Brett K. Lord-Castillo, Tomas M. Follett, Andrew Weiss, Bruce R. Mate, Dawn J. Wright

Tracking the Great Whales: An Arc Marine Case Study

Proceedings of the 27th Annual ESRI User Conference, San Diego, CA, Paper 1569, 2007
Track: Ocean, Coastal, and Marine Resources (OCE)
Session: Marine Data Model

Abstract

Arc Marine presents a new framework for collaboration in oceanic and coastal biogeographic analysis. With a common data structure, researchers can share results without conversion, automatically integrate environmental datasets, and develop common tools. This case study explores how Arc Marine supports the scientific process for the Oregon State University Marine Mammal Program (OSU-MMP). The OSU-MMP has used satellite telemetry to track nine cetacean species across two ocean basins. Customization of the model creates a geodatabase fit for data exploration, hypothesis testing, and research permitting compliance. The standardized framework allows for automated import of ARGOS downloads and easy conversion of standard environment rasters such as HDF MODIS or netCDF. Custom applications and tools conform to the core classes of Arc Marine. Thus, other projects using common tag types can utilize the same data loading tools and analysis tools without an extensive geodatabase redesign and reprogramming process.

Quick Reference

Introduction

Since 1983, Bruce Mate, Director of the Oregon State University Marine Mammal Institute (OSU-MMI), has used satellite telemetry tags to track the movements of the great whales. These investigations have unlocked the migratory routes and habitats of Right whales, Blue whales, Humpback whales, Fin whales, Gray whales, and Sperm whales. By mapping the distributions and abundance of whales throughout their migration, feeding, and breeding activities, the Marine Mammal Institute hopes to identify anthropogenic activities which stifle the recovery of the species. This research will thus ultimately lead to solutions to enhanced recovery of the great whales (Sherman 2006).

To enhance the effectiveness of the OSU-MMI satellite tagging program, the institute is in the midst of a geographic information systems project to create a geospatial repository of the tagging efforts and results which they have generated over the past seven plus years. These data sets cover six species of whales tracked with ARGOS satellite platform transmitter terminals (PTTs) over multiple ocean basins. As the GIS project advanced, a series of geographic information science questions began to emerge. This paper approaches one of the most critical of these questions, how can a geographic information system enhance the research advantages of satellite telemetry? Secondarily to that question, how can the Arc Marine data model be applied to the GIS project to answer this primary research question?

Advantages of Satellite Telemetry

In pursuing the movement and ranges of the great whales, satellite telemetry provides four key advantages: timeliness, continuous coverage (Lagerquist, Stafford, and Mate 2000), relationships to environmental data (Block 2005), and autonomous profiling of the animal's environment (Boehlert et al 2001). Timeliness or responsiveness allows remote sensed observations to translate into key management information in a matter of hours or even minutes as opposed to the months needed to fully realize the returns from marine surveys. The continuous coverage of satellite telemetry means knowledge of not only an animal's current location, but also where the animal was last week, next month, next season, or possibly even next year. The individual can be followed as it moves from breeding grounds along migration routes to summer ranges. This not only allows the identification of key habitats and migration routes, but also the timing of migration, feeding, and breeding.

The spatial character of these individual movement paths allows precise coordination with dynamic environmental datasets (Block 2005). This allows a deeper understanding of the natural variability in whale movements and how this variability relates to season, changing environmental conditions, and underlying geography. Finally, the tagged whale is able to act much like an autonomous profiling glider (Boehlert et al 2001), moving through its critical habitat and recording the environmental conditions from the animal's point of view. This provides a remotely observed view of the critical factors that drive the decision processes which determine when and where the individual whale moves. With an understanding of encountered microhabitats as well as timely mapping of critical habitats and migration routes, the marine resource manager is ultimately better equipped to respond to dynamic intersections of endangered whales with disruptive anthropogenic activities.

Arc Marine Case Study

While keeping in mind the overarching research questions, this project should be viewed primarily as a case study in the application of the Arc Marine data model to solve an information systems problem. Several key steps and questions are addressed, in particular customization of the Arc Marine data model to meet the needs of a subfield and of a specific research lab. Further, two major implementation questions are discussed: which programming languages to use for development and whether to develop an extension, stand-alone application or toolbox. Finally, the study returns to the research enhancements brought about by the transition to Arc Marine and the tools available from the larger development and research communities to meet future objectives.

Methods

This case study is developed on ArcGIS 9.2 using SQL Server 2005, ArcSDE on SQL Server, and Microsoft Access. The developmental databases have been instantiated in personal geodatabases while production is carried out in ArcSDE and SQL Server. Data model customization was facilitated by UML diagramming in Microsoft Visio. The legacy database resides on a personal geodatabase in Microsoft Access as well as additional data in Excel spreadsheets and text files. Schema development is based on the core Arc Marine data model.

The Arc Marine data model supports a wide range of industries and scientific disciplines that define their place of industry or research in the estuaries, coasts, seas, and deep ocean. Arc Marine is, foremost, a schema to support data collection on dynamic and multidimensional marine phenomena in a manner that models the real world in an object-oriented geodatabase (Wright et al 2007). This schema creates distinct community-wide advantages by increasing data interoperability, reducing analytic complexity, and facilitating tool development and dataset exchange. For the purposes of this case study, the most important structural aspect of the data model is the availability of standardized classes to represent model entities and the relational joins built in to the model which provide guaranteed relationships between data tables for complex querying.

As this case study is intended for implementation in the ArcGIS 9.2 environment, the choice of programming languages for development came down to the .NET framework (in particular VB.NET) and Python. While there are other languages to consider including C#, C++, and Java, the easy access to geoprocessor scripting in these two languages and to geodatabase records through the geoprocessor made VB.NET and Python the natural options in ArcGIS 9.2. These choices were further emphasized by existing expertise in the development team and reusable existing code base for database querying and the download of satellite telemetry results.

A critical element to this development language question is the question of the program form for the processing and analysis tools. Three options were considered in various combinations. The first option is a standalone processing and analysis application developed outside of ArcGIS that would only use the ArcGIS Engine APIs to provide mapping functions. The second option is an extension accessed via toolbar or menus within ArcMap. The extension would include data processing and researcher level analysis tools. The final option is a scripting and model toolbox that could be called both inside and outside of ArcGIS.

Results and Discussion

Arc Marine Customization

The Marine Mammal Institute researchers have placed several key specifications on this customization of Arc Marine, many of which extend beyond this lab to general use of the data model by the animal tracking community. These specifications can be summed up as follows:

- All raw data is retained, with preserved relationships to derived results and processing choices such as filters for invalid data. For example, time interval surfacing counts must be linked back to the ARGOS binary streams encoding the counts.

- Data acquisition quality and other auxiliary information are archived; the structure of this information can change depending on the tag type.
- The database stores operational information including animal approaches, photos, and tag deployment information, all linked to the appropriate measured data.
- Data filtering and querying choices by researchers are written to an audit trail, but not the resulting analytic snapshots.
- Hardware details are stored for future reference (tag components, manufacturers, raw data bit structure, etc.).

This customization involves a combination of subclasses, additional information tables, and the adaptation of existing data model objects to specific purposes. Although the database is primarily used for online analytical processing (OLAP) of the data repository, the complexity of the satellite transmissions combined with the storage of raw data and researcher decision processes calls for a heavy normalization of the database. Further, even with a large multiyear satellite tagging program, the number of data points is relatively small and heavy denormalisation may not be necessary to reach reasonable query response times. For this reason, the customization relies throughout on a snowflake database schema. Of the core Arc Marine objects, the three critical objects to this customization are the LocationSeries feature class, Vehicle class, and MarineEvent class. The LocationSeries and Vehicle class customizations handle the core issue of modeling animal locations from satellite telemetry while the MarineEvent class is used to address the first three of the specifications listed above.

LocationSeries, a subclass of InstantaneousPoint, allows for the spatial and temporal sequencing of a series of points moving through space. This class holds the critical geometry of satellite locates from the ARGOS telemetry messages. Animal tracks (as Track feature classes) are composited from the interpolation of movement between points. As such, tracks are calculated as on-the-fly features and not stored in the geodatabase. Animals themselves are not found in the core data model. Instead, an animal may take on several different classes depending on the animal's behavior compared to the object model. Most often, this representation is as a MeasurementPoint representing a single observation of the animal in a survey or as a Series of LocationSeries points and associated Track representing the movement of a single animal. Yet, point modeling was not appropriate for representing a tagged animal. Complex multi-dimensional data are difficult to connect to single points, particularly when the data comes from multiple sensors that are collecting between satellite fixes. Much of these data are associated with a time and an instrument rather than a specific location. Thus, the tracked animal is modeled as an instrument carrying Vehicle following an interpolated Track.

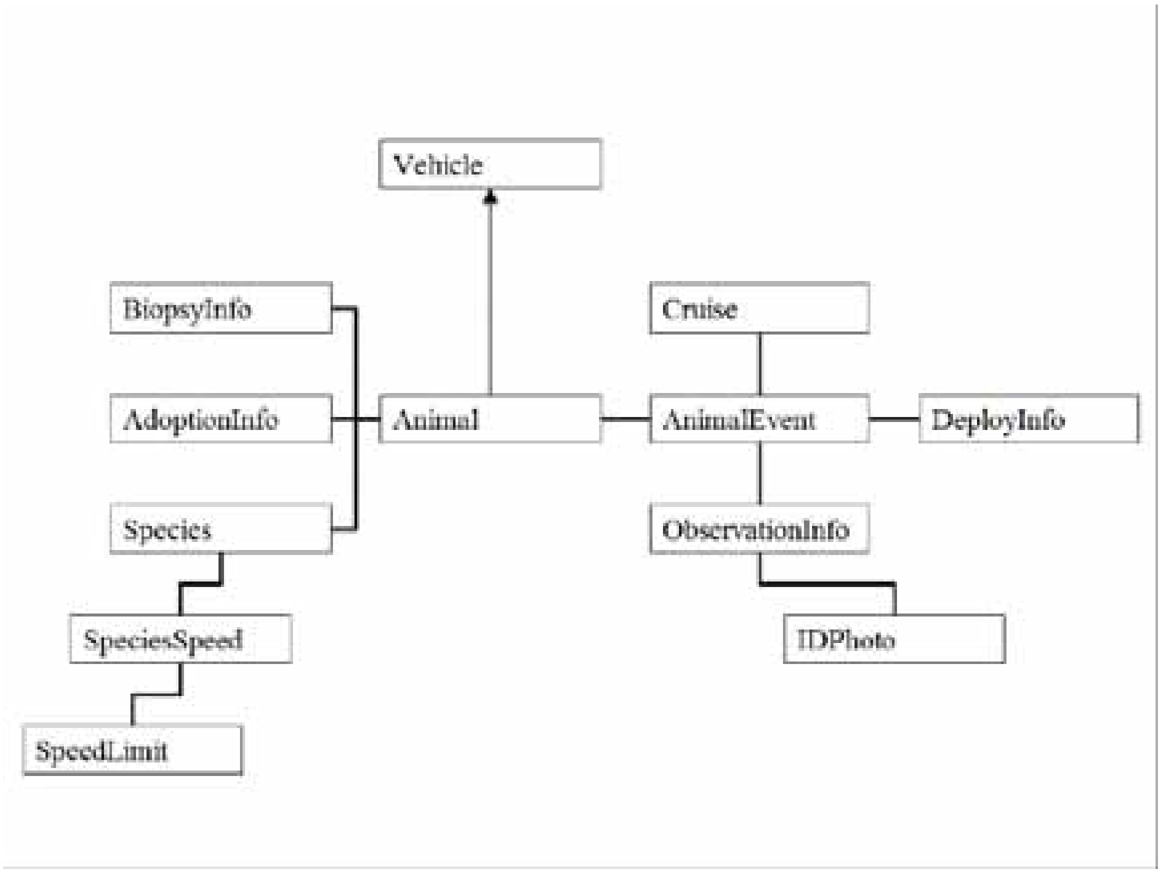

Figure 1. Snowflake schema of the Animal database class

The Animal subclass of the Vehicle class acts as the fact table in a snowflake schema which links together much of the database (Figure 1). In addition to its inherited attributes, the class also carries traits specific to animals: sex, genotype, estimated length, social group, and species. Within the Marine Mammal Institute program, the animal is also linked to biopsies (and related genetic information), adoptions (and related administrative information), and species specific attributes such as maximum swim speeds. As a Vehicle, an Animal can carry MeasuringDevices (in this case, the satellite tags) that relate directly to MeasuredData. These data, though, are often derived from raw satellite telemetry data that can carry a poor accuracy location or no location at all. Additionally, with ARGOS fixes there are two possible locations. The final additional subclass, AnimalEvent, relates the animal to these alternative locations, auxiliary information (such as receiving satellite or number of messages), and additional spatially oriented fieldwork information including tag deployments, ship approaches, and photographs.

MarineEvent is meant to be used for linear referencing of time or distance mileposts (M-values) along linear features like coastlines or ship tracks. In this case study, the linear feature, the animal's movement path, is an interpolated feature, but the timestamp M-values are used not only for dynamic segmentation on these derived paths, but also for time dependent analyses such as diel movement patterns. As these timestamped events mark interactions between the tracked whale and its environment, the subclass is termed the AnimalEvent. AnimalEvent inherits FromLocation and ToLocation mileposts (or timeposts) from its parent class. The most significant change is a relationship to the Animal subclass through VehicleID, relating the event to a vehicle animal.

This AnimalEvent dimension also carries subdimensions depending on the type of AnimalEvent. Table 1 lists the subdimension tables used in this customization, but is by no means a comprehensive list. As new tag types are added with different auxiliary attributes, new subdimension tables will be added.

Table 1. Context-dependent subdimensions of AnimalEvent

DeployInfo	Deployment of a measuring device onto an animal
TransmitEvent	Received satellite transmission with raw binary
ArgosInfo	Auxiliary information, ARGOS locations
ObservationInfo	Auxiliary information, field observation (links to photograph files)
DerivedInfo	Auxiliary information, interpolated or derived location
GPSInfo	Auxiliary information, FastlockGPS locations

These subdimension tables each carry a one to zero or one relationship with the AnimalEvent table, thus the joins from the animal to event information are context-specific (Figure 2). Though context-specific joins increase the complexity of query building, this aspect should be handled seamlessly by the interacting analysis tool. In exchange, the cardinality of the subdimensions is significantly reduced (particularly low frequency events such as DeployInfo). As will be addressed later, developers should program to the AnimalEvent interface and retrieve the attribute fields from the joined subdimension table rather than calling to specific named subdimension fields. For this reason, attributes specifically needed for analysis should be stored in MeasuredData and joined through MeasuringDevice to Animal. Locational information should be stored in the geometry of an InstantaneousPoint (normally a LocationSeries point) and appropriately flagged to reflect if the point is invalid, derived, etc. Only auxiliary information and archived data should be stored in AnimalEvent subdimension tables.

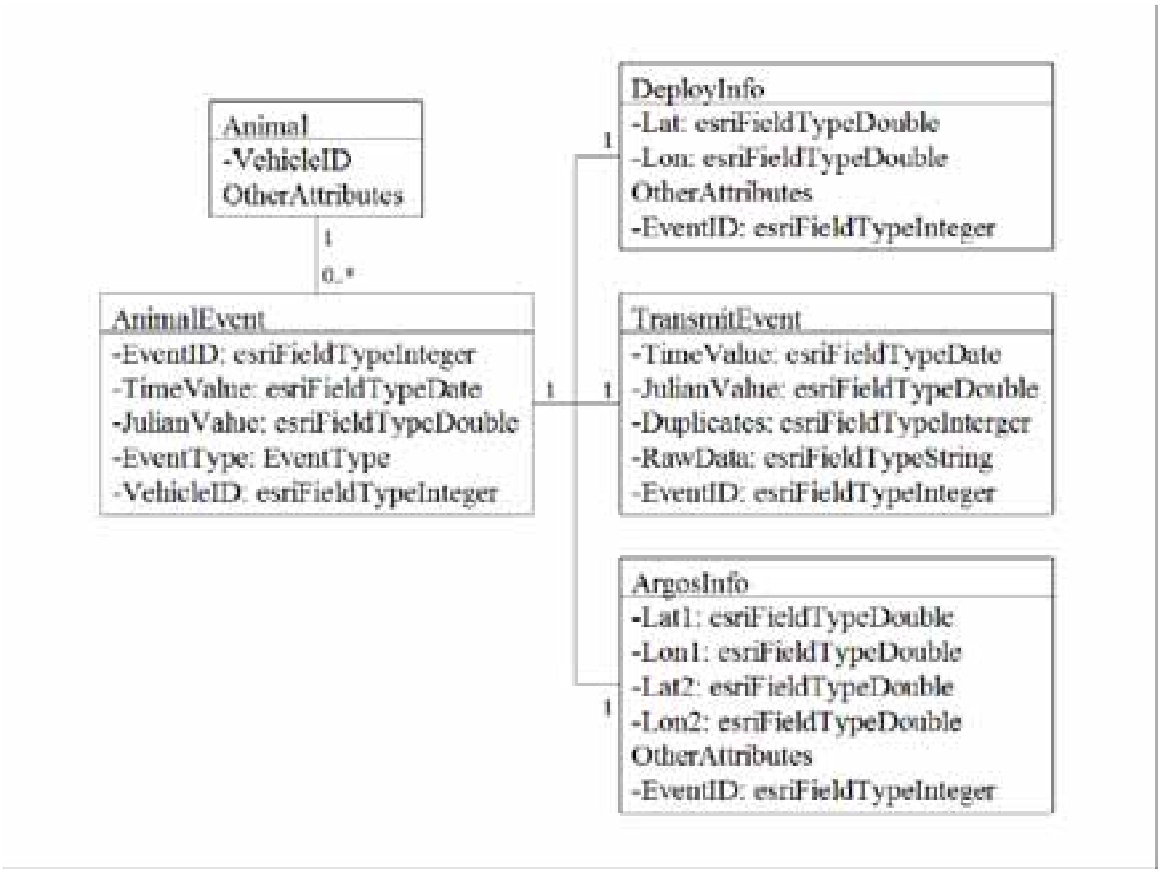
Animal
-VehicleID
OtherAttributes
1
0..*
AnimalEvent
-EventID: esriFieldTypeInteger
-TimeValue: esriFieldTypeDate
-JulianValue: esriFieldTypeDouble
-EventType: EventType
-VehicleID: esriFieldTypeInteger
1
1
1
1
DeployInfo
-Lat: esriFieldTypeDouble
-Lon: esriFieldTypeDouble
OtherAttributes
-EventID: esriFieldTypeInteger
TransmitEvent
-TimeValue: esriFieldTypeDate
-JulianValue: esriFieldTypeDouble
-Duplicates: esriFieldTypeInterger
-RawData: esriFieldTypeString
-EventID: esriFieldTypeInteger
ArgosInfo
-Lat1: esriFieldTypeDouble
-Lon1: esriFieldTypeDouble
-Lat2: esriFieldTypeDouble
-Lon2: esriFieldTypeDouble
OtherAttributes
-EventID: esriFieldTypeInteger

Figure 2. Context-dependent snowflake schema of AnimalEvent subdimension tables

To store data filtering and querying choices by researchers, InstantaneousPoints are assigned a default processing flag that indicates the origin, validity, and priority of the point for analytic processing (but no points are discarded). A researcher then works with a snapshot of the point features by applying a series of filters with arguments. The first filter will normally execute a query string against the default flagged set, but the researcher may also first remove default processing ("unsetting" the flags). This sequence of filter functions and parameters are saved with a UserID and creation date, creating an audit trail of research decisions. As each filter applies a specific function with a specific parameter, the snapshot can be recreated simply by reexecuting the saved list against the default processing set.

Additional tag hardware information is stored in a series of normalized tables (TagType-Hardware-Supplier, BitStructure-BitDetail, Schedule-ScheduleType-ScheduleDetail) linked to MeasuringDevice. The PTT number, frequently used for internal reference, then becomes the name of the device.

Development framework

The choice of a language and framework for development is ultimately not an exclusive choice. With the geodatabase as a central connection between modules, it is possible to use a Python script toolbox linked to a Visual Basic based ArcMap extension, and all in coordination with a standalone .NET application utilizing multiple languages. In comparing the application frameworks, each of the three forms, toolbox, extension or stand-alone application, has distinct advantages and disadvantages.

The stand-alone application carries an immediate advantage in licensing and accessibility. Not every researcher is a GIS analyst; an individual researcher or lab may have a preference for analysis in MatLab, R, S+, Excel, or other statistical applications. When distributing the tools to the tracking community, licensing requirements are reduced to ArcGIS Engine Runtime rather than an ArcInfo or even ArcSDE license. Yet, this accessibility of the application is offset by accessibility of the code. VB.NET carries a high learning curve with less modular code than Python scripting. Development will likely be centrally driven and the development time for a full stand alone application can be considerably higher than for an extension. As well, there must be a limit on the analytical scope of the application. While .NET allows access to the wide range of ArcGIS APIs, a standalone application cannot replicate the full spatial analysis and mapping capabilities of ArcGIS. As a final advantage of the application, many operations, such as tag hardware inventory and documentation of ship operations, have no need for the full power of a GIS, and may even be hindered by the reduced querying abilities of ArcInfo. These types of activities may even be carried out by a technician or research assistant who does not need access to the larger geospatial dataset.

An ArcGIS extension carries lower programming overhead than a stand-alone application, but still does not have the modularity and code accessibility of scripting. An extension, though, does offer true one-stop access to processing and analysis. The extension affords the ability to handle auxiliary data and metadata, spatial analysis, mapping, and querying all within ArcMap or ArcInfo, but with the licensing requirements of those applications. While such an extension can incorporate import functions to third-party statistical applications, carrying out data management in ArcGIS can limit adoption by research centers without appropriate licenses. Realistically though, many research groups in animal tracking will have institutional access to these licenses. Once again, the main disadvantage of an extension is the development time and centrally driven development. Without widespread adoption of the specific extension, a situation may emerge where every research group is reinventing the wheel.

This leaves the final option, Python scripting. The most pressing disadvantage with developing around a Python toolbox is a lack of access to the full ArcGIS APIs. While geoprocessor access can carry out many of the critical analysis and database updating functions, it cannot handle mapping and visualization tasks. Python, though, is supported by a quick learning curve that allows rapid development of sophisticated applications. The language is also strongly supported in multiple scientific communities, leading to the independent development of advanced graphical user interfaces (GUIs), statistical modules, and even tools for cross-platform development with .NET and COM which may provide workarounds for the API access disadvantage.

The chosen solution uses aspects of each form, though with a focus on Python scripting. Building on existing code, the standalone application will handle research and technician level access to update hardware and operations information or to create filtered data snapshots for analysis. The first priority, though, is the construction of automated tools for data updating, filtering, querying and export to analysis datasets. Generating the procedures in Python creates a rapid, modular development path, while the scripts can serve as the underlying code behind ArcMap toolbar buttons or a full-fledged Python-implemented GUI.

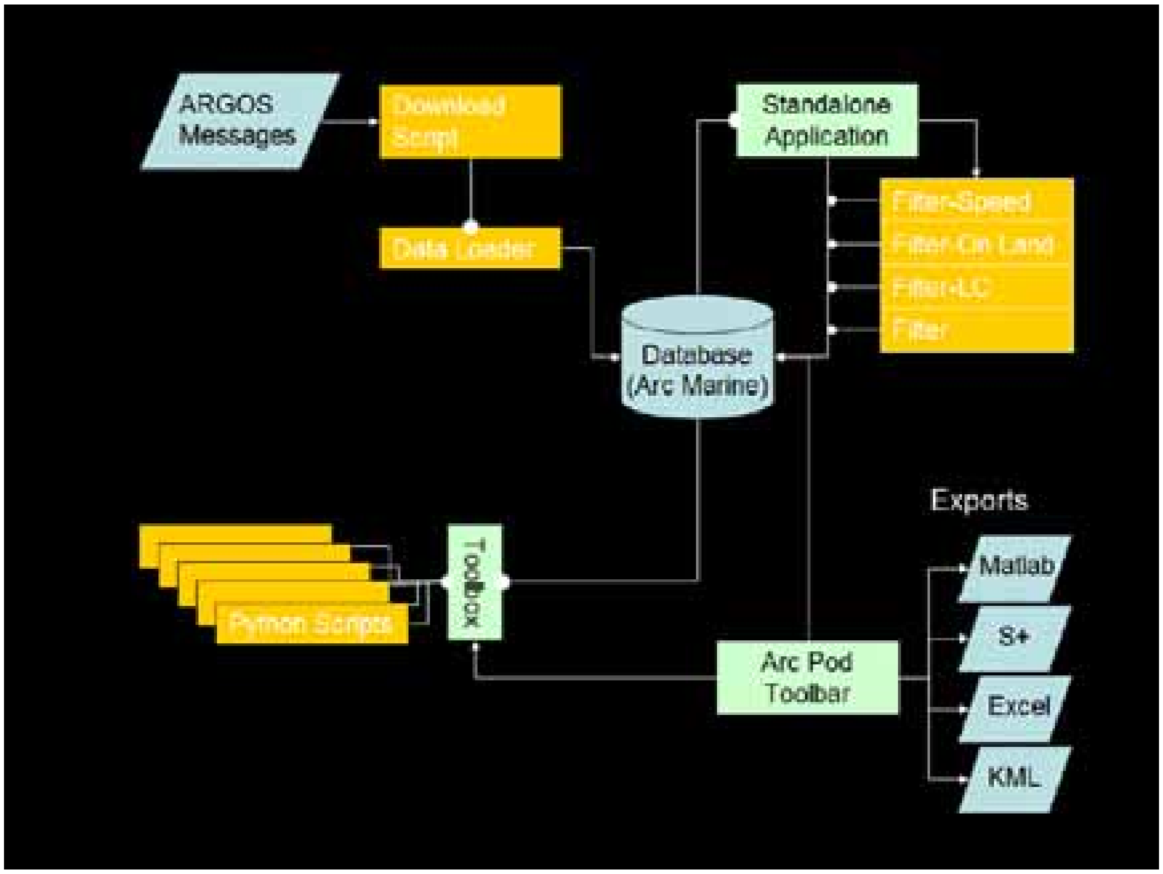

Figure 3. Case study application framework

Figure 3 depicts the five major sections of the application framework. All pieces of the framework center on the animal tracking customized version of Arc Marine, including snapshot LocationSeries point sets. In the upper left, automated daily updating is carried out by two Python scripts designed in a modular sequence. The first download script takes a series of connection parameters as an argument. That connection is used to download text results from Service Argos. The results are parsed to generate a Transmission container which holds, for each satellite transmission, PASS and DATA objects which respectively carry AnimalEvent (ArgosInfo) data and raw binary (TransmitEvent). The second data loader script takes this Transmission container and iterates through the PASS and DATA objects to construct LocationSeries points from the PASS data and new TransmitEvent records from the DATA object. MeasuredData records are then reconstructed from the new TransmitEvents (as well as other appropriate AnimalEvents such as GPSInfo for binary encoded Fastlock GPS results) according to stored procedures based on tag type. The binary translation procedures are separate from the data loader which is separate still from the download script. In this manner, changes to the tag program, database structure, or download service can each be dealt with separately without having to change the structure of the other components of the data update path. Each module simply has to generate return objects conforming to the argument requirements of the next module.

The next major component of the application framework, in the upper right, is the standalone application, or researcher/technician access level. This VB.NET based application implements only the key advantage of the standalone application: protected access to feature class snapshots and non-spatial information tables outside of ArcGIS. This application allows the execution of a limited number of short repeated operations in a low overhead program. More importantly, it ensures that the application of data selection filters takes place through a controlled portal where an audit trail is fully implemented. With a limited scope, the application also requires a minimal amount of development time, though it may be replaced in the future by a full-fledged Python application using Python implementations of Data Access Objects (DAO).

The lower left represents the current emphasis of development, the analytic toolbox. This toolbox represents a series of Python scripts which rely on expected interfaces to Arc Marine objects. Thus, these tools can be shared with other researchers and applied to shared datasets as long as each data table implements the same standard interface, in this case an interface based on the InstantaneousPoint class. Though early emphasis has been on procedural automated geoprocessing, more sophisticated scripting will be able to handle tasks such as metadata generation, editing, path generation, and movement modeling.

Finally, the lower right depicts the integrated application, or extension within ArcGIS. Rather than rewrite effective Python in Visual Basic or a similar language, this toolbar instead relies mostly on calls to proven geoprocessing scripts. Unlike the tools, which address Arc Marine object interfaces, the toolbar will directly access the geodatabase as well as available ArcGIS APIs. The most critical function of the toolbar will be the mapping tasks that are not handled effectively by ArcGIS Engine nor geoprocessing scripts, and menu based access to common export formats.

As identified by Rodman and Jackson (2006), currently available Python libraries allow for the eventual development of a standalone Python application in place of the present VB.NET implementation. In particular, the availability of the geoprocessor and DAO in Python allows for the combined use of the customized Arc Marine geodatabase, an external relational database holding non-spatial information, and external DAO read access to the geodatabase, allowing for faster and more complex queries (particularly *where* clauses) among AnimalEvents and auxiliary data tables. Like in Rodman and Jackson, this project uses WxPython as the primary GUI library and will use this library not only for stand-alone application development but also to develop advanced toolbox scripts and toolbar wizards. The native interface access of WxPython combined with the cross-platform development of Python (there is even a .NET implementation known as IronPython) also means that the entire framework can eventually evolve into a native look and feel cross-platform application. As a final note, three additional Python libraries of importance in tool development are Numpy, Matplotlib, and Makepy/pywin32. The first two libraries allow for the implementation of complex statistical operations, including an implementation of the plotting library of MatLab. Makepy/pywin32 is a utility that provides access to COM objects within Python, creating the potential for access to the ArcObjects COM API from within Python.

Interfaces

In order to maintain the modularity of code, and thus improve collaboration in tool development, programmers should attempt to program to an interface rather than to a specific object implementation. This case study introduced two such interfaces: InstantaneousPointUI and AnimalEventUI. AnimalEventUI is specific to this customization of the data model, while InstantaneousPointUI is a concept that should be applicable to any implementation of Arc Marine and provides a solid introduction to the concept of programming to the interface of core classes.

InstantaneousPointUI can be developed by examining the attributes of the InstantaneousPoint class. Based on its inheritance, this class contains OBJECTID, Shape, FeatureID, FeatureCode, CruiseID, TimeValue, ZValue, SurveyID, SeriesID, and PointType. Any InstantaneousPoint table can be expected to have these attributes, with all LocationSeries points in the table carrying PointType=4. Thus, the corresponding interface should implement read access to each of these attributes, and write access to attributes in non-key fields (TimeValue, ZValue, and the Shape geometry). Rather than creating encapsulated code for the class, the read and write behaviors are controlled by editing and update rules on the database. A tool should not, therefore, be written to access these key fields unless the tool itself implements update functions. The tool also should not rely on fields not present in the base InstantaneousPoint class; only TimeValue, ZValue, and Shape (and indexes as appropriate).

What does this imply for the development environment? While these rules may seem simple and straightforward, all too often custom tools are written to conform to specialized tables. As a result, the user has to go through a series of exports and transformations to even make a dataset usable by the tool. Any tool written to conduct analysis on LocationSeries points should require the InstantaneousPointUI. As a result, the tool will be guaranteed to function with any other LocationSeries point table. Once a research group has imported their animal tracking point observations to a LocationSeries table, no further transformations should be required to use a shared LocationSeries based tool.

For AnimalEventUI, the rules become more complicated. AnimalEvent also has context-dependent joins to subdimension tables. The linked subdimensions, though, is stored in the EventType field. Thus, with only access to the EventType domain and AnimalEvent, it is possible for a tool to gain access to the full event information through AnimalEventUI. The guaranteed components of AnimalEvent are:

- Index fields (MarineEventID, FeatureID, VehicleID)
- FromLocation (Inherited)
- ToLocation (Inherited)
- DataValue (Inherited)
- TimeValue
- JulianValue
- EventType
- Attribute list of the joined subdimension
- Record of the joined subdimension

Most AnimalEvent operations will involve either the creation of LocationSeries points from latitude and longitude information stored in subdimension tables or dynamic segmentation to assign locations to timestamped events stored in subdimension tables. As such, AnimalEventUI only needs to implement access to the index fields (to reach the related linear feature and animal), FromLocation and ToLocation (to dynamically segment the linear feature), TimeValue or JulianValue (for timestamp dynamic segmentation), and the Attribute list to find locational attributes. In order to implement AnimalEventUI, subdimension tables must also have the appropriate Lat or Lon prefix on locational attributes (a simple requirement when Service ARGOS results already include these prefixes on downloaded results). Through this interface implementation, any tool requiring the AnimalEventUI interface will be able to operate on any AnimalEvent table and its associated subdimension tables without additional transformation of the data. Thus, an operation as complex as data loading operations on multiple tag types can still be executed using a universal shared tool.

<u>**Conclusion**</u>

The major developments in this case study have direct implications in marine resource management, particularly ecosystem based management (McLeod et al 2005). As automation increases the timeliness of research, it also increases the responsiveness of management decisions. Rapid acquisition of whale positions means rapid assessment of anthropogenic threats for the manager in the field. A standardized data model with generalized tools increases the ease of multiple species analysis, and provides ready and repeatable access to results for analysts. Essentially, the management team is able to employ the exact same processes developed by the research team. Data model customization and multilevel application access not only means full retention of data, but also a full range of visualization in order to create clear concise reports and maps accessible through an application view customized to the resource manager or the public.

Overall, this means that GIScience enhanced satellite telemetry creates a faster and more accurate response when disruptive anthropogenic activity intersects with the real time critical habitat of a highly mobile species. The end result is improved ecosystem based management and enhanced recovery of the target species.

Acknowledgements

Financial support provided by the Education and Research Opportunities in Applying GIS and Remote Sensing in Coastal Resource Management program of the National Oceanic and Atmospheric Administration. (NOAA Grant # NA04NOS4730181)

Endnotes

Additional Arc Marine material including schemas and tutorial available at:

http://dusk.geo.orst.edu/djl/arcgis/

The current UML and XMI schemas for this case study are available at:

http://oregonstate.edu/~lordcasb/

References

Block, B. A. 2005. Physiological ecology in the 21st century: Advancements in biologging science. *Integrative and Comparative Biology* 45: 305-320.

Boehlert, G. W., D. P. Costa, D. E. Crocker, P. Green, T. O'Brien, S. Levitus, and B. J. Le Boeuf. 2001. Autonomous pinnipeds environmental samplers: Using instrumented animals as oceanographic data collectors. *Journal of Atmospheric and Oceanic Technology* 18: 1882-1893.

Lagerquist, B. A., K. M. Stafford, and B. R. Mate. 2000. Dive characteristics of satellite-monitored blue whales (Balaenoptera musculus) off the Central California Coast. *Marine Mammal Science*. 16(2): 375-391.

McLeod, K. L., J. Lubchenco, S. R. Palumbi, and A. A. Rosenberg. 2005. *Scientific Consensus Statement on Marine Ecosystem-Based Management*. Communication Partnership for Science and the Sea. Online. Available: http://compassonline.org/?q=EBM, 04/10/2007.

Rodman, L. C. and J. Jackson. 2006. "Creating Standalone Spatially-Enabled Python Applications Using the ArcGIS Geoprocessor," *Proceedings of the Twenty-Sixth Annual ESRI User Conference,* San Diego, CA, August 2006.

Sherman, L. 2006. Tracking the Great Whales. *Terra*. 1(2):2-8.

Wright, D. J., Blongewicz, M. J., Halpin, P. N., and Breman, J. 2007. "Arc Marine: GIS for a Blue Planet," Redlands, CA: ESRI Press, 275 pp.

Author Information

Brett K. Lord-Castillo

Graduate Research Assistant

Department of Geosciences

104 Wilkinson Hall

Oregon State University

Corvallis, OR 97331-5506

phone: 541-231-8462

fax: 541-737-1200

email: lordcasb-at-science.oregonstate.edu

Tomas M. Follett

Faculty Research Assistant

Coastal Oregon Marine Experimental Station

Hatfield Marine Science Center

2030 SE Marine Science Drive

Newport, OR 97365

phone: 541-867-0202

fax: 541-867-0128

email: tomas.follett-at-oregonstate.edu

Andrew Weiss

Faculty Research Assistant

Coastal Oregon Marine Experimental Station

Hatfield Marine Science Center

2030 SE Marine Science Drive

Newport, OR 97365

phone: 541-867-0394

fax: 541-867-0128

email: andrew.weiss-at-oregonstate.edu

Bruce R. Mate

Professor

Coastal Oregon Marine Experimental Station

Hatfield Marine Science Center

2030 SE Marine Science Drive

Newport, OR 97365

phone: 541-867-0202

fax: 541-867-0128

email: bruce.mate-at-oregonstate.edu

Dawn J. Wright

Professor

Department of Geosciences

104 Wilkinson Hall

Oregon State University

Corvallis, OR 97331-5506

phone: 541-737-1229

fax: 541-737-1200

email: dawn-at-dusk.geo.orst.edu