# 2020-06-25

Yannick opened the meeting announcing the agenda of a general round-table discussion.

He started by altering people to **two substantial upcoming changes that will affect all models and other repos**. Both involve active pull requests. Because of their large scope, spanning many repos, they require a coordinated merge. So, keep an eye out for a merge sprint that we will announce sometime in the next few weeks.

The first is a change in the format of the yaml configuration files. This concerns oops PR #667 and related PRs in other repos. This requires changes in all the yaml files that are straightforward but extensive. Yannick asked meeting participants to help out with this refactoring on the repositories they are working with.

The second substantial change is to split the MODEL traits into two separate collections of traits that are concerned with the model itself (MODEL) and those that are concerned with how the model interfaces observations (OBS). This is described in a discussion thread on the JEDI Models GitHub Team. Applications and other classes that were previously templated with MODEL will now need to be templated with MODEL and OBS if they involve observations. As described in the team discussion thread, there are examples of how to do this for the soca, fv3-jedi, and shallow-water models.

So, if you have not done so already, **Please make these changes in your model repos** so we all all ready for the upcoming merge sprint.

Chris H then asked if we can expect any other significant API changes. He appreciates the code improvements but notes that frequent API changes can make it hard on user/developers, particularly those who maintain model interfaces. Yannick acknowledged that we do want to get to a place where API changes will be less frequent. But, the reason that we're seeing many changes now is because we are preparing for the first public release of JEDI and we would like stable APIs to be in place by then. After the first release, the intention is to minimize further changes.

Chris H then gave an update on shallow water. Thanks to much help from Anna and Travis, LETKF is now working. Chris also thanked Anna for helping to keep the shallow-water model up to date with the latest API changes. He also reported a problem he is having with parallel CMake, first noticed by Travis. Chris H will create a new ZenHub issue to describe the problem in more detail. He thinks it has to do with a Fortran module that is used by many of the files in the repo. CMake is apparently not sorting out the dependencies correctly and is building things in the wrong order.

Mark O offered to help diagnose the problem. He recommended to update to the latest CMake. The latest versions, 3.17+, have better Fortran support. Ming mentioned CMake problems they were seeing with WRF. The build succeed with CMake version 3.6.2 but failed when they moved to 3.16.2. Mark O was surprised by this and suspected it may be a fortuitous and atypical case. CMake 3.6.2 doesn't have great Fortran support and they are good about making new versions backward compatible so he recommended again to use the latest one you can. Version 3.6.2 will not work with JEDI going forward.

Guillaume then asked if there is any adopted convention in JEDI for the range of longitudes: 0 to 360 or -180 to 180? Yannick and Steve H responded that there is no adopted standard currently but Steve noted that the OBS team is putting together a spread sheet that specifies conventions so this might be a good place to define it. Several mentioned that a conversion is done in some places of the code, such as bump, where the longitude is taken modulo 360 degrees. JJ suggested that this could be done in the Locations constructor. Wojceich will take a look at giving the Gaussian filter the ability to handle negative longitude values (ie, cast them into 0 - 360 degrees). Yannick encouraged someone to create a ZenHub issue to address this.

Jake then described a problem he is having on Cheyenne. For debugging purposes, he would like to have each MPI task write to a different output file but the PBS job scheduler puts all the standard output into a single file (and standard error in another file). This can be millions of lines for large jobs and difficult to extract useful information for debugging. Chris H thought that this cannot be changed. Mark O added that the concatenation of the stdout is really done by mpirun rather than PBS. Jake also added that CRTM generates a lot of output and wonders if that can be optionally reduced. Mark O is planning a pull request to address that.

One way to handle this is to have each MPI task write to a different file, tagged with the rank, instead of standard out. Yannick said this can be done in C++ and is partly implemented in JEDI. One way to get this to work with Fortran is to redirect the Fortran output through output channels that are defined in C++ but this has not yet been implemented. Ming said that WRF has flags to tell it to direct output to files tagged with the rank so maybe we could look there for inspiration.

Steve H reported that he and Ryan are making good progress with the ioda ObsSpace refactoring and associated ioda APIs. The ioda tests are now passing with the new structure but there is more to do. Although the code probably will not be ready by next Thursday, Steve thought it would be a good time to describe the changes in our next JEDI bi-weekly Focused Topic meeting on July 2. He asked the meeting participants whether they agree that this would be a good topic for next week and several responded affirmatively. Steve S later asked for clarification of when the Atlas focused discussion will take place. It was agreed that our next two focused topic discussions will be:

- July 2: Ioda ObsSpace refactoring and API
- July 16: The use of Atlas in JEDI

Emily then asked whether the changes in ioda will require changes in the data files and/or changes in the ufo code. In the ensuing discussion, she also asked a related question about where to put code to handle observation processing: ioda or ufo? More specifically, the code she has in mind will take raw satellite data with a large footprint and recast it with a smaller footprint, which will require some averaging of the raw data.

In response to both of these questions, Yannick and Steve emphasized the separation of concerns. The ufo should be independent of the obs file format - that should be handled exclusively by ioda. Furthermore, the internal representation of the data in ObsSpace should also be independent of the file format. So, the answer to the first question is that the new ioda obspace refactoring should not require extensive changes to the ufo code while removing the visibility (and concern) from ufo code to the obs data files. We can discuss this further next week.

Separation of concerns also means that ioda should not do any science. If the processing of the observations in question involves any scientific interpretation of the obs, then it should go into ufo. Emily, Jianjun, Yannick, and Steve agreed to set up a meeting with members of the JEDI and OBS teams to discuss this in more detail and determine whether the use case described by Emily involves a scientific interpretation of the data and where it should go.

In either case, Yannick and Steve thought that this use case could likely be handled in a similar way as the super-obbing discussed in our last meeting (June 11). The modifications to the raw observations can be handled in an analogous way to a variable change, taking one ObsSpace object as an input and returning the modified observations as another ObsSpace object.

Yannick then asked the group if anyone had any other comments or questions. After no one responded, the meeting was adjourned.