

Collaborative Tools



▶ **Agile Project Management and Collaborative Workflow**

- ◆ **git/GitHub**
- ◆ **git-flow**
- ◆ **ZenHub**

▶ **Documentation**

- ◆ **Sphinx/ReadTheDocs** (*high-level manuals, how-to's, etc*)
- ◆ **Doxygen** (*low-level code details*)
- ◆ **JEDI Wiki**

Mark Miesch (JCSDA)

And the JEDI Core Team

JEDI Academy - 10-13 June, 2019

Boulder, CO



The Way of a JEDI



▶ Collaborative

◆ A Joint Center (**JCSDA**)

- **Partners, collaborators, stakeholders, community**

◆ A Joint Effort (**JEDI**)

- **Distributed team of software developers, with varying objectives and time commitments**

▶ Agile

◆ Innovative

◆ Flexible (*future-proof*)

◆ Responsive to users and developers

◆ Continuous delivery of functional software

Part I: Agile Tools



▶ **git/GitHub**

- ◆ **Version control**
- ◆ **Enhancements and bug fixes immediately available to distributed community of developers**
- ◆ **Code review, issue tracking**
- ◆ **Community exports (Code distribution)
...and imports (ecbuild, eckit, fckit)**

▶ **Git-Flow**

- ◆ **Innovation**
- ◆ **Continuous Delivery**

▶ **ZenHub**

- ◆ **Agile project management**
- ◆ **Enhances GitHub's issue tracking and code review functionality**



Dashboard

GitHub, Inc. [US] | <https://github.com/orgs/JCSDA/dashboard>

Apps | JEDI | Software Engineering | Mac | Meetings | Outdoors | Garden | Transition | Colleges | Travel

Search or jump to... Pull requests Issues Marketplace Explore

JCSDA

Repositories [New repository](#)

Find a repository...

- JCSDA/ufo
- JCSDA/fv3-jedi
- JCSDA/ioda
- JCSDA/mpas
- JCSDA/jedi-docs
- JCSDA/ifu
- JCSDA/ifu-bundle
- JCSDA/soca
- JCSDA/ufo-bundle
- JCSDA/fv3-bundle
- JCSDA/oops
- JCSDA/crtm
- JCSDA/mpas-bundle
- JCSDA/docker
- JCSDA/docker_base
- JCSDA/bufr2nc
- JCSDA/fms
- JCSDA/fv3
- JCSDA/wrf-jedi
- JCSDA/singularity

Show more

Browse activity [View organization](#)

Recent activity

- Eliminated reference to UCAR repos and clarified FC environment varia...
JCSDA/jedi-docs · You opened this pull request
- Feature/refac obs opr
JCSDA/ufo · Your review was requested
- Feature/interp test bump
JCSDA/fv3-jedi · You opened this pull request

All activity

- xinzhang8noaa pushed to JCSDA/ufo 7 hours ago
 - 1 commit to `feature/refac_obs0pr`
eb35604 one can't override a nonaccessible deferred binding per F08...
- danholiday pushed to JCSDA/fv3-jedi 7 hours ago
 - 2 commits to `feature/radiancread`
0ed7934 working hofx for radiances
df854f5 towards bilin for crtms
 - 14 more commits »
- danholiday created a branch `feature/radian...` in JCSDA/ufo
 - JCSDA/ufo
Updated Apr 9
- xinzhang8noaa left 2 comments on pull request JCSDA/ufo#63 7 hours ago
 - xinzhang8noaa commented 7 hours ago
Googled following things: one can't override a nonaccessible deferred binding per F08/0052. (Intel Fortran doesn't yet recognize that.) Therefore, ...
- victordottir left 3 comments on pull request JCSDA/ufo#63 8 hours ago

**git - command line tool
(version control)**

**GitHub - Web-based
repository management
(branches, releases)**

**Changes to develop, master
branches handled via
pull requests**

GitHub Teams



JEDI · JCSDA Teams List

GitHub, Inc. [US] | <https://github.com/orgs/JCSDA/teams/jedi/teams>

Apps JEDI Software Engineering Mac Meetings Outdoors Garden Transition Colleges

[Learn more](#)

Find a team...

5 teams in the JEDI team		Members
JEDI-core		5 members 0 teams
JEDI-dev		3 members 0 teams
JEDI-models Models interfacing in JEDI		31 members 5 teams
FV3 Development		14 members 0 teams
LFRic JEDI-LFRic interfacing		7 members 0 teams
MPAS JEDI-MPAS interfacing		9 members 0 teams
Navy		4 members 0 teams
WRF Interfacing WRF with JEDI		2 members 0 teams
JEDI-obs Observation-related interfacing in JEDI		20 members 4 teams
UFO Unified Forward Operator development		18 members 3 teams
SOCA Sea-Ice Ocean Coupled Assimilation		13 members 0 teams

[Previous](#) [Next](#)

© 2018 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#) [Contact GitHub](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)

JEDI-models · JCSDA Discussion

GitHub, Inc. [US] | <https://github.com/orgs/JCSDA/teams/jedi-models/discussions>

Apps JEDI Software Engineering Mac Meetings Outdoors Garden

Search or jump to... Pull requests Issues Marketplace Explore

JCSDA JEDI JEDI-models Discussions Members 31 Teams 5 Repositories 17 Projects 0

JEDI-models
@JCSDA/jedi-models
Models interfacing in JEDI

1 member

[Unwatch](#)

Recent Pinned

Would a radiosonde with a single observation be useful for data assimilation interface tests?

MarekWasak
21 hours ago · edited -

That way we would be able to see the structure of B coming out in the analysis increments.

ytremolet 19 hours ago
Yes, as we develop more tests that should be included.

Reply...

Changes to IODA/UFO that affects all models running DA

danholdaway
3 days ago

Since the changes to UFO and IODA just merged everyone will need to replace "obsvalue": "ObsVal" with "obsvalue": "Observation" in your DA json files E.g. [JCSDA/fv3-jedi@7e88179](#) @ss421 @guillaumevernieres @byoung-joo

Show earlier comments

srherbener 2 days ago
@byoung-joo, @ytremolet

I got mpas-bundle to compile and I can see the test failure, but I'm not clear on how to resolve this. The problem is that the amsua_n19_obs.nc4 file says nobs = 12090 (860 records X 15 channels), and the amsua_n19_geovais.nc4 files says nobs = 806. I changed the obs reader for Radiance to record that nobs = 12090 and nlocs = 806 (one location per record that holds 15 obs, 1 for each channel). This was done so that nobs represents the number of unique observations and nlocs represents the number of unique locations. Also, this allows all 12090 observations to be placed into an ObsVector.

I can get the code to run past the interp_check routine by making the request to ObsSpace for the locations return every 15th location value (Lat, Lon, Time) which returns the 806 unique locations values. A down side of this "solution" is that you lose which locations are associated with which observations.

Unfortunately, the test crashes later on, perhaps because it thinks that nobs is 12090 and it's running past the end of the locations arrays (which are only 806 long), and the geovais arrays which are also 806 long. If I make it so that nlocs = 12090, then the association between obs and locations is preserved but the test still crashes supposedly because the geovais arrays are only 806 long.

Is it okay to use the "return every 15th location" solution for now, despite the need to assume that location 1 goes with obs 1 - 15, location 2 goes with obs 16 - 30, etc. The actual code for this solution figures out a step value based on nobs / nlocs so it automatically adjusts according to that ratio.

If we do go with this, something downstream is not right. Would this be as



Browser address bar: <https://github.com/JCSDA/ufo>

Repository: **JCSDA / ufo** Private

Unwatch 28 Star 0 Fork 1

Code Issues 33 Pull requests 1 ZenHub Wiki Insights Settings

JEDI Unified Forward Operator

Manage topics

454 commits 44 branches 0 releases 18 contributors Apache-2.0

Branch: develop New pull request Create new file Upload files Find file Clone or download

victordottir and **ytremolet** Bugfix for interpolation: set weights to 0/1 if the obs is outside of... Latest commit ac66a76 an hour ago

cmake	cmake clean-up	a year ago
docs	Adding in files for creating the "Building UFO in OS X" documentation. (9 months ago
src	Bugfix for interpolation: set weights to 0/1 if the obs is outside of...	an hour ago
test	Bugfix for interpolation: set weights to 0/1 if the obs is outside of...	an hour ago
tools	Feature/script fornewobs (#79)	17 hours ago
.gitattributes	Use git lfs	5 months ago
.gitignore	Feature/replace ad alloc (#77)	5 months ago
CMakeLists.txt	Feature/gnssro ropp1d forward (#67)	20 days ago
COPYING	First commit	a year ago
CPPI INT cfg	Feature/style check (#51)	2 months ago

git/GitHub (JEDI tips)



▶ **Work with JEDI bundles**

- ◆ **Clone bundle repo**
- ◆ **Let ebuild do the rest**
- ◆ **If that doesn't work, read the README file**
- ◆ **Get in the habit of running *make update* after ebuild**
- ◆ **Edit the CMakeLists.txt file to use your local version**

```
#ebuild_bundle( PROJECT ufo GIT "https://github.com/JCSDA/ufo.git" BRANCH develop UPDATE )  
ebuild_bundle( PROJECT ufo GIT "https://github.com/JCSDA/ufo.git" BRANCH feature/mystuff )
```

▶ **Cache your GitHub credentials**

```
git config --global credential.helper 'cache --timeout=3600'
```



▶ **LFS = Large File service**

- ◆ **Increases GitHub size limits for individual files from 100 MB to 2GB**
- ◆ **Cumulative storage purchased in 50 GB data packs**
- ◆ **Used for anything that isn't code (*data files, restart files, etc*)**

▶ **Transparent to the user**

- ◆ **When you push to GitHub, any files that are tracked by LFS will go to a remote server (the LFS Store)**
- ◆ **The GitHub repo will only contain a pointer to that file**
- ◆ **When you fetch/pull/clone an LFS-enabled repo from GitHub, LFS will check to see if you have the large files on your computer (local LFS cache). If not, it will retrieve them from the LFS Store as needed.**

Using Git-LFS



1) Extension to git

- ▶ brew install git-lfs

2) See if git-lfs is already enabled for that repo

- ▶ git lfs track

3) If not already sufficient, then add appropriate tracking patterns

- ▶ git lfs install **# only if step 2 returns nothing**
- ▶ git lfs track *.nc4

4) Add your large files to the repo

5) Make sure your files and patterns are tracked by git

- ▶ git add .gittattributes
- ▶ git add * **# new files**

6) commit, push, pull, fetch, clone and proceed as you would with any other repo

Git-Flow



Git Flow is:

▶ **A Philosophy**

- ◆ **Optimal for Agile Software Development**
 - **Innovation**
 - **Continuous Delivery**

▶ **A Working Principle**

- ◆ **Enforcement of branch naming conventions soon to come**

▶ **An Application (*extension to git*)**

- ◆ **Already installed in AMI and Singularity Container**
- ◆ **brew install git-flow-avh # (Mac)**
- ◆ **sudo apt-get install git-flow # (linux)**
- ◆ **<https://github.com/petervanderdoes/gitflow-avh>**

A state of mind,
git-flow is



The Git-Flow Manifesto

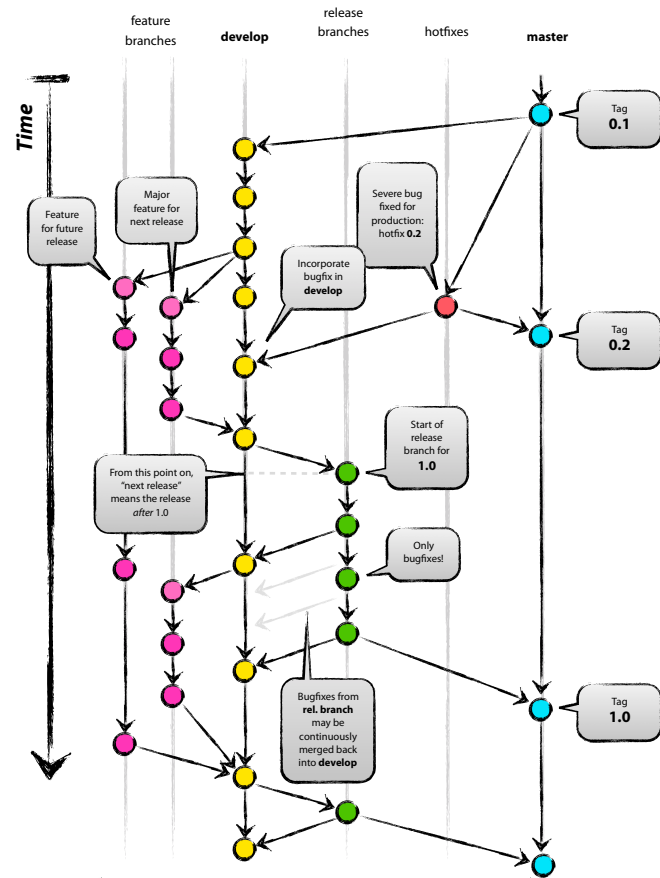


<http://nvie.com/posts/a-successful-git-branching-model/>

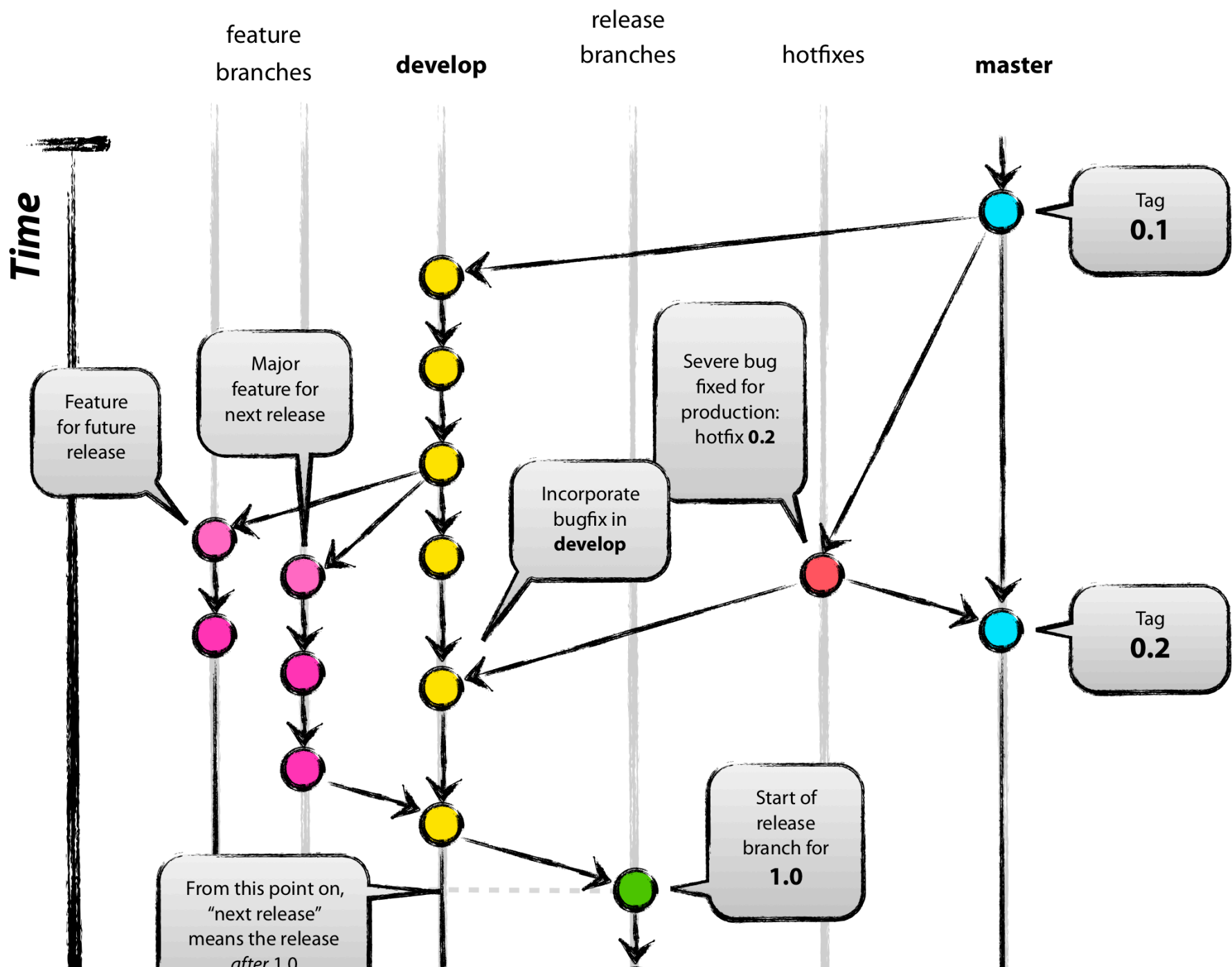
Vincent Driessen (2010)

git branching model

Highly Recommended!



Author: Vincent Driessen
Original blog post: <http://nvie.com/posts/a-successful-git-branching-model/>
License: Creative Commons BY-SA



The Git-Flow Manifesto: Takaways



- ▶ **master is for releases only**
- ▶ **develop**
 - **Not ready for public consumption but compiles and passes all tests**
- ▶ **Feature branches**
 - **Where most development happens**
 - **Branch off of develop**
 - **Merge into develop**
- ▶ **Release branches**
 - **Branch off of develop**
 - **Merge into master and develop**
- ▶ **Hotfix**
 - **Branch off of master**
 - **Merge into master and develop**
- ▶ **Bugfix**
 - **Branch off of develop**
 - **Merge into develop**

Life Cycle of a Feature branch



- 1) Enable git flow for the repo
 - **git flow init -d**
- 2) Start the feature branch
 - **git flow feature start newstuff**
 - Creates a new branch called feature/newstuff that branches off of develop
- 3) Push it to GitHub for the first time
 - Make changes and commit them locally
 - **git flow feature publish newstuff**
- 4) Additional (normal) commits and pushes as needed
 - **git commit -a**
 - **git push**
- 5) Bring it up to date with develop (to minimize big changes)
 - **git checkout develop**
 - **git pull origin develop**
 - **git checkout feature/newstuff**
 - **git merge develop**
- 6) Finish the feature branch (**don't use git flow feature finish**)
 - Do a pull request on GitHub from feature/newstuff to develop
 - When successfully merged the remote branch will be updated
 - **git remote update -p**
 - **git branch -d feature/newstuff**

***What if I can't install
git-flow?***

***Just be sure to use the
proper naming and
branching conventions***

**feature/mybranch
release/mybranch
bugfix/mybranch
hotfix/mybranch**

git/GitHub (more JEDI tips)




- ▶ **Follow git-flow naming conventions**
 - ◆ **Web hook will scold you if you don't**
 - ◆ **Git-hooks also available to prevent noncompliant pushes**
 - ◆ **Most development work occurs in **feature branches****
 - ◆ **git-flow extension can be installed with usual installers (homebrew, apt-get, yum)**
 - ◆ **Example: **brew install git-flow****
- ▶ **Don't push directly to **develop** or **master****
 - ◆ **Changes to these branches are handled via **pull requests****
- ▶ **Use git-LFS for large files**
- ▶ **What about forks?**
 - ◆ **For now, developers can work off the central repo**
 - ◆ **As the project grows, each partner/collaborator institution will maintain a fork (merge with central repo as needed)**
 - ◆ **Forking may also be useful for public releases**

Agile Software Development



▶ 12 Agile Principles

**Git-Flow helps with many of these
For the rest, we have ZenHub**




Early and continuous delivery of valuable software

1 ✓




Welcome changing requirements even late in development

2 ✓




Deliver working software frequently

3 ✓




Business people and developers working together daily

4




Build projects around motivated individuals and trust them to get the job done

5 ✓



The most effective method of conveying information is face-to-face conversation

6




Working software is the primary measure of progress

7 ✓




Sustainable development: maintain a constant pace indefinitely

8



Continuous attention to technical excellence

9 ✓




Simplicity: maximize the amount of work not done

10 ✓



Teams self-organize

11



Teams regularly reflect and adjust behaviour

12 ✓

Agile workflows: ZenHub



JCSDA / ufo Private

Unwatch 28 Star 0 Fork 1

Code Issues 33 Pull requests 1 **Z ZenHub** Wiki Insights Settings

JEDI Unified Forward Operator Edit

Manage topics

454 commits 44 branches 0 releases 18 contributors Apache-2.0

Branch: develop New pull request Create new file Upload files Find file Clone or download

victordottir and ytremolet Bugfix for interpolation: set weights to 0/1 if the obs is outside

- cmake cmake clean-up
- docs Adding in files for creating the "Building UFO in OS X"
- src Bugfix for interpolation: set weights to 0/1 if the obs is
- test Bugfix for interpolation: set weights to 0/1 if the obs is
- tools Feature/script fornewobs (#79)
- .gitattributes Use git lfs
- .gitignore Feature/replace ad alloc (#77)
- CMakeLists.txt Feature/gnssro ropp1d forward (#67)
- COPYING First commit
- CPPLINT.cfg Feature/style check (#51)

**Install browser extension
from <http://zenhub.com>
to see ZenHub tab on
each repo**

**available for
Chrome, Firefox**

Using ZenHub



The screenshot shows a GitHub repository page for 'JCSDA/ufo' with the ZenHub interface overlaid. The ZenHub interface displays several boards:

- New Issues:** 25 Issues - 21 Story Points. Issues include 'old-ufo #3 dot product in ObsVector and GeoVaLs is not yet distributed. (local pe value only)', 'old-ufo #5 Add time to marine UFO's.', 'old-ufo #6 Implement putdb in ObsSpace.cc', 'old-ufo #15 Remove hardcoded metadata from ufo_radiance_eqv', and 'old-ufo #21 Generic FG check'.
- Icebox:** 1 Issue - 0 Story Points. Issue: 'old-ufo #16 Read metadata for CRTM through interface'.
- Backlog:** 8 Issues - 9 Story Points. Issues include 'old-ufo #14 Access to metadata for CRTM' and 'old-ufo #17 Pass hooks between c++/Fortran for CRTM K matrix'.
- In Progress:** 8 Issues - 7 Story Points. Issues include 'old-ufo #13 Implement git-LFS' and 'old-ufo #33 Fix UFO test failures when running on Theia using ifort'.
- Review/QA:** 4 Issues - 0 Story Points. Issues include 'old-ufo #4 the st hofx i' and 'old-ufo #4 Featu'.

A callout box contains the following text:

All GitHub Issues and pull requests appear on the Zenhub boards

All ZenHub issues/tasks appear as GitHub issues

ZenHub Features



- ▶ **Customizable Project boards**
 - ◆ **Prioritize and organize tasks**
 - ◆ **Reviews/Feedback**
 - ◆ **Sprints (Milestones) and Epics**
- ▶ **Closely integrated with GitHub**
 - ◆ **Access boards directly from GitHub repos**
 - ◆ **ZenHub tasks are GitHub issues and vice versa**
- ▶ **Tasks/Issues**
 - ◆ **Assign up to 10 individuals**
 - ◆ **Labels, difficulty estimates, etc.**
 - ◆ **Can be linked to pull requests**
 - ◆ **Markdown supported (boldface, checklists...)**
- ▶ **Monitoring progress**
 - ◆ **Burndown charts**
 - ◆ **Velocity tracking**
 - ◆ **Release reports**
 - **Time estimate to deliver a specified set of features**

ZenHub Pipelines



▶ ***New Issues***

- ◆ ***Default landing spot***
- ◆ ***Issues should not stay here long***

▶ ***Backlog***

- ◆ ***Main “To Do” List***
- ◆ ***Arrange in order of priority (reviewed regularly by teams)***

▶ ***IceBox***

- ◆ ***Low-priority items that should be done at some point but do not require immediate attention***

▶ ***In Progress***

- ◆ ***Lets others know what you are doing to promote collaboration and avoid redundancy***

▶ ***Review/QA***

- ◆ ***Solicit feedback before you mark something as...***

▶ ***Closed***

ZenHub Issues/Tasks



JCSDA/jedi-docs#14 **Sphinx - unit testing** powered by | ZenHub

mmiesch commented on Apr 10
No description provided.

Sphinx - unit testing has no dependencies + add dependency

ytremolet changed the pipeline from **New Issues** to **Backlog** on Apr 15

mmiesch changed the pipeline from **Backlog** to **In Progress** on Apr 17

mmiesch self-assigned this on Apr 17

mmiesch connected this issue to [JCSDA/jedi-docs#55 Develop](#) 21 days ago

mmiesch changed the pipeline from **In Progress** to **Review/QA** via a connected PR [JCSDA/jedi-docs#55 Develop](#) 21 days ago

mmiesch added the **enhancement** label a minute ago

mmiesch set the estimate to 5 a minute ago

Write Preview AA B i [Rich Text Editor Icons]

Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from clipboard

Styling with Markdown is supported

Issue Details:

- Pipeline: Review/QA
- Assignees: mmiesch
- Labels: enhancement
- Milestone: No milestone
- Estimate: 5
- Releases: Not inside a Release
- Epics: Not inside an Epic
- Notifications: Unsubscribe
- 1 participant

Estimate

How complex is this issue?

Filter estimates (or type to create one)

0.5
1
3
5
8
13
21
40
75

Suggestion:
1 unit = 1/2 day
dedicated work

This pull request is connected to

Sphinx - unit testing
jedi-docs#14 opened 2 months ago by mmiesch

Disconnect

ZenHub Features



- ▶ **Milestones (Sprints)**
 - ◆ **Short-term (~ 2 weeks)**
 - ◆ **Focused work, often on 1-2 repos**
 - ◆ **Deliverables = specific functionality/features**

- ▶ **Epics**
 - ◆ **Long-term (indefinite)**
 - ◆ **Typically span multiple repos**
 - ◆ **Deliverables = releases, guiding vision**

- ▶ **Workspaces**
 - ◆ **Collect multiple repositories into a common board**

Project boards include *filters* to view only issues associated with Milestones, Epics or other attributes (assignee, label, repo, release...)

ZenHub: Sprint Retrospective



Sprint Retrospectives and other agile workflow components (Sprint Review, Release Planning, etc) are best done face-to-face, but one could in principle dedicate an issue or a pipeline to solicit further perspectives

ZenHubHQ / ZenHubHQ Private

Unwatch 16 Star 0 Fork 0

Code Issues 35 Pull requests 1 Boards Burndown Wiki Pulse Graphs

Retrospective --> August 1-15 #64

Open paigepaquette opened this issue 26 days ago · 0 comments

paigepaquette commented 26 days ago

Points to discuss

Add comments anonymously here 👍

- QA wait times
- Refactor updates?
- Social media update schedule
- ...

Things to start:

-
-
-

Things to stop:

-
-

Things to continue doing:

-
-

Actionables and keepers:

Metadata:

- Pipeline: New Issues
- Labels: Discussion
- Milestone: No milestone
- Estimate: No estimate yet
- Assignees: pnavarrc, olegzd, devinmcinnis, aupright, Mathieuu, brianleung11-3, azenMatt, paigepaquette
- Epics: Not inside an Epic

ZenHub: Burndown chart



Sprint 10 - V2 Improvements

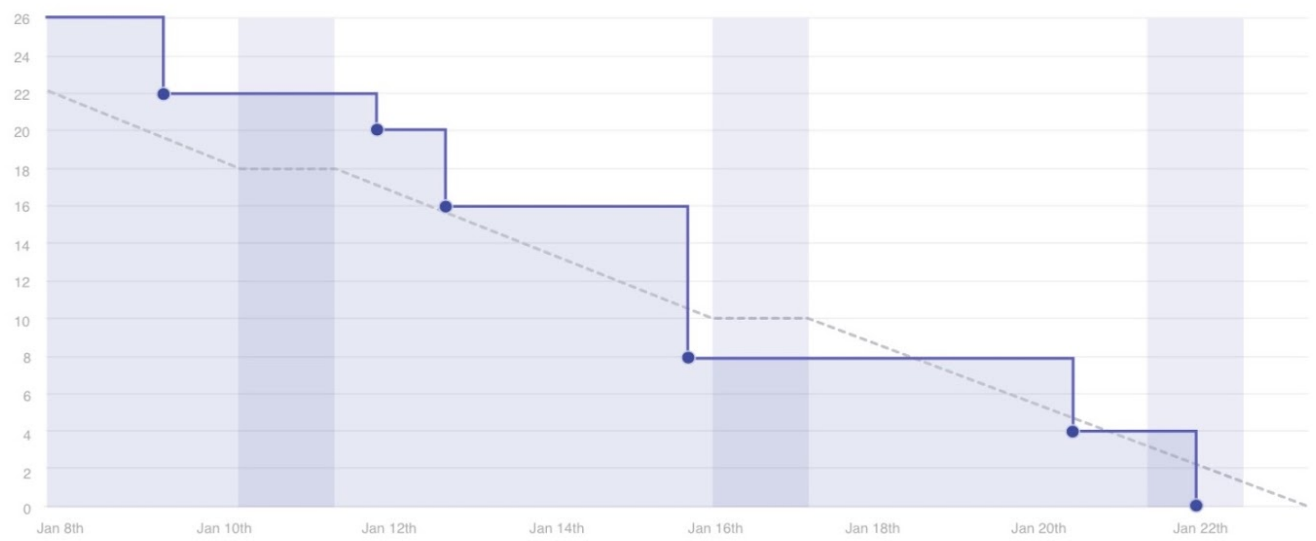
[Edit Milestone](#) [Change Milestone](#)

[Labels](#) [Hide Pull Requests](#)

[Burn Pipelines](#)

Start: **Jan 3, 2017** [Edit](#) Due: **Feb 28, 2017** [Edit](#)

Weekends Ideal Completed



26 Total Story Points
18 Completed Story Points / 8 Remaining Story Points

34 Total Issues and Pull Requests
28 Completed Issues and PRs / 6 Remaining Issue and PRs

ZenHub: Release Report



Project X

Combined projects across Q1 and Q2, working towards the next chapter of AppX.

Start date: Jan 1st, 2017 Desired end date: Jun 30th, 2017

Labels ▾

Release report

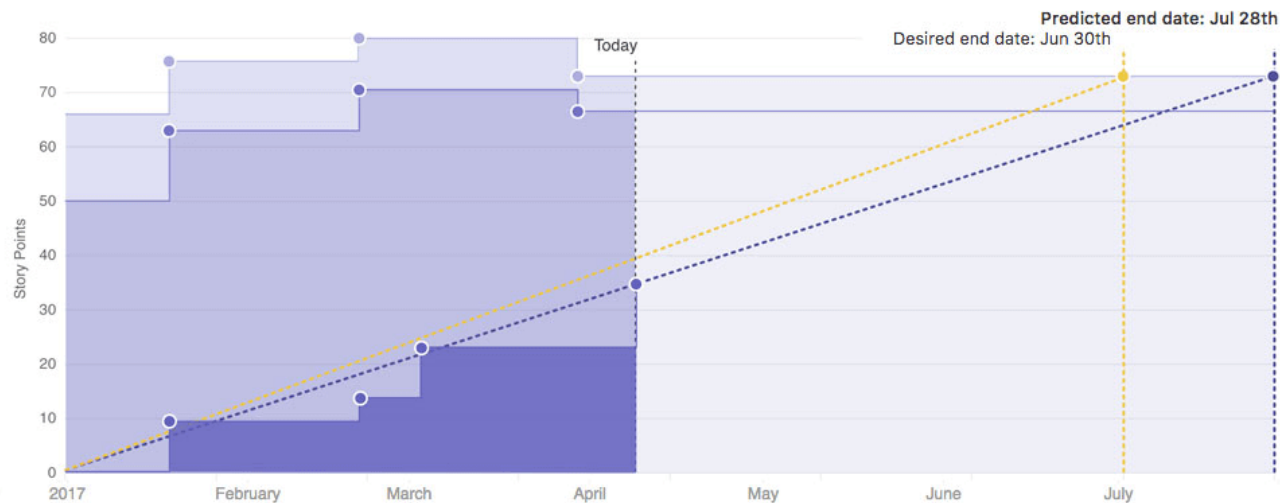
Completed points

Estimated scope

Total scope

Actual velocity

Desired velocity



The predicted end date is **Jul 28th** ⓘ

The predicted end date will be **28 days behind** the desired end date of Jun 30th

Total issues	7
Completed	3
Remaining	4

Total story points	30
Completed	21
Remaining	9

Part II: Documentation



▶ **Agile Project Management and Collaborative Workflow**

- ◆ **git/GitHub**
- ◆ **git-flow**
- ◆ **ZenHub**

▶ **Documentation**

- ◆ **Sphinx/ReadTheDocs** (*high-level manuals, how-to's, etc*)
- ◆ **Doxygen** (*low-level code details*)
- ◆ **JEDI Wiki**

Sphinx/ReadtheDocs



<https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/>

JEDI Documentation — JEDI D... x

Secure | <https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/>

Apps JEDI Software Engineering Mac Me

JEDI Documentation
latest

Search docs

Background
Working Practices
Developer Tools and Practices
JEDI Environment
Building, Testing, and Running JEDI

JEDI Documentation

Welcome to JEDI!

This documentation will help you get started with JEDI whether you are a user or a developer.

Table of Contents

- Background
 - JEDI High Level Requirements
 - JEDI General Methodology
- Working Practices
 - Branching and merging code
 - Forking and cloning repositories
 - Reviewing code
 - Testing
 - Creating documentation
- Developer Tools and Practices
 - Homebrew (Mac only)
 - Git flow
 - Git-LFS
 - Sphinx
 - Docker

Read the Docs v: latest

Publicly available
Targeted at users as well as developers

Sphinx/ReadtheDocs



The screenshot shows a Google search interface. The search bar contains the text "jedi working practices". Below the search bar, the "All" tab is selected. The search results show "About 4,060,000 results (0.52 seconds)". The top result is titled "Working Practices – JEDI Documentation 1 documentation" and includes the URL <https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/.../wor...>. Below the URL, the snippet reads "Working Practices¶. Branching and merging code · Forking and cloning ...".

Or, get there from
<http://academy.jcsda.org>

Sphinx/ReadtheDocs



Building, Testing, and Running x

Secure | https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/developer/building_and_testing/in...

Apps JEDI Software Engineering Mac Meetings Outdoors Garden Transition Colleges Travel Cooking Self EPO

JEDI Documentation
latest

Search docs

Background
Working Practices
Developer Tools and Practices
JEDI Environment

Building, Testing, and Running JEDI

- Building and compiling JEDI
 - Precursor: Git Configuration
 - Step 1: Clone the Desired JEDI Bundle
 - Step 2: Choose your Repos
 - Step 3: Run ecbuild (from the build directory)
 - Step 4: Run make (from the build directory)
- JEDI Testing
 - Running ctest
 - Manual Execution
 - The JEDI test suite
 - Tests as Applications
 - Initialization and Execution of Unit Tests
 - Anatomy of a Unit Test
 - Integration and System (Application) Testing
 - JEDI Testing Framework
- Adding a New Test
 - Step 1: Create a File for your Test Application
 - Step 2: Define A Test Fixture
 - Step 3: Define Your Unit Tests
 - Step 4: Register your Unit Tests with Boost
 - Step 6: Create an Executable
 - Step 7: Create a Configuration File
 - Step 8: Register all files with CMake and CTest
 - Adding an Application Test
- JEDI Configuration Files

Previous Next

Read the Docs v: latest

Sphinx



▶ **Sphinx**

- ◆ **The real workhorse behind the documents**
- ◆ **Python package**
- ◆ **Source code written with Restructured text**

▶ **Distribution plan**

- ◆ **ReadtheDocs for now to publish**
- ◆ **Sphinx Source code on GitHub (jedi-docs)**
- ◆ **Tagged versions of the doc repos will be linked to JEDI releases**

For more info on Sphinx see the corresponding page in the JEDI documentation, under *Developer Tools and Practices*

Doxygen



Doxygen

Used in JEDI for:

- ▶ **Documenting functions and subroutines (C++ and F90)**
- ▶ **Documenting classes and structures (C++ and F90)**
- ▶ **Viewing namespaces and modules**
- ▶ **Generating Class Hierarchies**
- ▶ **Generating Call diagrams**
- ▶ **Any other documentation that involves specific blocks of code**

For example Doxygen documentation (fv3-bundle)

See

<https://github.com/june2019>

Doxygen Implementation Plan



▶ **User/Developers (this means you!)**

- ◆ Please place appropriate Doxygen comments in source files
- ◆ (optionally) test functionality by compiling with Doxygen config files provided by JEDI team (feel free to customize, but please don't commit your changes)
 - Find Doxyfile (the plan is to have one in the Documents directory of every repo)
 - > **doxygen**
 - View results in html directory

▶ **JEDI Core Team**

- ◆ Will supply the Doxyfile config files
- ◆ Will publish html files for develop and master versions of repos (generated automatically, triggered by pull requests)
- ◆ Tagged versions linked to releases
- ◆ Please be patient - We're still working on this

Documenting Fortran Source Code



```
!! _____  
!  
!> \brief Example function  
!!  
!! \details myfunction() takes a and b as arguments and miraculously creates c.  
!! I could add many more details here if I chose to do so. I can even make a list:  
!! * item 1  
!! * item 2  
!! * item 3  
!!  
!! \date A long, long, time ago: Created by L. Skywalker (JCSDA)  
!!  
!! \warning This isn't a real function!  
!!  
subroutine myfunction(a, b, c)  
    integer, intent(in)      :: a !< this is one input parameter  
    integer, intent(in)      :: b !< this is another  
    real(kind=kind_rea), intent(out) :: c !< and this is the output  
    [...]
```

Documenting C++ Source Code



```
// -----  
/*! \brief Example function  
*  
* \details **myfunction()** takes a and b as arguments and miraculously creates c.  
* I could add many more details here if I chose to do so. I can even make a list:  
* * item 1  
* * item 2  
* * item 3  
*  
* \param[in] a this is one input parameter  
* \param[in] b this is another  
* \param[out] c and this is the output  
*  
* \date A long, long, time ago: Created by L. Skywalker (JCSDA)  
*  
* \warning This isn't a real function!  
*  
*/  
void myfunction(int& a, int& b, double& c) {  
    [...]
```

Useful Doxygen Commands



- ▶ `\brief`
- ▶ `\details`
- ▶ `\param`
- ▶ `\return`
- ▶ `\author`
- ▶ `\date`
- ▶ `\note`
- ▶ `\attention`
- ▶ `\warning`
- ▶ `\bug`
- ▶ `\class <name> [<header-file>]`
- ▶ `\mainpage`
- ▶ `\f$... \f$` (**inline formula**)
- ▶ `\f[... \f]` (**formula block**)
- ▶ `\em` (**or * ... ***)
- ▶ `\sa` (**see also**)
- ▶ `\typedef`
- ▶ `\todo`
- ▶ `\version`
- ▶ `\namespace`
- ▶ `...` (**url**)
- ▶ `\image`
- ▶ `\var`
- ▶ `\throws` (**exception description**)

Many more described here:

<https://www.stack.nl/~dimitri/doxygen/manual/commands.html>

Sample output: "man page"



◆ testStateInterpolation()

template<typename MODEL >

```
void test::testStateInterpolation ( )
```

Interpolation test.

testStateInterpolation() tests the interpolation for a given model. The conceptual steps are as follows:

1. Initialize the JEDI **State** object based on idealized analytic formulae
2. Interpolate the **State** variables onto selected "observation" locations using the `getValues()` method of the **State** object. The result is placed in a JEDI **GeoVaLs** object
3. Compute the correct solution by applying the analytic formulae directly at the observation locations.
4. Assess the accuracy of the interpolation by comparing the interpolated values from Step 2 with the exact values from Step 3

The interpolated state values are compared to the analytic solution for a series of **locations** which includes values optionally specified by the user in the "StateTest" section of the config file and a randomly-generated list of **Nrandom** random locations. Nrandom is also specified by the user in the "StateTest" section of the config file, as is the (nondimensional) tolerance level (**int**) to be used for the tests.

This is an equation:

$$\zeta = \left(\frac{x - x_0}{\lambda} \right)^{2/3}$$

Relevant parameters in the ****State*** section of the config file include

- **norm-gen** Normalization test for the generated **State**
- **interp_tolerance** tolerance for the interpolation test

Date

April, 2018: M. Miesch (JCSDA) adapted a preliminary version in the feature/interp branch

Warning

Since this model compares the interpolated state values to an exact analytic solution, it requires that the "analytic_init" option be implemented in the model and selected in the "State.StateGenerate" section of the config file.

Corresponding code



```
// -----  
/!* \brief Interpolation test  
*  
* \details **testStateInterpolation()** tests the interpolation for a given  
* model. The conceptual steps are as follows:  
* 1. Initialize the JEDI State object based on idealized analytic formulae  
* 2. Interpolate the State variables onto selected "observation" locations  
* using the getValues() method of the State object. The result is  
* placed in a JEDI GeoVaLs object  
* 3. Compute the correct solution by applying the analytic formulae directly  
* at the observation locations.  
* 4. Assess the accuracy of the interpolation by comparing the interpolated  
* values from Step 2 with the exact values from Step 3  
*  
* The interpolated state values are compared to the analytic solution for  
* a series of **locations** which includes values optionally specified by the  
* user in the "StateTest" section of the config file in addition to a  
* randomly-generated list of **Nrandom** random locations. Nrandom is also  
* specified by the user in the "StateTest" section of the config file, as is the  
* (nondimensional) tolerance level (**interp_tolerance**) to be used for the tests.  
[...]
```

Corresponding code (cont.)



[...]

*

* This is an equation:

* $\zeta = \left(\frac{x-x_0}{\lambda}\right)^{2/3}$

*

* Relevant parameters in the **State** section of the config file include

*

* **norm-gen** Normalization test for the generated State

* **interp_tolerance** tolerance for the interpolation test

*

* \date April, 2018: M. Miesch (JCSDA) adapted a preliminary version in the

* feature/interp branch

*

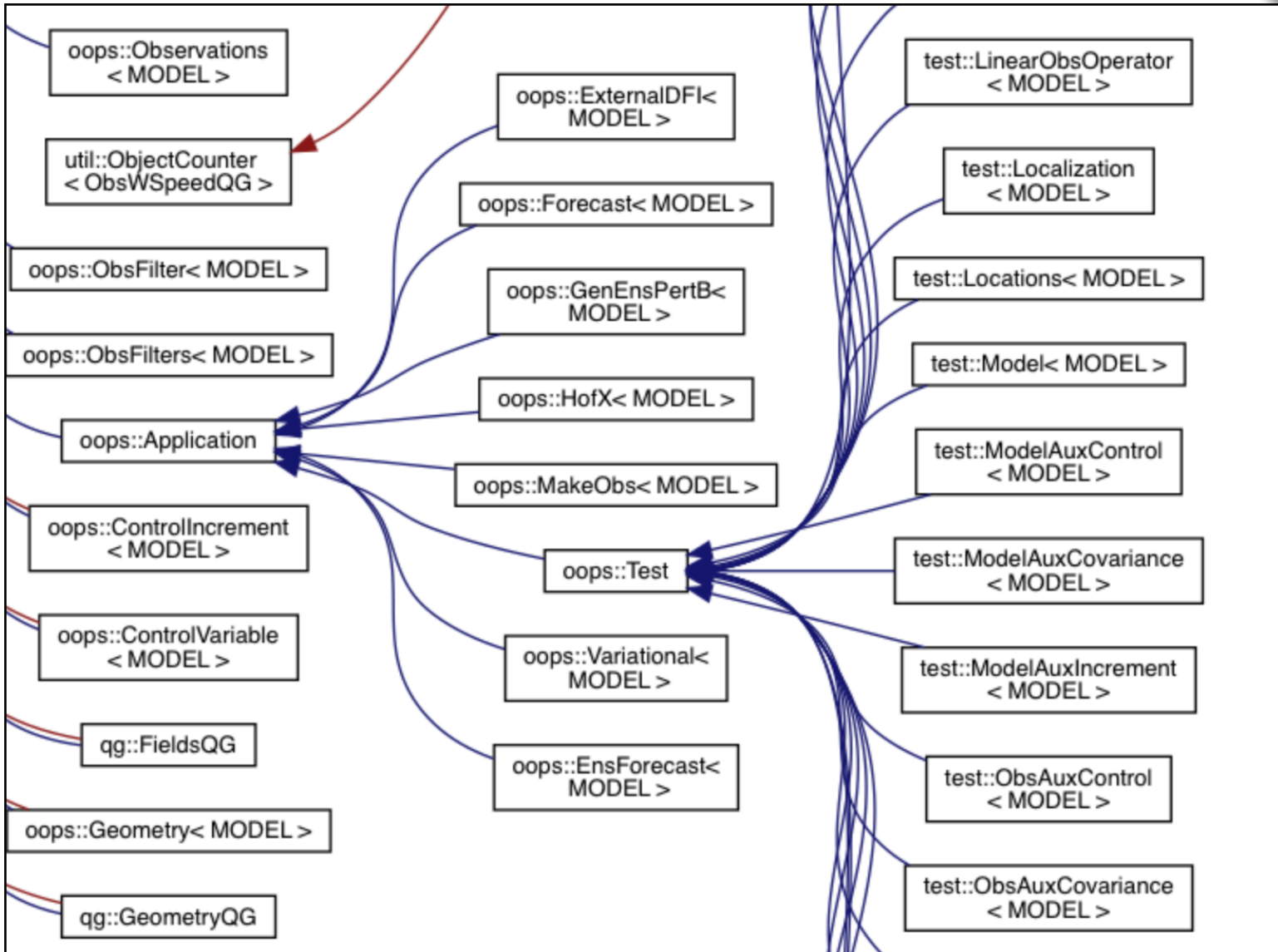
* \warning Since this model compares the interpolated state values to an exact analytic

* solution, it requires that the "analytic_init" option be implemented in the model and

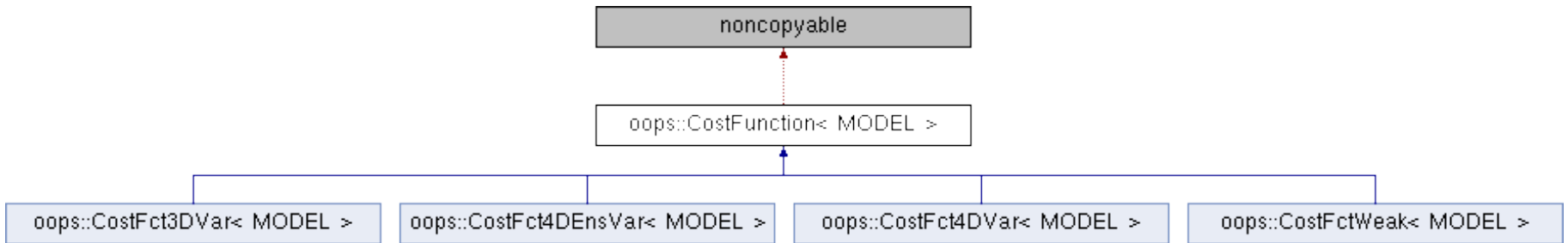
* selected in the "State.StateGenerate" section of the config file.

*/

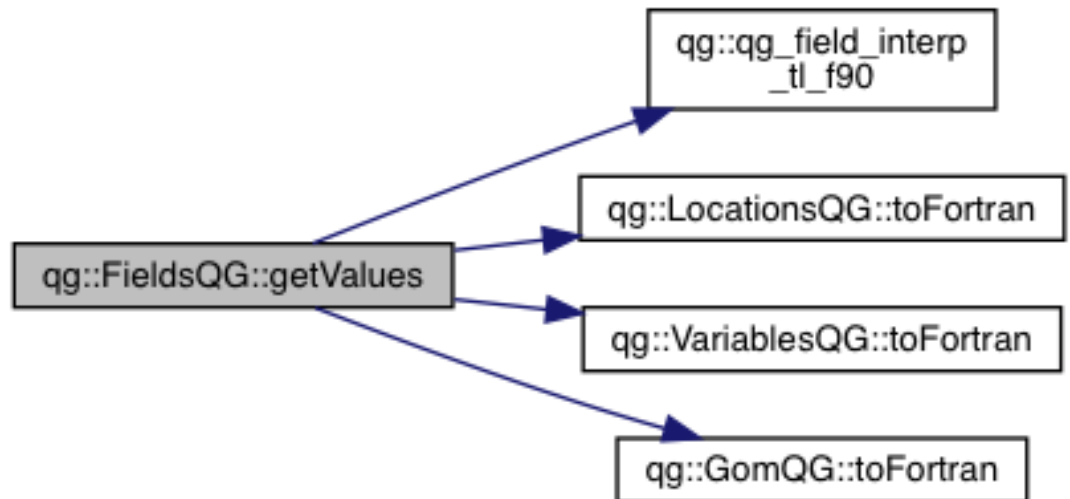
Sample output: class hierarchy



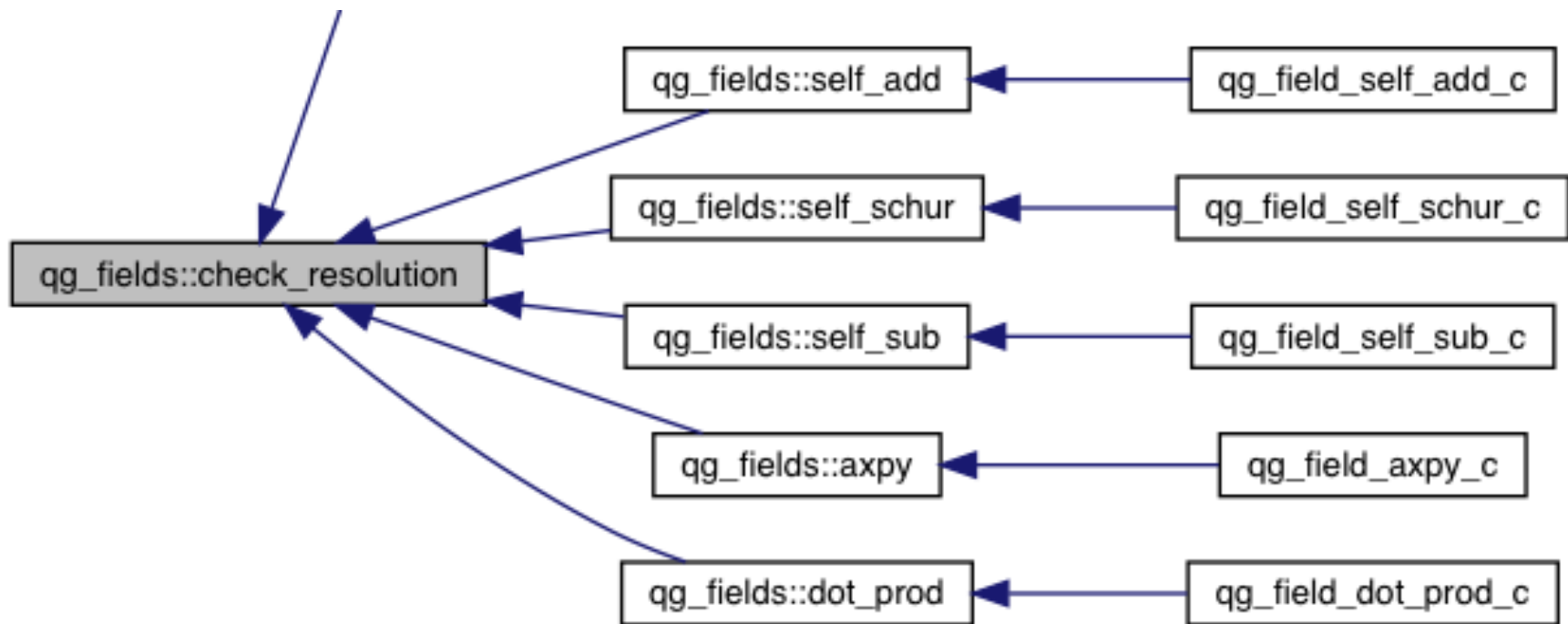
Sample output: inheritance, call graphs



Clickable boxes!

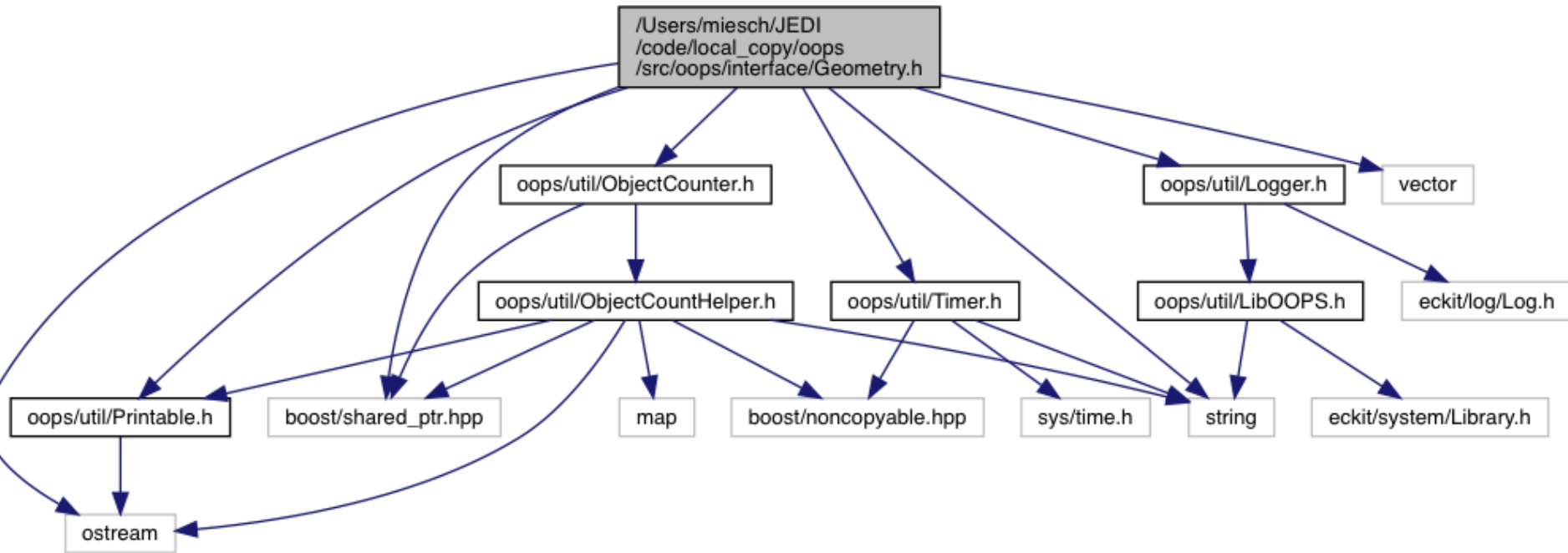


Sample output: caller graphs



Note that these traces end in `_c` (this is a Fortran routine)
Doxygen has trouble with C++ / Fortran binding
Look for corresponding `_f90` routine to follow further

Sample output: include diagrams



Can get complicated!

Other documentation



In a few cases, other sorts of documentation (often pdf) may be available in the Documents directory of a repo

Example: oops

Generally, we plan to link to these pdfs from the Doxygen pages

A Two Level Quasi-geostrophic Model

Mike Fisher, ECMWF

February 8, 2018

1 Introduction

This note describes a simple two-level quasi-geostrophic model, intended for use as a “toy” system with which to conduct idealised studies of data assimilation methods. In developing the model, the emphasis has been placed on speed and convenience rather than accuracy and conservation.

2 The Continuous Equations

The equations of the two-level model are given by Fandry and Leslie (1984) (see also Pedlosky, 1979 pp386-393), and are expressed in terms of non-dimensionalised variables:

$$\frac{Dq_1}{Dt} = \frac{Dq_2}{Dt} = 0 \quad (1)$$

where q_1 and q_2 denote the quasi-geostrophic potential vorticity on each of the two layers, with a subscript 1 denoting the upper layer:

$$q_1 = \nabla^2 \psi_1 - F_1(\psi_1 - \psi_2) + \beta y \quad (2)$$

$$q_2 = \nabla^2 \psi_2 - F_2(\psi_2 - \psi_1) + \beta y + R_s \quad (3)$$



JEDI - JEDI - wiki.ucar.edu

Secure | <https://wiki.ucar.edu/display/JEDI/JEDI>

wiki.ucar.edu Spaces Calendars Create

JEDI

Pages Edit Save for later Watch Share

JEDI

Created by UCAR Webmaster, last modified by Yannick Tremolet on May 01, 2018

Joint Effort for Data assimilation Integration

The long term objective of the Joint Effort for Data assimilation Integration (JEDI) is to provide a unified data assimilation framework for research and operational use, for different components of the Earth system, and for different applications, with the objective of reducing or avoiding redundant work within the community and increasing efficiency of

More Information

- [Project Plans](#)
- [Fortran Interfaces](#)

Abstract Layer

Interpolations

Observation Operators

IODA

Recently Updated

Space tools

IntroToDoxygen_041....pdf

Show All

◆ Targeted at developers

◆ Discussion of current progress, issues

◆ Resources for code sprints and other events

Warning: Less polished than ReadtheDocs (no guarantee that everything is up to date)

JEDI Wiki: Weekly Meeting Notes



May 3, 2018 - JEDI - wiki.ucar.edu

Secure | <https://wiki.ucar.edu/display/JEDI/May+3%2C+2018>

Apps | JEDI | Software Engineering | Mac | Meetings | Outdoors | Garden | Transition | Colleges | Travel | Cooking | Self | EPO

wiki.ucar.edu | Spaces | Calendars | Create

Blog | Calendars

PAGE TREE

- Project Plans
- Abstract Layer
- Interpolations
- Observation Operators
- Interface for Observation Data Access (IODA)
- Fortran Interfaces for JEDI
- How-to Articles
- Adding models into OOPS/JEDI
- File lists
- Software Development Methodology
- JEDI Weekly Meeting Notes
 - April 12, 2018
 - April 19, 2018
 - April 26, 2018
 - May 3, 2018**
 - May 10, 2018
 - May 17, 2018
 - May 24, 2018

Space tools

Pages / JEDI / JEDI Weekly Meeting Notes

May 3, 2018

Created by Mark Miesch, last modified by Stephen Herbener on May 03, 2018

Most of today's meeting was concerned with the reorganization of the ObsSpace classes in oops, ufo, and ioda.

Xin started the discussion by pointing out multiple places in ufo where code is duplicated, both in terms of the file structure and the code itself. He then went on to illustrate several examples of conditional execution based on if/else if statements. This could be cleaned up substantially and optimized with a more object-oriented approach.

For this reason, there is an effort at JCSDA (led by Xin, Steve, and Yannick) to reorganize the ObsSpace data structure in order to:

- Reduce duplicated subroutines
- Simplify the APIs
- Re-design ObsSpace data structure

ObsSpace Reorganization

Xin Zhang
JEDI Core Team
5/3/18

PDF

Then Steve shared similar concerns and efforts, focusing in particular on the reading and writing of data in ioda:

IntroToDoxygen_041....pdf

Show All



Resources

Resources: GitHub & ZenHub



JEDI Documentation - access link from
<https://academy.jcsda.org>

Extensive GitHub documentation & tutorials
<https://help.github.com>

Lots of Great Github Cheat Sheets

<https://education.github.com/git-cheat-sheet-education.pdf>

<https://jan-krueger.net/git-cheat-sheet-extended-edition>

<https://patrickzahnd.ch/uploads/git-transport-v1.png>

ZenHub Guides

<https://www.zenhub.com/guides>

Resources: Git-Flow



JEDI Git Flow page

https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/developer/developer_tools/getting-started-with-gitflow.html

The Git Flow manifesto (all you need to know about the philosophy):

<http://nvie.com/posts/a-successful-git-branching-model/>

Git Flow cheat sheet:

<https://danielkummer.github.io/git-flow-cheatsheet/>

Git avh (a fork of the original, with added features):

<https://github.com/petervanderdoes/gitflow-avh>

Atlassian git-flow tutorial (philosophy and application):

<https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

Resources: Git-LFS



JEDI Git-LFS page

https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/developer/developer_tools/gitlfs.html

GitHub's Help page:

<https://help.github.com/articles/about-git-large-file-storage/>

Tutorial:

<https://github.com/git-lfs/git-lfs/wiki/Tutorial>

Installation? Already installed in the JEDI singularity container

Binaries available for download on:

<https://git-lfs.github.com>

Or, on a Mac:

`brew install git-lfs`

Resources: Doxygen



JEDI Doxygen page

https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/developer/developer_tools/doxygen.html

Doxygen Users Manual

<http://www.stack.nl/~dimitri/doxygen/manual/index.html>

Installation? Already installed in the JEDI singularity container

Binaries available for download on:

<http://www.stack.nl/~dimitri/doxygen/download.html>

Doxygen Installation (Mac)



```
brew install doxygen
```

You may be prompted to also install Doxywizard and Graphviz - we recommend you say yes to both... If Graphviz does not install for some reason, you can install it manually:

```
brew install graphviz
```

You'll need this for generating graphs

Similar commands for linux package managers, e.g.

```
sudo apt-get doxygen
```