IOOS Code Sprint 2019 Activity Summary

Code Sprint Overview:

The IOOS Program Office, in collaboration with the Great Lakes Observing System (GLOS), hosted the inaugural IOOS Code Sprint October 8 - 10 in Ann Arbor, MI (for more information and materials, see: https://www.glos.us/code-sprint/).. The Code Sprint brought together over 45 software developers, data managers, and technical professionals from the IOOS Data Management and Communications (DMAC), the Canadian IOOS (CIOOS), Ocean Observatories, and other communities, to advance capabilities to publish physical, chemical, and biological ocean and lake observations. Most of the work centered on improving open source software libraries hosted on IOOS' GitHub organization (https://github.com/ioos). Specifically, work was done to standardize the software used for QARTOD implementation, query and plot timeseries datasets hosted on ERDDAP servers on a mobile-friendly app, format and display biological datasets in Darwin Core schema, and leverage emerging commercial cloud technologies like 'functions-as-a-service' for data publishing needs. The Sprint combined parallel breakout sprint workgroup time with short lightning talk sessions and a few tutorials on technical topics of interest, including Python code packaging best practices, Docker, and the Python-based Pangeo platform for big data geoscience. The Code Sprint was hosted at a coworking venue in downtown Ann Arbor that provided flexible space for teaming and full-group presentations. A slide deck summarizing the sprint activities described below is available at the Code Sprint website.

Sprints:

Note: all sprint activities used a custom GitHub label '**iooscodesprint**' for issues/PRs worked on during the sprint.

ERDDAP Development

Summary: The ERDDAP development sprint was focused on discussing needs for ERDDAP functionality enhancements and strategies for accelerating some aspects of ERDDAP development, rather than hands-on coding. Some notable ideas included: Google summer of docs for ERDDAP to develop real-world use case documentation, planning an international ERDDAP meeting or ERDDAPHackWeek, porting some elements of ERDDAP development roadmap to GitHub issues for uptake by potential contributors, and adding ERDDAP 'best practices' documentation and examples on official ERDDAP GitHub, among others. There was also discussion of specific functionality needs for ERDDAP, such as: derived/expression variables to support things like datetime variables from DD, HH, MM, ss tabular datasets, and longitude discontinuity support, unstructured grids & FMRC, and adding support for additional client-side charting frameworks in ERDDAP output (eg. Vega). Bob Simons gave an intro tutorial on working in ERDDAP Java code and provided feedback on his plans to work on some features that were discussed.

Repos/Issues/PRs:

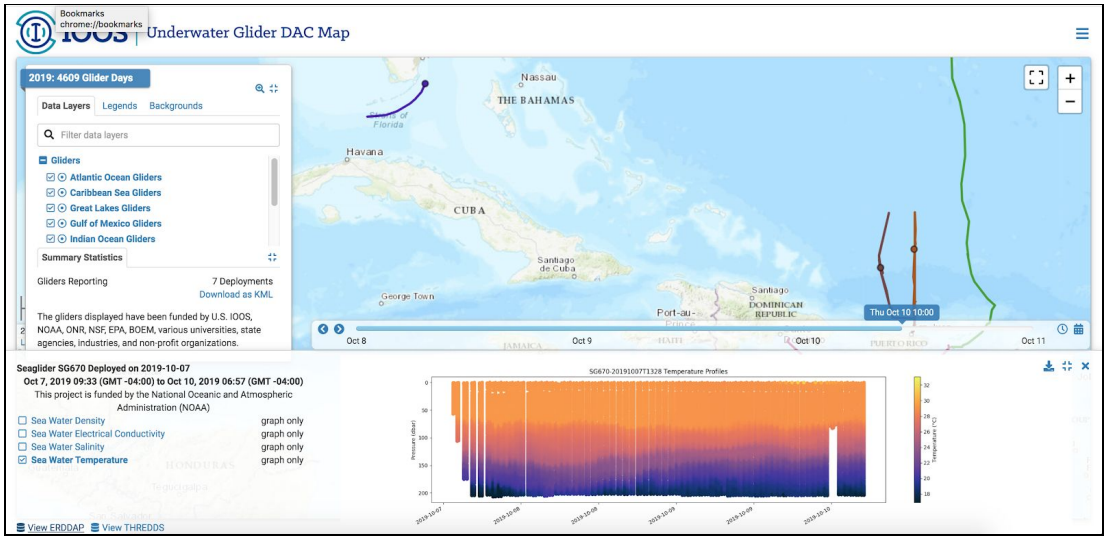
erddapy:

ERDDAP RESTful API creates some non-trivial URLs that are readable in most modern browsers but may fail when reading directly via packages, like *pandas*. During the sprint we added an optional parsing in pandas using the package request, which can read virtually any valid URL.

- https://github.com/pandas-dev/pandas/pull/28874 (allows us to pass any ERDDAP URL directly to pandas, including password protected servers via a session object.)
- https://github.com/ioos/erddapy/issues/96

Cloud Migration Challenges and Solutions

Summary: Cloud applications for DMAC workflows. The sprint combined discussion of present challenges, such as working with high-latency object storage systems common among cloud providers, and innovative cloud-only capabilities, such as functions-as-a-service, with some time for hands-on solution development. After comparing notes, the group worked on implementing AWS Lambda-based solutions for a few DMAC-related issues, including validating UDUnits strings for dataset compliance purposes and plotting Glider DAC timeseries profiles on-demand. Lambda 'layers' were used heavily in implementing these solutions.



Example timeseries profile plot on Glider DAC that can be produced using Lambda functions on AWS

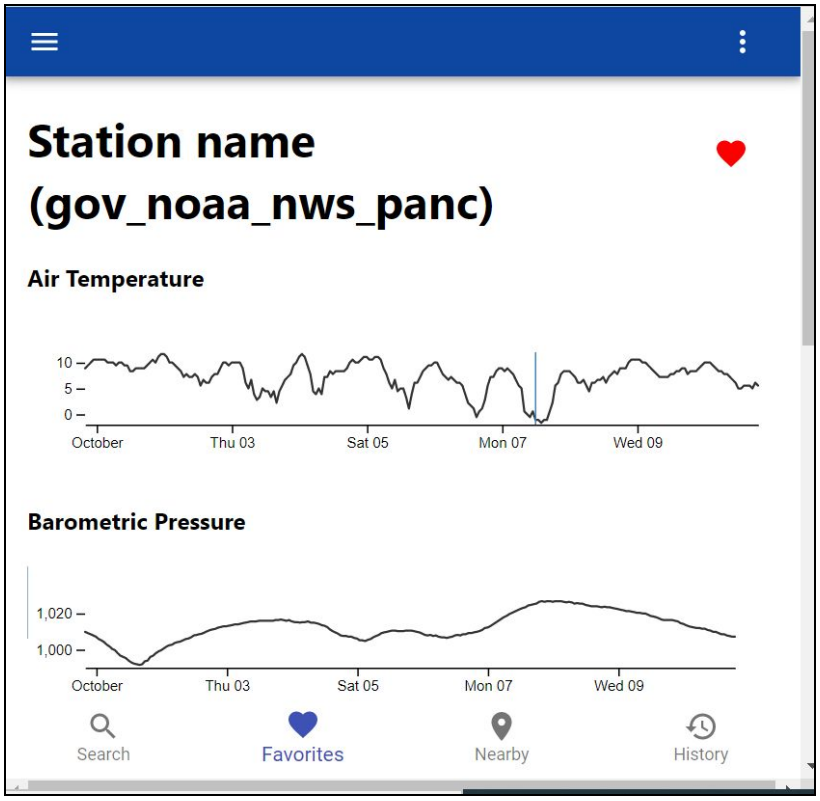
Repos/Issues/PRs:
Glider DAC Status: https://github.com/ioos/glider-dac-status

- Code Sprint Issues and PRs

Common IOOS Mobile/Web App Development

Summary: Develop a mobile-friendly application that allows users to explore recent sensor data from an ERDDAP server. This project consists of three different Javascript components that allow text-based search/filter of ERDDAP datasets, charting of timeseries data, and UI controls for mobile-friendly use (including favorite stations). The app can be customized on deployment to point at a particular ERDDAP install or set other global configuration variables.

The group developed a few demonstrations to illustrate the full capabilities of the envisioned system, including the ability to produce more advanced charts such as curtain plots of oceanographic profile timeseries data. Future work might include additional charting capabilities, embeddable charts, QARTOD flag representation, etc.



Screenshot of the ERDDAP Realtime Web App prototype

Repos/Issues/PRs:
ERDDAP Realtime Web App: https://github.com/akbstone/erddap-realtime-app

- Code Sprint Issues and PRs
- Contributors

ERDDAP JS Parser: https://github.com/akbstone/erddap-parser-js

- Code Sprint Issues and PRs
- Contributors

ERDDAP JS Timeseries Charting Library:
https://github.com/akbstone/erddap-timeseries-chart-js

- Code Sprint Issues and PRs
- Contributors

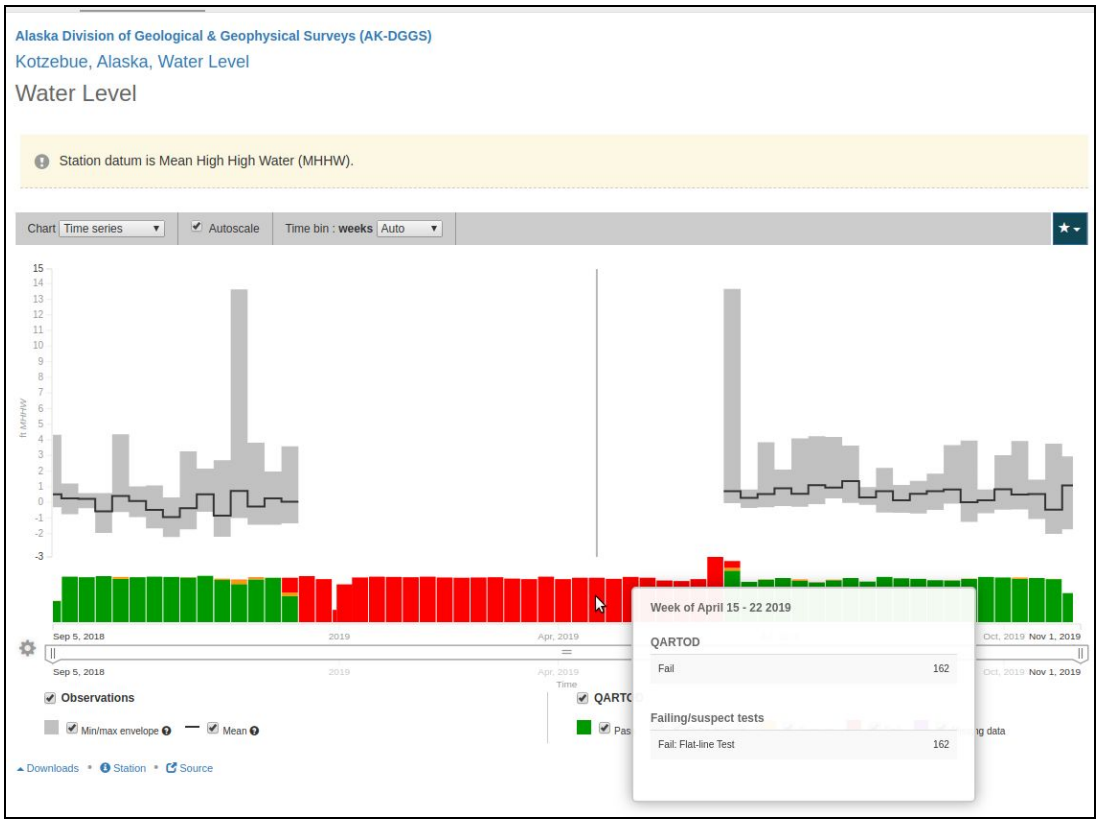
Contributions:

- Matt Brown (Mbrownshoes): Time series charts interaction
- James Munroe (jmunroe): ERDDAP parser tests and realtime app data requests
- Scott Bruce (sjbruce): ERDDAP parser tests and example output
- Michael Christensen (christensenmichael0): React code for realtime app
- Bob Fratantonio (Bobfrat): Dockerized real time app and worked on deployment pathway
- Dave Foster (daf): Method for customizing app settings on deployment, got erddap-* libraries working together, helped folks get set up

- Tylar Murray (7yl4r) : ERDDAP parser tests, created NPM package for erddap-parser-js (https://www.npmjs.com/package/erddap-parser)
- Joe Smith (JoesephPaulSmith): Time series chart tests
- Tad Slawecki (tslawecki): Researched progressive web apps and implemented a number of PWA improvements to real time app. Branch here: https://github.com/tslawecki/erddap-realtime-app
- Vicky Rowley: Researched and helped with dockerization and deployment pathway

QARTOD Library Implementation

Summary: Developing the ioos_qc library. Sprint consolidated multiple, distributed 'QARTOD' GitHub libraries into a single IOOS organization Python repo (ioos/ioos_qc). Documentation for the repo (https://ioos.github.io/ioos_qc/) was updated, test implementation and test configuration definitions in the API were improved, and best practices and approaches for choosing test configurations were discussed. Additional topics discussed included the development of an open source tool for QC configuration management, and machine learning applications to QC test parameter selection.



View of QARTOD flag status alongside water level timeseries plot at an example station in the AOOS portal

Repos/Issues/PRs:
ioos_qc: https://github.com/ioos/ioos_qc (Consolidated Python-based quality control library)

- Code Sprint Issues and PRs

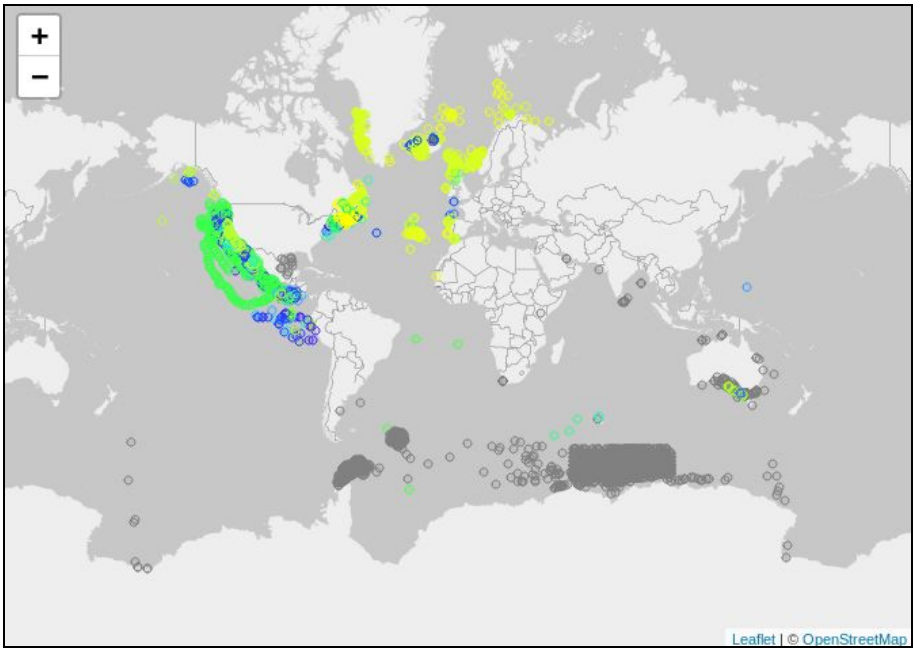
Deprecated the following community QC repositories:
https://github.com/ioos/qartod/pull/16 (library deprecation)
https://github.com/asascience-open/QARTOD/pull/71 (library deprecation)
https://pypi.org/project/ioos-qartod/ (last PyPI release with the deprecation warning)
https://github.com/conda-forge/ioos_qartod-feedstock/pull/5 (last release conda-forge with the deprecation warning)

To Do:

- Data Demo Center example
- Review docs links and ensure the docs mention it a "blessed" qartod.

Bio Data Management

Summary: The Biological Data breakout group generally consisted of members who brought 'raw' datasets and worked towards populating these datasets into OBIS using the Darwin Core metadata standard and GBIF's Integrated Portal Toolkit (IPT). OBIS-USA and GBIF node manager Abby Benson (USGS) reviewed the Darwin Core standard and the flexibility of characterizing surveys or animal tracks using a combination of Occurrence, Event and MeasurementOrFacts table. Additional discussion focused around the complementary nature of various portal endpoints for: 1) complete dataset archive (ie DataOne); 2) direct data transformation and access (ie ERDDAP); and 3) biogeographic discovery and analysis for given taxa across datasets (ie OBIS for marine or GBIF for marine and terrestrial). The Ecological Metadata Language (EML) standard can be consumed and shared across all three endpoints minimizing effort to describe data and maximizing discovery and utility for mentioned purposes. Bioschemas.org, which includes an XML schema for Taxon borrowing elements from Darwin Core, was also mentioned for its ability to enhance discovery of datasets through Google's Dataset Search, especially if paired with metadata description in CKAN a la the IOOS Catalog. This IOOS community is ideal for building out the necessary technical glue for minimizing effort of researchers to document and publish datasets for answering pressing questions about the status and distribution of marine biodiversity, particularly in a rapidly changing world (eg re-occurrence of marine heat wave in the US Pacific). CIOOS and the Ocean Tracking Network were especially active with developing a new stream of datasets to align with Darwin Core. A repository was fed with various scripts to transform data for feeding into the IPT, extract from OBIS and map (statically and interactively). Members will reconvene sometime within the next month to share code and experience.



Global map of OBIS Balaenoptera musculus occurrence data visualized in R/leaflet. More info: https://marinebon.org/ioos_bio_data_scripts/obis_subset/obis_subset.html

Additionally, some face-to-face time at the Code Sprint yielded progress connecting the MBON Portal with external Infographics developed for Integrated Ecosystem Assessments reports. A live demonstration of this capability can be seen at the links below:

- https://marinebon.github.io/fk-iea/bio.html (click on 'Marine Species' and look for plots under 'MBON IOOS Portal') or go directly to: https://marinebon.github.io/fk-iea/modals/fauna.html

Repos/Issues/PRs:

- Code: https://github.com/marinebon/ioos_bio_data_scripts
- Description: https://marinebon.org/ioos_bio_data_scripts/

ERDDAP Configuration

Summary: Swapping notes, tips and tools used for ERDDAP configuration and administration. This session started with tools-sharing presentations by several participants, and moved from there into discussion of techniques to manage ERDDAP's datasets.xml config file. An outcome of that discussion was for Bob to add the ability for the GenerateDatasetsXml script to accept a JSON-object of parameter values for easier use. Other topics included: working with data stored on cloud object storage systems such as AWS S3 and recent enhancements in ERDDAP 2.0 that allow local caching or virtual filesystem clones of S3 data for improved read performance, nThreads and other performance tuning options for reading cloud-hosted data,

useful metadata interchange formats like the NCCSV format to represent netCDF-like metadata in a CSV file and finally the IOOS Gold Standard ERDDAP repository as an ERDDAP bootstrapping tool for new users.

Links:

- BCO-DMO ERDDAP config tool - Matt Biddle (coming soon)
- Axiom/Kyle ERDDAP config tool: https://github.com/jessicaaustin/kyles-erddap-config-tool
- Rich's approach for creating datasets.xml for a bunch of similar time series data (Python/Jinja2) https://github.com/rsignell-usgs/erddap_tools

To Do:

- Bob will modify GenerateDatasetsXml to accept JSON object for parameters/args

Data Cataloging/Data Discovery

Summary: A discussion of tools available for 'civic hacker' use to discover and retrieve datasets with minimal knowledge of the nuts and bolts supporting such queries in the IOOS Catalog or other data systems. OpenAPI was discussed as a way to define a super-API wrapping CKAN's existing one, but with additional capabilities.

IOOS Client Libraries/GitHub Management

Summary: Discussion and plan for cleaning up the IOOS GitHub organization and archiving/deleting old repositories. Plans and path forward for maintaining some functionality in the pyoos package.

Repo status hackpad reference: https://hackmd.io/rYrSEAybSKKHMsvHNidGMg

Results: All recommended actions in the hackpad have been executed and IOOS Organization now has 102 repositories with most/all unmaintained repos moved to 'Archive' status or deleted.

To Do:

- Filipe and Emilio to coordinate re: pyoos readers - Filipe may investigate NERRS-sectomy from pyoos of NERRs SOAP service reader/client to standalone library.
    - Alternate approach would be to advocate to NERRS central data management office (CDMO) to replace SOAP services with ERDDAP (might require funding)

Tutorials:

Python Project Skeleton

Recorded Presentation:
https://drive.google.com/file/d/1enneRSleetDkUi-uPqUTUOacaUXf64PO/view
Slides: https://www.glos.us/wp-content/uploads/2019/11/Tutorial-1-Slides.pdf

IOOS develops and maintains a significant number of python packages in multiple repositories and across organizations (RPS, Axiom, IOOS, etc). Most of those packages are in "good shape" but some are lacking tests, auto-publication of docs and artifacts, outdated testing matrix, etc. In order to help developers to fix those issues we held a tutorial on how to "create a package from start to PyPI publication." We demonstrated a boilerplate package with the bare minimum metadata, testing, and documentation, that would ensure the package success and community adoption. The project itself is available here: https://github.com/ioos/ioos-python-package-skeleton for copying-and-modifying as one starts a new project.

Docker

Recorded Presentation:
https://drive.google.com/file/d/1X964Wffk49J-udle4C-dyslHfYQp3Mdo/view
Slides: https://www.glos.us/wp-content/uploads/2019/11/Tutorial-2-Slides.pdf

Tutorial by Ben Adams of RPS Oceans covered general use of Docker including tips and tricks for running pre-built Docker containers, making Docker contains using Dockerfile, monitoring running containers using open source tools, and deploying multi-tiered applications using Docker Compose.

Pangeo Workshop

Rich Signell led an overview of Pangeo and hands-on tutorial based on GitHub/Pangeo Binder resources.

Recorded Presentation:
https://transcripts.gotomeeting.com/#/s/13168a78bc2fd479b7021491784259de0d1ef314ecfeacf7645c131e7e5f83ab
Slides: https://drive.google.com/open?id=1by2de1FqfSRlgQwPZY336rWxIxB9FGT3
Tutorial: https://github.com/rsignell-usgs/pangeo-tutorial/tree/ioos