OceanCube Operations and Maintenance

Introduction:

As with many projects, OceanCube started off with a certain level of expectations and has grown in complexity and capability. OceanCube's initial design provided the capability to scrape the internet for daily distributed oceanographic model data and to display this data graphically via a web interface. The graphic representations are provided in 2 forms. The 1st form is as a cube with each depth layer displayed as an image of the parameter (currents, salinity, water temperature) with the composite of these layers forming the cube that can be rotated to observe the displayed layers from differing viewing angles. The second form presented a 2D graphic of the parameter at a time and depth. Both options allow for a movie-loop progression through the model data based on forecast times. The OcenCube has since expanded to provide client-side interpolation of the model data for a user defined region or cross-section, generation of temperature, salinity, current and sound velocity profiles for any point, and the display of near real-time trackline data from reporting operational vehicles. OceanCube has now been combined into the cubeNET project that extends capability by providing near realtime access to oceanographic instrumentation data. These data are retrieved via communication channels, loaded into a relational database, and linked to a Gafana application for display and download. This document will focus on the major software components of the OceanCube and generally describe the function of each component.

General Overview:

The first step in generation of an OceanCube model web interface is in the gathering of the data. There are generally 2 forms of data to display which each has slightly different processing requirements. The most frequent type of data is oceanographic model forecast data which predicts the expected conditions in the future starting at the time of the model run. Examples of these are the HYCOM and AMSEAS models that forecast or predict the oceanographic environment out to 96 hours. The other major type of model data is in the form of observation data that represents a generalization of the oceanographic data based on recent observations. Examples of these are the GOES-16 satellite observation data and HF-RADAR. For the observation type of model data, the generated graphics start at the last or most recent observation and goes backward in time for previous observation data over the past hours or days. There is also a 3rd type of data for which graphics are generated and displayed across multiple model web interfaces. This data type includes the bathymetric data layer which is fixed and unchanging and the chlorophyll data graphic which is single graphic derived daily from VIIRS satellite data if sufficient coverage exists.

Frequently Used Arguments:

Most of the software modules that are used to generate the model web interfaces are written as python scripts. Among these scripts there are some commonly used arguments. These are described below:

a. **areadID**

The areaID argument is used throughout nearly all of the OceanCube applications and is actually a key into the main OceanCube configuration file, cube.json (currently located at /user/braud/cubeenv) . AreaID is needed to acquire numerous parameters that may be specific for a particular area and model. Below is the entry for the areaID "USM_HYCOM":


```
{
  "STREAK_LENGTH": "0.5",
  "CHLOROPHYLL_COLOR_STATIC": "1",
  "AREA_ID": "USM_HYCOM",
  "HEIGHTS_MIN": "-4",
  "TEMP_BOTTOM": "0",
  "HEIGHTS_MAX": "4",
  "TEMP_MAX": "100",
  "AREA_NAME": "USM Test Range, MS",
  "CURRENTS_MIN": "0",
  "Y_AXIS_TICKS": ".025",
  "MODEL_AREA_ID": "",
  "DERIVED_FROM": "",
  "X_AXIS_TICKS": ".025",
  "MODEL": "HYCOM",
  "X_AXIS_GRIDLINE": ".25",
  "CURRENTS": "1",
  "DISTRIBUTION_STATEMENT": "T",
```

"HEIGHTS_CONTOUR": "1",

"HEIGHTS_INC": "1",

"TEMP": "1",

"BOTTOM": "0",

"SALINITY_MIN": "30",

"SALINITY": "1",

"DISTRIUBTION_REASON": "Admin/Ops",

"STREAK_SPACING": "0.1",

"CURRENTS_INC": ".25",

"SOUTH": "29.5",

"CLASSIFIED_BY": "",

"TEMP_MIN": "50",

"SALINITY_BOTTOM": "0",

"EAST": "-88",

"RESOLUTION": "3.5",

"COASTLINE": "DEFAULT",

"TEMP_INC": "5",

"CLASSIFICATION": "UNCLASSIFIED",

"CURRENTS_CONTOUR": "0",

"FILL_LAND": "1",

"CURRENTS_VECTORS": "0",

"SALINITY_INC": "2",

"HEIGHTS": "1",

"DEPTHS": "0,2,4,6,8,10,12,15,20,25",

"TEMP_CONTOUR": "0",

"DISTRIBUTION_OFFICE": "University of Southern Mississippi",

"AREA_TITLE": "University of Southern Mississippi Ocean Cube",

```
    "CURRENTS_BOTTOM": "0",
    "ACTIVE": "1",
    "NORTH": "30.5",
    "TEMP_OVERLAY": "CURRENTS",
    "SALINITY_MAX": "40",
    "Y_GRID_ANNO": ".25",
    "SALINITY_OVERLAY": "CURRENTS",
    "CHLOROPHYLL": "CURRENTS",
    "STREAK_KEY_MAX": "3",
    "WEST": "-89.4",
    "X_GRID_ANNO": ".25",
    "Y_AXIS_GRIDLINE": ".25",
     "PLATFORMS": "IVER3-3089,Triton_1,Triton_3,Triton_4,Triton_5",
     "HIGH_RES": "GLP_NGOFS2",
    "WAVES": "0",
    "CURRENTS_MAX": "1",
    "POINT_LIST_DESTINATION":
"WaveRider,HRSN,WYCM6,USMR1,PTBM6,PNLM6,GDXM6,CRTA1,DPIA1,KATA1",
    "SALINITY_CONTOUR": "0",
    "HEIGHTS_OVERLAY": "CURRENTS"
  },
```


Some of the most important parameters in this list are the MODEL (HYCOM) and the bounding rectangle of the area represented by “NORTH”, “SOUTH”, “EAST” and “WEST”. Many of the parameters in this file are used for the display generation. They define what data will be plotted, whether overlays such as current vectors will be used, and what depth layers will be displayed. A recently added parameter in this list is “HIGH_RES” which is a list of areaIDs that are also keys to sets of parameters within the cube.json file. This creates a parent/child relationship that allows for the display of the bounding box of the child model (typically data at a higher resolution) and a way to hyperlink to the child model from within the parent.

Another recently added parameter is “PLATFORMS”. This is a list of the platform names stored in the OceanCube relational database. These names are used to perform database queries from the OceanCube client to retrieve up-to-date trackline and instrumentation data from communicating collection platforms.

The “POINT_LIST_DESTINATION” parameter is actually a list of identifiers into a second configuration file, IMD_POINTS_0.json (also currently located at /users/braud/cubeenv), that contains parameters for a station or point that will be displayed across all of the generated model graphics. These parameters determine the location, symbology, color, size, and label for a displayed station as well as a hyperlink that may be used by the OceanCube client software to obtain additional information for a station. Below are the parameters stored in the IMD_POINTS_0.json file for the WaveRider station:


```
{
   "AREA_ID": "WaveRider",
   "DESCRIPTION": " USM WaveRider Buoy",
   "AREA_NAME": "USMWR",
   "IMD_LABEL": "USM WaveRider Buoy: USM-WR",
   "IMD_SIZE": "9",
   "LONGITUDE": "-88.79614",
   "IMD_SYMBOL": "c",
   "ACTIVE": "1",
   "LINK": "http://oceancube.usm.edu:3001/d/Qj2-9m3Gk/waverider?orgId=6",
   "IMD_POINT_TYPE": "USM Instrumentation",
   "LATITUDE": "30.07574",
   "IMD_COLOR": "green",
   "MAX_DEPTH": "20",
   "PLATFORM" : "Buoy 74112"
},
```


b. -dtg YYYYMMDDHH

This parameter is used to represent a datetime parameter where YYYY represents the 4-digit year, MM represents the 2-digit month, DD represents the 2-digit day, and HH represents the 2-digit hour in the range of 00 to 23. The -dtg is only used for python applications where the parameter may be optional. The data retrieval applications that are run periodically as "cron" jobs can not use a fixed datetime argument and therefore generates a value based on the current date and time.

c. -start NNN

This is an optional integer argument allows for the model web application to start on a TAU other than 0 (valid time of the model).

d. -end NNN

This is an optional integer argument allows for the model web application to end on a TAU other than the maximum TAU for the model.

e. -nproc NNN

The number of simultaneous processes or processors to use.

OceanCube Data Flow:

a. -start NNN

This is an optional integer argument allows for the model web application to start for a TAU other than 0 (valid time of the model).

OceanCube Data Flow

The wiring diagram in Figure 1 include all of the major components of the OceanCube system used to generate the model web interfaces for the various incoming model data. The objective in the early stages of processing is to gather the model data from various sources and manipulate the data into structures and formats such that a common downstream processing flow can be used.

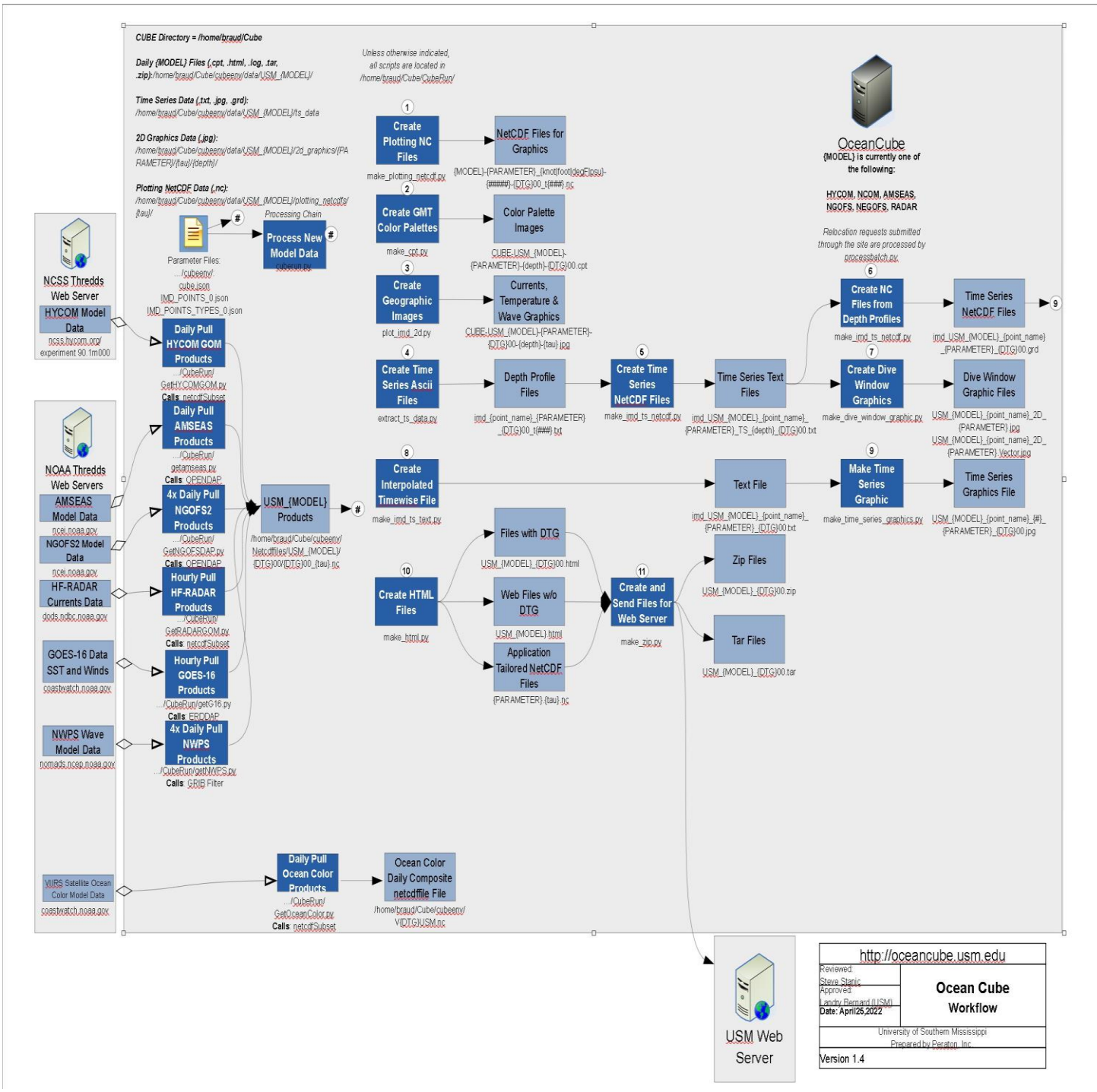

Figure 1. OceanCube Wiring Diagram

1. Gathering the Data

The data from the various models is acquired through the running of tailored python applications for each model. These applications are setup as "cron" jobs that currently run on the Cthulhu Linux server operated by the University of Southern Mississippi. As "cron" jobs, they are run at the frequency of the incoming data at the approximate times that new data is expected to become available. When the requested data is not yet available from the remote data supplier, the request fails and a time delay is implemented prior to repeated tries. After the data is downloaded, some manipulation is normally needed to prepare the data for the common downstream OceanCube processing. In the simplest case if the data can be obtained through a site that supports netcdf subsetting, a single netcdf file containing all the oceanographic parameters for an area can be downloaded with a single call. This is the case for the HYCOM and GOES-16 model data. In a slightly different case, if the data is distributed through a DODs server, all of the data can be downloaded with a single call but it must be re-written into a netcdf file for further processing. This is the case for the AMSEAS model. In the case of the NGOFS2 data, each TAU (time increment) of a model run is stored as a separate file and must be downloaded individually. Once all of the TAUs are downloaded, they are combined into a single netcdf file. Other format conversions must be made for HF-RADAR data which is obtained as ascii data and NWPS which is obtained in a GRIB format. If the data is unstructured, such as can be obtained from the NGOFS2 field data, higher resolution grids can be created by gridding the denser points available in the shallow water and producing netcdf formatted files. All of these data gathering modules have an optional parameter "-launch". If this parameter is used, it indicates that once the data has been downloaded, further processing to complete the creation of the model web pages for this data is to be initiated. Depending on the data type, there are two similar processing paths that are taken. The CubeRun module is used to complete the processing of all oceanographic forecast model data and RadarRun is used for the observational models that process recently acquired historic data. RadarRun is used to process HF-RADAR and GOES-16 data. The data gathering modules and the frequency of their execution are portrayed on the left side of the wiring diagram in Figure 1.

2. Making Adjustments

The next step in the processing path is to produce numerous individual netcdf files that will latter be used for each graphic image produced. A single netcdf file is created for each parameter type (salinity, temperature, etc.), at each depth, for each TAU. During this step, unit conversions are also performed. Degrees Centigrade are converted to degrees Fahrenheit. Meters of height are converted to feet. Meters per second are converted to knots. In addition to the unit conversions, for vector data (such as wind and currents) two forms of netcdf files representing the data are produced. The first form consists of netcdf files that contain the 'u' and 'v' components of the vectors, and the second form are files for the speed and direction of the vectors. The module, make_plotting_netcdf is responsible for the generation of all of these individual netcdf files and is depicted at the top of Figure 1 as step 1.

3. Normalizing Colors

If left on its own, the program used to create the individual graphic depictions for data types (Generic Mapping Toolkit, GMT) would generate the full range of colors for each graphic produced. This would create inconsistencies between the colors used for each TAU and each depth for a parameter. To correct this problem, color pallets for each parameter are created by scanning all of the individual netcdf files of a parameter and building a single-color pallet that spans the minimum to maximum values found for that parameter. The module responsible for generating the normalized color pallet for each data type is make_cpt and is depicted as step 2 in Figure1.

4. Generating images

For step 3 in the wiring diagram, the graphic images for each parameter, at each depth, for each TAU are produced by the plot2d software. This module uses the GMT library for producing the image. In addition to the main function of producing the colored visualized image of each of the plotting netcdf files, this program is responsible for all of the annotation associated with that image. This includes the header, footer, title, graticules, tick marks and legends. It also produces the land mask, vector current overlay (if selected) and plots the location of the stations or points of interest. Many of the specific configuration parameters for the area plot are stored in the .json files that are associated with this areaID. Figure 2 depicts aplot2d graphic produced for surface salinity for the NGOFS2 model data for the USM test range.

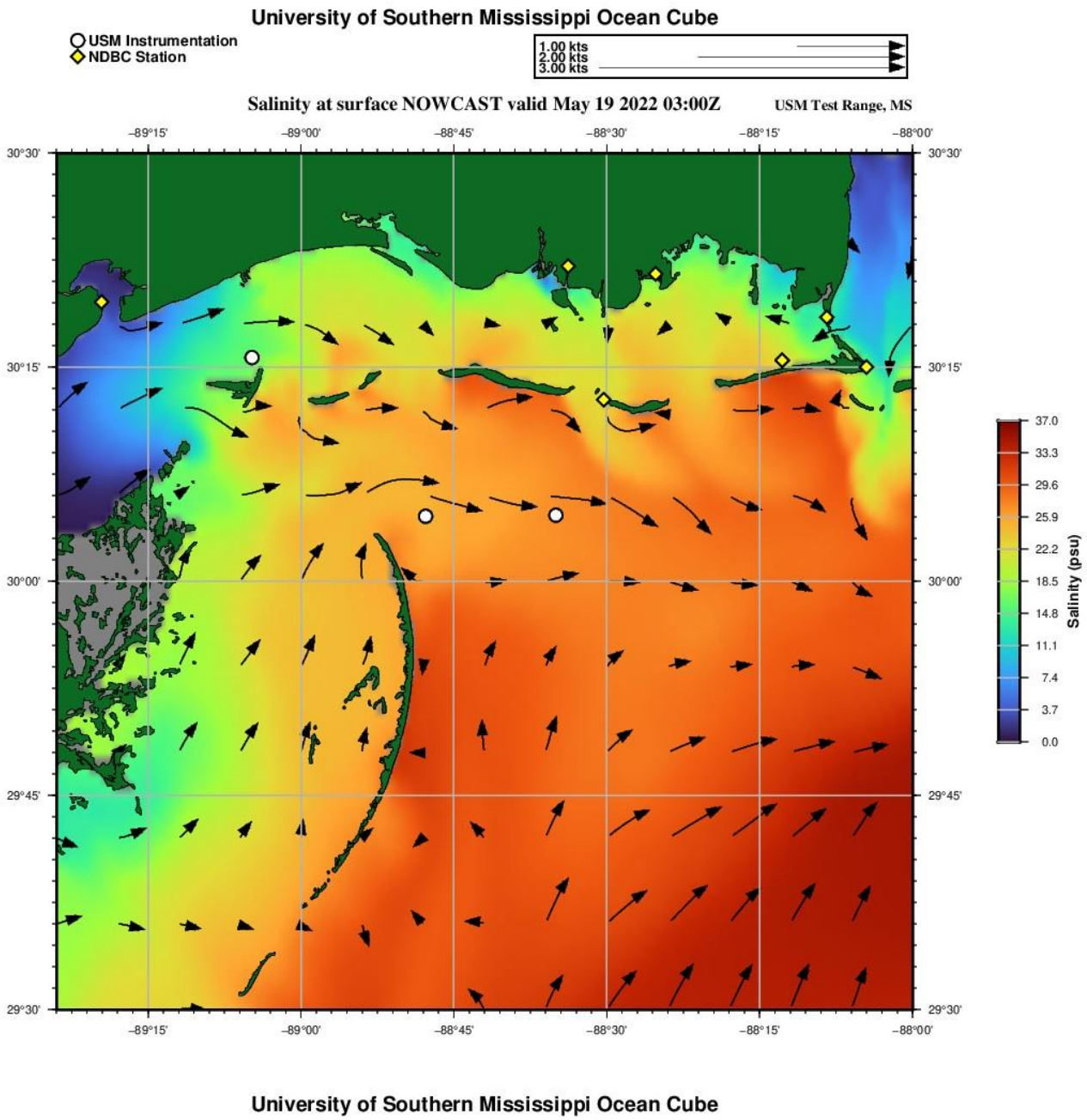

Figure 2. Image generated by plot2d

5. Building Supplemental Graphics

The modules 4 through 9 as depicted in the Figure 1 represent a throwback to the limitations of the original OceanCube design. In the initial concept, the client-side software had little to no access to the underlying data or netcdf files that are used to generate the graphics. Only the completed images were downloaded. To enhance the usefulness to the user, fixed graphics are generated at the station locations based on the interpolation of the underlying netcdf files to depict variations of a parameter over time or over time and depth at that station's location.

Once the decision was made to allow for the download of the underlying parameter data as netcdf files to the client side, the client-side capabilities were vastly enhanced. Had that been the original design, the functionality of modules 4 through 9 would have certainly migrated to the client-side and the capability to generate these graphics could have extended to any point in the area rather than at fixed station locations. Figure 3 depicts a sample of the imagers that are produced by modules 4 though 5 and modules 8 and 9 of the wiring diagram. This graphic is a plot of the salinity over time at the location of the USMR1 buoy for the duration of the NGOFS2 model run.

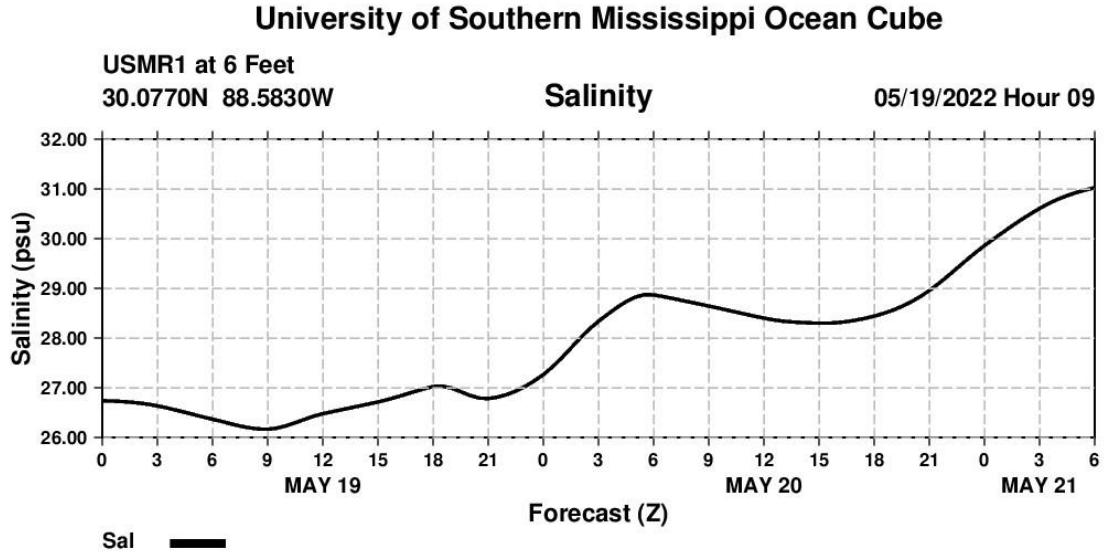

Figure 3. Graph Image of Salinity vs. Time at a location

In addition to modules 4 through 5, modules 6 and 7 are used to generate the 2D dive window graphics as depicted in Figure 4. As the dive window name implies, this graphic was originally developed as a planning tool for divers to assess the currents direction and magnitude at depth over time for a location.

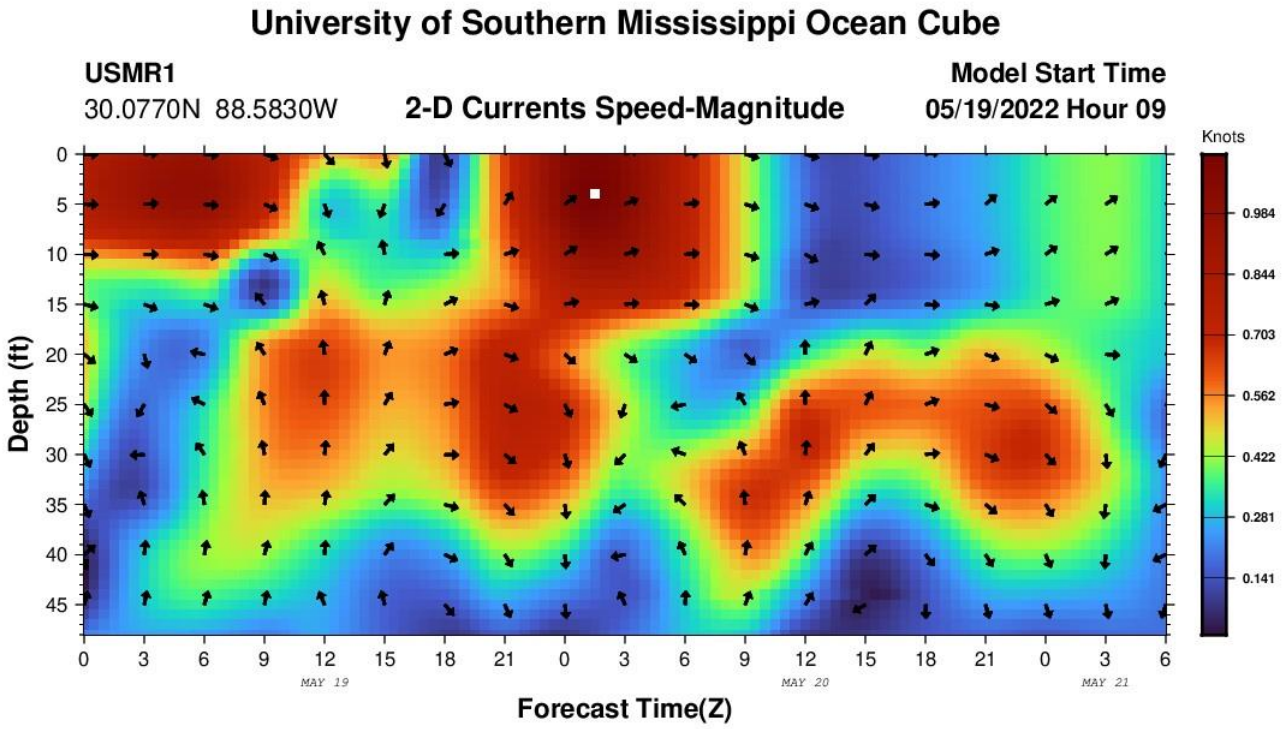

Figure 4. Dive Window Image at a location

6. Generating the HTML

Module 10 of the OceanCube wiring diagram is the creation of the root page for each of the model web interfaces. The output file is given the name of the areaID with the .html extension. This step is performed by the make_html python script. The output of this program is a html file that contains both the mark up code to define the layout and structure of the web page along with the thousands of lines of javascript code that provide all of functionality of the client-side application. Some of the functionality has been mentioned previously, but a full description is beyond the scope of this document. Please refer to the online documentation available for download in the banner section of the cubeNet web for a full description of OceanCube capabilities. Much of the recent added capability of the OceanCube takes place through the expansion of this software module to increase the functionality of the javascript code that runs within the user's browser.

7. Packaging and Distribution

In step 11, nearly all of the files needed to assemble the web site have been created. The main exception is the netcdf files containing all of the underlying model data that were used to

create the graphics. As it steps through the assembly of all of the required files used by the web application, the module make_zip calls the module densify to create a single netcdf file for each data type for each TAU. If multiple depth layers are available for the data type, they are consolidated into a single netcdf file. (This is driven by an original requirement to rapidly generate a graphic display of profile data for any point in an area.) The make_zip module assembles all of the .jpg files that have been previously generated, any required javascript libraries, ancillary data files for bathy and chlorophyll, the newly created netcdf files and the root html file. A single .zip file is created from this assembly that can be distributed to the web server. This .zip file is then copied and extracted to the web server in 2 locations. The primary location is the directory on the website that is identified by the areaID. This accomplishes an immediate update of the web site with the latest data from the model. A second copy is made and extracted to a new directory under the archive section on the web site with the areaID name combined with the datetime. This creates a permanent record of the web application that is available for the future and will not be updated as new model data becomes available.

Summary:

This document provides a very high-level description of the processes involved in the generation of web interfaces for the OceanCube. It identifies the primary configuration files that allow the operator of the OceanCube system to make many changes such as creation of new areas or the addition and display of points of interest within an area without the need for programming changes. The wiring diagram identified in Figure 1 is explained and a walkthrough of the process flow with a description of each of the modules involved is presented. Samples of the server-side generated graphics are provided as a demonstration of the types of graphics produced and made available to the web interface. Finally, the creation of the total web application and its distribution to the web server is described.