: Executing
[2024-04-09T15:01:20]     INFO Executing: runscript.ungrib >runscript.ungrib.out 2>&1
[2024-04-09T15:01:20]     INFO   in /home/maddenp/ungrib/run
[2024-04-09T15:01:33]     INFO 20240408 06Z ungrib run via local execution: Final state: Ready
[2024-04-09T15:01:33]     INFO 20240408 06Z ungrib run: Final state: Ready
```



```
~ % ls -l ~/ungrib/run/FILE*
-rw-r----- 1 maddenp maddenp 818178824 Apr  9 15:01 /home/maddenp/ungrib/run/FILE:2024-04-08_06
~ %
```

Powered By GSL

Summary

uwtools is a tangible Python package that you should install today to help you with workflow tasks.

The tools are designed to help scientists be productive, and to help applications increase the automation of common tasks.

Graph-oriented drivers are more flexible than many existing solutions that are often hard-coded, procedural, and/or brittle.

Helps with hierarchical testing for research, and reliability and recovery in real time use cases

Powered By GSL

Upcoming Plans

- Integrate uwtools into existing applications
  - Short Range Weather
  - Land DA
  - Seasonal Forecast System (SFSv1.0)
- Build more drivers
  - FV3 global and coupled; regional is available now
  - JEDI
  - Preprocessing
  - UPP
- Build more generic tools
  - Moving files across platforms
  - Interfacing with ecFlow

Acknowledgements

This research was supported in part by

- NOAA cooperative agreement NA22OAR4320151
- An FY22 WPO JTTI award: *Implementation of a Unified Workflow for the Unified Forecast System (UFS) Short Range Weather (SRW) Application*
- The OCIO Software Engineering for Novel Architectures program award to NOAA GSL
- The EPIC Contract
- The NOAA R2O Project
- In-kind contributions from NOAA EMC, NOAA GSL, and George Mason University

Powered By GSL