HAFS Code Management and Community Engagement

Evan Kalina[1,2,3] and Bin Liu[4]

...with contributions from the HAFS application team and community

[1]NOAA/GSL, [2]CU/CIRES, [3]DTC, [4]IMSG at NOAA/NWS/NCEP/EMC

HAFS Code Management Goals

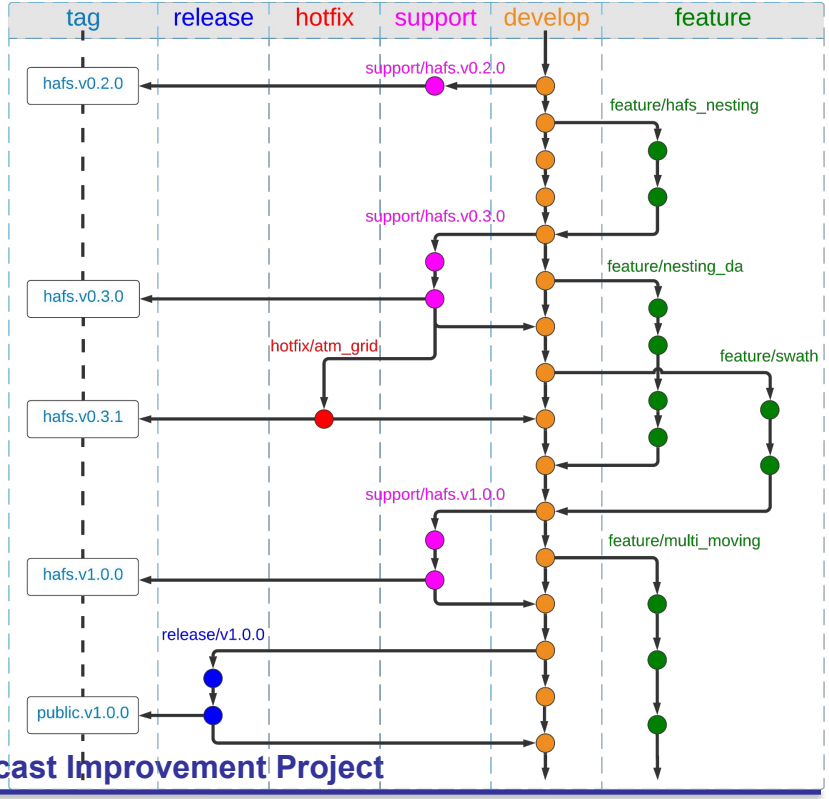
- Ensure latest HAFS developments are available to the community.
- Provide a standard, easy-to-follow set of guidelines for the community to contribute to HAFS development.
- Detect and fix bugs before they reach the main HAFS development branch.
- Minimize time spent reviewing pull requests.
- Engage in code management and development practices that are consistent with other UFS applications and submodules.

HAFS Code Repository Structure

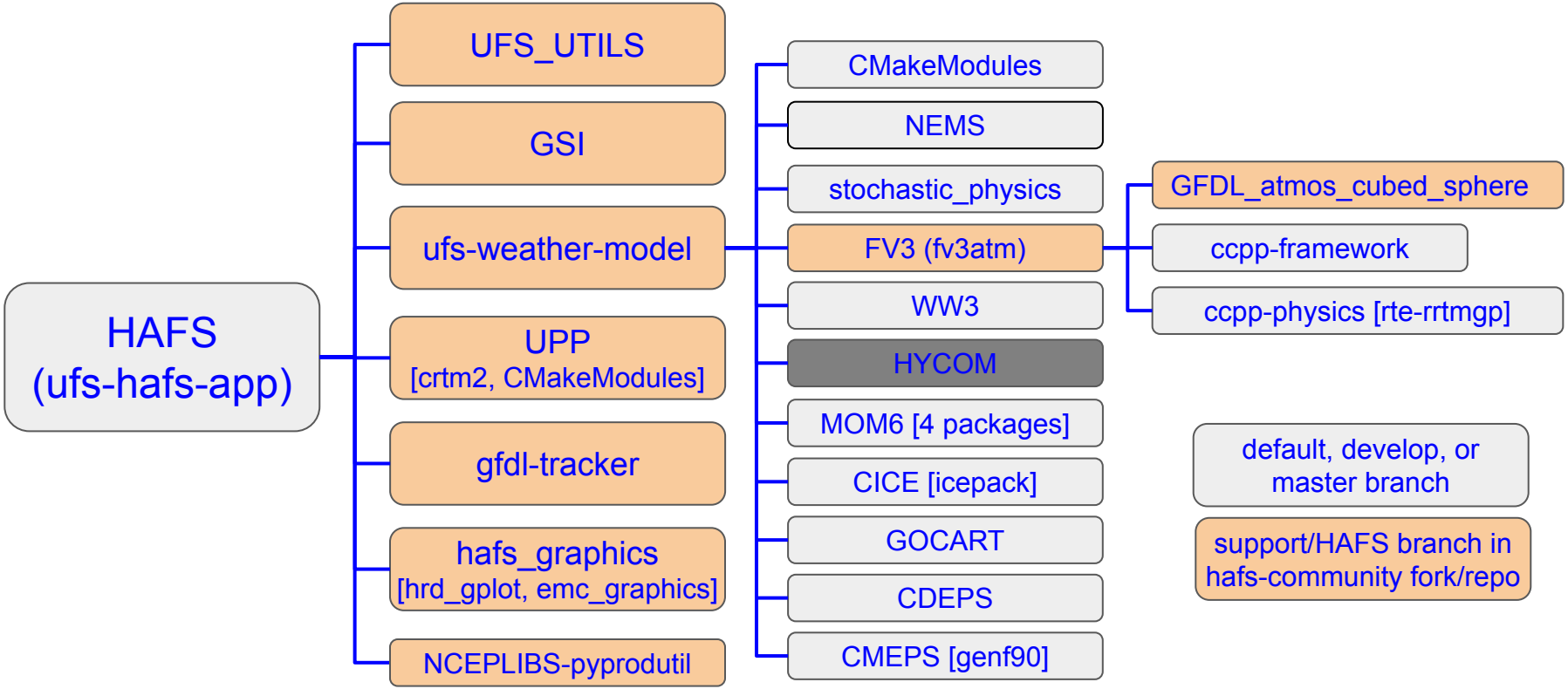
The authoritative HAFS repository:

- https://github.com/hafs-community/HAFS
- Maintained by EMC and DTC with community/application team contributions
- HAFS forking/branching/tagging rationale
  - **develop**: the main development branch
  - **support/[name]**: branches used by operations or real-time parallels
  - **release/vx.x.x**: public release branches
  - **tag**: **hafs.v0.2.0**, **hafs.v1.0.0**, **public.v1.0.0**, etc.
  - **hotfix/[name]**: temporary bug fix branches
  - **feature/[name]**: feature branches for adding major new capabilities or enhancements
    - Small/individual efforts should fork the repository

Example HAFS Branching Diagram



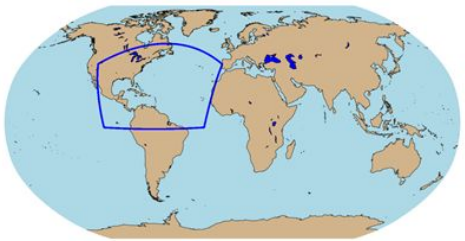
HAFS Submodules



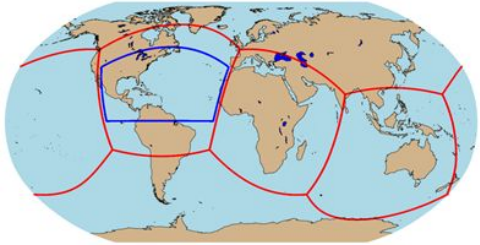
HAFS Workflow: Current Capabilities

- The current HAFS workflow is based on:
  - HWRF/HMON workflows
  - FV3CAM regional workflow
  - FV3GFS global workflow
- Supports both on-demand triggering and continuous cycling
- Supports both standalone regional and global-nesting configurations
- Supports both basin-focused (static) and storm-focused (relocatable) configurations

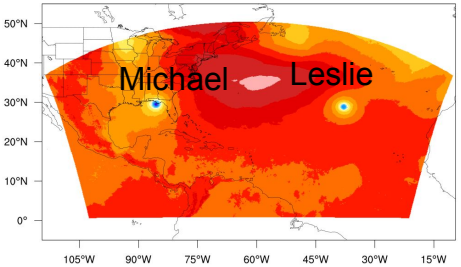
regional



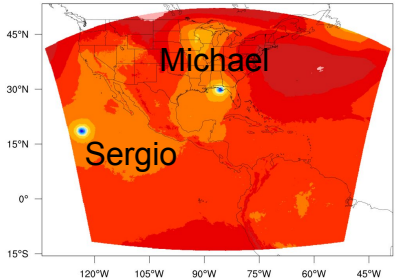
global-nesting



basin-focused



storm-focused

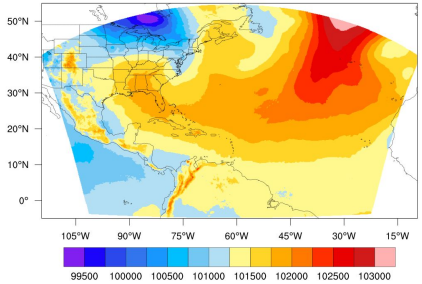


2018100718Z F066

HAFS Workflow: Current Capabilities

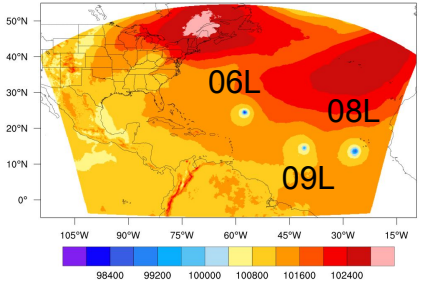
- Supports zero-storm (TC genesis, or any other regional/global-nesting application), one-storm, and multiple-storms
- Supports C768 as well as other resolutions (C96, C192, C384, C1152, C1536, etc) for both regional and global-nesting configurations
- Includes TC specific pre-processing and post-processing workflow elements
- Supports WCOSS (Cray and Dell) as well as NOAA RDHPCS (Jet, Hera, Orion)

zero-storm

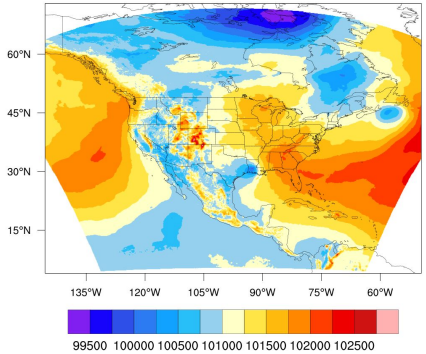


2019062512Z

multiple-storm

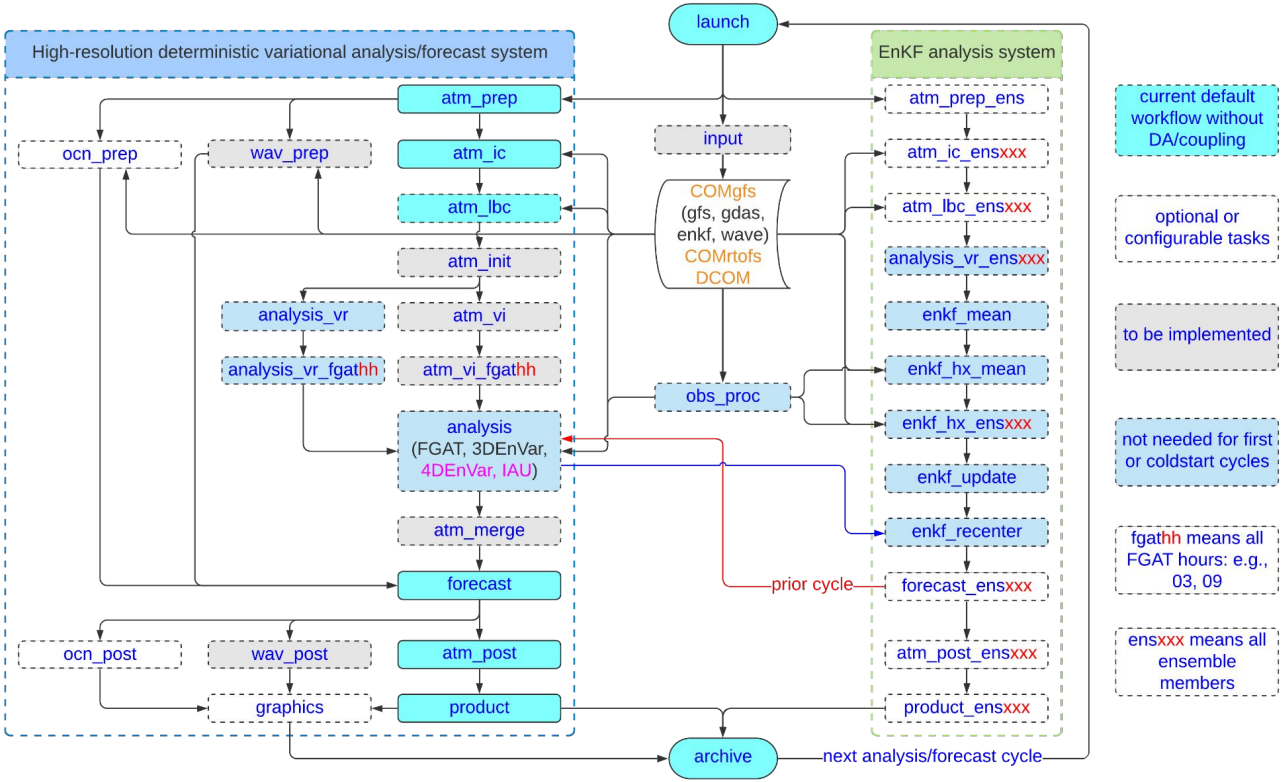


2018091000Z
Florence, Helene, Isaac



2019071400Z
for a CONUS domain

HAFS Workflow (with DA and coupling)



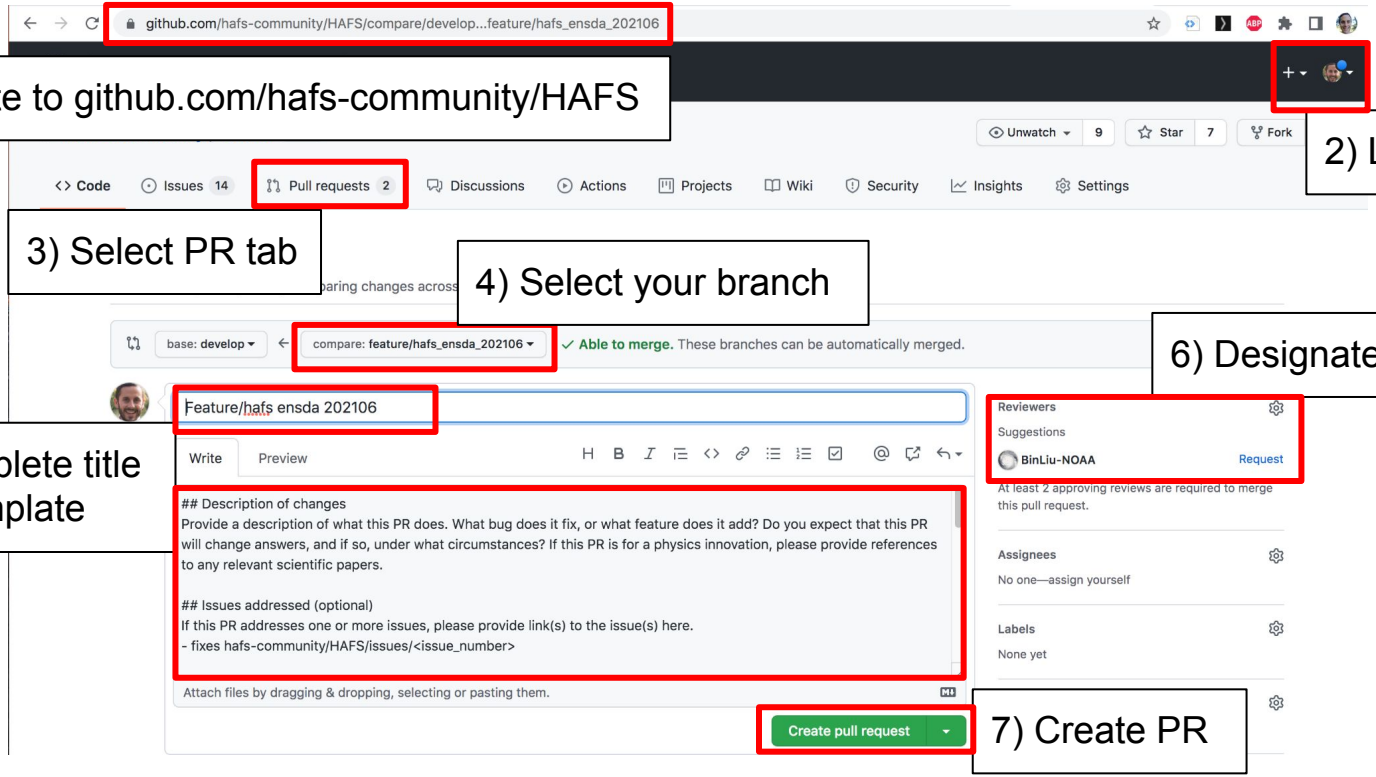
How to Contribute to HAFS Development

- The governance policy explains how the community can contribute.
  - Before starting development
    - Discuss plans during the biweekly HAFS developers meeting
    - Create a new issue in the repository
    - Either create a new feature branch in hafs-community (for collaborative efforts) or fork the repository (for individual work).
      - Applies to submodules as well
      - Start from develop or support/HAFS branch (for submodules)

HAFS Code Repository Governance

- During development
  - Commit your work back to the repository in manageable chunks
  - Provide meaningful commit messages
  - Keep your fork(s) or branch(es) synced with develop, support/HAFS
- Submitting Pull Requests (PRs)
  - Sync first
  - Retest and ensure your development is working
  - Fill out the PR template on GitHub and submit
  - DTC and EMC staff will run regression tests

Submitting a Pull Request



1) Navigate to github.com/hafs-community/HAFS

2) Log in

3) Select PR tab

4) Select your branch

5) Complete title and template

6) Designate reviewers

7) Create PR

Resources for HAFS Developers

- See the HAFS User Quick Start and HAFS Developers Guide.
- Documentation for using CDEPS in HAFS is also available.
- We are working on unifying this documentation under Read the Docs.
- Repository governance is in the hafs-community/HAFS Wiki.
- The GitHub discussions page under hafs-community/HAFS is actively monitored.
- Join the next biweekly HAFS Developers call on Monday, Nov. 29th at 12 ET.
  - Email evan.kalina@noaa.gov to be added to the calendar invite.