Software Design Proposal
Scientific Data Management System

Alex Fremier
Associate Professor
University of Idaho College of Natural Resources

Colby Blair
Computer Science Undergraduate
University of Idaho Computer Science Department

December 9th, 2011

Contents

List of Figures

1 Project Summary

In today's scientific environment, there is a growing focus on data storage and processing. Some of the largest problems scientists face are related to the data deluge, and the inability to conceptalize problem and solutions from large tracts of data. The result is many attempts by scientific professionals to design software, leading to many bad software designs. Vast amounts of expensively collected data never get processed, because the software designs take so much maintenance. Data is often duplicated, often in different locations, lost, or never makes it from collection to analysis. The overall lack of system designs to handle the data further hinders analysis.

This proposal is for a project that helps solve these design problems for many scientists. The project is a data harvester and organizer tool. It is a local web interface that allows scientists to collect, query, organize, and share data with other researchers. Many of these scientists work with similar data sets, ask different questions, and need immediate search tools. The proposed tool would first allow users to visualize and filter data, and secondly prepare and ship the data off for analysis. It also helps users visualize data graphically, to help the conceptualize, organize, and refine data, before transporting and processing it. This project proposes using PTAG and othe ecological data sets from the Columbia Basin to test design, but should be exdendable to many different users with very different data sets.

The tool will not overlap with the functionality of existing tools like PTAGIS or DART. It will carefully handle data sharing, choosing opt-in, lock-down policies by default.

1.1 Background

In the Columbia Basin today, millions of federal dollars are spent on PTAG and othe systems to collect environmental data. This data includes fish location, ecological community composition, and abiotic data. Yet, the project results from most research are far from concrete. Despite the lack of understandable results, important decisions that affect the local environment and economy have to be made. Decisions like whether or not to conduct major habitat restoration projects.

The Columbia Basin is just one small example. Bioinformatics is another data intesive study that generates more data that it can process. Its estimated that less than 10% of the data collected by bioinformatic researchers at the University of Idaho actually makes it through proccessing [1]. The rest has to be filtered as best as can be managed, and the low value data trimmed out. The problem with 10% data use, is that not just the fat, but the meat and bone has to be cut away. Either significantly more data must be analyzed, or significantly less should be collected. At the very least, storing the data in an easily readable format can show where the gaps are. So far, our experience and research shows similar shortfalls in data analysis in the Columbia Basin.

1.2 Problem Statement

One of the biggest problems researchers in the Columbia Basin have is getting their data somewhere meaningful. Central databases like PTAGIS offer a central storage and clients can push data there, but they don't offer useful tools for managing data. Once the data is pushed, it is hard to access and manipulate. Researchers are often in remote locations, and have low bandwidth connections. Creating a robust tool will guarantee ease of use for the data, no matter the location.

Many researchers have significant data management tasks before even thinking about pushing data to the cloud. Once they are ready, they need seemless ways to synchronize data to and from

the cloud. They also may want to query their data, and filter it into small subsets. Most researchers don't have time to learn new programming language or interfaces. They need a data management tool that has a user interface that is familiar to use cases they already understand. Once they have created a data subset, they will want to share it, save it, copy it, and compare it with other data sets. Getting the right data to the right place in the right amount of time is crucial.

1.3 Objectives

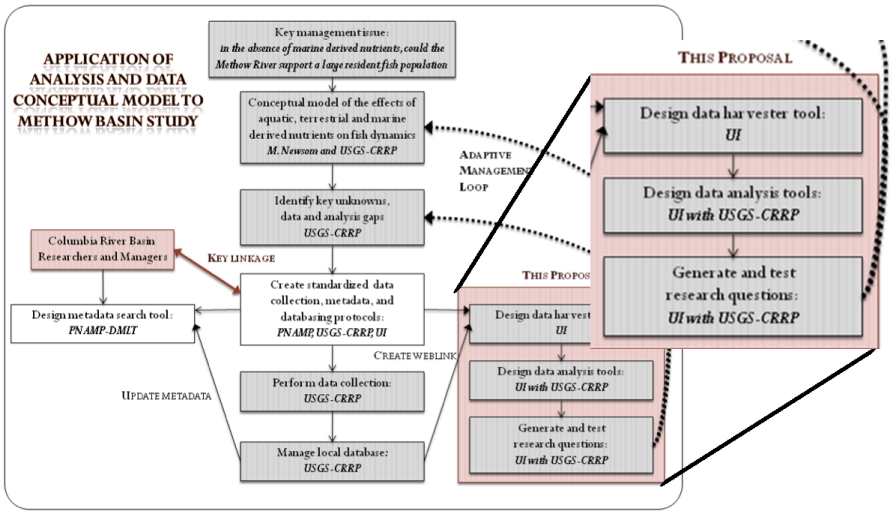

Figure 1: This proposal software's application domain in USGS-CRRP

The proposed project will tackle the UI data harvester portion of the grant funded USGS-CRRP project (Figure 1), directed by Alex Fremier of the UI College of Natural Resources. The data management tool will be a web-like GUI that can be installed locally on the clients' computers. The tool will contain a expandable meta data server that can be connected to others, for a **decentralized** data storage cloud. The cloud will consist of the many instances of the tool, acting as a cloud of **decentralized databases**. It will contain an interface to an existing or custom **networking protocol** that will allow for many different data formats to be synchronized across the cloud. The project will also have graphic interfaces for **seamless data management**.

The decentralized cloud model allows some serious advantages to researchers. They can see exactly what their data looks like, before they submit it. They can filter out bad data before it consumes bandwidth, and can retract undesirable data from the cloud, even after synchronizing it. It also frees them from central storage service management and fees, and **encourages internode and inter-research communication**. The tool can be setup on multiple hosts, and can be used for the benefits of the centralized cloud model. Each client can decide what topology suites them best.

The decentralized cloud model implements a distributed database model. Distributed databases increase availability and reliability. They are more easily expandable, and can better protect from data loss from local disasters or malicious attacks. Moving data to where it is in highest demand also increases query performance. Offloading archive data to remote site with more resources preserves

local resources. Replicated datasets can guarantee better availability. By staging data locally and filtering it before allowing it to be exchanged, the autonomy of the organization is better preserved.

The network protocol will allow incremental synchronization of data from host to host, even in less reliable environments. The tool will create an **outreach** from researchers in high availability areas to those in low availability, low bandwidth areas, and back. The network protocol will have support for major data formats (i.e. SQL, CSV, more), and allow users to send incremental pieces of the data. In the case of very large data sets or low bandwidth, the receiver of the data can still use it. Whether the data takes more time to transfer, or never completes.

The seamless interface will allow users to sort their data needing only basic knowledge of computers. Users will be able their mouse to select datasets and apply filters to them. The tool will allow them to query local or remote data, or to dynamically join both. Queries will create data subsets that users can bring to a workspace. The tool will be able to sort data however the user likes, on the fly. It will then be able to graph the ranges in the subset in most ways the user could want to sort them.

Once the user has manipulated their data subset to their satisfaction, they will be able to save it in the meta data server's format, or in a set of major data formats. The tool will also have analysis modules that will allow them to run analysis on the local host. These modules will be extensible to running analysis jobs on other designated compute hosts, like workstations, clusters, or even Amazon's EC2. The tool will come only with basic functionality for local analysis modules, but will be extensible to the heavier compute options.

2 Project Description

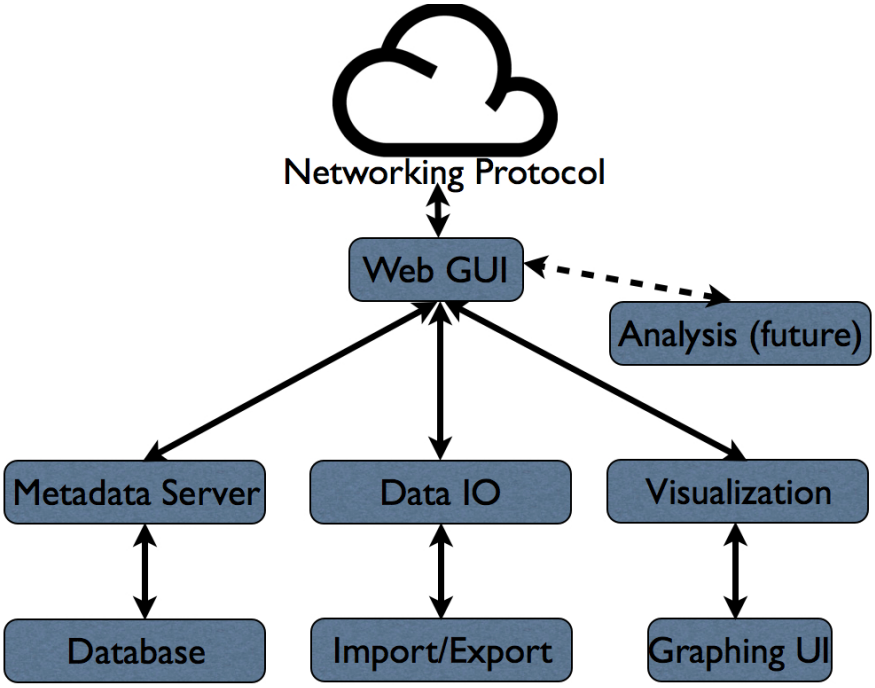

Figure 2: Software tool's component layout

2.1 Methodology

The team will develop the software in whatever style they like, but will report weekly to the customer with progress reports and new project projections. Project leaders may occasionally travel with Alex Fremier to meet with other users, but only optionally. Weekly trends will give projections, and a customer acceptance test will conclude at the end of the semester.

2.1.1 Software Requirements

This project will be written a web interface, preferably using a Model View Control (MVC) Design pattern like Ruby on Rails. The web gui will implement a RESTful interface, and will implement modules for inputing, filtering, visualizing, and analizing data.

The project will contain 4 different module sections:

1. Metadata server and database
2. Data input / output
3. Visualization and selection interface
4. Networking protocol

5. A stub for an analysis pipeline

The metadata server module will allow the user to specify what format they will like to use (MySQL, PostGreSQL, SQLite, Casandra). It will help them manage backups, both locally and offsite, and the scheduling of backups. It will allow them to select which offsite datasets or entire hosts to mirror, and help them automate the management of data collisions. The interface will also allow them to monitor the health of offsite hosts, and view statistics about the network health going to the host.

The data input / output modules will help users input various different data standards (CSV, R language tables, more) into the tool's common metadata format, which will be similar to an SQL table format. These modules will also allow users to apply a few useful filters to the data, similar to how Microsoft Excel can. When an data input session start, the data will be staged as a sandbox dataset, and the user will be able to sort, apply filters, and delete data similar to a spreadsheet, until the data is formatted correctly.

The data visualization and selection modules will help users with graphing their data in an array of charts, and have spatial data represented on 3D plots. Users will be able to select ranges from the charts, and have the resulting datasets cached for them for future processing. Users will be able to join multiple datasets, split datasets, and describe relations between different datasets. They will be able to create lookup tables between datasets on the fly, allowing them the maximize data usability while minimizing data redundancy.

The networking modules will provide different ways to syncronize data. These can utilize current network protocols, as well as try to implement new ones. Examples of existing protocols can include rsync, http, ftp, scp, SOAP XML, built in database protocols, and more. The data should make sense, however, in whatever incremental state it is in. If a large transfer, or a small transfer on a slow connection, fails, then the receiving host should see the transfered pieces of data in a dataset, and see a visible gap (i.e. through the visualization tools). The client can then choose to resend, and the minimal amount of retransmits should occur to finish the transmission.

The distribution will be installable on Windows, Linux, and Mac, and will include any third party source needed. It will be downloadable to customers via a repository.

3 Customer

The customer will be Alex Fremier of the UI's College of Natural Resources. If this project proposal is accepted, specific customer requirement will be drawn and presented at the beginning of the Spring 2012 semester. An acceptance test will be specified along with a full list of function tests.

4 Bibliography

References Cited

[1] Foster, James. *Visualizing Human Microbiome Ecosystems*. University of Idaho: Computer Science Colloquium, December 7th 2010. Seminar.