PURPOSE

The coastal geomorphology research group (CGRG) has accumulated large volumes of data and is exploring options to view available data, query it, and update available datasets. The general idea is that all data should be visible to everybody accessing the site, but the ability to upload and/or download data should be limited to a list of authorized users. The purpose of this project was to set up a data repository that allows authorized users to upload and download data.

OBJECTIVES

In order to achieve this purpose, the following objectives must have been met:

1. Develop a coastal geomorphology website using PHP.
2. Store and manage a diverse assemblage of data types, including LiDAR (light-detection and ranging), bathymetry, aerial and satellite imagery, and geophysical data (*e.g.* ground-penetrating radar, seismic, electromagnetic induction).
3. Map the available data in an interactive map console.
4. Select upload and download capabilities for authorized users.
5. Map the publications of the coastal geomorphology research group.

WEBSITE DEVELOPMENT

The first step in this project was to redevelop the existing marine and coastal website. The current site is very out of date and did not include the necessary mapping functionality. The redeveloped website was launched at geog489-16.tamu.edu/coastal.php (Figure 1). It is managed by Microsoft Internet Information Services (IIS), and is composed of several PHP (pre-hypertext processor) pages. There are several reasons for using PHP in this project.

The first significant advantage of incorporating PHP into this project is that I was able to generate a header and footer PHP file that can be inserted into each web page. This means that I can update a single header file and single footer file and the changes are pushed to each individual page. Making the website more dynamic and easily updatable was a request by the client, and this was a step towards that goal. The second advantage of using PHP in this project is that it is a secure way to connect to, query, and return information from Microsoft SQL Server database containing the available datasets.

The website graphic design was updated using cascading style sheets (CSS), JQuery, and Javascript. Each design element was contained within a hierarchy of tags that allowed me to format the pages accordingly. The CSS file was used to control the layout and design properties of the pages. This was used over hard-coding the styles because it is more dynamic and easily updatable. JQuery and javascript were used to make the website and webmap more interactive. In addition to this interactivity, OpenLayers, the webmap API used in this project, employs javascript syntax for its functionality.

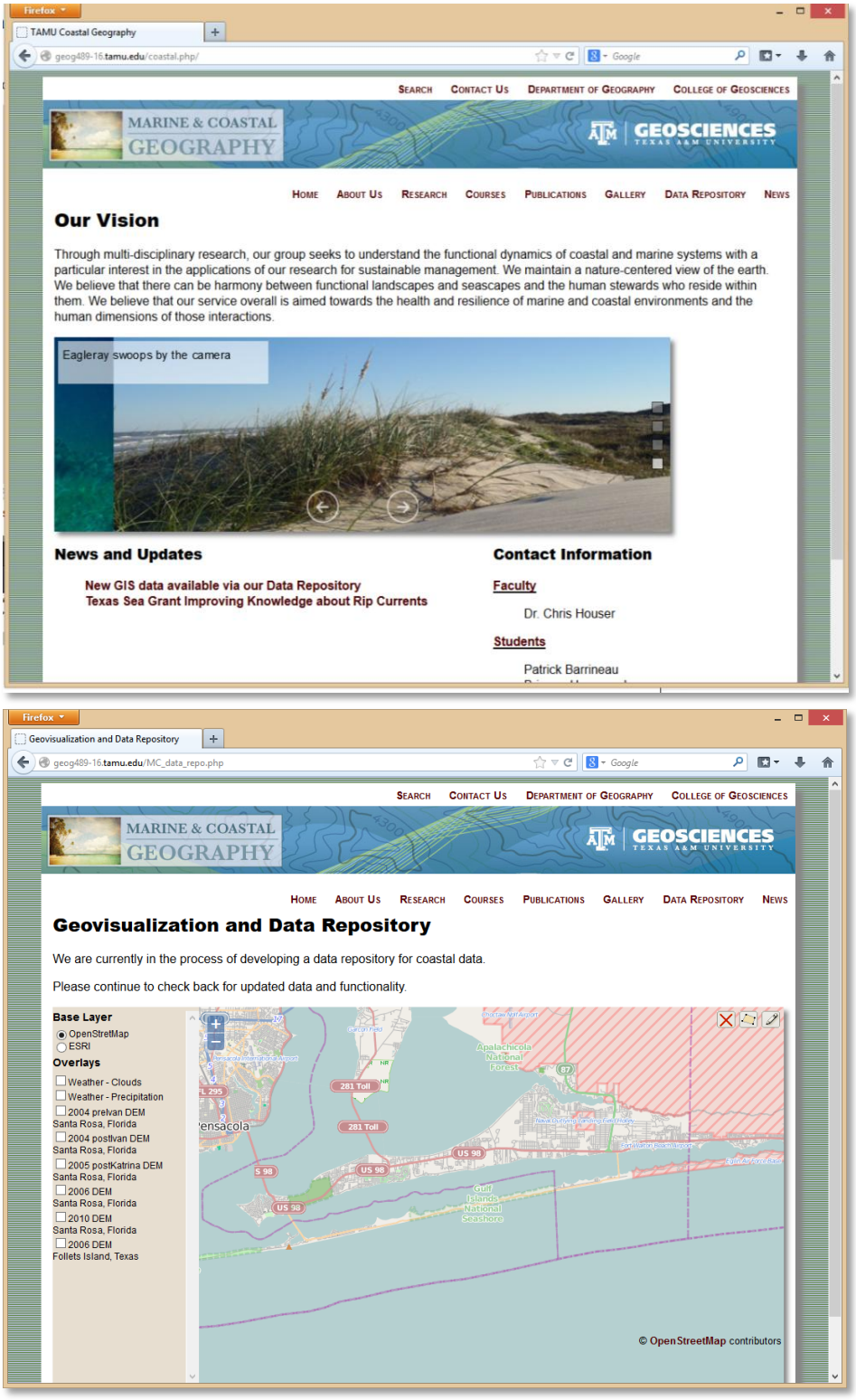

Figure 1: New TAMU coastal geomorphology research group website.

DATABASE MANAGEMENT

Data was managed using ArcGIS Server 10.2, and available datasets were managed using SQL Server. This server database package enables the site to host raster and vector datasets and custom python geoprocessing scripts (future work). The advantage of using ArcServer is that it can store and organize both raster and vector datasets. These map services can either be pre-cached or dynamically cached. Pre-cached datasets are advantageous for this project because they limit the required processing power of the client and internet connection. The disadvantage of pre-caching the tiles is that they occupy server storage. Pre-cached tiles were used for this project in order to minimize demands from the client and maximize performance.

Once a map service was generated and tiles were cached (Figure 2), the available dataset was input into a SQL Server database table (Figure 3). This table contains the name, date, access url, and other metadata information about each dataset. This table was accessed via PHP pages (Figure 4 left) and the results were returned formatted as JSON (javascript object notation; Figure 4 right). The JSON object was then consumed by the OpenLayers API and the available datasets were added to the webmap as display options.

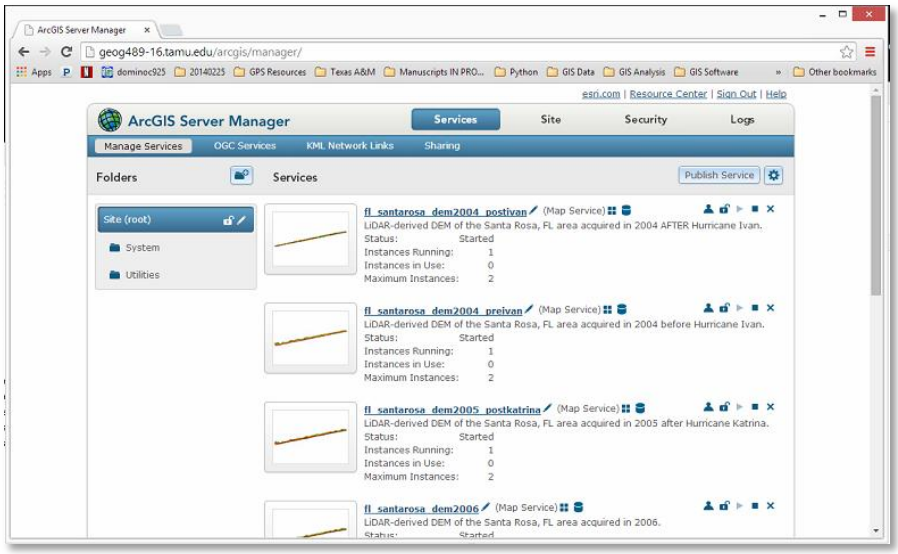

Figure 2: ArcGIS Server available web map services.

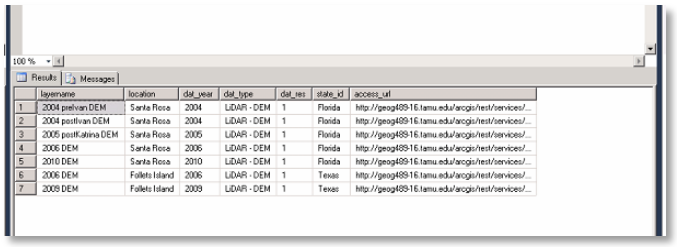

Figure 3: SQL Server table containing the available datasets.

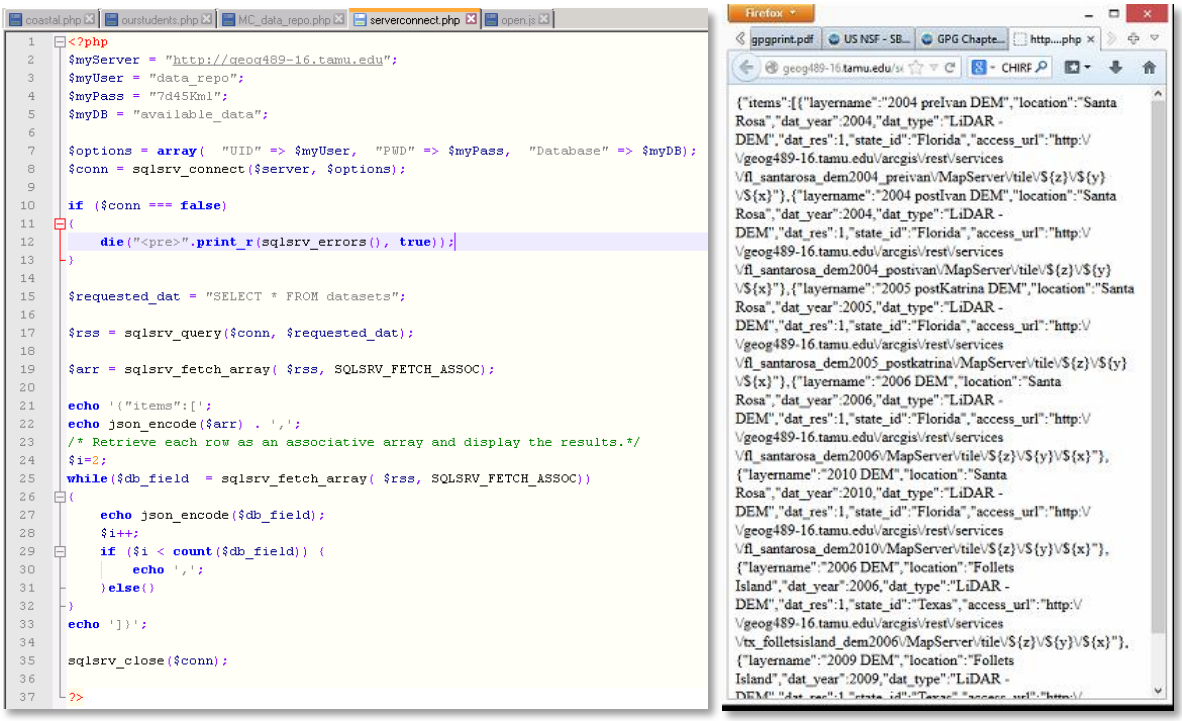

Figure 4: PHP script to query SQL database (left) and resulting formatted JSON object (right).

WEB MAPPING using OPENLAYERS

OpenLayers web map javascript API was used over Google Maps or other sources because it is 100% open source. The map functions similar to Google Maps, in that you can toggle layer visibility, display map layers, and draw on the map. I believe that the syntax is more intuitive and easier to adapt to your custom application. Another significant advantage of using OpenLayers is that the data is continuously updated by municipalities and private sources. This means that it is more likely to have the correct updated version before Google Maps.

The OpenLayers API is written in many different languages. For this project I chose to use javascript because it is easily interpretable and open source. This is one area where the functionality of PHP shines through. PHP contains a built-in function to convert the returned data string to a JSON file. Called “json_encode”, this function consumes a raw query string and returns the correctly formatted JSON object for each database table entry. The JSON object returned from SQL Server (Figure 4 right) is consumed by the javascript and is looped over to add the available datasets to the web map (Figure 5).

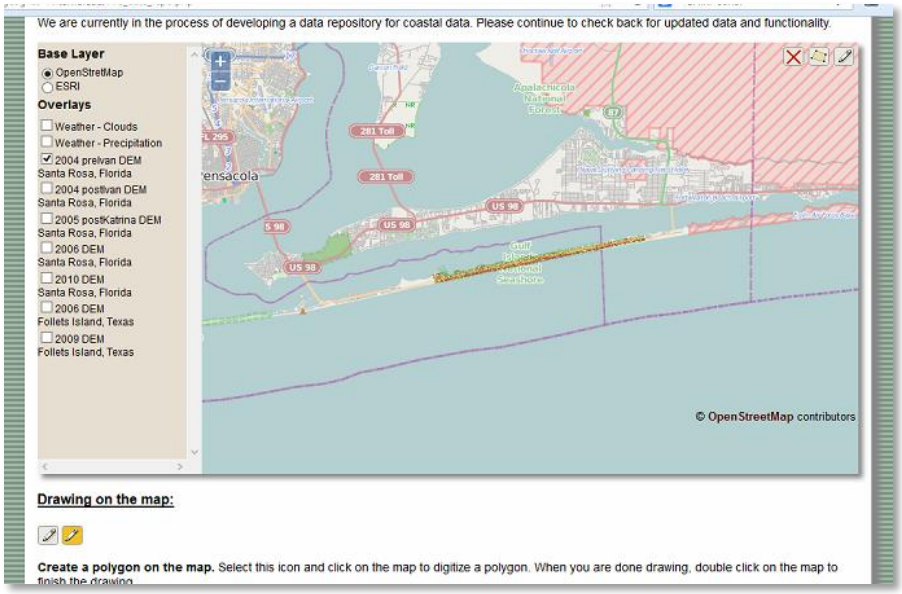

Figure 5: OpenLayers webmap displaying an available raster dataset.

RESULTS

The project outlined above met three of the five stated objectives. The new coastal geomorphology research group was successfully developed and the website is more database-driven than before. Raster and vector data types were stored and managed using ArcServer 10.2, and available data was added to an interactive web map. The most significant hurdle in this project was learning PHP and using it to establish a connection to and query from a database table.

I believe that PHP is a valuable skill in that it allows you to establish a secure database connection and dynamically format multiple web pages with a single PHP file. In addition, it is open source, which is advantageous to users entering the private job market that are not using Windows-based servers.

Learning the OpenLayers javascript API was more intuitive than the Google Maps API. This likely stems from the fact that it is open source and has a much broader developer base. I believe that OpenLayers is a much more user-friendly mapping API that does not limit the number of map requests and is more adaptable. The functionality of OpenLayers is not fully utilized by the current system; however, future work will be to incorporate this functionality to query data from the database.

Two objectives were not met for this project: 1) upload and download capabilities for authorized users, and 2) map publications. The upload and download objective requires a user management system. This was not implemented because efficiently developing the website and generating the PHP connection and resulting JSON took more time than originally anticipated. Publications were not mapped yet, as we do not have geographic locations assigned with every publication. Additionally, we are yet

unsure about the copyright laws regarding referencing published works where the publishers are profiting.

FUTURE WORK

The CGRG is in need of a data repository, from which we can upload, store, and download the currently available data. The results of this project represent a good first effort in addressing this need. Client, Dr. Chris Houser, is satisfied with the progress on this project up to this time, but acknowledges the need to further the capabilities of the system in the future.

Future work will include:

1. Migrating from ArcGIS Server (proprietary) to GeoServer (open source).
2. Migrating from Microsoft IIS (proprietary) to Apache (open source).
3. Migrating from SQL Server (proprietary) to MySQL and PostGres.
4. Development of a user-management component/system to track the authorized actions of users accessing the page.
5. Mapping the geographic foci of CGRG publications.
6. Development of an FTP site for authorized data upload and download.

Future work is primarily focused around migrating from proprietary software to open source software. However, the most significant future developments will be mapping the CGRG publications and the development of a user management system, which will serve as the foundation for upload and download capabilities of authorized users.

The project has advanced my understanding of web GIS and development in several capacities. First, it has provided me the opportunity to gain firsthand knowledge of setting up and administering a corporate-level server. This is a valuable skill requested or required by many GIS job openings. Second, implementing PHP and OpenLayers has provided valuable insight into the comparison between the C# and Google Maps combination that we employed in labs throughout the course. I believe that PHP and OpenLayers are more broadly applicable skills in the job market. PHP is often explicitly requested by employers, and familiarity with a variety of APIs diversifies the marketability to perspective employers.

CONCLUSIONS

Although the project did not achieve all of the stated objectives, it met the majority of them. The client was satisfied with the project results, and future work will continue as we develop a user- management system and add the ability to interactively query (geographic and non-geographic queries) the available data.

The project has enhanced my skills by providing applied lessons in server management and administration and interactive web GIS development. The skills from this project

(*e.g.* PHP, OpenLayers, javascript, HTML) will make me more marketable in the GIS community and will better enable me to contribute to the current knowledge base of web GIS.