Adding a Workflow Component

HWRF Python Scripts Training

College Park, MD

January 22, 2016

Goals of exercise

- Become familiar with adding a workflow component
- Understand interaction between scripts and Rocoto workflow manager
- Consider dependency changes necessary for adding new components
- Demonstrate the basics, not generate a new HWRF component (large undertaking) or make things overly complicated

Things to Consider When Adding a Task

- How much Python scripting is needed for your workflow addition?
  - Does it need it's own set of utilization scripts in ush?
  - How about new produtils functionality?
- What can you leverage from what already exists at all levels?
- How does it change data input/output of other tasks?
- Does it need its own object defined in hwrf_expt.py?
- Does it need configuration options?
  - A whole new section or just a new variable in an already existing section
- Should you be able to turn it on/off? (Answer is nearly always yes for HWRF)

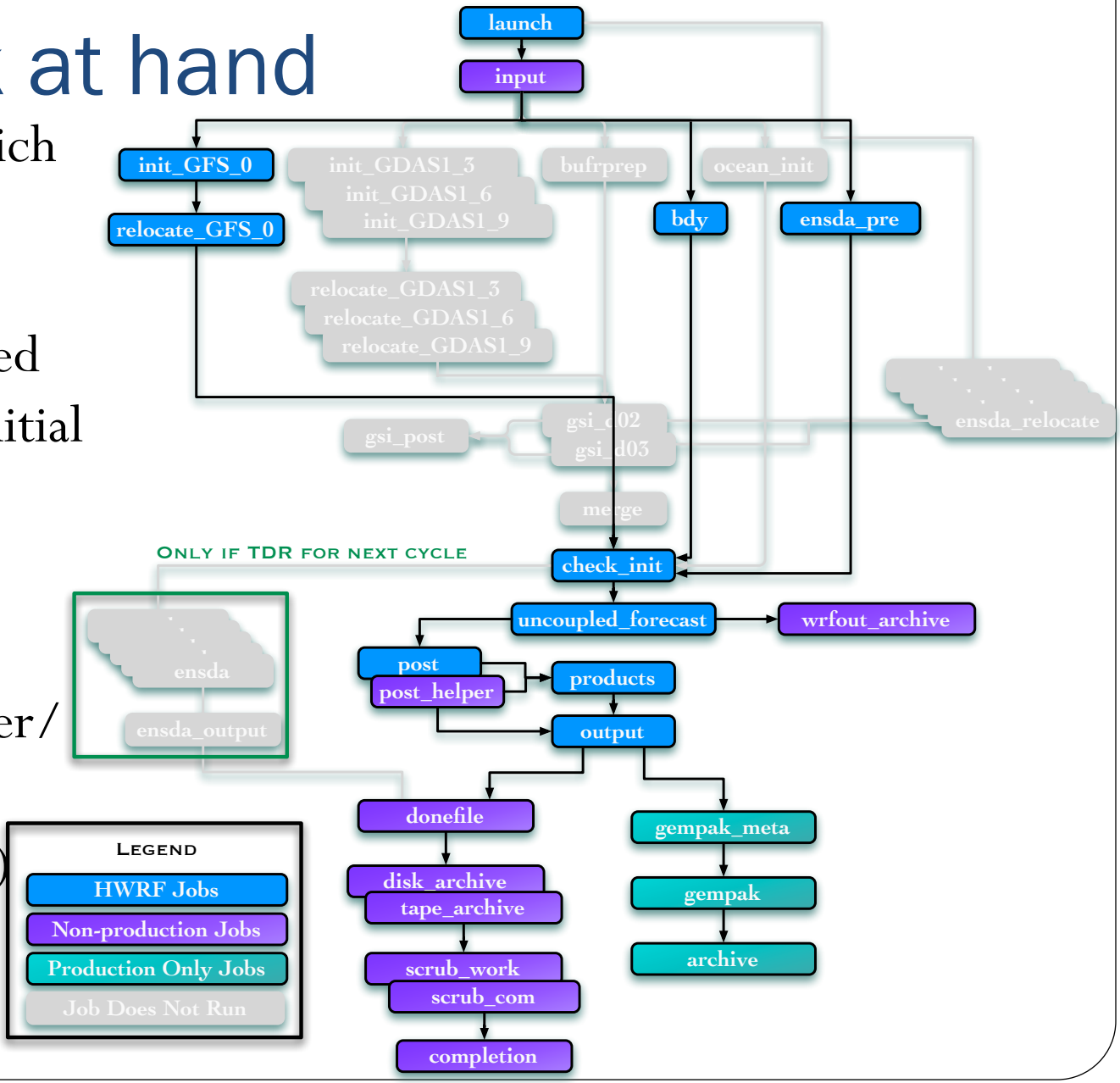
You are not alone!!!!

Bring your idea, concerns, questions up to the HWRF Developers Committee!

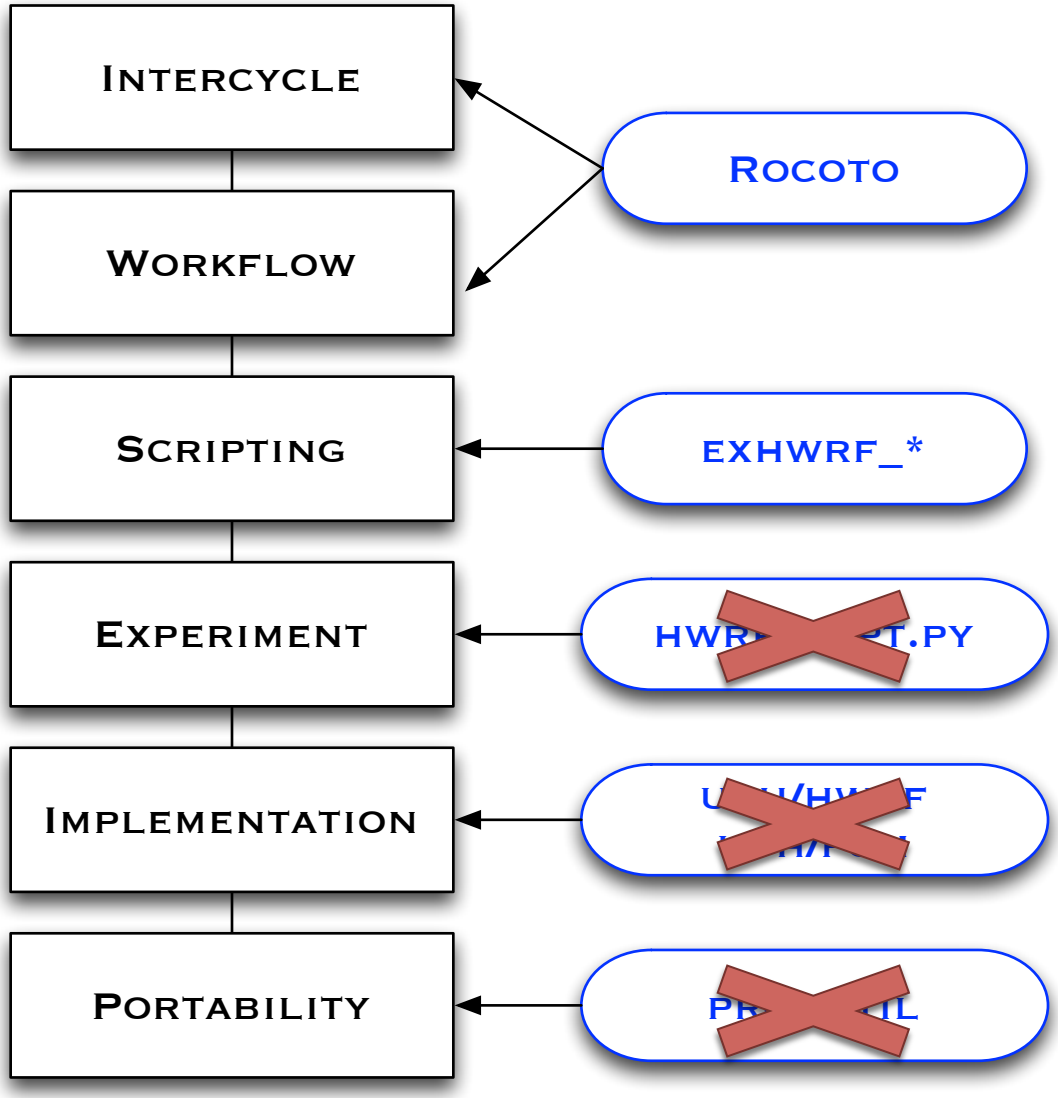
We'll help!

The task at hand

- Configuration in which we turn off most components and initialize an uncoupled forecast from GFS initial conditions
- Remove the NHC Products from the products task (gribber/copier/tracker stay grouped as products)



What ingredients do we need?



Since we are working with HWRF components that already and run by default, we don't need to define them in the layers below the experiment level

The approach

- Currently have a single script that handles gribber, copier, tracker, and NHC products
- The Rocoto workflow handles the submission of the script by submitting it as one batch job that handles all the work
- A simple way to accomplish the goal is to create 2 scripts and have Rocoto submit two batch jobs
  - One for NHC Products
  - One for all the other products
- Other solutions are possible, but are more advanced
  - Call the same script multiple times from Rocoto with different arguments/environment variables to specify which set of tasks to run (similar to exhwrf_init.py)
  - Make the option to run either together or separately configurable

Scripts

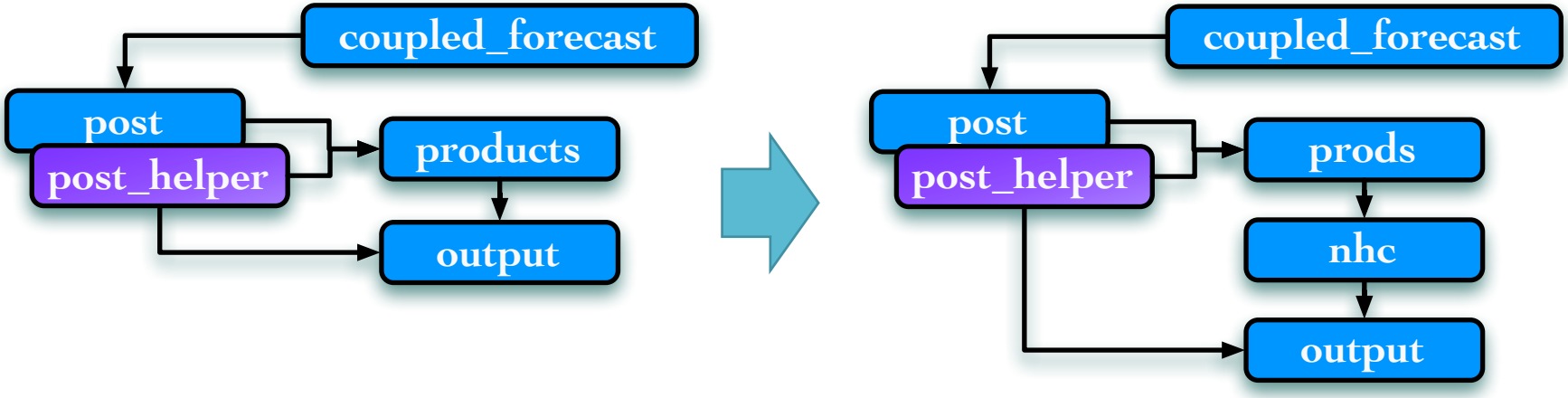
- exhwrf_products.py is responsible for running all the tasks
- Create two new scripts
  - exhwrf_nhc.py – includes only the products() function and the run methods from exhwrf_products.py
  - exhwrf_prods.py – includes everything else from exhwrf_products.py

Rocoto Components

- Create Rocoto ent files for submitting the new scripts in rocoto/tasks
  - Can use the old products.ent file as a template for creating two new task files named prods.ent and nhc.ent
  - New components need to be added as XML entities to the tasks/all.ent file
- Add new components to the workflow template rocoto/hwrf_workflow.xml.in
  - Won't run if the new task aren't in the workflow!

Consider Dependencies

- Products depends on Post job either running or completed
- The NHC products, however, runs after all of the other "products" internal jobs have finished
  - NHC Products depends on new prods job completing
  - Output now depends on NHC Products completing



General Guidelines

- Define the problem
- Add necessary ush/produtils/scripts for running the component
- Define new objects in hwrf_expt.py if needed
- Set new configuration options if needed
- Add Rocoto tasks ent files
- Add tasks to workflow template
- Test to make sure things work (and other things didn't break)