**Seafloor Mapping Database Requirements and Use Cases**

January, 2020

1. **Motivation**

The quickly growing number of AUV and LASS surveys need to become more discoverable and accessible, and their interconnectedness and repeat mapping require a new level of documentation and organizational discipline. This all can be done without changing the current file-based structure on the SeafloorMapping share by capturing metadata such as geographic location and server storage paths in a database and presenting it through a queryable web interface.

As our data volume grows, this need is becoming critical even for experienced users. It will be useful for users less familiar with the surveys or MB-System to access the current data products without digging around on the server. It is necessary for making sense of the data archive in perpetuity, and is advisable in the spirit of FAIR data principles (findable, accessible, interoperable, reusable).

This is not intended to change the structure or processing of the sensor data itself. Those will remain as files on the SeafloorMapping share.

Because that share on Titan is web-accessible, metadata and data products can be mined for populating the database and viewed through the query interface.

The quality[[1]](#footnote-1) of particular survey data influences the prioritization - or even inclusion - for working with it and make data products. Currently this is tracked by users in their own spreadsheets and datalists. It needs to become an assessment that is accessible and clearly presented.

Likewise, whether a survey is repeated over a previously mapped area also needs to be tracked.

1. **Functional requirements**

Relational database with geographic capability (could be Microsoft SQL-Server or PostgreSQL with PostGIS extension[[2]](#footnote-2) as STOQS)

Maintained and backed up by IS (does this preclude using PostgreSQL?)

Web-based query, appearing like STOQS (campaign list) or the samplesDB (multiple queryable items), from Canyon Head (language would be Django if PostgreSQL; what language if MS SQL Server?)

Interface for populating the database needs to be useable by knowledgeable AUV data processing staff on Mac computers, and not expensive (argument for using PostgreSQL).

1. **Use cases**

Retrieve a list of AUV or LASS surveys with a query on Canyon Head (e.g., web-based), by geographic area (e.g., lat/long box), at a place by name, on an expedition, between dates, for a PI, …

Construct from that list an MB-System datalist that includes the accepted prioritization1 for surveys.

View a JPG thumbnail map made from each mission’s ZTopo.grd from the query results page, and download a full-rez version.

View survey sites on a map in the browser (as survey start points or as tracklines as with STOQS). Download a KML file made from a mission or regional compilation for use in GoogleEarth.

View the path to the mission directory, or the expedition’s Figures directory, or the current working directory. Click to open a Finder window or terminal window to that directory (or if that’s not recommended, just allow the path to be copied).

View or download a mission’s Notes file[[3]](#footnote-3). (Getting to the directory, above, may be enough.)

If the mission is part of a regional compilation[[4]](#footnote-4), find where on SeafloorMapping the most current working directory is (e.g., where navadjust work is being done…may be years later).

See what data or processing issues should be considered. Predict what prioritization to give a mission in a datalist.[[5]](#footnote-5)

Record the submission of data to repositories and the DOIs received. Record publication citations and DOIs resulting from the data.

1. **Processes (some may be just dreamin’)**

Generate a current set of ZTopo\_mission.grd netCDF grid and .jpg files to reside in each mission directory (which should include the mission name in their filenames for uniqueness of downloaded files). These steps are done now as processing steps, but could also be updated with a recurring scheduled script. Script could also make a KML file, which currently isn’t done.

Capture and populate the path to mission directory, for thumbnail web display of ZTopo.jpg and to download ZTopo\_mission.kml. An update\_status code can trigger relevant scripts.

Run gmt grdinfo on ZTopo.grd in mission directories to populate DB with the four corners of the grid (or other geometry fields); or have a load script mine .inf files.

As new missions accumulate, populate DB with new records. Send an alert if new missions are discovered in the SeafloorMapping directories but aren’t yet in the DB. Have also a web form for manual entry, or a load script to extract other info from Notes file2.

Generate JPG thumbnail, KML file, and GIS products (geoTiffs and ASCII raster files) for each of the regional compilations (needs to be updated as changes are made; update\_status code can trigger script).

1. **References**

FAIR Data Principles  
<https://www.force11.org/group/fairgroup/fairprinciples>

Microsoft SQL spatial data types  
<https://docs.microsoft.com/en-us/sql/relational-databases/spatial/spatial-data-types-overview?view=sql-server-2016>

PostgreSQL  
<https://www.postgresql.org>

PostGIS extension for PostgreSQL  
<https://postgis.net>

PostGIS developers manual  
<https://postgis.net/docs/manual-3.0/>

Django   
<https://www.djangoproject.com>

STOQS database (default view is a list of campaigns aka expeditions)  
<https://stoqs.mbari.org>

STOQS data model  
<https://github.com/stoqs/stoqs/blob/master/doc/stoqs_model.png>

SamplesDB query web page  
<https://mww.mbari.org/samplesDB/Queries/>

Example of SamplesDB query return page (unfortunately the tech used here is obsolete)  
<https://mww.mbari.org/samplesDB/Queries/returnTable.asp?Conjunction=AND&qType=1&qDiveName=272&qRovName=Tiburon&qSampleRefName=&qSampleRefNameLike=&qSampleID=&qDate=&qDate1=&qDate2=&qYYYYDDD=&qYYYY=&qWaypoint=&qWaypointLike=&qChiefScientist=&qCollector=&qEqpt=manipulator&qKeyword=&qLat1=&qLat2=&qLon1=&qLon2=&qDepth1=&qDepth2=&qAnal=&B1=Submit+query>

SamplesDB design diagram  
<https://mww.mbari.org/SamplesDB/documentation/Default.htm>   
 (select “Design diagram”)

Rolling Deck to Repository (R2R) data quality assessment  
<https://www.rvdata.us/about/quality-assessment>

1. Quality is a subjective thing. Data quality might be degraded because of any number of sensor problems or difficulties processing the data. It is envisioned that the quality would be presented here more as a warning to users that there could be a problem, than populating a set of fields with a pre-set vocabulary. [↑](#footnote-ref-1)
2. The format for latitude and longitude with PostgreSQL appears to be degrees minutes seconds written as if decimal degrees. Is this a deal-killer? [↑](#footnote-ref-2)
3. The Notes file in each survey directory is a running log of the day, the operations checklist for that survey, and the first documentation for each survey. Some of it might be predictable enough for a load script to discover. [↑](#footnote-ref-3)
4. Assumes there’s just one regional compilation, because the navigation for all the surveys in it is defined by the current navadjust project. The path to it and GIS and etc products would have to be updated in the DB if previous compilations are deprecated. End products of repeat surveys should be stacked rather than gridded with the quality prioritization. [↑](#footnote-ref-4)
5. For now, quality assessment is proposed as a comment field where a list of issues could accumulate. It also could be a suggested order of magnitude for datalist prioritization. Our issue tracking could be expanded to include specific metrics, as done by the Rolling Deck to Repository (R2R) database. A user can always change the prioritization for a survey in their own gridding datalist. [↑](#footnote-ref-5)