

Getting Started with HWRF Development

Christina Holt & Ligia Bernardet

22nd September 2014

HWRF Developers meeting





Repository Code Management

The repositories for the HWRF and other relevant components serve the important purpose of maintaining a unified code and for transitioning development to the operational centers

The community uses the same code as the operational centers

Everybody has access to all developments

This is meant to make your life easy! We're here to help.



Outline

- Computing resources
- Checking out the code
 - What you get
 - Where you get it
 - What to expect the first time
- Building HWRF
 - System requirements
 - Installing
- Running HWRF
- Development procedures



Computing Resources

- HFIP PIs can apply for accounts/projects on NOAA's Jet
 - Follow instructions at <https://rdhpcs-s.noaa.gov/acctmgmt>
 - Let Robert Gall (robert.gall@noaa.gov) know you're applying
 - Contact Nysheema Lett (Nysheema.Lett@noaa.gov) for a NOAA email address if you don't have one
 - Jet Questions go to Jet Help Queue (rdhpcs.jet.help@noaa.gov)
- If you need help determining the amount of resources to ask for, please email Christina or Ligia



Checking out HWRF

The DTC community HWRF system is available to checkout.

- To checkout the “top of the trunk” code use the command:
`svn co https://svn-dtc-hwrf.cgd.ucar.edu/trunk HWRF`
- This creates a top level directory called HWRF/.

```
exec  
graphics  
jobs  
kick_scripts  
nwport  
parm  
README  
README.FGAT_JOBS  
README.fix  
README.rocoto  
README.gsi  
rocoto  
scripts  
sorc  
test  
testfile  
ush  
wrappers
```

New for everybody this
year!
Python scripts & Rocoto
workflow



SORC Directory Structure

- Notice that there are no component directories.
- The SVN “externals” functionality populates this directory with the component directories on checkout.
- The file `.externals` contains:

```
gfdl-vortextracker https://svn-dtc-gfdl-vortextracker.cgd.ucar.edu/branches/HWRF
hwrp-utilities https://svn-dtc-hwrp-utilities.cgd.ucar.edu/branches/HWRF
ncep-coupler https://svn-dtc-ncep-coupler.cgd.ucar.edu/branches/HWRF
pomtc https://svn-dtc-pomtc.cgd.ucar.edu/branches/HWRF
UPP https://svn-dtc-unifiedpostproc.cgd.ucar.edu/branches/HWRF/
WPSV3 https://svn-wrf-wps.cgd.ucar.edu/branches/HWRF/
WRFV3 https://svn-wrf-model.cgd.ucar.edu/branches/HWRF/
```



Additional Components

- To get GSI, you must check it out separately

```
cd src/
```

```
svn co https://gsi.fsl.noaa.gov/svn/comgsi/trunk GSI
```

```
~~ OR ~~
```

```
svn co https://svnemc.ncep.noaa.gov/projects/gsi/trunk EMCGSI
```

The EMC and Community GSI repos are temporarily out of sync, so Community Users are encouraged to grab a compiled copy (specific to Jet) of the EMC GSI from disk

- HYCOM also requires an additional checkout

```
cd src/
```

```
svn co https://svn-dtc-hycom.cgd.ucar.edu/trunk HYCOM
```

- Experimental component, not supported by DTC or EMC



Checkout: Passwords & Usernames

- The first time each repository is accessed on a particular machine, SVN will prompt you for your username and password information.
- You will have at least two username/password combinations. One for GSI and another for the remaining repositories.



Checkout: Passwords & Usernames

When checkout for the first time on a particular machine, SVN will prompt you with a message like this:

```
Error validating server certificate for 'https://svn-dtc-hwrf.cgd.ucar.edu:443':
```

- The certificate is not issued by a trusted authority. Use the fingerprint to validate the certificate manually!
- The certificate hostname does not match.
- The certificate has expired.

```
Certificate information:
```

- Hostname: localhost.localdomain
 - Valid: from Thu, 21 Feb 2008 06:32:25 GMT until Fri, 20 Feb 2009 06:32:25 GMT
 - Issuer: SomeOrganizationalUnit, SomeOrganization, SomeCity, SomeState, --
 - Fingerprint: 86:01:bb:a4:4a:e8:4d:8b:e1:f1:01:dc:60:b9:96:22:67:a4:49:ff
- ```
(R)eject, accept (t)emporarily or accept (p)ermanently?
```

**Type “p”**



# Checkout: Passwords & Usernames

**Next you will be asked for your password.**

```
Authentication realm: <https://svn-dtc-
hwrp.cgd.ucar.edu:443> dtc:hwrp
Password for 'stark':
```

**If the default username is most likely not correct –just hit return & enter the correct username:**

```
Authentication realm: <https://svn-dtc-
hwrp.cgd.ucar.edu:443> dtc:hwrp
Username:
```

**Hit return again, and enter the appropriate password.**

```
Authentication realm: <https://svn-dtc-
hwrp.cgd.ucar.edu:443> dtc:hwrp
Username: stark@ucar.edu
Password for 'stark@ucar.edu':
```



# Compiling HWRF on Jet

Before you begin the compilation process, you must have the following modules loaded on Jet to use the build system:

```
module purge
module load intel
module load mvapich2
module load netcdf
module load pnetcdf
```

Refer to [HWRF Users' Guide v3.6a](#) if you would prefer to compile each component manually



# Compiling HWRF

```
GSI/ OR EMCGSI/
WRFV3/
WPSV3/
UPP/
pomtc/
ncep-coupler/
hwrp-utilities/
gfdl-vortextracker/
executables.lst
Makefile
build/
README
```

- In the source code directory, type make to compile all eight components:  
cd HWRF/src  
make
- This will take take some time.
- Once the components are compiled, complete the build with:  
make install
- Install places the executable in the common directory HWRF/exec/

after running `make install`

```
cp /mnt/lfs2/projects/hwrp-vd/Mingjing.Tong/GSI_HWRF/src/global_gsi HWRF/exec/hwrp_gsi
```



# HWRF Directory Structure

|                            |                                                                                   |
|----------------------------|-----------------------------------------------------------------------------------|
| <code>exec/</code>         | Component executables installed by <code>make install</code>                      |
| <code>kick_scripts/</code> | EMC run scripts (going away soon)                                                 |
| <code>jobs/</code>         | Mid-level job scripts used with <code>kick_scripts/</code> (also going away soon) |
| <code>nwport/</code>       | EMC utilities                                                                     |
| <code>parm/</code>         | Configure files to define an experiment                                           |
| <code>rocoto/</code>       | GSD workflow manager [ <a href="#">Documentation</a> ]                            |
| <code>scripts/</code>      | High-level Python scripts                                                         |
| <code>sorc/</code>         | Source code for each component                                                    |
| <code>ush/</code>          | Low-level Python scripts                                                          |
| <code>wrappers/</code>     | Community scripts to run each HWRF component by hand                              |



# Running HWRF

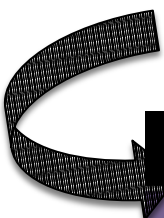
- Link the fix files
  - README.fix explains, and is always up to date with the latest version of EMC GSI
- More information on the Python re-write is here:
  - <https://wiki.ucar.edu/display/DTCHWRF/DTC+HWRF+Scripts+Home>
- Two options:
  - Wrapper scripts
    - Instructions are available in [HWRF Users' Guide v3.6a](#) (HWRF UG)
  - Rocoto
    - Documentation available here: <http://rdhpcs.noaa.gov/rocoto/>
    - More details for using with HWRF: `HWRF/README.rocoto`
    - Training can be provided at a later HWRF Dev. Meeting



# Summary

- You have now built the “top of trunk” version of the code.
- It is sufficient for running HWRF
- If you want to develop code, however, you would need a working branch where you can commit your developments
- You **should not commit to the trunk** without the

HWRF Developers Committee approval

A thick, black, curved arrow with a textured, fabric-like appearance points from the text above towards the table below.

| Name               | Organization | Email Address               |
|--------------------|--------------|-----------------------------|
| Ligia Bernardet    | DTC          | Ligia.Bernardet@noaa.gov    |
| Christina Holt     | DTC          | Christina.Holt@noaa.gov     |
| Vijay Tallapragada | NCEP/EMC     | Vijay.Tallapragada@noaa.gov |
| Sam Trahan         | NCEP/EMC     | Samuel.Trahan@noaa.gov      |



# Developers Committee

- Telecon on Mondays at noon (ET)
- Forum for discussion on plans and updates for development
  - Including testing, evaluation, and technical aspects
- We will send out an agenda each Friday before the Monday meetings
  - Time left for open discussion
  - Let us know if you'd like to add an agenda item
- [hwrf\\_developers@rap.ucar.edu](mailto:hwrf_developers@rap.ucar.edu) is the mailing list for exchanging information about HWRF development





# Support/Communication

- A number of email lists exist for receiving information about component development
  - It is up to the users to add themselves to the WRF developers mailing list
    - Go to [mailman.ucar.edu/mailman/listinfo/wrf-developers](http://mailman.ucar.edu/mailman/listinfo/wrf-developers)
  - Most others come to you by default
- Developers may request to be removed from any of these email lists at any time.
- These are not discussion lists, but inform users of commits to SVN repository



# Development Branches

- Once you have a development branch, you should commit regularly (svn commit)
- HWRF development continues in parallel
- Keep your branch up to date with trunk (svn merge)
- Your regular updates will prevent you from diverging too far from the trunk
- Integration of your developments is more streamlined
- Transition to operations is smoother



# Examples for Merging

- Slides 22-32 provide more information on the repository structure and an example of merging the trunk (or branches/HWRF if an external repository) into your personal branch
- Great resource for svn: <http://svnbook.red-bean.com/>

# Pushing Development Back onto the Trunk

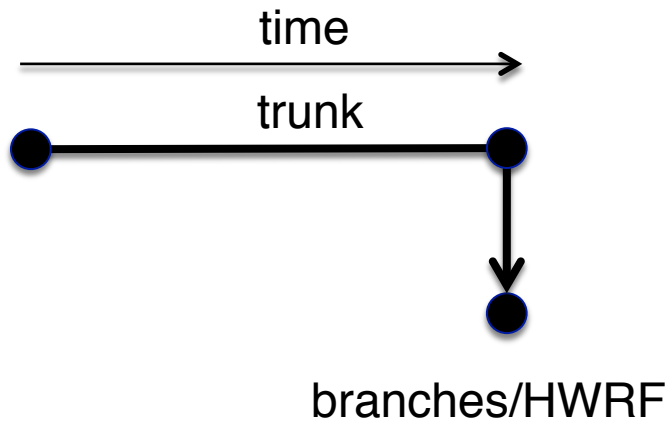


- Keep in touch!
  - Let us know what you're planning to develop
  - We can help you make better development decisions to make the transition easier, and keep you from recreating the wheel
  - As your developments are happening, keep us in the loop
- Don't let your code diverge too far
  - Working on a copy of code that is a year old and then trying to merge up to the trunk is VERY discouraged!
- When you've made incremental changes that should go into the trunk of the HWRF repo, let us know!
  - The HWRF Developers Committee should be notified by email of all planned changes
  - Once changes are approved, we can discuss the procedures for regression tests and merging up to the trunk



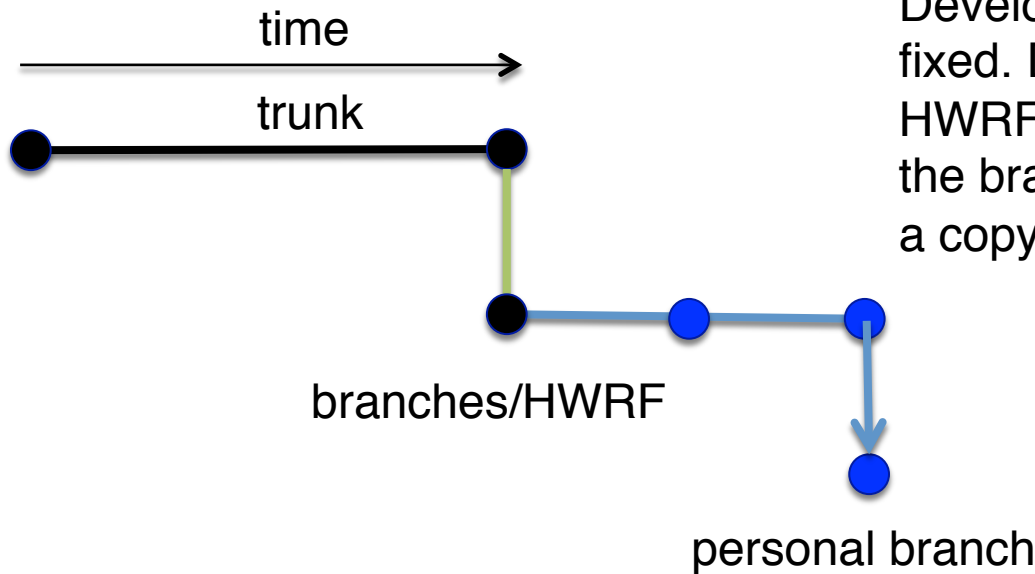
Questions?

# Creation of branches/HWRF



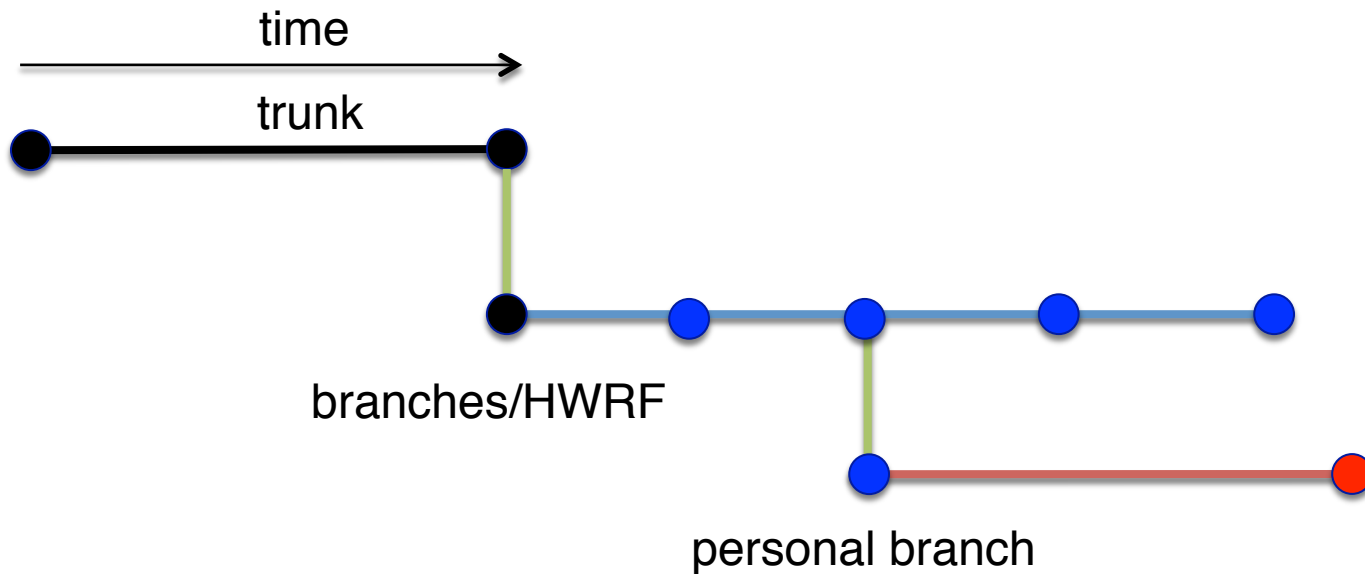
At some point in the trunk's time evolution, a branch called HWRF is created. At this creation, both the branch and the trunk are identical.

# Creation of personal branch



Development on the trunk is held fixed. Development on the branch HWRF continues. At some point in the branch HWRF's time evolution, a copy is made for a personal branch.

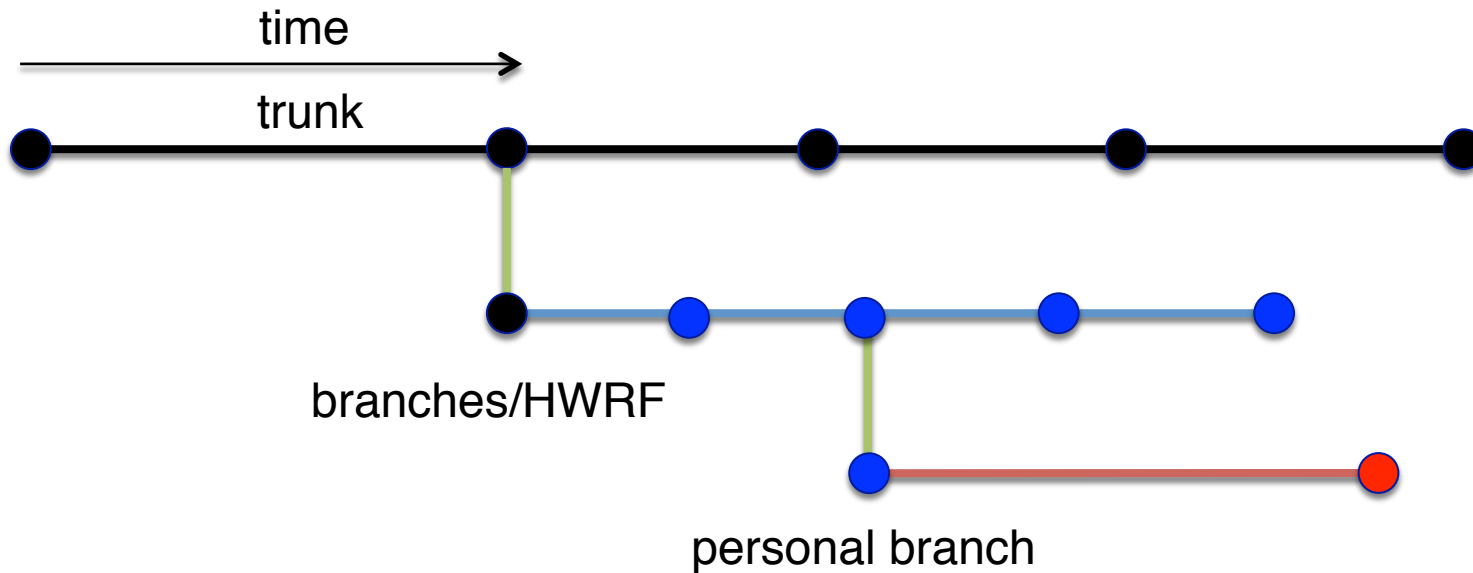
# Parallel development



Development continues independently on the branch HWRF and the personal branch. Notice that you may have multiple commits to the branch HWRF from other developers, while only a single commit is made to the personal branch. The two branches have diverged.

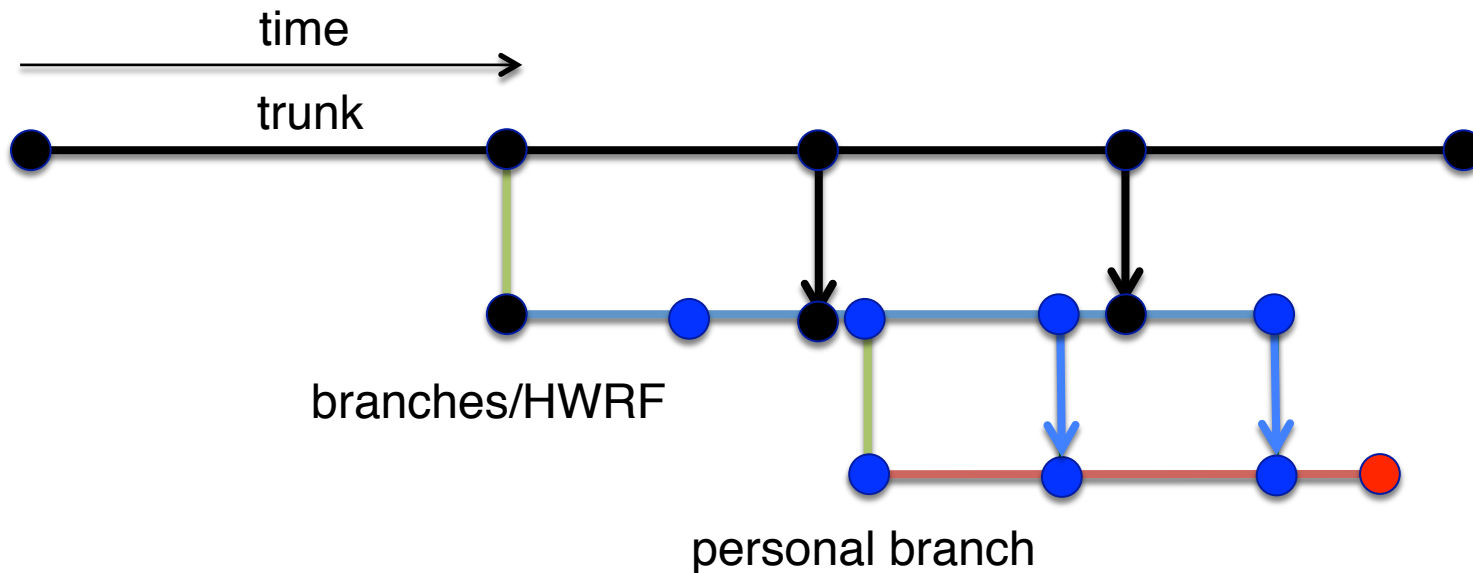


# More Parallel development



In the case of external components such as WRF, UPP, WPS, etc., additional development is almost guaranteed to occur on a regular basis to the trunk. Now all three (trunk and two branches) have diverged.

# Merging



- DTC is responsible for updating branches/HWRF (black arrows)
- Developers are responsible for updating personal branches (blue arrows)
- Start by merging everything new in branches/HWRF into your personal branch. Resolve any conflicts that arise from the merge, and test.



# Updating your branch

- This recipe covers updating a personal branch of one of the components in the src directory.
- It does not cover updating the top level HWRP directory such as the scripts and parm directory. This is slightly different and will be covered later.
- Two different types of updates:
  - Updating your branch and your working copy
    - Update your working copy:
      - `cd HWRP && svn update (--ignore-externals)`
      - Update your branch with your working copy:
        - `cd HWRP && svn commit .`
    - Updating your branch with branches/HWRP



# Updating Branches

There may come a time during your development, where you need to include some update to the code from another developer. We will start with an outline of the simple method for doing this, and work through an actual case later.

1. Save your work
  1. Be certain your work is at a stable point.
  2. Go into each of the directories where you've been working and run an `svn commit` at the top of that directory.
  3. This will commit all of your work to your personal branch for that component.



# Updating Branches

2. Checkout a fresh copy of your top level branch
  1. `svn co https://svn-dtc-hwrf.cgd.ucar.edu/branches/personal_branch_name`
  2. This will automatically update the code in any component where you **don't** have a personal branch
3. Manually update your personal branches by from branches/HWRF back onto your personal branch.
  1. `svn merge http://component/branches/HWRF .`
  2. Repeat with all components that have personal branches.



# Updating Branches - Example

Now lets consider a specific example.

- Suppose we have a user named Ligia with a personal branch of the top level HWRF repo called `ligia`.
- She has personal branches for the two components WRF and hwrf-utilities.
- Since checking out her current copy of `HWRF / branches / ligia`, she has done development only in the WRF component.
- Since then, Sam has updated the top HWRF trunk, and the UPP and WRF components. Ligia needs those updates for her own work.



# Updating Branches - Example

Ligia conducts the following steps to update her branch.

1. Save her work.

1. `cd ligia/sorc/WRFV3`
2. `svn commit`

2. Checkout a new copy of [HWRF/branches/ligia](https://svn-dtc-hwrf.cgd.ucar.edu/branches/ligia)

1. `svn checkout`  
<https://svn-dtc-hwrf.cgd.ucar.edu/branches/ligia>
2. This checkout updates the top of the HWRF branch, and all of the components in which Ligia doesn't have personal branches, (e.g. everything but WRF and hwrf-utilities).



# Updating Branches - Example

Ligia conducts the following steps to update her branch.

3. Update the remaining components (.e.g. WRF & hwrp-utilities) by merging in from [branches/HWRF/](https://svn-wrf-model.cgd.ucar.edu/branches/HWRF/).
  1. `cd ligia/sorc/WRF`
  2. `svn merge`  
<https://svn-wrf-model.cgd.ucar.edu/branches/HWRF/>
  3. `svn commit`
  4. `cd ../hwrp-utilities`
  5. `svn merge`  
<https://svn-dtc-hwrp-utilities.cgd.ucar.edu/branches/HWRF/>
  6. `svn commit`
  7. Always merge from [branches/HWRF/](https://svn-wrf-model.cgd.ucar.edu/branches/HWRF/)