2019-08-22

The meeting opened with an update from the ufo code sprint that is now occurring in Boulder.

Lou began the discussion by reporting that the implementation of the radar obs operator into ufo is largely complete. Yunheng added that the GeoVaLs and obs files are done and ready for testing. They just have to generalize their interpolation routines and develop tests. Lou also mentioned that the wind profiler needs a bit more work. They expect to have the radar obs operator fully implemented by the end of the sprint, possibly as soon as tomorrow. Lou was also asking about tests for different types of interpolation. Though it was not discussed at the meeting, there are plans in the works for generalizing and extending the interpolation options and tests in oops.

Yannick then mentioned that most of the other work at the sprint has been concerned with QC filters, including generic interfaces and specific implementations.

JJ has been adding piping to get diagnostics from the H(x) operators to the QC filters and will be testing this new functionality today. After that is working, he plans to extend this to include data that is computed by ufo but is not formally part of the H(x) class. Lou asked him to clarify what he means by piping. JJ remarked that this does not refer to system-level data flow - rather, it was just intended as a way to describe how the data is passed among different levels in the C++ and Fortran class hierarchies.

Cory is adding more ctests and is working on a deriv check filter.

Emily is working on domain checks and has created a new yaml file to connect IR sounders with related QC. She is now working out the details of how to implement this. She has created a QC flowchart for IR sounders used in GSI, attached here:

Yannick praised this as a very useful reference and encouraged others to develop similar diagrams for other obs types.

At the code sprint, Yannick has been generalizing the QC filters to include additional actions such as recording meta-data after the QC has been applied.

Hui has created new background checks and is now working toward more complex and comprehensive background checks. She has also been working with the new functionality in ioda to group observations by metadata (see below) and how this can be used to facilitate QC. She is starting with refraction QC in GNSSRO but plans to extend this to other obs types.

Ryan has finished implementing a generic difference filter for all QC checks and plans to move on to other generic functions and interfaces.

Then Phil shared his screen and gave an instructional demonstration of how to effectively use ZenHub to plan a sprint, using this sprint as an example. When planning a sprint, the first step is to define a **Milestone** in ZenHub. Then, the team can create new issues or identify existing issues that they want to address during the code sprint. Each of these issues can then be selected and added to the Milestone. Then, to view and manipulate the issues in a sprint, one can filter the ZenHub board to only illustrate those issues connected with the code sprint. To see an example, navigate to the ZenHub board for ufo and select **August 2019 Code Sprint** from the drop-down **Milestones** menu. Teams can move issues from one column to another as the work proceeds.

Then Phil showed a burndown chart for this code sprint, which can be accessed from the ZenHub board by selecting **Reports Burndown report** from the menus on the left. This shows a graph. The horizontal axis is time, covering the duration of the sprint. The vertical axis is some measure of the work that was planned for the sprint. In agile workflows, this is often measured in units called **story points**. A good rule of thumb is that one story point is roughly equivalent to a half day of work by one person. So, if the issue in question can be resolved in an afternoon, you can assign it one story point. Or, if it will take a week of dedicated work, you might assign it 10 story points. This does not necessarily mean that it will be done in a week because all of us work on multiple projects so the odds of having an entire week available to devote to a single task may be slim. One can also use a relative scale when assigning story points - for example, if you know one task will take much more time than another, then just give it more story points.

You can assign story points to issues in ZenHub by selecting the issue and then accessing the drop-down **Estimate** menu on the right. The drop-down menu follows a nonlinear (Fibonacci) sequence, mimicking the reality of the workplace in which some issues are easy to address and others will take much more time. When planning a code sprint, one may wish to break up large issues into smaller tasks that can be more easily completed during the sprint.

Though estimating story points is not an exact science, it can be very useful for planning and reporting purposes. The burndown chart shows how much has been accomplished already in the sprint (story points completed, plotted as a blue line and listed below the graph) and how much remains to be done. This can be compare to the pace needed to complete all the issues assigned to the sprint (overplotted in grey). If the pace is too slow (substantially above the grey line), then it makes sense to reassess the objectives of the sprint and exclude some issues that may be lower priority. Or, if the pace is faster than expected, the team may wish to add issues.

Burndown charts are also useful to show stakeholders what has been accomplished over the course of a code sprint or some other event or unified goal (often called an epic in ZenHub). But, Phil cautioned that the utility is only as good as the information put in - inaccurate story point estimates or incomplete documenting of tasks can detract from the utility of burndown charts and related reports.

So, the take-away messages to all JEDIs, particularly for a code sprint but also more generally:

- **Document what you're working on by creating an issue in ZenHub**
- **Estimate how difficult or time-consuming the task is (i.e. assign story points)**
- **(Particularly for project leads) Monitor the burndown chart to see when you might need to re-assess the scope of your sprint (or other event/objective) based on the time available**

We then moved on from the ufo code sprint to get an update from the remaining JEDI group in Boulder.

Clementine brought the group's attention to a pull request in SABER that eliminates the **ecbuild_find_mpi()** and related statements in the CMakelists.txt file. This was causing build problems on some systems where ecbuild failed with a message that it could not find MPI. After a discussion with the JEDI core team, we determined that these statements are not necessary. The MPI functionality in SABER (and other repos) is implemented through fckit and eckit so the build process doe not require this check. Since the meeting this has now been merged. Please let us know if this causes any problems.

Clementine also mentioned that she is relocating the code in oops that creates the MPI communicators. She expects to issue a pull request soon.

Hailing has updated the GNSSRO converter and is making use of the new ioda functionality to group observations by record number. She is also working with Mark O to assimilate COSMIC2 data. She is also working on super-refraction QC and obs error inflation.

Mark O mentioned that H(x) for 3DVar in fv3-jedi does not currently do quality checks. Hailing had started to address this but it was never merged into fv3-jedi. Mark O intends to re-open this issue and work with Hailing to address it. He recommended that we should prioritize this highly moving forward.

Steve V is testing and benchmarking code for ODC

Steve H then formally announced something that several others at the meeting had already referred to, namely a new functionality that has recently been merged into ioda that allows users to specify how observations should be grouped together. This is controlled from the config (yaml) file through a new obsgrouping keyword in the ObsDataIn section of ObsSpace. This takes the name of a variable that will be used to group the observations together onto a single MPI task (processor element). Steve warns that to use this new feature you need a full DA flow that includes a model instantiation - it won't work for ufo tests since the grouping is not applied to the geovals file.

Steve H also reported that he has successfully built and tested the python interface for ODC. He has introduced an odc-bundle for building and testing this package from ECMWF, which is implemented by means of a python package called odyssey.

Ming wrote a Fortran program to convert prep bufr to netcdf. This works with conventional data. Anna asked if this allows for separate files for different data types and Ming responed yes. Anna asked how fast it is and Ming said it can convert a 30 MB bufr file from GDAS in 2 min. Steve H added that this is part of a broader effort to consolidate the ioda converters, making them more generic and reducing code redundancy. Ming was asked if radiances were included and Ming responded that he would coordinate with Hailing to implement this because she has already implemented such a converter.

Maryam has made progress on implementing automated CI testing for JEDI through Amazon's CodeBuild service. She can now run tests with both gnu and intel compilers using docker images. And, preliminary results suggest the tests run faster than with Travis CI. She first implemented this functionality in SABER but she is now working with SOCA to see if it can speed up the tests there.

Yannick asked if CodeBuild allows us to see the output as in Travis. Maryam said yes and directed interested parties to her SABER pull request to see examples.

During the code sprint, Anna has been implementing a new generic filter in ufo that will allow users to use the results of computations in their QC filters.

Lou brought up a potential use case for this. He said that if you have reflectivity data on the convective scale then you don't want to use other types of satellite data such as radiance because they may provide conflicting information. Anna and Yannick commented that cross-observation functionality like this might be difficult to fit into the current structure of oops but agreed that situations like this should be addressed at some point in the future.

The floor then turned to the UKMO. Steve said he has been mainly working with the LFRic model so had little to report to the JEDI group. He said they are still having problems with eckit 1.1 that have been a bit slow to resolve due in part to vacation time taken by several team members.

Then, from NRL, Sarah reported that they are working on the GetValues routines for TLAD. She then brought up the covariance structures in the model interfaces and asked if they would eventually be moved to SABER. Yannick responded that yes, much of this code will eventually be moved to SABER but we want to make sure we retain the capability for model developers to define their own covariance matrices that may be customized for their models.

Rahul reported that SOCA has been updated to work with the elimination of the config_mod in oops. However, he suggested that it still might be a good idea to put the config_mod Fortran module back into oops even though most of the models no longer use it. This will make it easier for remaining modules like LFRic to transition. We can wait until no models use it before we completely eliminate it. Yannick agreed and said he would put it back in today. When that is done, we should be able to merge the PR.

Rahul also has been working on making the SOCA interfaces more consistent with fv3-jedi so the two models can work together more smoothly in the future.

Steve S (UKMO) wanted to clarify the point about config_mod and asked if they had to do anything on the LFRic side to prepare for the PR in oops. The response was no - after we put config_mod back into oops, the PR should not break LFRic and the team can take their time in updating that functionality.

Chris H said he almost has 3DVar working for the shallow water model but he ran into an obstacle in the implementation of the Obs Filter. After some advice from Anna and Yannick, he now knows the way forward - he intends to implement an instantiateObsFilterFactory() as in fv3-jedi. It should be possible to implement this as a template in ufo so the models could call it with their traits rather than requiring all models to implement this factory. Yannick said we will work on this but it will have to wait until after the code sprint.

Meeting adjourned.