From Raw Data to Observing Products with SSDS: Fishes to Wishes

Kevin J. Gomes
Francisco Chavez
John Graybeal

Monterey Bay Aquarium Research Institute (MBARI)
7700 Sandholdt Road
Moss Landing, CA 95039-9644
kgomes@mbari.org

Abstract **- The Shore Side Data System (SSDS) (http://ssds.mbari.org) has been collecting real-time oceanographic data from Monterey Bay for over two years. While the system is not yet ready for public distribution, its data may be accessed readily by web browsers, external computer software (via web services), and the HOOVES desktop application.**

One of SSDS' hallmarks is its ability to accept data from new platforms and instruments, and immediately provide access to that data via public interfaces. With the support of a well-designed data observing infrastructure such as SIAM, advanced features like variable- and instrument-based data access are also instantly available. The system can accept data from any type of instrument and make it available in the original format, or (for described data) as individual data items.

Many users in MBARI's science and engineering community have taken advantage of the SSDS interfaces to view and automatically process near-real-time and archival data ingested by the system. With its Java Enterprise Architecture-based services, SSDS provides ready access to data and its descriptive metadata—including data provenance and geospatial and temporal metadata.

In this paper we use SSDS as an example to consider all parts of the data life cycle in a modern observatory data system. We identify lessons and make recommendations for future observing system developers. We will particularly emphasize topics applicable to the ocean observing communities such as OOI and IOOS.

I. INTRODUCTION

Several years ago, the Monterey Bay Aquarium Research Institute (MBARI) made the decision to pursue technology and science developments in the area of ocean observatories. The Monterey Ocean Observing System (MOOS) project [i] was started and, to support the development of the observatory, three sub-projects were created to build the different components of an ocean observatory. The MOOS Mooring Controller (MMC) project [1] is focused on building the hardware and electronic infrastructure to support the unique requirements of disconnected, low-bandwidth ocean observing assets. The Software Infrastructure and Applications for MOOS (SIAM) project [2] is building the software infrastructure for handling plug and work instruments to enable automated configuration and control of those same assets. The Shore Side Data System (SSDS) [3] project is building the data management system to track the information coming from the observatory and its instruments and provide user access to those managed data.

The MOOS project has been iterative in its development and has deployed several milestones in each sub-project, resulting in operating observatory assets for over two years. SSDS has been collecting and serving operational data for over two years, and scientific data for one year.

During the development and deployment of these various subcomponents we have learned valuable lessons about ocean observatories and data management. In this paper we discuss how the SSDS project was built to meet the requirements of managing ocean observing data, how real science applications have used SSDS, the lessons that we have learned so far, and from those, the recommendations we would make to future observatory projects like the Integrated and sustained Ocean Observing System (IOOS)[ii] and the Ocean Observatories Initiative (OOI)[iii].

II. THE SHORE SIDE DATA SYSTEM (SSDS)

A. The Requirements

In order to effectively manage data coming from the observatory, the SSDS had to meet many requirements that were defined by both the engineers and scientists that were utilizing the MOOS program. A short summary of the main requirements are listed here:

- Capture observatory and instrument lifecycle data: The data management system needs to capture all observatory configuration information and track the history of those configurations. This includes knowing which nodes were deployed where, what their configurations were, and what instruments were connected to those nodes. It also must track when instruments were connected and removed from nodes. Client interactions with the observatory infrastructure and its instruments must be collected for system monitoring and troubleshooting.

- Return data in raw format: The data management system must capture the data in its native, or "raw", format (the format that comes from the instrument) and be able to return that data in the same format. This was required to maintain functionality of the existing suite of tools, which depended on that native format.

- Simple tools for viewing data: The data management system needs to provide simple tools for viewing the data. This includes things like converting binary data to ASCII representation for viewing and simple time series plots. Although this sounds simple, when you consider that the data management system must accept data in any raw format, the problem becomes more complex.

- Support queries for data: The data management system must make query tools available that allow users to search for observatory and instrument lifecycle data, as well as the instrument data itself.

- Support data merging: From the above queries, the system should also be able to return data from different instruments and different formats, merged on a common variable (usually time).

- Track data lineage and processing histories: The data management system should also provide the capability for users to track the history of data sets and the processing steps that were used to create them. This is critical to judge both the content of the data and its quality.

- Return data in various formats: When users have discovered the data they are interested in, the data management system should return that data in a format of the user's choice (usually chosen from a list of common formats).

B. The Architecture

After analyzing the requirements, it was clear that a loosely coupled, distributed, web-enabled architecture was the best fit. To leverage the existing open source application and web servers, a J2EE[iv] architecture was chosen. The requirements meant that SSDS would need to be able to collect and manage not just data from the observatory, but also distributed data products that were created using the original observatory data, so many characteristics of the J2EE architecture would be well suited.

A data model was designed to allow SSDS to track metadata for both observatory and distributed data sets. Figure 1 shows a diagram of a subset of this data model, but highlights the key objects of DataProducers and DataContainers. Those, in combination with the RecordDescriptions and RecordVariables, allow SSDS to "look into" any data set, assuming it can be parsed. With this metadata, SSDS can fulfill the requirements demanded of it. This data model is represented in Java classes and is mapped to a relational database for persistence and quick access.

With the metadata at the core of SSDS, mechanisms were needed to put metadata into, and retrieve it from, the SSDS. Incoming data and metadata are wrapped up into packets and published via the Java Messaging System (JMS)[v] which is a very straightforward way for clients to get their information into SSDS. The metadata is sent in XML format that conforms to the SSDS schema and describes all the information about the data and its lifecycle.

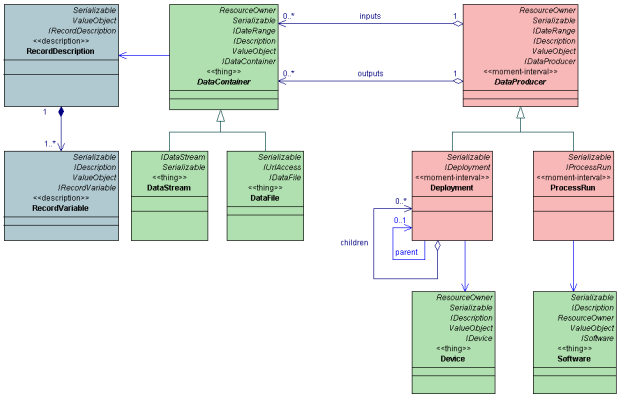

Figure 1 - Simplified SSDS Data Model

The SIAM project has been critical in enabling the generation of metadata and subsequent publishing to the SSDS. With the Plug and Work architecture of SIAM, all instruments can be self describing (containing the XML descriptions in the instrument itself) and observatory metadata is also made available to the SSDS. When one of these self-describing instruments is plugged into the observatory, the infrastructure metadata (what the instrument is plugged into, when it was connected, and so on) is sent to the SSDS as well as the instrument's self-description (XML). Soon after the metadata is sent, the data are also published to, and stored in, the SSDS. Now that SSDS contains both the metadata and data from the observatory, it can satisfy the various queries and user access requirements.

In addition, the data model allows for distributed data sets to also be registered in the SSDS. Users can register their data sets and describe them to SSDS (again in XML) and then SSDS can fold those into the query and access tools that are also rendering the observatory data. With this metadata captured, we can now connect up an instrument's raw data to other data sets and data systems to enable data merging and interoperability.

A series of web page and web service access points have been created to allow the users to query and retrieve data from the SSDS. Users can query for individual variables and other metadata and use some basic visualization tools to examine the data in the context of other data.

III. SCIENCE USES

To give specific examples of the scientific uses of the SSDS, we will highlight two main areas of the observatory and see the functionality that SSDS provides in those areas. First, we will discuss the data management for the various moorings that have been deployed in Monterey Bay. Second, we will focus on the Autonomous Underwater Vehicle (AUV), which is operating at regular intervals and collecting data from the Monterey Bay.

A. Mooring Data Management

MBARI has two moorings deployed in Monterey Bay and is a partner on a third mooring through the Center for Integrated Marine Technologies (CIMT)[vi] program. The SSDS is managing data from these three moorings. The metadata describing the configuration and data coming from the moorings is sent to the SSDS, which collects and stores that metadata and the subsequent data produced by the moorings. Users are then able to create custom data products by using the generic access and visualization tools made available from the SSDS.

Several developers have written software processes in languages such as Perl, Visual Basic, and Matlab, which further process the data after reading it from the SSDS web services. Some of these processes read the data in its raw format from SSDS, while others query for specific variables that SSDS extracts from the raw data stream. The data is then processed further, for example converted to other formats, and sent to other applications or data systems. For example, an external process obtains data from all three moorings through the SSDS web services, and then converts the data to the correct format for the National Data Buoy Center (NDBC)[vii].

Scalability and flexibility are critical in this context. Because of the SSDS and SIAM infrastructure, MBARI can deploy new moorings, or change the configuration of existing moorings and those changes are reflected in SSDS --and the user's access applications--immediately. Frequently with moorings, instruments are periodically replaced as part of a normal maintenance and cleaning schedule. With the SSDS's access services, if the instrument that a user is reading temperature from is replaced, the user's downstream queries for that temperature, still function correctly, without any programmatic intervention. This helps immensely in maintaining the flexibility and scalability of the data system.

B. AUV Data Management

The SSDS is also managing the data from MBARI's Autonomous Underwater Vehicles (AUV) and its post analysis products. AUV missions are run regularly and upon the AUV's return, the data from all the instruments is automatically offloaded and cataloged, and the metadata is published to the SSDS. Another process (external to SSDS) produces further data products, which are themselves registered with the SSDS. The resulting data products are then available through the SSDS services and user interface tools.

The Hierarchical Ocean Observatory Visualization and Editing System (HOOVES) application is an SSDS tool that gives users powerful access to these complex data sets. Figure 2 shows the hierarchical view of the deployment of an AUV and the instruments that were on the vehicle during the mission. This gives users access to the configuration of the vehicle, and the relevant information about each of the instruments deployed on that mission.

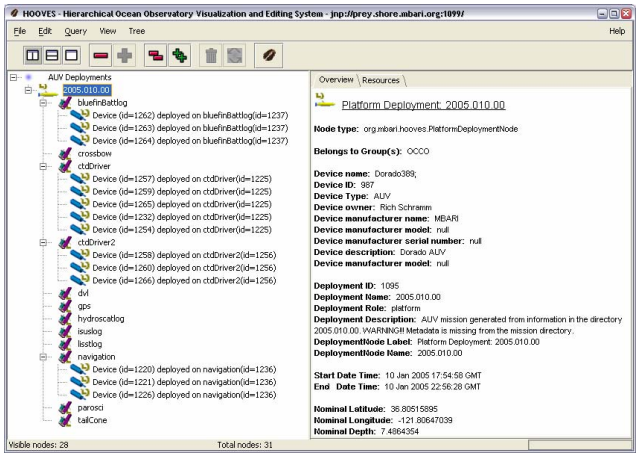

Figure 2 – AUV Deployment Configuration

Figure 3 shows a view of the processing history of the AUV instrument data. The raw instrument data starts on the left, and the various processing steps and subsequent data products are shown as they progress to the right. This allows the user access to both the data at each processing step, as well as the information about the processing steps themselves (date of processing, software version, etc.).

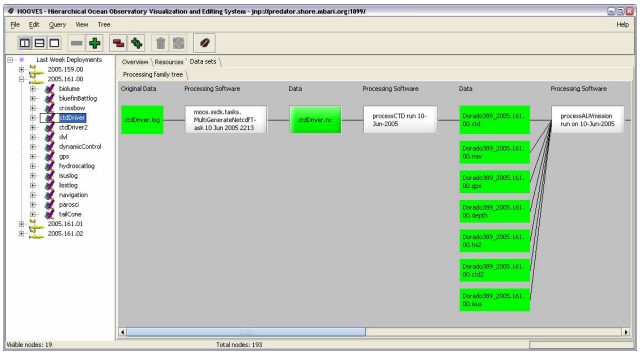

Figure 3 – Data and Processing History Display

As shown in Figure 4, HOOVES can plot different variables from different processed outputs to allow users to analyze both the raw and post processed data from each mission.

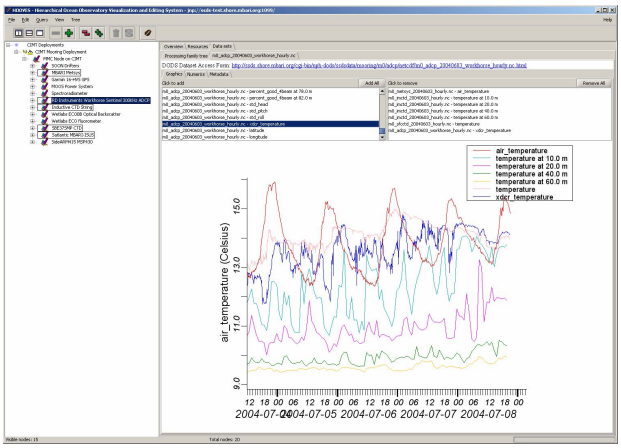

Figure 4 – Merging Data From Different Sources

On request, HOOVES will display the composite numerical data from the plots (as shown in Figure 5). This enables users to copy the data into their favorite display or analysis application.

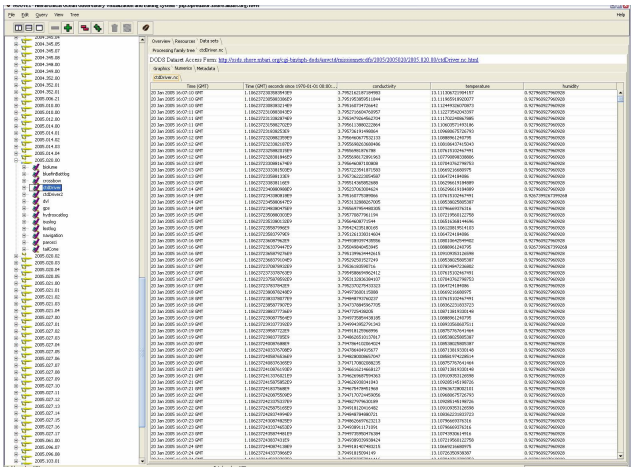

Figure 5 – Numerical Data From Plots

By allowing the AUV to keep its files in their native binary format and simply describing that parseable format to SSDS, the science user and operations crew can get immediate feedback to judge the quality of data coming from the vehicle. This allows equipment problems to be discovered quickly and remedied before the following mission.

The HOOVES application can also display associated non-numerical products, like images, that are registered in the SSDS. Figure 6 shows plots that were created from the AUV data processing software and registered with SSDS.

Although SSDS does not create these advanced data products internally, once these products are registered with the SSDS, they are immediately available to the rest of the user community. This gives the science user the capability of storing the heterogeneous products that are created from data processing and making those available to the users. Users will then be able to discover these post-processed products by following the process histories of the raw instrument data.

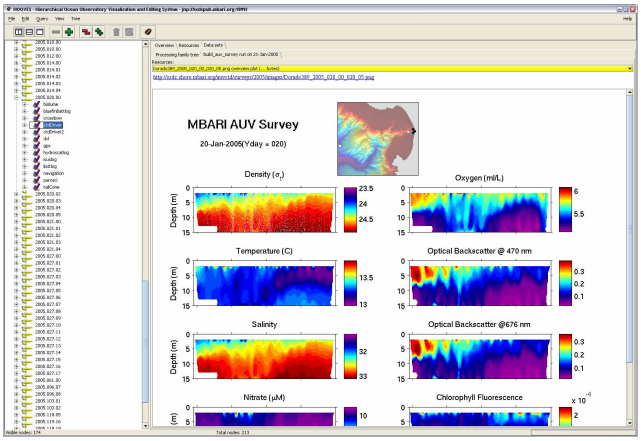

Figure 6 – SSDS Links Non-Numerical Products

IV. LESSONS

Throughout the SSDS project many lessons have been learned about data management for ocean observatories. We found that while our initial requirements gathering was accurate, the priorities of the requirements were quite different from their original ranking.

A. Saving All Raw Data is Important

Although deemed somewhat important at the beginning, the requirement that SSDS must return all raw data turned out to be fundamental to the acceptance of the SSDS. Most of our users were very familiar with the instruments they wanted to put in the observatory and were quite reluctant to have a data management system between themselves and that instrument. This is due, in large part, to the amount of applications and custom data processing that they were currently using on the data streams coming from those instruments. Therefore, it was important that the data management system not interfere with that capability at all.

One of the SSDS' unique features is that as a first step towards common data management, users could let their data be managed by SSDS in a non-intrusive way, giving the users direct access to their raw data via a web (URL) link. In this way, users could maintain control over their data, and read it by simply pointing their custom applications and processing to a URL, instead of a file. This was a crucial first step in getting the users interested in a data management system. If they could run all the same processes and tools they were used to running on their data, but also get the functionality of data merging, querying and visualization from the data management system, they were indeed interested.

Many of the existing data management solutions we investigated demanded some sort of standard data format that all users have to translate their data into before they can utilize the system. Users were reluctant to do so for several reasons. The first, most obvious reason is that the time it takes to do these data translations takes valuable time directly away from the amount of science they can accomplish. With the number of different data management systems available and in development, how can the user know that the effort invested in data conversions, will not have to be repeated to enable more interoperability of their data. Our users are very interested in what data management systems have to offer, but don't have the resources to convert their data to the various formats all these systems require. The SSDS does its best to allow users to keep their data in its native format and then automatically provide the translations to interoperate with other data management systems and user access tools (like OPeNDAP, Live Access Server, Ocean Data View, etc.).

We also discovered that users valued a central location for retrieving their raw data. As more and more systems and instruments came on line, users were happy to know that they could always go to one location (SSDS) to find all the data. This simplified access point has been an important aspect of the success of the SSDS.

B. Metadata is Hard to Capture

While metadata is critical to the data management system, it is very difficult to capture. The SSDS system provides several avenues for users to get metadata into and out of SSDS, but it is by no means simple to do so. For example, creating a description file for an instrument requires (at a minimum) identifying and copying an XML file describing a similar instrument, hand-editing the fields which are different for this instrument, validating the modified file, checking the file into a configuration control system, and putting a copy of the file in the right place—either in the metadata attached to the instrument, or in a directory where it can be submitted to SSDS.

The lack of common metadata authoring and interface tools, in combination with the general complexity of metadata, make it difficult for the users to accurately and consistently put all the correct metadata into the data management system.

The SSDS development team is currently working on tools to assist the users in authoring metadata for the SIAM instruments, as well as the metadata for distributed data sets that are registered with the SSDS. Still, in order to capture the processing history metadata in an automated fashion, much more work needs to be done. Obviously, documentation is critical, as are good Application Programming Interfaces (APIs) such as web services. Still, a developer must be experienced in these APIs, and the SSDS data model, to effectively integrate SSDS data registration in their automated data processing. Tools should be developed to help the non-programmer connect their processing to the SSDS, and these tools will be the subject of future work.

C. Functionality Will Not Satisfy All

Although a rigorous process was followed to capture the requirements for the SSDS, in the end, it is always going to be difficult, if not impossible, to build one system that meets all the users' needs. We found this to be especially true in the areas of query and data retrieval formats.

Of course, with the number and variety of users who will want to access data and metadata from the SSDS, the number of potential queries is extremely large, and finding the crucial set of queries to implement is difficult. No matter which ones a data management system implements, it will not be able to provide all queries in the exact way the users would like to have them. Similarly, the number of desired formats for returned data is invariably larger than a fixed list of formats that will be provided. Hence, there will always be some users who will have to translate the format of the data provided by the SSDS into a custom format for their specific needs.

In summary, it is very difficult for a data management system design team to anticipate every query and data format that the users will require. Either users have to work in the framework of what the data management system provides, or the data management system must provide the capability for the users to add or specify custom queries and format translations to the data management system itself.

D. Address Naming and Semantic Mismatches

Even in systems that enforce some standard data format (like netCDF [viii]), interoperable data access is often hampered by the lack of common naming and semantics. We are finding that common representations are key to powerful data management system functionality, and the lack of them hinders the effectiveness of SSDS.

The SSDS provides the capability to do very detailed and powerful queries, but without this common representation, these queries can be fragile. For example, a user setting up data processing can query for data from a specific instrument type (like 'CTD') that is deployed on a specific platform type ('mooring'). A query for all "CTD" instruments connected to platforms with "mooring" in the name will fail if a new mooring is deployed that is named "buoy".

To this end, the SSDS team has been looking into common naming standards, but found that although many might meet some end user needs, none can meet all the needs. In addition, as oceanography and science progresses, new terms and naming standards will emerge, and to include those terms in new standards is often a time consuming process.

In our experience, it seems much more likely that mappings between all the different naming conventions and semantics will have to exist to allow for data system interoperability. If not, every individual data management system will have to accommodate different naming conventions, something we view not likely to happen.

E. Cultural and Procedural Shifts Are Difficult

Often technology and systems developments cause cultural shifts that impose more rigor on the users than before, and this can cause the slow adoption of new developments. In our case, the users now have to register their devices and describe their data formats in XML files stored in the instruments themselves, and someone needs to provide the resources to perform these tasks. In all cases, science teams are limited in their available resources, and allocating a resource to meet this new overhead can be taxing. If the data management system, or infrastructure, does not provide an immediate benefit to the instrument or science user, there is not much motivation to undertake the additional effort to implement this functionality.

In addition, new policies and standards are often critical to the success of new systems and those, too, can impose more work on the data source provider. To be successful, it is imperative that the data management system provide immediate value to the user, and minimize the amount of work the user must take on to adopt the system.

V. RECOMMENDATIONS

A. Accommodate All Raw Data

In order to accommodate a large user base, the data

management system and infrastructure should accommodate the storage of all data that users would like to register or store. This is critical for legacy applications and data sets, and also lowers the barrier of entry for data producers who do not have a large number of resources to retool their data generators.

If the data management system can accommodate any data source and provide automatic translations (assuming the data is described to the system and the data is parseable), this will allow a faster integration time for data generators like instrument manufacturers. This in turn allows more users to supply data to the data management system, which will help ensure that the data management system is widely used.

It is also critical that the data management system provide the capability and tools to allow the users to track the processing history of data sets. As data products become more and more complex through processing steps, it is crucial to know the history of that processing so that if the underlying data (or metadata) changes, all down stream processing can be notified of that change.

B. Develop Effective Metadata Capture Tools

Due to the difficulty of capturing complete metadata, a data management system must provide easy-to-use tools that intuitively direct the user to provide the appropriate information. The process must be simple and fast for the user, yet result in the complete specification of needed information.

Due to the fact that this capture process can be labor-intensive, the tools must keep the user informed and engaged. When providing any particular piece of metadata, the user must understand why that information is useful, and, why it cannot be obtained by any other means. The biggest obstacle to user acceptance of metadata entry systems is not simply the process or time required per se, but the user's conviction that the process could have been smarter and easier, or that the data was needed at all.

A few basic guidelines will go a long way toward making the metadata capture process successful. We propose the following "user-friendly" requirements (with examples in parentheses):

- Never ask for the same information twice (user name and phone number)
- Pre-populate metadata entry and default values whenever possible
    - Use all previously captured data to pre-complete forms (instrumentation details)
    - Use operational processes to generate operational metadata (deployment and configuration information)
    - Utilize services to capture metadata from external providers (data systems, instrumentation providers, or calibrators)
- When automation is not possible, simplify data entry
    - Templates, cut and paste, and data import technologies (generic data interface descriptions, copying previous entries)
    - Pre-fill drop down selection lists with previous entries for that field
    - List all possible entries in constrained fields
    - Make modification and validation of constrained fields as transparent as possible (accept new terms, but validate them off-line)
- Document why each metadata field is needed, and how it will be used

C. Favor Loose Coupling, Flexibility And Extensibility

As mentioned in the lessons section, it is extremely difficult (if not impossible) to build one data management system to meet all the users needs. There will always be a data format requested that was not expected, or a naming convention not seen before. The oceanographic data management system should allow for a loosely coupled infrastructure of federated data management systems, which tend to be more domain specific. This allows the developers of the individual systems to meet the needs of their direct customers and keep their system bounded, but also allow for the interoperation with other systems and sharing of their data. This would require that some external entity provide mechanisms for data systems to map their data descriptions and formats to other data systems.

We also recommend that future data management systems provide a plug-in style of architecture so that users can add their own custom queries and data translations. This ensures that the data management system will meet the needs of the largest number of users, and also prevents the duplication of user efforts. If someone can create a custom query or data format translation and plug that into the data management system, others can then use that same functionality to meet their needs.

Web services are quite applicable to these types of architectures and are, in fact, being investigated by both the IOOS Data Management and Communications (DMAC)[ix] and Ocean Research Interactive Observatory Networks (ORION)[x] cyberinfrastructure groups, as well as LOOKING [xi], the OOI cyberinfrastructure exploration project.

D. Support Semantic and Knowledge Interoperability

As mentioned before, naming and semantic impedance mismatches are a tricky problem. Scientists searching for, or using data, need their terminology and understanding to "match" the terminology and understanding of the rest of the community. To take a simple example, a scientist looking for "sea surface temperature" needs to find "Ocean Temperature" measured at 0 meters, and the scientist using a data item called "sea surface temperature" needs to know what that term really means in that data set.

There are basically two ways to solve this problem, standardization or mapping. The data system can impose some sort of standard naming convention that all data sets must conform to, but this is fragile in that these naming

standards will often not meet all user's needs and do not change quickly enough. The other solution, which we recommend, allows users to use a naming and semantic scheme that meets their needs, but provides an infrastructure that enables interoperability by mapping the terms to those used in other systems. This allows individual data system developers and managers to keep their systems well defined and tractable, while still providing the interoperability the system's users desire.

Although mapping for semantic interoperability is in the early stages in the marine sciences, the efforts of the Marine Metadata Interoperability (MMI) Project[xii], and many individual domain projects, are beginning take root. We recommend that these or similar techniques be improved to the point they can be adopted as part of the basic infrastructure for ocean observing data systems.

VI. CONCLUSIONS

Since undertaking ocean observatory systems development, MBARI has learned many lessons in all facets of observing systems. For data management, the SSDS project has given us valuable experience and insight into what it takes for a data management system to be successful. From these lessons, we feel strongly that the data management system should be loosely-coupled, flexible, and extensible; it should accommodate all raw data; give the user effective metadata capture tools; and provide mechanisms for creating and applying semantic interoperability. Hopefully our experiences will help improve the design and shorten the development time of future data management systems for oceanography, ultimately giving the oceanography community very effective data management tools to improve our understanding of the ocean that surrounds us.

Acknowledgments

We would like to thank the entire MOOS development team and its project manager, Keith Raybould. Specific thanks go out to MOOS team members Tom O'Reilly, Kent Headley, and Mark Chaffey. We thank our always supportive science and operations teams in the mooring, AUV, and MMI projects, especially John Ryan, Hans Thomas, Erich Rienecker, Duane Thompson, and Luis Bermudez. Of course, thanks also go to the SSDS development team of Andrew Chase, Mike McCann, Brian Schlining, and Rich Schramm. SSDS work is funded by the David and Lucille Packard Foundation.

REFERENCES

[1] M. Chaffey, L. Bird, J. Erickson, et. al., "MBARI's buoy-based seafloor observatory design", MTS/IEEE Oceans 2004 Conference Proceedings, November 2004.

[2] T. O'Reilly, K. Headley, et. al., "Software Infrastructure and Applications for the Monterey Ocean Observing System: Design and Implementation", MTS/IEEE Oceans 20004 Conference Proceedings, November 2004.

[3] J. Graybeal, K. Gomes, et. al., "MBARI's operational and extensible data management for ocean observatories", The Third International Workshop on Scientific Use for Submarine Cables and Related Technologies, Tokyo, 2003.

Web References

[i] Monterey Ocean Observing System (MOOS): http://www.mbari.org/moos

[ii] Integrated and sustained Ocean Observing System (IOOS) : http://www.ocean.us

[iii] Ocean Observatories Initiative (OOI): http://www.orionprogram.org/OOI/

[iv] Java to Enterprise Architecture (J2EE): http://java.sun.com/j2ee/index.jsp

[v] Java Messaging System (JMS): http://java.sun.com/products/jms/index.jsp

[vi] Center for Integrated Marine Technologies (CIMT): http://cimt.ucsc.edu/research.html. Data for the CIMT mooring is available at http://ssdspub.mbari.org:8080/access/cimt.jsp

[vii] National Data Buoy Center (NDBC): http://www.ndbc.noaa.gov/ Monterey data through NDBC is available at http://www.ndbc.noaa.gov/Maps/Monterey_Bay.shtml

[viii] network Common Data Form (netCDF): http://www.unidata.ucar.edu/packages/netcdf/

[ix] IOOS Data Management and Communications (DMAC): http://dmac.ocean.us/

[x] Ocean Research Interactive Observatory Networks (ORION): http://www.orionprogram.org/

[xi] Laboratory for the Ocean Observatory Knowledge INtegration Grid (LOOKING): http://lookingtosea.ucsd.edu/

[xii] Marine Metadata Interoperability Project (MMI) : http://marinemetadata.org