Florida State University

College of Arts and Science

SAMOS 2.0:

High Resolution Meteorological and Oceanographic Data Processing

By Geoff Montee

February 2012

A project submitted to the

Department of Computer Science

In partial fulfillment for the requirements for the

Degree of Master of Science

Major Professor: Dr. Daniel G. Schwartz

Committee: Dr. Mark Bourassa, Shawn Smith, Dr. Andy Wang

Acknowledgements

As very few people can create something significant without the help of others, I would like to acknowledge some people who have made SAMOS 2.0 possible. First off, I would like to thank Shawn Smith and Mark Bourassa at the Center for Ocean-Atmospheric Prediction Studies (COAPS) for allowing me to work on the project and for providing the guidance that allowed me to actually get as far as I have with it. I would also like to thank my professors in Computer Science at Florida State University: both those who are on my committee and those who have taught me in classes. Both groups have helped me develop the skills to get me to this point in life. I would also like to thank Bill Fanning at the University of Rhode Island and Toby Martin at Oregon State University for their work on developing new transmission formats to encapsulate high resolution data for SAMOS 2.0. I would also like to thank my wife for supporting me through this endeavor—especially during the all-night coding sessions that are almost inevitable in graduate school.

Abstract

This document outlines the next generation of the Shipboard Automated Meteorological and Oceanographic System processing system.

The first generation of SAMOS data processing accepts 1-minute averaged data from volunteer research vessels using e-mail as the main method of delivery. The data files, which are currently accepted in a couple different formats, undergo automated and visual quality control and are converted to netCDF format. Metadata is stored in the SAMOS database, and updates to this metadata must be made manually.

The second generation of the SAMOS system is intended to improve upon several key aspects of the original, by: providing access to the raw data recorded onboard the vessels, which is typically higher resolution (e.g. data measured at a frequency of 1 Hz or more) to be used in the earlier stages of the processing; automatically updating metadata upon file delivery using self-describing files; and providing more versatility in original file format options available to ships using a carefully-designed programming interface with generic components.

Access to the raw data provides several advantages to ship operators and the SAMOS Initiative. For ship operators, providing the raw data to be processed by the SAMOS system, rather than one-minute averaged data, means that the ship's Data Acquisition Systems (DAS) are required to do less processing onboard. More importantly, every ship operator does not have to expend the man hours to develop and maintain one-minute averaging code for their DAS. For the SAMOS Initiative, it means more consistent methods for processing the data from its raw form to the final SAMOS data product. In addition, the high frequency data allows the SAMOS system to calculate values that are not feasible with averaged data, such as true winds, radiation, and moisture.

1. Introduction

1.1 SAMOS 1.0

The original SAMOS system started as a small project, with big aspirations for its future. Over time, features have been added to the existing system in order to further the goals of the project. However, with the current system, the practical limitations of building onto some of the existing code are quickly being reached.

The goal of the SAMOS Initiative is to provide meteorological and near-surface oceanographic data recorded onboard research vessels to the public in a standard, platform-independent format. The end products are divided into 3 categories of quality control: preliminary, intermediate, and research. In each case, the data is provided in the Network Common Data Form (netCDF) standard format developed by Unidata. This binary file format is portable between platforms, and allows for storage of metadata and data in the same file. The netCDF files generated by the SAMOS system contain essential metadata in each file's header and quality-control flags for each data point. Data is typically sent to the SAMOS Data Center once a day—usually shortly after midnight UTC.

Preliminary quality data is created by: converting the vessel's one-minute averaged data into pre-defined units for each particular variable; assigning each variable a pre-defined identifier; fetching the vessel and instrument metadata from a SAMOS database that is manually maintained; writing the converted data and stored metadata to a netCDF file for each day of data sent by the vessel; and performing automated quality control on the resulting files. These files are distributed to the public as version 100 files.

Intermediate quality data is created by merging all files for a given ship and day into a single daily file for that ship. In the case that multiple data points exist for a given minute, the data point with the best quality "flag" is used. This merge process currently happens on a 10-day delay to allow time to receive data delayed by ship-to-shore communication problems. There are also plans to implement statistical quality control during this process. These files are distributed to the public as version 200 files.

Research quality data is created when analysts visually inspect the intermediate quality data and flag additional data points where they suspect the data is invalid. These files are distributed to the public as version 300, 301, 302, etc. files, depending on how many times the original intermediate file is viewed by the analyst.

Data quality can be flagged for consistent issues that may warrant notifying the ship operators, such as instrument malfunction. However, data can also sometimes have fleeting quality issues that are expected, such as when a bird lands on the met tower, causing disturbances in the wind data measurements. Regardless of the quality of the data, a data point is never removed from the netCDF file (aside from duplicate time elimination during the merge process). Instead, an appropriate flag is set for the data point in the file.

Through the years that the SAMOS Initiative has been in operation, the SAMOS Data Center has also discovered several key concerns that, if addressed, could lead to a much better product. One of the biggest hurdles in the SAMOS 1.0 system is metadata management. Any changes to vessel and instrument metadata must be made manually. Since the system processes data in an automated manner using files sent from the vessel, it would be useful for the system to also automate metadata processing in a similar manner. This adds more complexity, but the end result would be a more flexible automated system.

The SAMOS 1.0 system receives one-minute averages from research vessels. Onboard the vessel, raw data is typically recorded at a frequency of around 1 Hz, in millisecond resolution. The vessels then average the data themselves and send the result to the SAMOS Data Center. This is not an ideal arrangement for any party involved, primarily because it results in more work for the vessel operators who have to average the data. In addition, the SAMOS Initiative can do much more with the raw data. It would be much more useful for the SAMOS system to receive the raw data and perform all quality control and reductions. This would result in a more consistent product for end-users of the data.

1.2 Motivation for SAMOS 2.0

As stated in the previous section, there are disadvantages in the way that metadata and data are currently provided to the SAMOS Data Center from vessel operators. The SAMOS 2.0 system is primarily aimed at overcoming these disadvantages.

First, in SAMOS 1.0, vessel and instrument metadata must be manually entered into the system by the vessel operator or an analyst at the SAMOS Data Center. SAMOS 2.0 fixes this issue by supporting self-describing files that include metadata about the source vessel and instruments. This allows less human intervention and more automation in the processing.

Second, SAMOS 1.0 requires vessel operators to form one-minute averages from the raw data before sending it to the SAMOS Data Center to be processed. SAMOS 2.0 provides support for raw data in millisecond resolution. Thus, the burden on vessel operator's to maintain averaging code is no longer an issue.

Access to the raw data also provides advantages to the SAMOS Initiative. Data gathered using finer time resolution allows for more accurate calculations of other kinds of data from the source data. For example, true wind speed and direction can be calculated from platform-relative wind speed and direction, and vessel speed, course, and heading.

Additionally, since the SAMOS 2.0 system does all of the reductions and averaging, the end-users can expect more consistency among the data products, even if the source vessels have different data acquisition systems and instruments.

1.3 Significance

The improvements to the SAMOS system are very significant. For vessel operators, the SAMOS 1.0 system essentially acts as a means to quality-control their data and to provide it to the public in a standard, well-defined format. The SAMOS 2.0 system adds a few other major benefits for vessel operators. The system works with the raw data to derive more complex data, average it, and keep the metadata up-to-date. Basically, if a vessel operator can adopt a Data Acquisition System that can work with a self-describing file format compatible with SAMOS 2.0, they could save themselves a lot of effort and make their data more useful to the public.

The benefits affect end users as well. The addition of true winds calculations will provide more accurate data for their use. In SAMOS 1.0, many ship operators calculate true winds on their DAS. However, the calculations are complicated and aren't always done correctly, due to their nonlinear nature and their use of vectors. The addition of the true wind calculations to SAMOS 2.0 removes this burden from ship operators and provides more consistency and accuracy for users of the data. There are also plans to add other complex data calculations to SAMOS 2.0, such as equations intended to deduce fluxes and dew points. There is incredible potential for powerful quality-control algorithms as well when dealing with high resolution data, and accommodating these algorithms would result in more reliable data.

For the SAMOS Data Center, SAMOS 2.0 means more automation in the system from start to finish. The days of manually entering vessel and instrument metadata into the database through a web form should be limited in the future so that analysts and developers can spend more of their precious time on their other tasks, such as analyzing the data for quality and providing real-time feedback to vessel operators.

2. System Design

2.1 SAMOS 1.0 Design

The SAMOS 1.0 system has undergone many significant changes over the years. Many components have been added or modified to provide a better data product to the end users, or to provide benefits to vessel operators and SAMOS Data Center personnel. For the sake of simplicity, this document will only cover the current version of the system, rather than discussing the history of the design and the system's various iterations.

Most of the SAMOS 1.0 system serves its purpose well. The original system can logically be divided into several components:

1. Database and Metadata Management

   The database used by the SAMOS system is hosted on a MySQL server. Metadata is stored in this database and is primarily manually updated through a web interface designed specifically for that purpose. The same database stores information that tracks the progress of specific original and daily files through the processing, as well as quality-control information for any resulting files. The database is also used to verify trusted senders (and even blacklisted ones) upon data delivery. This component is used by every other component of the SAMOS system.

2. Data delivery

   The current method of data delivery is implemented using e-mail. Files are sent to a specific e-mail address, which has all of its incoming messages routed to a script which handles the data processing after verifying that the sender is trusted.

3. Data conversion

   If the data file is in one of the system's compatible formats (either JGOFS or SAMOS formats), then the data is converted to an intermediate format (COAPS ASCII format). All data of interest to SAMOS is given pre-determined variable identifiers consistent across the SAMOS system. Unit conversions and sanity checks (e.g. range and type checks) are also performed. Data collected on different days but hosted in the same original file is split into individual, daily file pieces for the particular day, using UTC time. The result of this step is a version 5 COAPS ASCII file.

4. Preliminary NetCDF Generation

   Daily file pieces in the intermediate COAPS ASCII format are converted to another, more verbose intermediate format. This format basically consists of a series of commands, each of which uses a sub-program to add one item to the netCDF file. The result of this step is a version 5 command file and a version 10 netCDF file for each day of data in the original file.

5. Preliminary NetCDF Screener

   The Prescreener program performs some automated quality control on the resulting NetCDF file. The results of this step are a single version 11 and a single version 100 netCDF file for each day of data in the original file. The version 100 is distributed to the public as a preliminary file.

6. Daily NetCDF File Merging

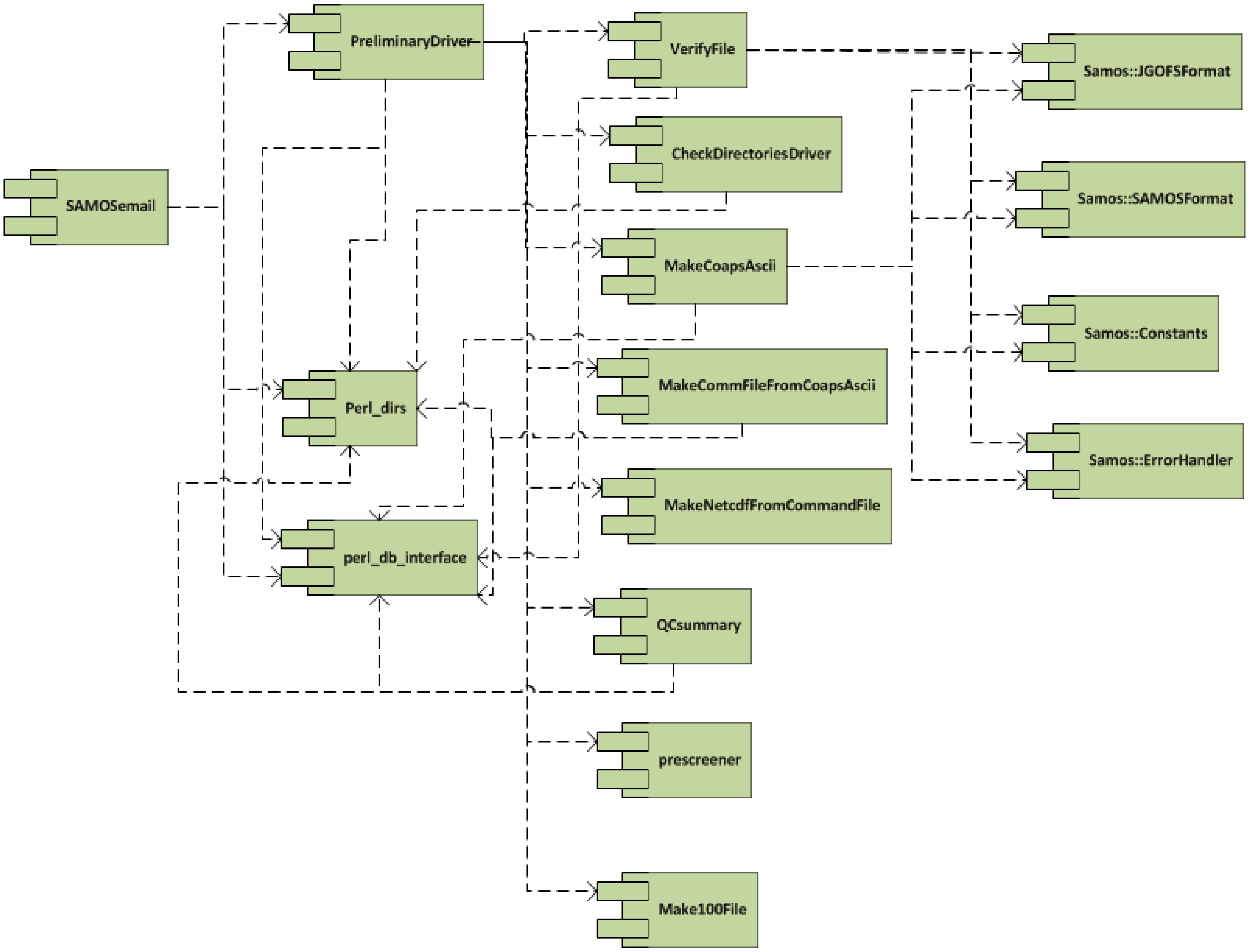
   After a 10-day delay, all preliminary netCDF files for a given ship and day are merged into one netCDF file and duplicate data is eliminated. More automated quality control is in development for this portion of the processing. The result of this step is a version 200 netCDF file, which is distributed as an intermediate file.

7. Visual Quality Control

   An analyst visually checks the data for a given day and flags any data that appears suspect. The result of this step is a version 300 or greater, depending on how many visual quality control iterations have been performed on the file. This file is distributed as a research file.

The components of the system that create the netCDF data products (items 4-7) are great at what they do and probably have a lot more life in them as long as they see regular maintenance. Their whole purpose is to allow for a mechanism to put the one-minute averaged data into the netCDF data product, while maintaining metadata and quality control information. These requirements remain consistent in SAMOS 2.0, so the mechanisms to fulfill these requirements may remain the same for a while longer.

However, the other components (items 1-3) require some redesigns to handle the needs of the next generation SAMOS. The resulting system will be a type of hybrid system. The original SAMOS processing pathways and all of its manual metadata management will remain intact to support vessels already contributing to the SAMOS Initiative.



[Figure 2.1: Component diagram of SAMOS 1.0 preliminary processing.]

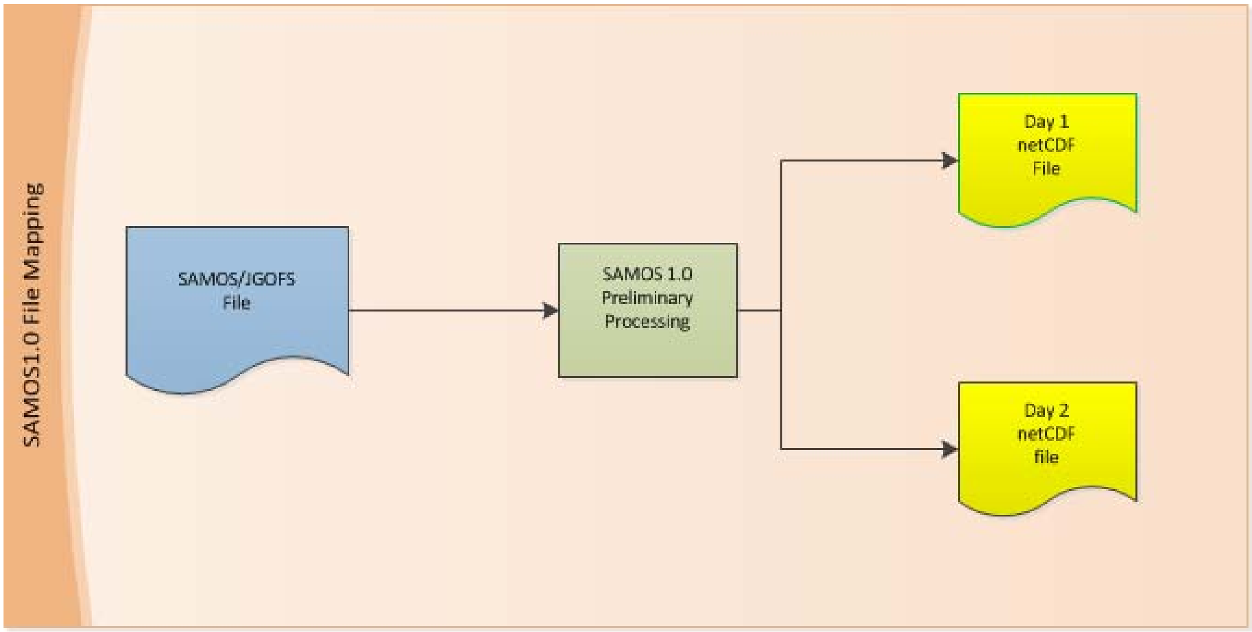
2.2 SAMOS 2.0 as a Hybrid System

The beginning of the processing will have two separate entrances. One will handle the processing for the high frequency data. The other will be the classic pathway for those vessel operators who still send 1-minute averaged data in either of the legacy formats: SAMOS or JGOFS. These two pathways will join at the Preliminary NetCDF Generation step (see section 2.1).

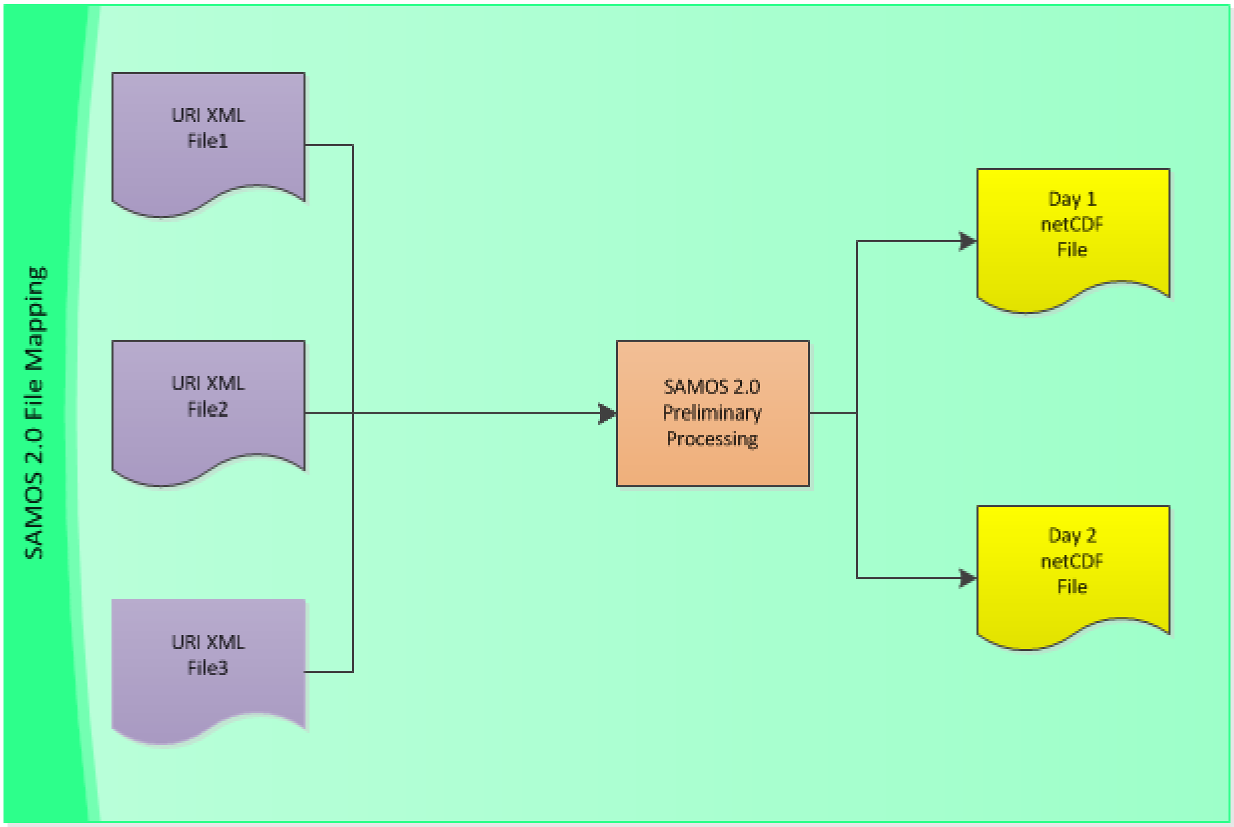
The Database and Metadata component of the system is heavily used in every component of the system. Taking into consideration that the metadata storage and updating have changed substantially for the SAMOS 2.0 system, changes have to be made to the metadata retrieval mechanism in the latter components of the system. However, most of this is handled through a custom database Application Programming Interface (API) written in Perl. So, the changes are relatively isolated in this case.

Not only is the database component used heavily in the processing, but there are also many web applications and tools that access the MySQL database. Many of these tools have hard-coded SQL queries. The metadata retrieval code for these tools will have to be updated eventually.

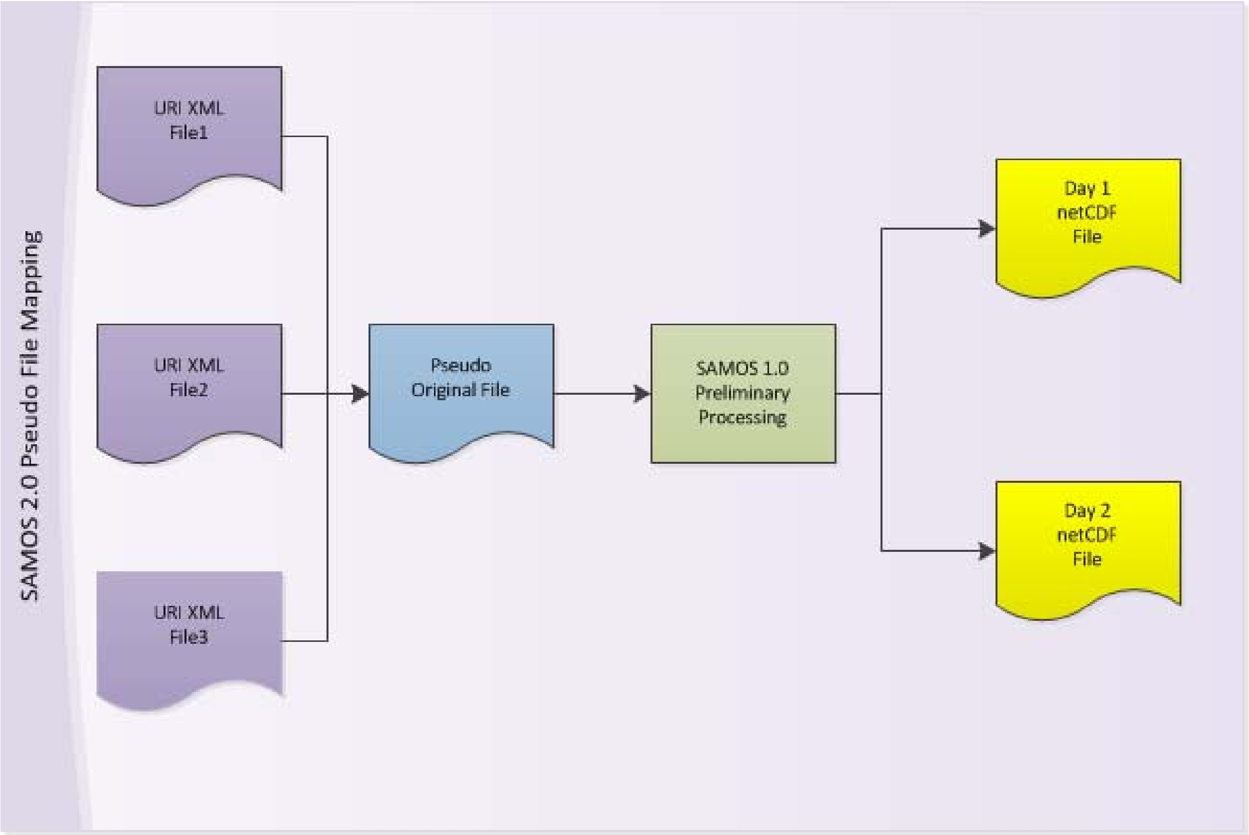
The database is also heavily used to track the status of a file as it goes through the processing steps. For the sake of simplicity, this is kept the same in SAMOS 2.0. However, there are extra tables behind the scenes to track SAMOS 2.0-specific information. For example, in SAMOS 1.0, a single original file provided as input to the system will output one or more netCDF files (one for each day of data in the original file). In SAMOS 2.0, one or more (in most cases it will actually be more than one) original file(s) as input will result in one or more netCDF files as output. The difference is essentially that SAMOS 1.0 has a one-to-many original file to netCDF file mapping (see figure 2.2), while SAMOS 2.0 has a many-to-many mapping (see figure 2.3). To get around this, multiple SAMOS 2.0 original files are mapped to a single SAMOS 1.0 original file in the database (see figure 2.4). Therefore, there are no changes to the data tracking code for SAMOS 1.0 components.



[Figure 2.2: SAMOS 1.0 file mapping.]



[Figure 2.3: SAMOS 2.0 file mapping.]



[Figure 2.4: SAMOS 2.0 pseudo file mapping.]

2.3 SAMOS 2.0 Components

The components of the new portion of the hybrid system that preclude the Preliminary NetCDF Generation are: Data Delivery (see figure 2.6); Data Extraction and Metadata Updating (see figure 2.7); and Data Processing and Averaging (see figure 2.8).

2.3.1 Data delivery

It is expected that e-mail will remain a potential method of data delivery for the SAMOS 2.0 system. In addition to e-mail, methods using tools such as rsync are also being considered. It is also desired to move to a single point-of-contact for ships to use when delivering data.

The SAMOS project is not the only organization to which ships send their data. It is expected that ships will eventually send their data to the Rolling Deck to Repository (R2R) project, probably via e-mail or some other push mechanism. The SAMOS system will then mirror the R2R data repository by connecting to their servers on a regular basis, probably via rsync.

It is also a possibility that data sent to R2R through e-mail will simply be forwarded to SAMOS upon reception. The details in this component are still under development, but the SAMOS project remains flexible in this area. However, using R2R as the single point-of-contact in the future will simplify the data distribution process for vessel operators who currently send their data both to R2R for archival and to the SAMOS Data Center for processing.

The only currently accepted delivery method relies on e-mails with the data files attached that are sent to a specific email address. Upon receipt, the attachments are extracted and uncompressed (if necessary). The file names are appended with the time of receipt, to guarantee unique names. The new files are then placed in a directory, known as the extraction queue, to be processed later.

2.3.2 Data Extraction and Metadata Updating

At regular intervals, the system will check for newly-arrived files in the extraction queue directory. When a new file is found, the vessel and instrument metadata will be extracted from the file. If any of the metadata seems unreasonable, then the file will be rejected and an analyst will be notified. The following are examples of circumstances that should be reported:

a.) The file is empty;
b.) The ship is not in the SAMOS system;
c.) The file says that the data is for a time in the future;
d.) A metadata mapping seems illogical (e.g. using units of “Celsius” for the variable “Pressure”);

e.) There is no metadata in the file;
f.) There is no data in the file;
g.) The file format is unrecognized.
h.) A metadata mapping overlaps a mapping currently in the database.

Extracted vessel metadata primarily consists of coordinate information that is used to determine the location of the various instruments, also known as talkers, on the ship, relative to the ship's center-line. While other vessel metadata is read, not all of it is updated in real-time, as some values are expected to be static. Static vessel metadata includes the vessel call sign and vessel name, for example.

Talker metadata is also extracted. According to the National Marine Electronics Association (NMEA) 0183 standard, a "talker" is a device that transmits a data string to a "listener." In the context of SAMOS 2.0, a talker can be a device that transmits a data string or an instrument that measures some kind of data, and passes this data to another device for transmission.

With that in mind, a talker can have "children" talkers. This happens very often. For example, the R/V Endeavor has a humidity/temperature probe, a barometric pressure sensor, two sea temperature probes, two radiation sensors, and a rain gauge—all of which transmit their data to a device that forms a data string for a given point in time with measurements from each of their devices. The resulting data string has values for air temperature, relative humidity, atmospheric pressure, sea temperature, long wave atmospheric radiation, short wave atmospheric radiation, precipitation accumulation, wet bulb temperature, and dew point temperature. The various measuring devices are child talkers of the device that forms the data string. The relationship between talkers and variables measured form a tree structure, with variables as leaf nodes (see figure 2.5).

Metadata recorded for talkers includes instrument make, instrument model, instrument version, instrument serial number, and the coordinates of the instrument.

The final metadata type is variable metadata. In the above case, the variables would be air temperature, relative humidity, atmospheric pressure, sea temperature, long wave atmospheric radiation, short wave atmospheric radiation, precipitation accumulation, wet bulb temperature, and dew point temperature.

The SAMOS 2.0 system identifies a variable with its "known variable" based on the "SAMOS name" associated with the field in the source file. Known variables are pre-configured vessel-independent variables that tell the SAMOS system how to handle a specific kind of measurement. For example, it tells the system that wind speed measurements should be converted into meters per second. SAMOS names are vessel-independent names for known variables that the SAMOS system expects. An example SAMOS name is "PL_WDIR" or "platform-relative wind direction" for the vessel-relative wind direction variable. If a variable in a data string does not have a known variable association or its SAMOS name is invalid, then its metadata and data is ignored. For example, dissolved oxygen is sometimes measured onboard research vessels, but it is not currently supported by the SAMOS system. Therefore, dissolved oxygen variables would be ignored during extraction.

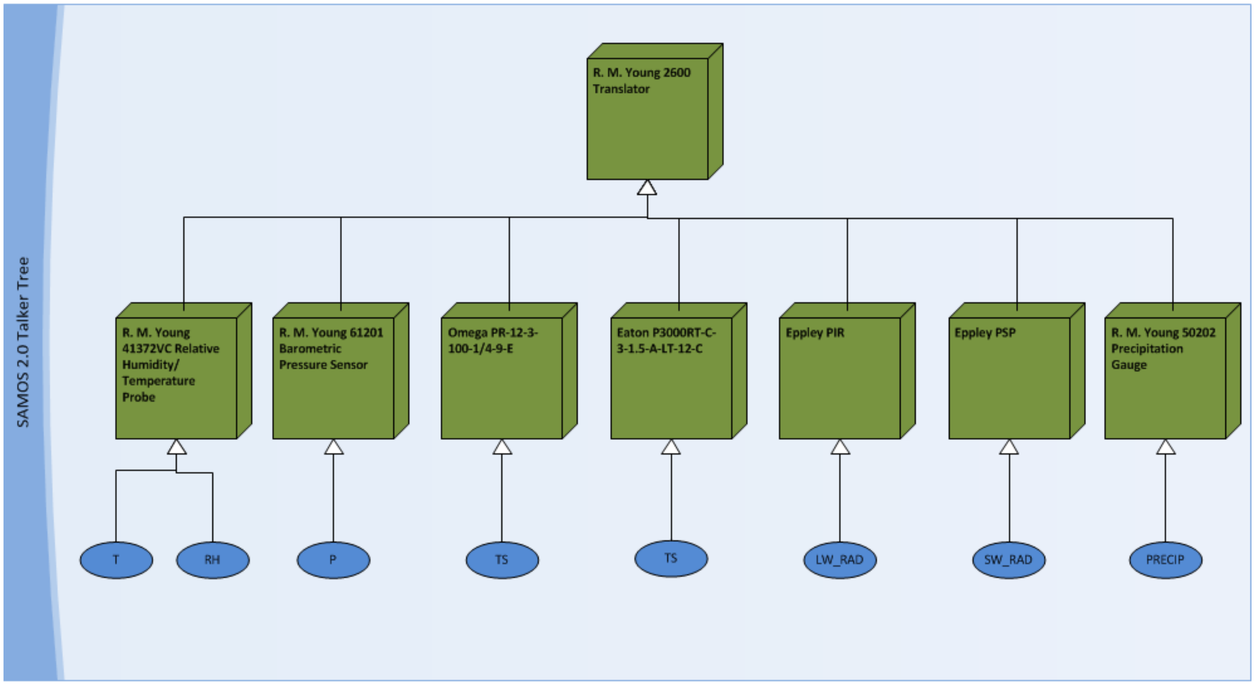
Variable metadata includes the source talker (the talker that forms the measurement, not the talker transmitting the string), the identifier associated with the measurement given by the vessel operator, the units of the measurement and its known variable association.

While the metadata is being checked, the process will keep track of metadata values that differ from the values currently in the SAMOS system for the particular ship, and if necessary, the variable with which it is associated. The SAMOS system keeps track of valid time ranges for all metadata values. The valid time ranges use a resolution of milliseconds, using the minimum and maximum timestamps of the data in the file in which the metadata is included. This feature is an improvement over SAMOS 1.0, as the old system keeps track of metadata with a time resolution of one day. However, the improvement cannot be realized while SAMOS 1.0 code is still used in the latter stages of processing. These time ranges are also automatically consolidated for attributes that have the same value, if the time difference between the two time ranges is less than a value that can be configured in the SAMOS 2.0 configuration file.

Here are some of the possible cases that will come up, and what process should be carried out thereafter:

1.) A metadata attribute appears that has not been seen before:
    a.) Insert the attribute name into the database.
    b.) If the attribute value appears to be invalid, alert an analyst. (Note: this feature is speculative, and is not currently implemented.)
    c.) Otherwise, insert the new value into the database with the valid start time equal to the start time of the data in the file and the valid end time equal to the end time of the data in the file.
2.) A metadata attribute appears that has been seen before:
    a.) If there is already a value for the time range of the data in the file:
        1.) If the old value is the same as the new value, set the valid start time to the minimum of the two timestamps and the valid end time to the maximum of the two time ranges.
        2.) If the old value is different, alert an analyst that there is overlapping metadata.
    b.) If there is no value for the time range of the data in the file:
        1.) If the attribute value appears to be invalid, alert an analyst. (Note: this feature is speculative, and is not currently implemented.)
        2.) Otherwise, insert the new value into the database with the valid start time equal to the start time of the data in the file and the valid end time equal to the end time of the data in the file.

After the metadata is inserted into the database, the data is extracted from the file and inserted into the database in a temporary table.


SAMOS 2.0 Talker Tree
R. M. Young 2600 Translator
R. M. Young 41372VC Relative Humidity/ Temperature Probe
R. M. Young 61201 Barometric Pressure Sensor
Omega PR-12-3-100-1/4-9-E
Eaton P3000RT-C-3-1.5-A-LT-12-C
Eppley PIR
Eppley PSP
R. M. Young 50202 Precipitation Gauge
T
RH
P
TS
TS
LW_RAD
SW_RAD
PRECIP

[Figure 2.5: An example talker tree in SAMOS 2.0. Variables are identified using short-form SAMOS names.]

2.3.3 Data Processing and Averaging

At a scheduled time, the system will process and average all data that has been extracted from the collected data files. The first step is to quality control the data. There isn't currently any quality control implemented, except crude range-checking on the averaged data later in processing. These ranges used in this quality control method are defined for each known variable. It would be useful to implement type checks, range checks and other automated quality controls on the raw data. There are currently plans to implement navigation quality control on the raw data based on code from the R2R project.

After quality control, the system performs calculations on certain variables of the high resolution data to derive other variables. This currently only includes a true-winds calculation, which takes in the ship heading, ship course, ship speed, ship-relative wind direction and ship-relative wind speed, and outputs earth-relative wind direction and speed (see section 2.3.3.1). This may include other calculations in the future, such as flux and dew point calculations. Since data points for two variables are unlikely to be time stamped identically when using millisecond resolution, the system allows a little bit of a difference between two points for these calculations. The allowed maximum time difference can be configured in the SAMOS 2.0 configuration file.

At some point in the future, the system may also use equations gathered from the metadata included in a data file to perform calculations in this step. One use for this feature would be to calculate the desired variable from the actual raw measured data sent by the instrument, which is often measured in a unit such as Volts. The ship could send the raw data, as well as any calibration information needed to compute the desired data.

Next, the system will average all of the extracted and calculated data. The code supports scalar and vector averages (see section 2.3.3.2), depending on the variable in question. The data is time-stamped in millisecond resolution, so magnitude and direction data points for vector averaging won't always have the exact same timestamp. As with the complex data calculation algorithm, a bit of a time difference is allowed, and the maximum difference is configurable. The averaged data is then stored in another temporary table.

Most of the data is expected to be recorded approximately every second and time-stamped with millisecond resolution. The one-minute averages are formed by averaging the data from the start of the minute (e.g.: HH:MM:00.000) and the end of the minute (e.g.: HH:MM:59.999-60.999 depending on leap seconds) and the resulting averaged timestamp is at the start of the minute (e.g. HH:MM).

After averaging, the system will convert all of the averaged data to the common units (usually a combination of SI units) desired by SAMOS for each particular variable, and then write the data to a version 5 COAPS ASCII file to be passed into the SAMOS 1.0 system.

2.3.3.1 True Winds Calculations

True winds are earth-relative wind direction and speed measurements that are calculated from platform-relative wind speed and direction, and vessel heading, speed, and direction. The code used to calculate true winds is based on the WOCE-MET true winds calculation code.

The following process is used for the calculation:

$h_\theta = vessel\ heading$

$R_\theta = zero\ reference\ angle$

$P_\theta = platform - relative\ wind\ direction$

$C_\theta = vessel\ course\ over\ ground$

$A_\theta = apparent\ wind\ direction = \ h_\theta + R_\theta + P_\theta$

$P = platform - relative\ wind\ speed$

$C = vessel\ speed$

$A = apparent\ wind\ speed = P$

$C'_\theta = convert\_to\_math\_coordinates(C_\theta)$

$$A'_\theta = convert\_to\_math\_coordinates(A_\theta)$$

$$C = convert\_to\_true\_wind\_speed\_units(C)$$

$$A = convert\_to\_true\_wind\_speed\_units(A)$$

$$T_u = T'_u = eastward\ component\ of\ true\ wind\ vector = |A| \cos A'_\theta + |C| \cos C'_\theta$$

$$T_v = T'_v = northward\ component\ of\ true\ wind\ vector = |A| \sin A'_\theta + |C| \sin C'_\theta$$

$$|T| = true\ wind\ speed = \sqrt{{T_u}^2 + {T_v}^2}$$

$$T'_\theta = \tan^{-1} \frac{T_v}{T_u}$$

$$T_\theta = true\ wind\ direction = convert\_from\_math\_coordinates(T'_\theta)$$

The method to convert angular values to and from the math coordinates can be a bit tricky. If the angle is measured clockwise to true north, which is common for vessel heading, vessel course, and wind direction in the case of the oceanographic reference standard, then the following is used:

Convert to:

$$\theta' = 90 - \theta$$

Convert back:

$$\theta = 90 - \theta'$$

If the angle is measured clockwise from true north, which is common for wind direction in the case of the meteorological reference standard, then the following is used:

Convert to:

$$\theta' = 270 - \theta$$

Convert back:

$$\theta = 270 - \theta'$$

Another troublesome aspect of the true winds calculation is finding the wind direction and the wind speed variable mates and vessel course and vessel speed variable mates to pair up. However, the code prioritizes speed and direction variables that are measured on the same talker. If a speed-direction pair can't be found on the same talker, then the code checks all talkers and returns the first two variable matches that it finds. It may be useful to build a smarter selection system based on past quality control results, once quality control is designed.

2.3.3.2 Vector Averaging

Scalar averaging is a very basic process, so that won't be reviewed in detail. However, vector averaging is more complex. The averaging code used for scalars and vectors is based on IMMA code designed to generate 11-minute averages from SAMOS netCDF Files.

The process for vector averaging is as follows.

For each point in time:

$$V_\theta = vector\ direction$$

$$V_m = vector\ magnitude$$

$$V'_\theta = convert\_to\_math\_coordinates(V_\theta)$$

$$V_u = x - axis\ component\ of\ vector = |V_m| \cos V'_\theta$$

$$V_v = y - axis\ component\ of\ vector = |V_m| \sin V'_\theta$$

To get the average:

$$V_{sum\_u} = \sum V_u$$

$$V_{avg\_u} = \frac{V_{sum\_u}}{n}$$

$$V_{sum\_v} = \sum V_v$$

$$V_{avg\_v} = \frac{V_{sum\_v}}{n}$$

$$|V_{avg\_m}| = average\ vector\ magnitude = \sqrt{{V_{avg\_u}}^2 + {V_{avg\_v}}^2}$$

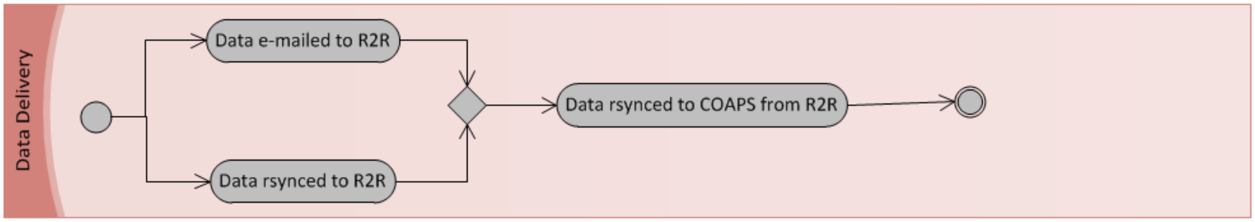
$$V'_{avg\_\theta} = \tan^{-1} \frac{V_{avg\_v}}{V_{avg\_u}}$$

$$V_{avg\_\theta} = average\ vector\ direction = convert\_from\_math\_coordinates(V'_{avg\_\theta})$$

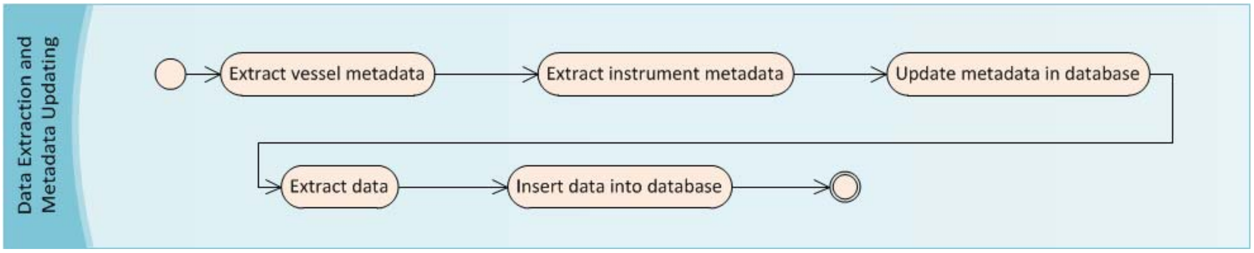
The conversions to and from the math coordinates are done similarly to the method in true winds. The exact formula of the conversion depends on the variable and the reference standard (see section 2.3.3.1).

Before the vector averaging can be done, the magnitude and direction variables need to be paired. Similarly to the pairing method for true winds, variables measured by the same talker are prioritized (see section 2.3.3.1). If no pairs are found on the same talker, the first match found is used.

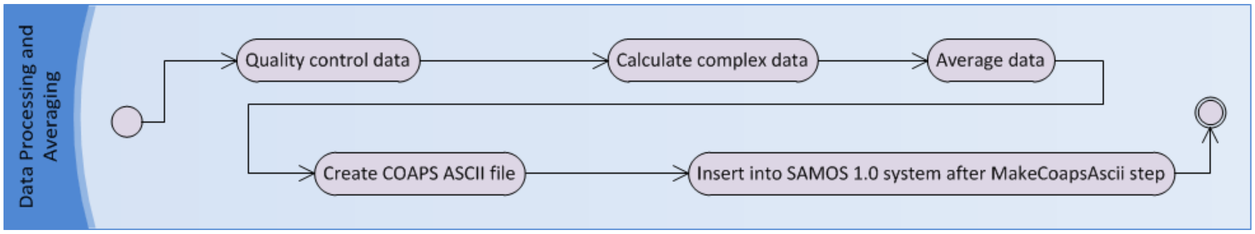
Another complexity is that the magnitude and direction data points may not always have the same timestamps, since millisecond resolution is very fine-grained. Therefore, the system allows a configurable max time difference between two data points. The system pairs the two data points that have the smallest time difference, as long as that difference is less than the maximum.



[Figure 2.6: Activity diagram for Data Delivery phase of SAMOS 2.0.]


Data Extraction and Metadata Updating
Extract vessel metadata
Extract instrument metadata
Update metadata in database
Extract data
Insert data into database

[Figure 2.7: Activity diagram for Data and Metadata Extraction phase of SAMOS 2.0.]


Data Processing and Averaging
Quality control data
Calculate complex data
Average data
Create COAPS ASCII file
Insert into SAMOS 1.0 system after MakeCoapsAscii step

[Figure 2.8: Activity diagram for Data Processing and Averaging phase of SAMOS 2.0.]

2.4 Self-describing Files

The SAMOS 2.0 system currently supports only one self-describing file format—an XML-based format sent by the R/V Endeavor at the University of Rhode Island. The file format is described below, using XPATH notation.

1.) /datafile
The root node of the document tree.
2.) /datafile/meta_vessel
The node that contains the vessel metadata.
3.) /datafile/meta_vessel/name
The node that contains the vessel name.
4.) /datafile/meta_vessel/call_sign
The node that contains the vessel call sign.
5.) /datafile/meta_vessel/imo_number
The node that contains the vessel IMO number.
6.) /datafile/meta_vessel/coordinate_system
The node that contains the nodes to describe the coordinate system.
7.) /datafile/meta_vessel/coordinate_system/x_positive
The node that contains the x_positive coordinate.
8.) /datafile/meta_vessel/coordinate_system/x_origin
The node that contains the x_origin coordinate.
9.) /datafile/meta_vessel/coordinate_system/y_positive
The node that contains the y positive coordinate.
10.) /datafile/meta_vessel/coordinate_system/y_origin
The node that contains the y origin coordinate.
11.) /datafile/meta_vessel/coordinate_system/z_positive
The node that contains the z positive coordinate.
12.) /datafile/meta_vessel/coordinate_system/z_origin
The node that contains the z origin coordinate.
13.) /datafile/meta_vessel/coordinate_system/measurement_units
The node that contains the units used in the coordinate system.
14.)/datafile/meta_vessel/coordinate_system/survey_reference
The node that contains the survey information.
15.)/datafile/meta_vessel/coordinate_system/description
The node that contains a description of the coordinate system.
16.) /datafile/meta_cruise
The node that contains the nodes that define the cruise metadata.
17.)/datafile/meta_cruise/cruise_id
The ID that identifies the cruise.
18.)/datafile/meta_cruise/principle_investigator
The PI of the cruise.

19.)/datafile/meta_cruise/chief_scientist

The chief scientist of the cruise.

20.) /datafile/meta_data

The node that contains the nodes that define the metadata.

21.) /datafile/meta_data/talker

The node that contains the nodes that define the “root” talker of the data file.

22.)/datafile/meta_data/talker/make

The make of the talker.

23.)/datafile/meta_data/talker/model

The model of the talker.

24.)/datafile/meta_data/talker/version

The version of the talker.

25.)/datafile/meta_data/talker/serial_number

The serial number of the talker.

26.)/datafile/meta_data/talker/name

The name given to the talker.

27.)/datafile/meta_data/talker/reference_standard

The reference standard of the data provided by the talker. If this is set, then no /datafile/meta_data/data_sentence nodes are expected. The only currently supported value is “NMEA 0183”.

28.) /datafile/meta_data/talker/location

The node that contains the nodes that define the location of the talker.

29.)/datafile/meta_data/talker/location/x

The x location of the talker.

30.)/datafile/meta_data/talker/location/y

The y location of the talker.

31.)/datafile/meta_data/talker/location/z

The z location of the talker.

32.)/datafile/meta_data/talker/calibration

The node that contains the nodes that define the calibration of the talker.

33.)/datafile/meta_data/talker/calibration/calibration_date

The calibration date of the talker.

34.)/datafile/meta_data/talker/calibration/coefficient

The node that contains information about a calibration coefficient for the talker.

35.)/datafile/meta_data/talker/calibration/coefficient/name

A calibration coefficient name.

36.)/datafile/meta_data/talker/calibration/coefficient/value

A calibration coefficient value.

37.)/datafile/meta_data/timestamp

The node that contains nodes to define the timestamp.

38.)/datafile/meta_data/timestamp/name

A name to give the time variable.

39.)/datafile/meta_data/timestamp/samos_name
The SAMOS name of the time variable.

40.)/datafile/meta_data/timestamp/format
The format of the timestamp.

41.)/datafile/meta_data/timestamp/units
The units of the timestamp.

42.)/datafile/meta_data/timestamp/precision
The precision of the timestamp.

43.)/datafile/meta_data/timestamp/reference_frame
The reference frame of the timestamp.

44.)/datafile/meta_data/timestamp/reference_standard
The reference standard of the timestamp.

45.)/datafile/meta_data/timestamp/source
Contains the same information and child nodes as/datafile/meta_data/talker, except that its data refers to the talker that provides the time.

46.)/datafile/meta_data/data_sentence
The node that contains nodes that define a data sentence. This is not required if the data sentences use a reference standard, such as NMEA 0183.

47.)/datafile/meta_data/data_sentence/id
The ID of the data sentence.

48.)/datafile/meta_data/data_sentence/value_delimiter_hex
The hexadecimal value that represents the character separating fields in the sentence.

49.)/datafile/meta_data/data_sentence/source
Contains the same information and child nodes as/datafile/meta_data/talker, except that its data refers to the talker that provides the data sentence.

50.)/datafile/meta_data/data_sentence/field
The node that contains nodes that define a field in a data sentence. Fields are expected in the sentences in the same order that they occur in the definition.

51.)/datafile/meta_data/data_sentence/field/name
A name to give the variable. This must be unique for the sentence.

52.)/datafile/meta_data/data_sentence/field/samos_name
The SAMOS name of the variable.

53.)/datafile/meta_data/ data_sentence/field/format
The format of the field.

54.)/datafile/meta_data/data_sentence/field/units
The units of the fields.

55.)/datafile/meta_data/data_sentence/field/precision
The precision of the field.

56.)/datafile/meta_data/data_sentence/field/reference_frame
The reference frame of the fields.

57.)/datafile/meta_data/data_sentence/field/reference_standard
The reference standard of the field.

58.)/datafile/meta_data/data_sentence/field/source

Contains the same information and child nodes as/datafile/meta_data/talker, except that its data refers to the talker that provides the field.

59.)/datafile/data

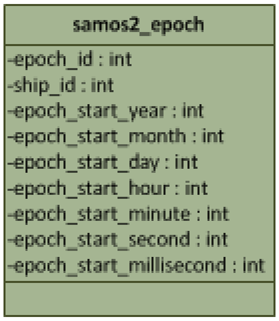
A node that contains a data sentence.

60.)/datafile/data[@time]

The timestamp of the data sentence.

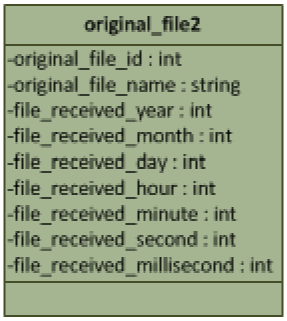
2.5 SAMOS 2.0 Database

This section will discuss the various tables added to the database for SAMOS 2.0. The tables are represented by graphics created in Microsoft Visio 2010, which doesn't have SQL types by default; therefore, the type names shown are only approximate.



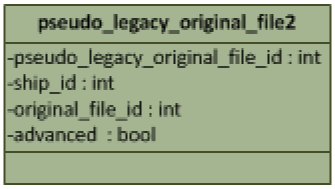
samos2_epoch
-epoch_id : int
-ship_id : int
-epoch_start_year : int
-epoch_start_month : int
-epoch_start_day : int
-epoch_start_hour : int
-epoch_start_minute : int
-epoch_start_second : int
-epoch_start_millisecond : int

The purpose of the "samos2_epoch" table is to record the date which SAMOS 2.0 is enabled for each ship. Once SAMOS 2.0 is enabled for a particular ship, the SAMOS 1.0 code in the latter stages of processing attempts to fetch SAMOS 2.0 metadata, rather than SAMOS 1.0 metadata. There currently is no method to run SAMOS 1.0 and SAMOS 2.0 for the same ship simultaneously.



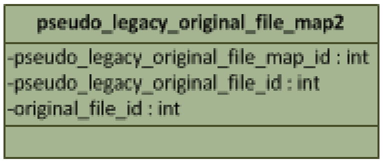
original_file2
-original_file_id : int
-original_file_name : string
-file_received_year : int
-file_received_month : int
-file_received_day : int
-file_received_hour : int
-file_received_minute : int
-file_received_second : int
-file_received_millisecond : int

The "original_file2" table is used to store files that are input into the SAMOS 2.0 system. This is similar to the "original_file" table in SAMOS 1.0.



pseudo_legacy_original_file2
-pseudo_legacy_original_file_id : int
-ship_id : int
-original_file_id : int
-advanced : bool

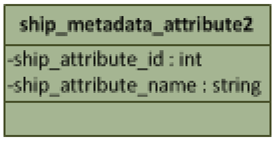
The "pseudo_legacy_original_file2" table is used to keep track of "fake" original files in the SAMOS 1.0 data tracking system (see figures 2.2, 2.3, 2.4). The "original_file_id" field here is a foreign key that refers to the ID of the "original_file" table in SAMOS 1.0. The "advanced" field here is a Boolean value that is set to false when the file is still pending in the data tables (described later), and true when the data has been sent into the SAMOS 1.0 system.



pseudo_legacy_original_file_map2

-pseudo_legacy_original_file_map_id : int
-pseudo_legacy_original_file_id : int
-original_file_id : int

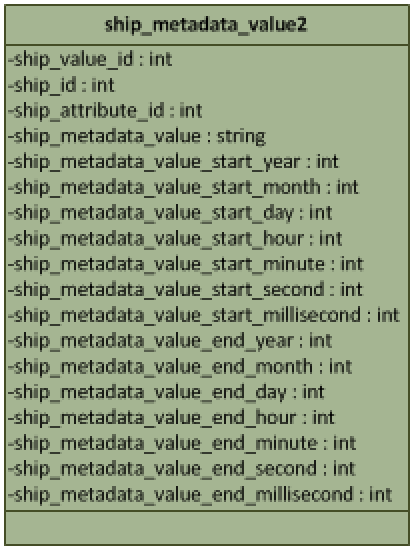
The “pseudo_legacy_original_file_map2” table is used to map SAMOS 2.0 original files (through “original_file2”) to pseudo original files (through “pseudo_legacy_original_file2”), which in turn is mapped to SAMOS 1.0 original files (through “original_file”). The “original_file_id” here is a foreign key that refers to the ID of the “original_file2” table. This table essentially forms the many-to-many original file-to-netCDF file mapping, to compensate for SAMOS 1.0s one-to-many mapping.



ship_metadata_attribute2

-ship_attribute_id : int
-ship_attribute_name : string

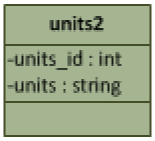
The “ship_metadata_attribute2” table is used to name metadata attributes for ships.



ship_metadata_value2

-ship_value_id : int
-ship_id : int
-ship_attribute_id : int
-ship_metadata_value : string
-ship_metadata_value_start_year : int
-ship_metadata_value_start_month : int
-ship_metadata_value_start_day : int
-ship_metadata_value_start_hour : int
-ship_metadata_value_start_minute : int
-ship_metadata_value_start_second : int
-ship_metadata_value_start_millisecond : int
-ship_metadata_value_end_year : int
-ship_metadata_value_end_month : int
-ship_metadata_value_end_day : int
-ship_metadata_value_end_hour : int
-ship_metadata_value_end_minute : int
-ship_metadata_value_end_second : int
-ship_metadata_value_end_millisecond : int

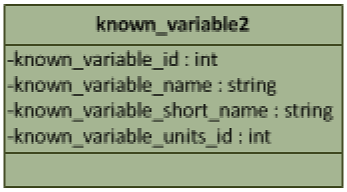
The “ship_metadata_value2” table is used to name metadata attribute values for a given ship, attribute, and time range.



units2

-units_id : int
-units : string

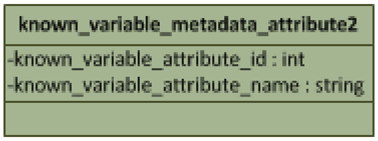
The “units2” table stores the units for variables in SAMOS 2.0. Both known variable and ship-specific variable tables get their units from this table. Its purpose is to cut down on redundant string fields in those variable tables. Rather than storing the string “celsius” for a bunch of temperature variables, it is stored once in the units2 table, and every field and table that references it will instead store an integer value that acts as a foreign key to this table.



known_variable2

-known_variable_id : int
-known_variable_name : string
-known_variable_short_name : string
-known_variable_units_id : int

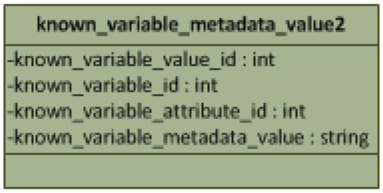
The “known_variable2” table defines a known variable for SAMOS 2.0. The table is similar to the “known_variable” table in SAMOS 1.0. The main difference is that SAMOS 2.0 doesn’t have “extra variables” defined for multiple instruments that measure the same things (such as TD, TD2, TD3). The “known_variable_units_id” is a foreign key that refers to the “units2” table. This represents the units into which a variable’s data should be converted before going into the netCDF file. Known variables are vessel-independent variable definitions. These definitions tell the SAMOS 2.0 system how to process a type of measurement. These do not have any talker association, as they refer to a type of measurement, and not a specific measurement from a specific ship.



known_variable_metadata_attribute2

-known_variable_attribute_id : int
-known_variable_attribute_name : string

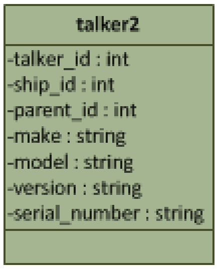
The “known_variable_metadata_attribute2” table stores metadata attribute names for known variables.



known_variable_metadata_value2

-known_variable_value_id : int
-known_variable_id : int
-known_variable_attribute_id : int
-known_variable_metadata_value : string

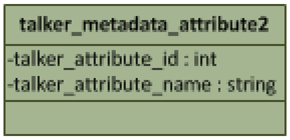
The “known_variable_metadata_value2” table stores metadata values for known variables and attributes. There currently are no time ranges for these attributes.



talker2

-talker_id : int
-ship_id : int
-parent_id : int
-make : string
-model : string
-version : string
-serial_number : string

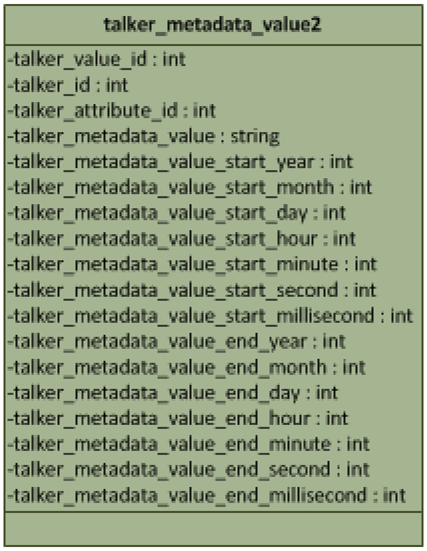
The “talker2” table stores information on talkers. A talker is an instrument that sends data, in NMEA 0183 terminology. The “parent_id” field is either set to NULL, or to the value of “talker_id” of its parent talker—the talker to which it is connected (see section 2.3.2 for a discussion of the talker hierarchy and figure 2.5 for a visual aid).



talker_metadata_attribute2

-talker_attribute_id : int
-talker_attribute_name : string

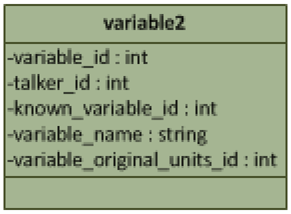
The “talker_metadata_attribute2” table stores metadata attribute names for talkers.



talker_metadata_value2

-talker_value_id : int
-talker_id : int
-talker_attribute_id : int
-talker_metadata_value : string
-talker_metadata_value_start_year : int
-talker_metadata_value_start_month : int
-talker_metadata_value_start_day : int
-talker_metadata_value_start_hour : int
-talker_metadata_value_start_minute : int
-talker_metadata_value_start_second : int
-talker_metadata_value_start_millisecond : int
-talker_metadata_value_end_year : int
-talker_metadata_value_end_month : int
-talker_metadata_value_end_day : int
-talker_metadata_value_end_hour : int
-talker_metadata_value_end_minute : int
-talker_metadata_value_end_second : int
-talker_metadata_value_end_millisecond : int

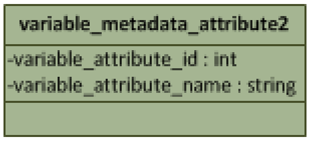
The “talker_metadata_value2” table stores metadata values for a given talker, attribute, and time range.



variable2

-variable_id : int
-talker_id : int
-known_variable_id : int
-variable_name : string
-variable_original_units_id : int

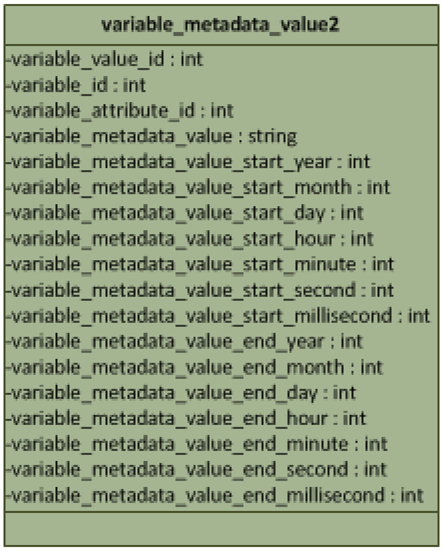
The “variable2” table stores information on a given variable for a talker. A variable must always have a parent talker, given by its field “talker_id” (see figure 2.5).



variable_metadata_attribute2

-variable_attribute_id : int
-variable_attribute_name : string

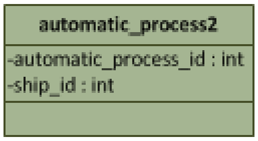
The “variable_metadata_attribute2” table stores metadata attribute names for variables.



variable_metadata_value2

-variable_value_id : int
-variable_id : int
-variable_attribute_id : int
-variable_metadata_value : string
-variable_metadata_value_start_year : int
-variable_metadata_value_start_month : int
-variable_metadata_value_start_day : int
-variable_metadata_value_start_hour : int
-variable_metadata_value_start_minute : int
-variable_metadata_value_start_second : int
-variable_metadata_value_start_millisecond : int
-variable_metadata_value_end_year : int
-variable_metadata_value_end_month : int
-variable_metadata_value_end_day : int
-variable_metadata_value_end_hour : int
-variable_metadata_value_end_minute : int
-variable_metadata_value_end_second : int
-variable_metadata_value_end_millisecond : int

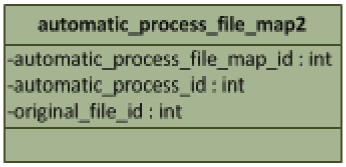
The “variable_metadata_value2” table stores metadata values for a given variable, attribute, and time range.



automatic_process2

-automatic_process_id : int
-ship_id : int

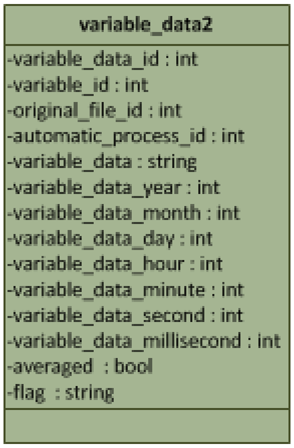
The “automatic_process2” table stores information on automatic process runs of a ship’s data. It is used to keep track of which original files correspond to which run of the automatic processing. An automated processing run is an iteration of stage 3 (see section 2.3.3) of the SAMOS 2.0 processing.



automatic_process_file_map2

-automatic_process_file_map_id : int
-automatic_process_id : int
-original_file_id : int

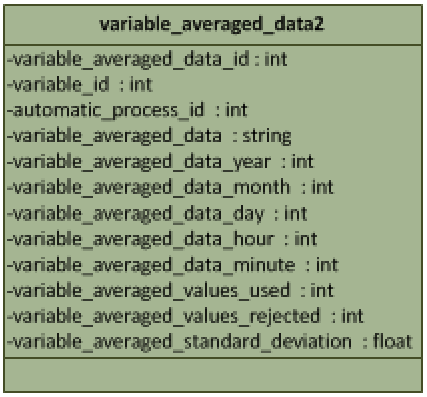
The “automatic_process_file_map2” table keeps track of which original files were processed during an automated processing run. The “original_file_id” field refers to the ID of the table “original_file2”.



variable_data2

-variable_data_id : int
-variable_id : int
-original_file_id : int
-automatic_process_id : int
-variable_data : string
-variable_data_year : int
-variable_data_month : int
-variable_data_day : int
-variable_data_hour : int
-variable_data_minute : int
-variable_data_second : int
-variable_data_millisecond : int
-averaged : bool
-flag : string

The “variable_data2” table stores data values and their timestamps after they have been extracted from a file or calculated through the automated processing. The “original_file_id” field is set to the ID of a file in “original_file2”, if it was extracted from a file. If not, it is set to NULL, and the value of “automatic_process_id” is set to the ID of the process that calculated the value in “automatic_process2”. Through that mapping, the group of original files that resulted in the value can be deduced. Examples of calculated values are true wind speed and direction. The “averaged” field is set to false if the value has not been averaged. Otherwise, it is set to true. The “flag” field is either NULL (not flagged), or is set to the value of the QC flag. Quality control is not currently defined, so no QC flag values are defined. This field is planned to be used in the future. Also note that “variable_data2” is a VARCHAR/string data type, not a float or integer, since variables of all data types should be supported. The conversion to its actual data type is handled by the SAMOS 2.0 application, depending on the variable.



variable_averaged_data2

-variable_averaged_data_id : int
-variable_id : int
-automatic_process_id : int
-variable_averaged_data : string
-variable_averaged_data_year : int
-variable_averaged_data_month : int
-variable_averaged_data_day : int
-variable_averaged_data_hour : int
-variable_averaged_data_minute : int
-variable_averaged_values_used : int
-variable_averaged_values_rejected : int
-variable_averaged_standard_deviation : float

The “variable_averaged_data2” table stores average values that have been calculated from data stores in the “variable_data2” table. The “automatic_process_id” field refers to the automatic processing run that resulted in the data point. The number of values used and rejected when creating the average, as well as the standard deviation, are also stored.

2.6 SAMOS 2.0 Configuration File

A configuration file is used to fetch configuration information for SAMOS 2.0. This is slightly different from SAMOS 1.0. SAMOS 1.0 used primarily Perl include files. However, SAMOS 2.0 is written in C++, which is a compiled, rather than an interpreted language. The use of an include file for SAMOS 2.0 would require a re-compilation after every change to configuration values. Thus, a configuration file interface was built to simplify this aspect.

The default configuration file is named “samos2.conf”. The configuration file supports:

1.) String values in single quotes (‘’) and double quotes (“”).
2.) Escaped quotes using a backslash (\) within quotes.
3.) Variable interpolation within a variable definition using the syntax:

```
var1=”somestring”
var2=”astring$(var1)andsomeotherstrings”
```

A list of configuration variables defined for SAMOS 2.0 are:

1.) perl
The executable to be used for perl scripts.
2.) rootdir
The root directory for SAMOS.
3.) codesdir
The codes directory for SAMOS.
4.) scratchdir
The scratch directory for SAMOS.
5.) ftpdir
The FTP directory for SAMOS.
6.) uid
The user ID to be used for chown/chgrp.
7.) gid
The group ID to be used for chown/chgrp.
8.) analyst_email
The email address to send analyst alerts.
9.) from_email
The email address from which alerts should be sent.
10.) web_url
The root web domain to be used for links to SAMOS information.
11.) dbhost
The domain of the database server.
12.) dbuser
The user for database logins.
13.) dbpass

The password for database logins. Note: the config file should not be world-readable, since it has this password.

14.) dbname

The database name to use on the database server.

15.) errordir

The directory used for error logs.

16.) incomingdir

The directory used to place original files as they are received.

17.) verifieddir

The directory used to place original files after verification.

18.) unverifieddir

The directory used to place original files if they fail verification.

19.) processingdir

The directory used to place files during processing.

20.) processingquickdir

The directory used to place files during quick (preliminary) processing.

21.) processingresearchdir

The directory used to place files during research processing.

22.) autoqcdir

The directory used to place files after auto QC processing.

23.) autoqcquickdir

The directory used to place files after quick auto QC (prescreener).

24.) autoqcresearchdir

The directory used to place files after research auto QC.

25.) publicdir

The directory used to place files that are publicly available.

26.) publicquickdir

The directory used to place files that are publicly available after quick (preliminary) processing.

27.) publicresearchdir

The directory used to place files that are publicly available after research processing.

28.) visualqcdir

The directory used to place files after visual QC.

29.) visualqcquickdir

The directory used to place files after quick visual QC.

30.) visualqcresearchdir

The directory used to place files after research visual QC.

31.) ftpdir

The directory used to place files on the public FTP.

32.) ftpquickdir

The directory used to place files on the public FTP after quick processing.

33.) ftpresearchdir

The directory used to place files on the public FTP after research processing.

34.) ftpintermediatedir

The directory used to place files on the public FTP after intermediate processing.

35.) archivedir

The directory used to archive files.

36.) smtp_server

The SMTP server to use for sending outgoing mail.

37.) umask

The umask to use before creating files.

38.) fmode

The mode to use when creating regular files.

39.) dmode

The mode to use when creating directories.

Future Work

3.1 Future additions to SAMOS 2.0

There are a lot of features that could potentially be added to SAMOS 2.0 in the future. It would be useful to end users for the SAMOS Data Center to develop additional algorithms to calculate more derived values from data, similar to how true winds is currently calculated from platform-relative wind velocity, ship velocity and ship heading. Some potential calculations to add are those to derive the dew point as well as fluxes.

It would also be useful for end users and vessel operators if new automated quality control algorithms were developed to better flag the quality of the high resolution data. End users would have higher quality data to work with as a result. The SAMOS Data Center could also provide better feedback to vessel operators on the operation of their instruments based on this data.

The only self-describing file format supported by SAMOS 2.0 is an XML-based format that is used by the R/V Endeavor at the University of Rhode Island. There are currently plans to develop support for a CSV-based format used by a team at Oregon State University, who is currently working on acquisition of a new research vessel.