

University of Nevada, Reno

**Environment for Large Data Processing and Visualization Using
MongoDB**

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in
Computer Science & Engineering

by

Rui Wu

Dr. Sergiu Dascalu, Dr. Frederick C. Harris, Jr /Co-advisors

December, 2015

Copyright © 2015 by Rui Wu

All rights reserved.

Abstract

Data means treasures to both scientists and business people. Scientists can discover significant rules and theories beneath data. People involved in business can find their potential customers and improvement suggestions from exploring data. To uncover these valuable treasures, we need effective and efficient tools. There are some traditional quality data management, processing, and visualization tools and systems. However, most of them are no longer as powerful as before when the data size grows larger. These tools and systems may respond slower or even stop working because of increasingly large data volumes. We are now in the digital era. Data is generated at an amazingly high speed. Remarkably, 90% of the data in the whole world has been generated in the last two years. To manage, process, and visualize large data, we need new tools and systems.

In this thesis, we introduce the ELDP&V system. It is designed to manage, process, and visualize large data. We used MongoDB to record file paths and stored files in a filesystem to manage data. By using this method, we do not need to create different schemas for different files. The system offers users basic data processing methods, such as distribution frequency histograms. It also provides scientific models for users for data processing. ELDP&V can visualize data with line charts, bar charts, and 2D maps. Most of the visualization results are interactive. We implemented ELDP&V with some new ideas and design workflows, which led to higher performance. For example, in a test we performed the system has visualized 13,455,368 records in 50 seconds. We tested the same dataset with some traditional web-based tools. They stopped working or produced

some errors. When ELDP&V visualizes a time-based file, it displays the chosen items based on timestamps. This means the loading time will not be affected by the file size.

The thesis presents background on big data, outlines ELDP&V's goals and characteristics, the proposed system's design, and demonstrates its prototype in action. It also contains a comparison with related work and provides several pointers to directions of future work.

Keywords: large data, data management, data processing, data visualization.

Acknowledgements

I am really grateful to my advisor Dr. Dascalu and my co-advisor Dr. Harris for their selfless guidance. They take care of me as their own child not only in my research but also in my daily life. I also want to thank Dr. Hiibel for serving on my committee and offering precious feedback.

I would also like to thank my parents, Dan Yao and Zhangao Wu. They supported me to pursue my dream to study cutting-edge computer science theories and technologies here, in the USA, even if I am the only child in my family. My uncle, Dr. Min Yao, also helped me a lot. When I came to America, I lived in his house for half year and he drove me to the university and back every day.

Last but not least, I want to thank my friends Eric Fritizinger, Lisa Palathingal, Chase Carthen. We met at University of Nevada, Reno and I never felt lonely because of these my new “family” members.

This material is based upon work supported by the National Science Foundation under grant number IIA-1301726.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Table of Contents.

| | |
|---|----|
| Chapter 1 INTRODUCTION..... | 1 |
| Chapter 2 BACKGROUND..... | 4 |
| 2.1 Big Data..... | 4 |
| 2.1.1 Big Data Definition..... | 4 |
| 2.1.2 Big Data Challenges | 5 |
| 2.2 Interactive 2D Visualization..... | 5 |
| 2.2.1 Interactive 2D Visualization Definition..... | 6 |
| 2.2.2 Prevalent Libraries | 7 |
| 2.2.3 Traditional Workflow vs New Workflow | 9 |
| 2.3 Data Processing..... | 11 |
| 2.3.1 Basic Data Processing..... | 11 |
| 2.3.2 Models..... | 11 |
| 2.4 Database | 12 |
| 2.4.1 Traditional Relational Database Management System | 12 |
| 2.4.2 NoSQL Database | 13 |
| Chapter 3 ELDP&V GOALS and CHARACTERISTICS | 16 |
| 3.1 ELDP&V Goals..... | 16 |
| 3.2 ELDP&V Characteristics | 17 |
| 3.3 Scenarios | 18 |
| Chapter 4 ELDP&V DESIGN | 20 |
| 4.1 Requirements Specification..... | 20 |
| 4.2 Use Case Diagram | 22 |
| 4.3 Use Cases | 25 |
| 4.4 Flowchart..... | 27 |
| 4.5 Glossary..... | 29 |
| Chapter 5 ELDP&V PROTOTYPE..... | 31 |
| 5.1 ELDP&V Main Components | 31 |
| 5.1.1 Data Visualization..... | 31 |
| 5.1.1.1 CSV Visualization | 31 |
| 5.1.1.2 NetCDF Visualization..... | 41 |
| 5.1.2 Data Processing..... | 50 |
| 5.1.2.1 Basic Method | 50 |
| 5.1.2.2 Scientific Models | 53 |
| 5.1.2 Data Management | 56 |
| 5.2 ELDP&V Video Demo | 56 |
| Chapter 6 COMPARISON WITH RELATED WORK..... | 57 |
| Chapter 7 FUTURE WORK | 62 |
| Chapter 8 CONCLUSIONS..... | 65 |
| REFERENCES | 68 |

List of Tables

| | |
|---|----|
| Table 3.I. Fish Hydrologist Persona | 18 |
| Table 4.I. Functional Requirements of ELDP&V | 20 |
| Table 4.II. Non-Functional Requirements | 22 |
| Table 4.III. Upload Files Use Case | 25 |
| Table 4.IV. Zoom in Line Chart Use Case | 26 |
| Table 4.V. Overlay Visualization Results on Map | 26 |
| Table 4.VI. ELDP&V Glossary | 29 |
| Table 6.I. Comparisons between Different Tools and Systems..... | 60 |

List of Figures

| | |
|--|----|
| Figure 2.1. Line & Bar Graph..... | 7 |
| Figure 2.2. Comparisons between Traditional and New Client/Server Web Application Data Visualization Workflow | 10 |
| Figure 2.3. Table Objects Example in RDBMS [SQL RDBMS Concept 2015]..... | 13 |
| Figure 2.4. Comparison between SQL and NoSQL Database [MongoDB 2015]..... | 14 |
| Figure 4.1. ELDP&V Use Case Diagram | 23 |
| Figure 4.2. ELDP&V Flowchart..... | 28 |
| Figure 5.1. Traditional CSV Visualization Workflow..... | 32 |
| Figure 5.2. Screenshot of CSV Line Chart Visualization..... | 34 |
| Figure 5.3. Multi-processes CSV Visualization Outputs..... | 36 |
| Figure 5.4. Final Visualization Result Image | 37 |
| Figure 5.5. New CSV Visualization Workflow | 38 |
| Figure 5.6. Time Consumption Comparison between Traditional and New Workflows .. | 40 |
| Figure 5.7. New Workflow Time Consumption | 40 |
| Figure 5.8. Visualize 560640 Records with Different Number Processes, When Process Number Grows, Performance Goes Up and Goes Down | 41 |
| Figure 5.9. NetCDF Visualization Workflow..... | 42 |
| Figure 5.10. Line Chart Visualization Results Screenshot | 44 |
| Figure 5.11. Bar Chart Visualization Results | 44 |
| Figure 5.12. NetCDF File 2D Map Visualization..... | 48 |
| Figure 5.13. 2D Map Visualization Google Map Overlay..... | 49 |
| Figure 5.14. Frequency Distribution Histogram | 52 |
| Figure 5.15. Fish Model [Wu et al. 2015]..... | 54 |
| Figure 5.16. Fish Model Multi-runs..... | 55 |
| Figure 5.17. Data Management..... | 56 |
| Figure 6.1. Microsoft Excel Line Chart Visualization Example | 58 |
| Figure 6.2. Matlab 3D Visualization Example [Wu et al. 2014] | 58 |
| Figure 7.1. New Workflow Improvement..... | 63 |

Chapter 1 INTRODUCTION

Big data is a prevalent topic for both the academic and the industrial domains. There are many valuable opportunities and difficult problems brought by the digital era. Data is generated from different sources, such as scientific sensors and Facebook users. The data size can be super large. For example, there are 300 petabytes of data stored in Facebook warehouses and the daily incoming rate is around 600 terabytes [Facebook 2015]. Around 300 hours of video materials are uploaded to YouTube every minute [YouTube 2015]. There are precious research theories and commercial opportunities buried in the data. If we want to uncover these valuable treasures, we need good tools to handle big data.

The chosen tools should be able to manage and process big data. Because of the data size, most of the traditional popular tools are not capable of managing large volumes of data. There are many mature and popular Relational Database Management Systems (RDBMS), such as MySQL, PostgreSQL, and the Oracle Database. However, these RDBMSs are not suitable to large scale and high-concurrency applications [Han et al. 2011]. NoSQL database management systems, such as SciDB and MongoDB, are designed to manage big data. The most important reasons that we choose MongoDB instead of other databases are because the queries are faster than in relational databases and MongoDB can handle as many as 10 petabytes of data [MongoDB 2015].

Visualization is an effective method to bring vitality and luster to boring data. We do have some prevalent and mature visualization tools, such as Matlab [Azemi 1996], Mathematica [Savory 1995], GODIVA [Ma et al. 2004], Origin [Hao 2011], Mayavi

[Ramachandran and Varoquaux 2011], and R-software [Voulgaropoulou et al. 2012]. However, when the data size grows bigger, these tools may become slower and be less effective. Furthermore, for most of the traditional tools, users have to install some software in their computers to visualize their data. Also, users sometimes cannot interact with the visualization results. Our proposed system is named Environment for Large Data Processing and Visualization (ELDP&V) using MongoDB. The system uses stream data techniques to reduce lag and it is a web-based application, which means its users do not need to install any software on the client side. They only need to open a browser and explore our website. Users can also interact with the visualization results.

ELDP&V has three main parts: *data management*, *data processing*, and *data visualization*. For *data management*, we used a file system and database to manage all files. The file path in the file system is stored in the database. For *data processing*, ELDP&V has some basic methods (such as find maximum, minimum, and average value) and, because it is oriented on environmental data processing, it also has a fish model, which can predict fish density based on air temperature and precipitation. If users want us to add another model, they need to email us about the model's pseudocode. Then, we will add the model into our system. For *data visualization*, ELDP&V can visualize CSV and NetCDF files. Users can visualize the chosen file with line charts, bar charts, histograms, and 2D maps. Most of the visualization results are interactive. In the visualization part, we offer two visualization workflows. One is the traditional web-based client server data visualization workflow, the other is a new workflow proposed by us.

The thesis, in its remaining part, is organized as follows: Chapter 2 introduces the background of big data, data visualization, processing, and management; Chapter 3

presents ELDP&V's aim and characteristics; Chapter 4 describes the design of EDBP&V; Chapter 5 shows our system prototype and a video demo; Chapter 6 compares ELDP&V with several related systems; Chapter 7 introduces our future plan regarding ELDP&V; last, but not the least, Chapter 8 concludes the whole thesis with a summary of our contribution.

Chapter 2 BACKGROUND

This chapter introduces the background about big data, data visualization, data processing, and data management.

2.1 Big Data

2.1.1 Big Data Definition

Big data is a very popular and new topic. There are many definitions about big data. Here is a definition from the company SAS: Big data is a popular term used to describe the exponential growth, availability and use of information, both structured and unstructured [SAS 2015]. Here is another definition from IBM: Data, coming from everywhere; sensors used to gather climate information, posts to social media sites, digital pictures and videos, purchase transaction record, and cell phone GPS signal to name a few [IBM 2015].

Although, there are many big data definitions, big data has three known properties based on [Laney 2001]: growing volume; high input and output velocity; different varieties of data. People also call these three characteristics 3Vs (Volume, Velocity, and Variety). It is easy to tell the data volume is very large from the name “big data”. Because big data is always generated in real time, the input and output velocity is high. There are many data types, such as video, audio, text, and image. For each of these categories, it has different types too. For example, an image can be png, jpg, tif, gif and so on. As a result, variety is another big data property.

2.1.2 Big Data Challenges

Data is generated with an amazing speed: 90 per cent of all the data in our world was generated in the last two years [IBM 2015]. Big data brings us many valuable business opportunities and research topics. Also, big data is a severe challenge to traditional technology, software, and our lives.

Users complain about the performance of traditional Data Base Manage System (DBMS) to handle big data [Cudré-Mauroux et al. 2009]. Dr. Stonebraker and his cooperators designed and developed SciDB based on the requirements from different scientists. They used “array model” instead of “table model” [Stonebraker et al. 2009]. “Table model” is widely used in traditional DBMS. SciDB performs better than traditional DBMS on big data management based on the experiments in [Taft et al. 2014].

Some of the traditional data processing and visualization methods are not suitable for big data. It is unreasonable to try to handle all of a big file at once. ELDP&V extracts data from the chosen big file and it separates data into small slices. ELDP&V processes and visualizes a slice each time and at the same time ELDP&V starts the preparatory work for the next slice of data.

2.2 Interactive 2D Visualization

Our thesis only focuses on 2D visualization because our goal is to create a web-based application and most of the 3D models loading and visualization tools (such as WebGL [Tavares 2012]) and libraries are not mature. Here is a blog about the shortcomings of WebGL [Janne 2014]. Therefore, it is more practical to implement a 2D visualization tool than a 3D visualization tool.

2.2.1 Interactive 2D Visualization Definition

Visualization is an important and necessary method to find rules buried in data. It is also a straight method to display the data meaning. As [Heer et al. 2005] mentioned, it has been proven that visualization is an indispensable method to make complex data easier to understand.

“Interactive 2D Visualization” is a self-explanatory phrase. This means methods to visualize data with 2D graphs (such as line chart, bar chart, pie chart) and users can interact with the 2D outputs, such as zooming in, zooming out, and displaying detailed information when users click on some parts of the 2D graphs. In ELDP&V, most of the visualization results are interactive.

There are many effective and prevalent 2D visualization methods. We can use line charts to find the trend of data and also use bar charts to compare different parts of data. We can combine different 2D visualization methods to be more effective. Figure 2.1 displays a 2D visualization example with both line chart and bar chart. Users can observe the data trend and compare different datasets with the same graph. However, it does not mean the more combinations we have, the more effective the graph will be. Users may get lost in a very complex graph.

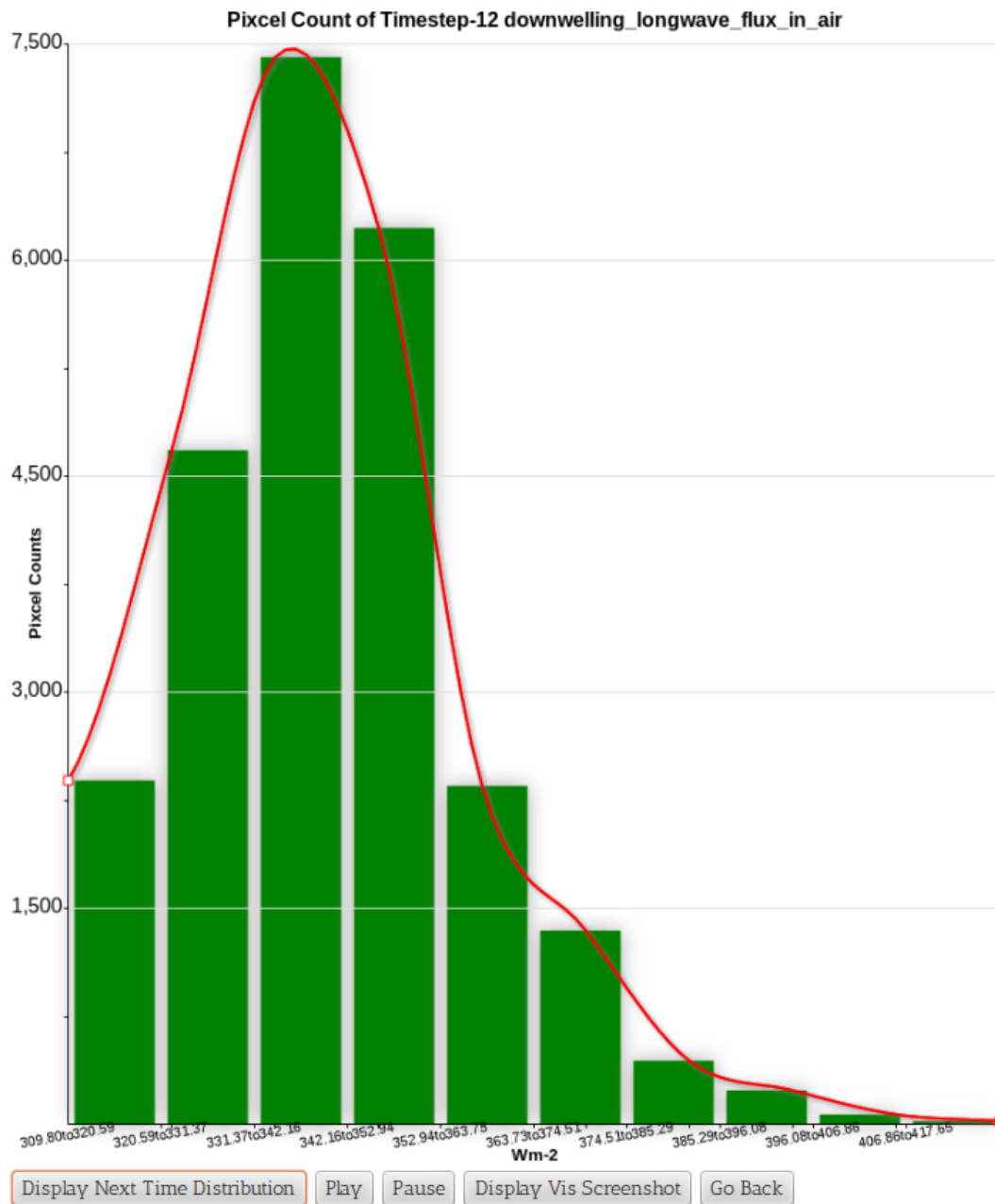


Figure 2.1. Line & Bar Graph

2.2.2 Prevalent Libraries

There are three prevalent libraries: D3, Dygraphs, and RGraph. D3 is a JavaScript library [D3 2015]. It enables users to manipulate or interpolate html elements in a very

easy and efficient way. For example, the following two parts of code turn a paragraph text color into green:

HTML:

```
var x = document.getElementById("myParagraph");  
x.style.setProperty("color", "green", null);
```

D3:

```
d3.select("#myParagraph").style("color", "green");
```

It is clear that D3 library is more efficient. Moreover, D3 is also good at visualizing data. There are a lot of beautiful visualization examples in [D3 2015]. D3 will not limit users' imagination, it offers programmers many powerful interfaces that can help programmers create their own visualization methods. This is an advantage, but also a disadvantage, because users need to know the D3 library very well to use it as their wishes.

Dygraphs is a JavaScript library focus data line chart visualization [Dygraphs 2015]. Users can visualize millions of records without feeling lagged and it is compatible with most of the browsers. The visualization results are interactive. Users can zoom in or zoom out of the line charts. ELDP&V uses this library mainly for line chart visualization and also for enabling users to download the chosen part of the data from the line chart.

RGraph is another very powerful JavaScript data visualization library [RGraph 2015]. It supports around 60 different kinds of data visualization methods. The library is easy to use and there are many examples in [RGraph 2015]. Users can customize the library too. For example, it is not hard to customize the library to react to users' clicks on a bar chart. ELDP&V mainly uses RGraph to visualize data with bar charts.

2.2.3 Traditional Workflow vs New Workflow

Most traditional web-based client/server visualization applications transfer data from the server side to the client side and use tools or JavaScript libraries to visualize data with different charts. As [Lee et al. 2014] mentioned, most prevalent visualization tools and libraries can only be run on the client side, such as Google Chart, Open Flash Chart (OFC), Adobe Flex, and D3.js. Also, only parts of the popular tools and libraries can process large datasets. Generally, the performance will drop dramatically when there are more than 200,000 data points. For Adobe Flex, the visualization results stop responding occasionally when there are more than 50,000 records. Furthermore, if users want to visualize larger datasets, they need more memory spaces, because most these tools and libraries put data into client side's machine memory spaces. The worst part is that we need to transfer all the data to the client side if we use this traditional workflow. If the datasets is very large or the network speed is very low, the data transfer time can be unbearable. Therefore, if we want to visualize real big data, a new method to visualize real datasets is badly needed.

In this thesis, we have proposed a new workflow to visualize large data for a web-based client/server application. The basic idea is to move as much work as possible to the server side. We visualize data in the server side with multi-processes and only transfer visualization results (images) to the client side. This is different from most of the prevalent tools and libraries workflow introduced in [Lee et al. 2014] that transfer data to client side and visualize in client side. Figure 2.2 compares the traditional and new client/server web application data visualization workflows. The main difference between

these two workflows is that the new workflow only transfers visualization result images, which can be around 10MB for 1TB data. The size of our visualization images are decided by the image resolutions. It is much faster than transferring all the data to the client side. For the new workflow, the client side will collect users' actions and transfer the action information to the server side. The server side will update the visualization results based on the action information. In this thesis, we used multi-core to visualize data. There are some limitations. For example, this workflow will not work if the data size is bigger than the server memory size, because we need to allocate memory for the chosen data. To solve this problem, we plan to use distributed system to improve ELDP&V performance in the future.

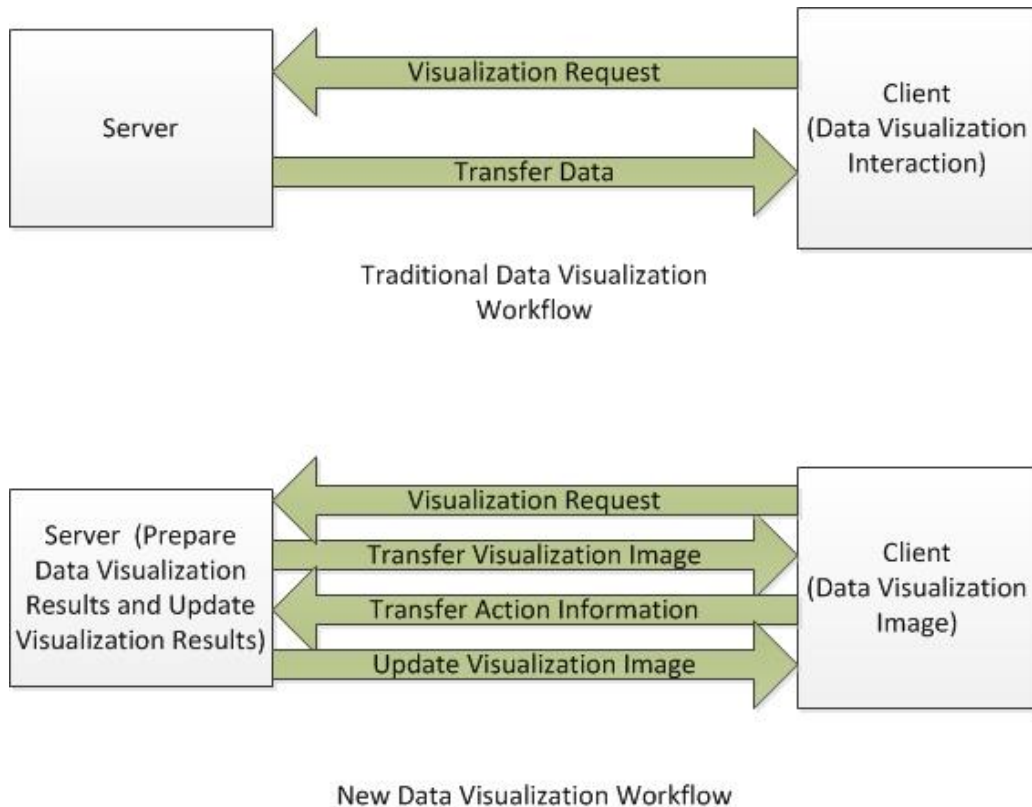


Figure 2.2. Comparisons between Traditional and New Client/Server Web Application Data Visualization Workflow

2.3 Data Processing

2.3.1 Basic Data Processing

In this thesis we define all the data processing methods that do not use professional models (such as climate model) as basic data processing methods. For example, ELDP&V obtains the maximum, minimum, and average values of a file. This definition means basic data processing methods can be complex too. For example, ELDP&V extracts a three dimensional temperature array from a NetCDF file. The system visualizes all the values frequency distribution with line charts and bar charts. In this example, the system needs to flatten, sort the array, and then record the number of different sections. The process will take a longer time for larger size datasets for normal methods. We implemented an effective method that reduces the negative impact of the data size, which is introduced in Chapter Five.

2.3.2 Models

Scientific models are widely used in solving problems in different research fields. Scientists use different models to represent objects, phenomena, or processes [Box and Draper 1987]. The common example is that climate scientists predict future weather with climate models. It is not strange that some weather forecasts are wrong; there is a famous saying that no models are accurate, but some of them are useful.

ELDP&V is designed to process data with models. For example, there is a fish model in our system now. Users can predict fish population density of a river with precipitation, air temperature, and elevation.

We believe that models are similar to functions in a program. We do not need to know how the function works in details to get results from the problem. It sounds easy, however, it is a big challenge for model creators and programmers who need to convert the models into programs.

2.4 Database

If we want to build a good web-based client/server application for processing big data, we need to choose a qualified database management system. There are two kinds of databases—RDBMS and NoSQL database. After Chapter 2.4.1 and Chapter 2.4.2 discussion, it will be clear why we choose NoSQL database.

2.4.1 Traditional Relational Database Management System

There are several mature and prevalent RDBMSs, such as MySQL, Oracle, and Postgres. These RDBMSs are created based on the relation model in [Codd 1972]. Structured Query Language (SQL) is used in RDBMS to manage data stored in the database and to process the data stream [Microsoft SQL 2015].

The most widely used data object in RDBMS is table. Figure 2.3 displays a table example stored in RDBMS. Each row of the table is a record stored in the database. Each column contains the information of all entities in certain field. For example, in Figure 2.3, “Name” column contains all the entities’ names.

| ID | NAME | AGE | ADDRESS | SALARY |
|----|----------|-----|-----------|----------|
| 1 | Ramesh | 32 | Ahmedabad | 2000.00 |
| 2 | Khilan | 25 | Delhi | 1500.00 |
| 3 | kaushik | 23 | Kota | 2000.00 |
| 4 | Chaitali | 25 | Mumbai | 6500.00 |
| 5 | Hardik | 27 | Bhopal | 8500.00 |
| 6 | Komal | 22 | MP | 4500.00 |
| 7 | Muffy | 24 | Indore | 10000.00 |

Figure 2.3. Table Objects Example in RDBMS [SQL RDBMS Concept 2015]

From the example, we can see that RDBMS is good at managing structured data. Also, RDBMS is mature and has many great relative tools. We believe it is better to choose RDBMS than NoSQL to manage data if users do not have a large size of data and do not need to manage their data across several sites.

RDBMS users can meet these problems [Stonebraker 2010]: 1) it is difficult to manage data across different sites; 2) it is hard to design the schema for data if the data does not have a rigid relational schema; 3) some enterprise RDBMSs are able to manage big data, but users need to pay a large amount of money for their licenses.

2.4.2 NoSQL Database

RDBMSs are great and mature. However, they are not suitable for large scale and high-concurrency applications [Han et al. 2011]. NoSQL database management systems are designed to manage big data. NoSQL databases can be run easily in distributed systems and it does not require the stored data has rigid relational schema.

NoSQL means “not only SQL”. Researchers and businessmen wanted to solve the

limitations of RDBMS. Therefore, they created NoSQL database in the early twenty-first century [Leavitt 2010]. The image below is from [MongoDB 2015]. It compares the difference between SQL and NoSQL Databases in detail.

| | SQL Databases | NOSQL Databases |
|------------------------------|--|--|
| Types | One type (SQL database) with minor variations | Many different types including key-value stores, document databases, wide-column stores, and graph databases |
| Development History | Developed in 1970s to deal with first wave of data storage applications | Developed in 2000s to deal with limitations of SQL databases, particularly concerning scale, replication and unstructured data storage |
| Examples | MySQL, Postgres, Oracle Database | MongoDB, Cassandra, HBase, Neo4j |
| Data Storage Model | Individual records (e.g., "employees") are stored as rows in tables, with each column storing a specific piece of data about that record (e.g., "manager," "date hired," etc.), much like a spreadsheet. Separate data types are stored in separate tables, and then joined together when more complex queries are executed. For example, "offices" might be stored in one table, and "employees" in another. When a user wants to find the work address of an employee, the database engine joins the "employee" and "office" tables together to get all the information necessary. | Varies based on database type. For example, key-value stores function similarly to SQL databases, but have only two columns ("key" and "value"), with more complex information sometimes stored within the "value" columns. Document databases do away with the table-and-row model altogether, storing all relevant data together in single "document" in JSON, XML, or another format, which can nest values hierarchically. |
| Schemas | Structure and data types are fixed in advance. To store information about a new data item, the entire database must be altered, during which time the database must be taken offline. | Typically dynamic. Records can add new information on the fly, and unlike SQL table rows, dissimilar data can be stored together as necessary. For some databases (e.g., wide-column stores), it is somewhat more challenging to add new fields dynamically. |
| Scaling | Vertically, meaning a single server must be made increasingly powerful in order to deal with increased demand. It is possible to spread SQL databases over many servers, but significant additional engineering is generally required. | Horizontally, meaning that to add capacity, a database administrator can simply add more commodity servers or cloud instances. The database automatically spreads data across servers as necessary. |
| Development Model | Mix of open-source (e.g., Postgres, MySQL) and closed source (e.g., Oracle Database) | Open-source |
| Supports Transactions | Yes, updates can be configured to complete entirely or not at all | In certain circumstances and at certain levels (e.g., document level vs. database level) |
| Data Manipulation | Specific language using Select, Insert, and Update statements, e.g. SELECT fields FROM table WHERE... | Through object-oriented APIs |
| Consistency | Can be configured for strong consistency | Depends on product. Some provide strong consistency (e.g., MongoDB) whereas others offer eventual consistency (e.g., Cassandra) |

Figure 2.4. Comparison between SQL and NoSQL Database [MongoDB 2015]

There are many famous NoSQL databases, such as SciDB and MongoDB. SciDB used array models to replace table models in RDBMS and is designed for scientific data [Cudré-Mauroux et al. 2009]. SciDB is good but not perfect, and it is not mature compared to other NoSQL databases such as MongoDB. MongoDB is designed for distributed system and can manage large size data. It is widely used, with some famous users, such as Expedia, City of Chicago, and MetLife [MongoDB Users 2015]. There are also some relative libraries, such as pymongo. Also, MongoDB supports many popular frameworks. For example, users can create their bank-end with Django with MongoDB or Flask with MongoDB.

Chapter 3 ELDP&V GOALs and CHARACTERISTICS

This chapter explains the main goals and characteristics of ELDP&V. We explain the reasons that we designed these characteristics for ELDP&V. We also offer two scenarios to explain how ELDP&V helps users visualize data.

3.1 ELDP&V Goals

ELDP&V is used to help users visualize, process, and manage large datasets. ELDP&V only focus on CSV and NetCDF files. These files can be generated by models or users. Users can upload a file or choose a file stored in the ELDP&V database.

ELDP&V visualizes CSV files with line charts. Users can zoom in or zoom out of the chosen part. ELDP&V enables users to download the chosen part of the data and take a screenshot of the visualization results.

ELDP&V visualizes NetCDF files with line charts, bar charts, and 2D maps. The line chart visualization part is similar to CSV files line chart visualization part. The maps part enables users to customize how they project data onto a map. This means that users need to choose which attributes of the chosen variable are longitude and latitude. Based on the user's choice, the system will convert values into colors and display them onto a map. If the chosen attribute is time serial data, users can play the data frame by frame, similar to a video. More details will be introduced in Chapter 5.

ELDP&V aims to handle big data. We used MongoDB [MongoDB 2015] to store data. We visualized data with D3 library [D3 2015], Dygraph library [Dygraphs 2015], and RGraph [RGraph 2015]. These database and libraries can handle big data. Our system can visualize more than 2000 records per second, including loading data, libraries, and plotting data.

3.2 ELDP&V Characteristics

ELDP&V is a web-based client/server application. This means users only need to open a browser and visit our website. There is no need to install any other software, which is really convenient. However, we need to transfer the visualization data from the server to the client. This may slow down the speed of the application, if data files are really large.

The most distinct characteristic of ELDP&V is that all the visualization results are interactive. For the line charts part, users can zoom in and zoom out along the x-axis and y-axis with the help of Dygraph library [Dygraphs 2015]. We interviewed some model scientists and they said sometimes they are only interested in parts of the data. Therefore, we added a function into our application: users can choose a section from the visualization results and then download chosen parts of the data. ELDP&V also enables users to take a screenshot of the visualization results.


Our application can process big data because we chose libraries that can handle big data and that were programed with consideration for big data problems. We used MongoDB to store data. MongoDB can handle petabyte size data [Big Data Explained 2015]. We used Dygraph library to plot line charts, which can plot millions of points in a

short time [Dygraphs 2015].

3.3 Scenarios

We believe it is easier for others to understand how ELDP&V works with examples of possible scenarios. Therefore, we have listed two scenarios below based on the persona in Table 3.I:

Table 3.I. Fish Hydrologist Persona

| | | | |
|-----------------|--|---|--|
| Persona | Environmental Scientist Focusing the Relationships Between Fish and Climate | | |
| Photo | |  | |
| Fictional Name | David Smith | | |
| Job Title | Professor, UNR | | |
| Goals and Tasks | Her main research field is fish propagation and water environment. She is very interested in fish population prediction models and the relationship between climate and water environment. | | |

First scenario: Elisabeth wants to figure out the changing rules of the fish population density of a river for the last 30 years. Elisabeth has the model inputs (a CSV file) of a fish model and she chooses to visualize them using ELDP&V. ELDP&V visualizes all four columns--time, water-temperature, precipitation, and fish population--of the uploaded CSV files in a line chart (time as x-axis and others as y-axes). There are

three checkboxes below the line chart--water-temperature checkbox, precipitation checkbox, and fish population checkbox. Elisabeth only wants to figure out the relationship between the fish population and water temperature. Therefore, Elisabeth unchecks the precipitation checkbox and the line chart only contains fish population and water temperature. Elisabeth finds that the fish population changes in a strange way from time A to time B, so she zooms in from Time A to Time B, takes a screenshot of this part by clicking the “Screenshot” button, and extracts this part of data by clicking the “Download” button.

Second scenario: Elisabeth wants to understand the trends of air temperature in Nevada. Elisabeth finds that there are air temperature records from 2000 to 2015 as a NetCDF file stored in the ELDP&V database. Elisabeth chooses the file and follows the instructions offered by ELDP&V: 1) choose the temperature from the variable list from the NetCDF to visualize; 2) choose the map visualization method; 3) choose lat and lon from the attribute list to be latitude and longitude. ELDP&V displays the air temperature value as a color in a map (the bigger number with the darker color). Elisabeth click on the map. ELDP&V finds out the latitude and longitude of the point, then the system displays a line chart of air temperature from 2000 to 2015.

Chapter 4 ELDP&V DESIGN

This chapter provides details on how we designed ELDP&V. It includes the requirements specification of our system, system-level design, examples of low-level design, and glossary of our system.

4.1 Requirements Specification

Table 4.I includes the functional requirements of ELDP&V.

Table 4.I. Functional Requirements of ELDP&V

| Functional Requirements | |
|-------------------------|---|
| R1 | ELDP&V application shall allow users to upload files. |
| R2 | ELDP&V application shall allow users to access files in the application database. |
| R3 | ELDP&V application shall allow users to visualize CSV files with line chart. |
| R4 | ELDP&V application shall allow users to visualize NetCDF files with line chart. |
| R5 | ELDP&V application shall allow users to visualize NetCDF files with map. |
| R6 | ELDP&V application shall allow users to choose which variable of NetCDF file to visualize. |
| R7 | ELDP&V application shall allow users to choose which attribute of the chosen variable to visualize with line chart. |
| R8 | ELDP&V application shall pop up a warning message when users do not choose reasonable attributes to visualize. |
| R9 | ELDP&V application shall allow users to specify other attributes except the chosen attributes for line chart visualization. |
| R10 | ELDP&V application shall allow users to take a screenshot of the visualization result. |
| R11 | ELDP&V application shall allow users to zoom in the chosen part of the line chart visualization results. |
| R12 | ELDP&V application shall allow users to download the chosen part of data as a CSV file. |

| | |
|-----|---|
| R13 | ELDP&V application shall allow users to zoom out by double clicking the line chart. |
| R14 | ELDP&V application shall allow users to start another visualization by clicking "GoBack" button. |
| R15 | ELDP&V application shall allow users to view map as a movie (each frame is based on the timestamp). |
| R16 | ELDP&V application shall allow users to view the help manual, if they have questions about how to use this application. |
| R17 | ELDP&V application shall allow users to process files with some basic methods. |
| R18 | ELDP&V application shall allow users to process files with models. |
| R19 | ELDP&V application shall allow users to email us feedbacks. |

Here are some examples about the Functional Requirements:

If users choose to visualize variable "temperature", but users do not choose the attribute "temperature" to visualize, the system will pop up a warning message as described in R8 of Table 4.I. If users choose to visualize variable "temperature" then users need to select two attributes of the chosen variable for line chart visualization. If users choose the same attributes (e.g. lon), the application will also pop up a warning message as described in R8 of Table 4.I.

For R19, if users cannot find suitable models, they can email us about their required model. Users can send us pseudocode, python code, or a python library of the model. We will add the model into our system as soon as possible.

Table 4.II contains ELDP&V non-functional requirements. We used the Flask framework [Flask 2015] to build ELDP&V backend, which is a python framework. Therefore, all the backend codes should be written in python. There are some great JavaScript libraries for data visualization, such as D3, RGraph, Google Map, and so on. With the help of these libraries, we can save a lot of time when implementing ELDP&V. To use these libraries, we chose to develop our system with JavaScript and HTML.

Table 4.II. Non-Functional Requirements

| Non-Functional Requirements | |
|-----------------------------|--|
| NFR1 | The back end of ELDP&V application shall be written in python. |
| NFR2 | The front end of ELDP&V application shall be written in HTML and JavaScript. |
| NFR3 | The line chart visualization part of ELDP&V application shall use D3 library and Dygraph library. |
| NFR4 | The 2D map visualization part of ELDP&V application shall use Google map API. |
| NFR5 | ELDP&V application shall visualize more than 2000 records per second including loading data and different libraries. |
| NFR6 | ELDP&V application shall overlay the 2D map visualization results on a Google map in the correct position. |
| NFR7 | ELDP&V application shall react quickly to users' interactions. |

4.2 Use Case Diagram

Figure 4.1 displays ELDP&V use case diagram. It is very useful to introduce ELDP&V from the users' view. The actor on the left side of Figure 4.1 represents our system users. The big rectangle on the right side of Figure 4.1 represents ELDP&V. All the ellipses in the rectangle are the functions offered to users by ELDP&V.

Upload files: Users can upload their files to the server side and then visualize or process the uploaded file. The ELDP&V server will store and manage the uploaded file.

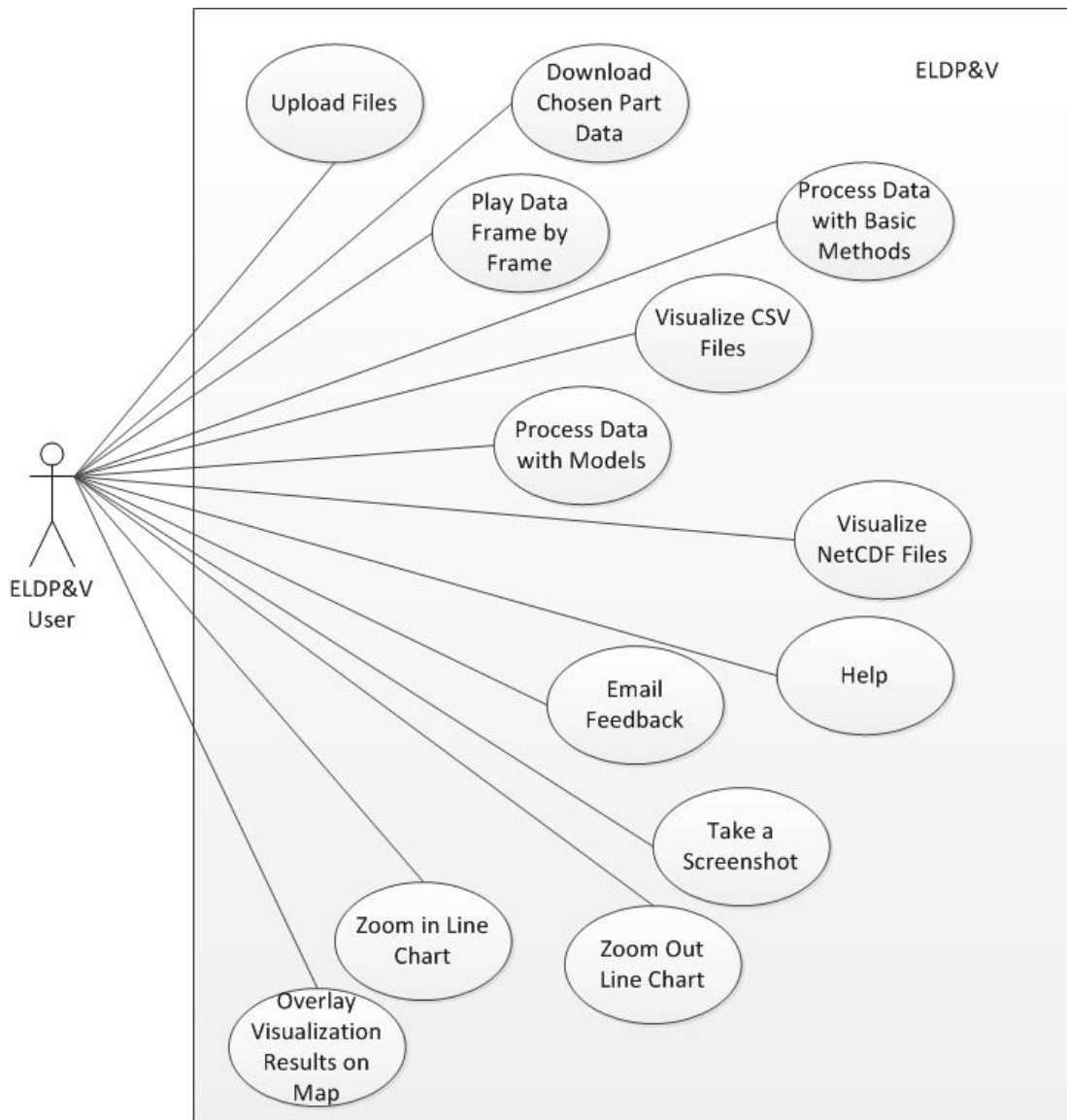


Figure 4.1. ELDP&V Use Case Diagram

Download chosen part data: Users can extract data from the chosen file. They need to choose a part of the line chart visualization results by clicking and holding the left mouse button, dragging along x-axis, and then releasing the left mouse button. After clicking the “downloadCSV” button, the system will offer a CSV file containing the chosen part data for users to download.

Play data frame by frame: If the chosen NetCDF variables have timestamp attribute, the system will allow users to update the visualization results frame-by-frame based on the chosen variable timestamp.

Process data with basic methods: Users can process the data with some basic methods, such as “Frequency Distribution Histograms”. Users can tell the frequency distribution of the chosen data. Some of our collaborators use this method to tell if the collected data is abnormal.

Visualize CSV files: Users can visualize CSV file with two workflows: the traditional workflow is good for small CSV files and has more interaction methods; the new workflow is fast with large CSV files but it has less interaction methods for the current version ELDP&V.

Process data with models: Users can process the chosen data with models. The current version ELDP&V only has a fish population density model.

Visualize NetCDF files: Users can visualize NetCDF files in a few simple steps. We tried our best to simplify the process and also add many instructions for the users in our website. Users can visualize NetCDF files with line charts, bar charts, and 2D maps.

Help: If users are confused about what to do or how to do something, they can find frequently asked questions and answers from the help page of our website.

Email feedback: If users have any suggestions or questions that can send us feedbacks from the contact page of our website. The system will send an email containing users’ suggestions and questions to our maintainers.

Take a screenshot: Users can take screenshot of the visualization results by clicking “Taking a screenshot” button.

Zoom in line chart: Users can zoom in on a line chart by clicking the left mouse button, dragging along x-axis, and then releasing the left mouse button.

Zoom out line chart: Users can zoom out on a line chart by double clicking the line chart.

Overlay visualization results on map: Users can visualize a variable of a NetCDF file with a 2D map. Then ELDP&V will overlay the 2D map on a Google map based on the location information of the chosen variable (latitudes and longitudes).

4.3 Use Cases

ELDP&V contains many use cases. In this section, we will only introduce three of them in detail. These use cases are “Upload Files”, “Zoom in Line Chart”, and “Overlay Visualization Results on Map”.

Table 4.III contains the details about how ELDP&V reacts when users upload their files.

Table 4.III. Upload Files Use Case

| | |
|------------------|---|
| Use Case: | Upload Files |
| Actors: | EBDP&V User |
| Preconditions | 1. EBDP&V is running. 2. EBDP&V user has logged in. |
| Flow of Events: | 1. The user clicks 'Upload' button. 2. EBDP&V pops up a window that enable the user to choose a file to upload. 3. The file format is supported and the file is uploaded to the server. |
| Postconditions: | A file is uploaded to the EBDP&V server. |
| Alternative Flow | 3 Users upload a unsupported file format, EBDP&V will lead users to an error message page. |

The second use case is about how users can zooms in on a line chart, which is displayed in Table 4.IV.

Table 4.IV. Zoom in Line Chart Use Case

| | |
|------------------|---|
| Use Case: | Zoom in Line Chart |
| Actors: | EBDP&V User |
| Preconditions | <ol style="list-style-type: none"> 1. EBDP&V is running. 2. EBDP&V user has logged in. 3. The user has chosen a file to visualize with line chart. |
| Flow of Events: | <ol style="list-style-type: none"> 1. EBDP&V obtains data from the chosen file. 2. EBDP&V displays visualization results with line chart. 3. The user clicks the mouse left button and move the cursor along x-axis. 4. EBDP&V zooms in the chosen part along x-axis. |
| Postconditions: | The chosen part of line chart visualization results is zoomed in. |
| Alternative Flow | <ol style="list-style-type: none"> 3. The user clicks the mouse left button and move the cursor along y-axis. 4. EBDP&V zooms in the chosen part along y-axis. |

The third use case is about how users can overlay visualization results on a google map. The use case also includes the situation that the chosen variable is time serial. If so, EBDP&V will enable users to update the 2D map, frame by frame, based on the timestamp.

Table 4.V. Overlay Visualization Results on Map

| | |
|---------------|---|
| Use Case: | Overlay Visualization Results on Map |
| Actors: | EBDP&V User |
| Preconditions | <ol style="list-style-type: none"> 1. EBDP&V is running. 2. EBDP&V user has logged in. 3. The user has chosen a file to overlay the visualization results on a 2D map. |

| | |
|------------------|---|
| Flow of Events: | <ol style="list-style-type: none"> 1. EBDP&V obtains data from the chosen file. 2. EBDP&V displays visualization results in a canvas. 3. EBDP&V overlays the canvas on a 2D map. |
| Postconditions: | The visualizarion results are overlayed in a 2D map. |
| Alternative Flow | <ol style="list-style-type: none"> 4. If the chosen variable is time sequential, EBDP&V will offer users a 'play' button. 5. The user clicks 'play button', the map will be updated frame by frame. |

4.4 Flowchart

This section introduces ELDP&V with a flowchart so there is a clear understanding of the system logic. It is easier to analyze ELDP&V system with the flowchart. More details are shown in Figure 4.2.

Figure 4.2 mainly consists two parts: data visualization and processing. ELDP&V is an event-driven system, which means different events will trigger different callback functions of the system [Kelton and Law 2000].

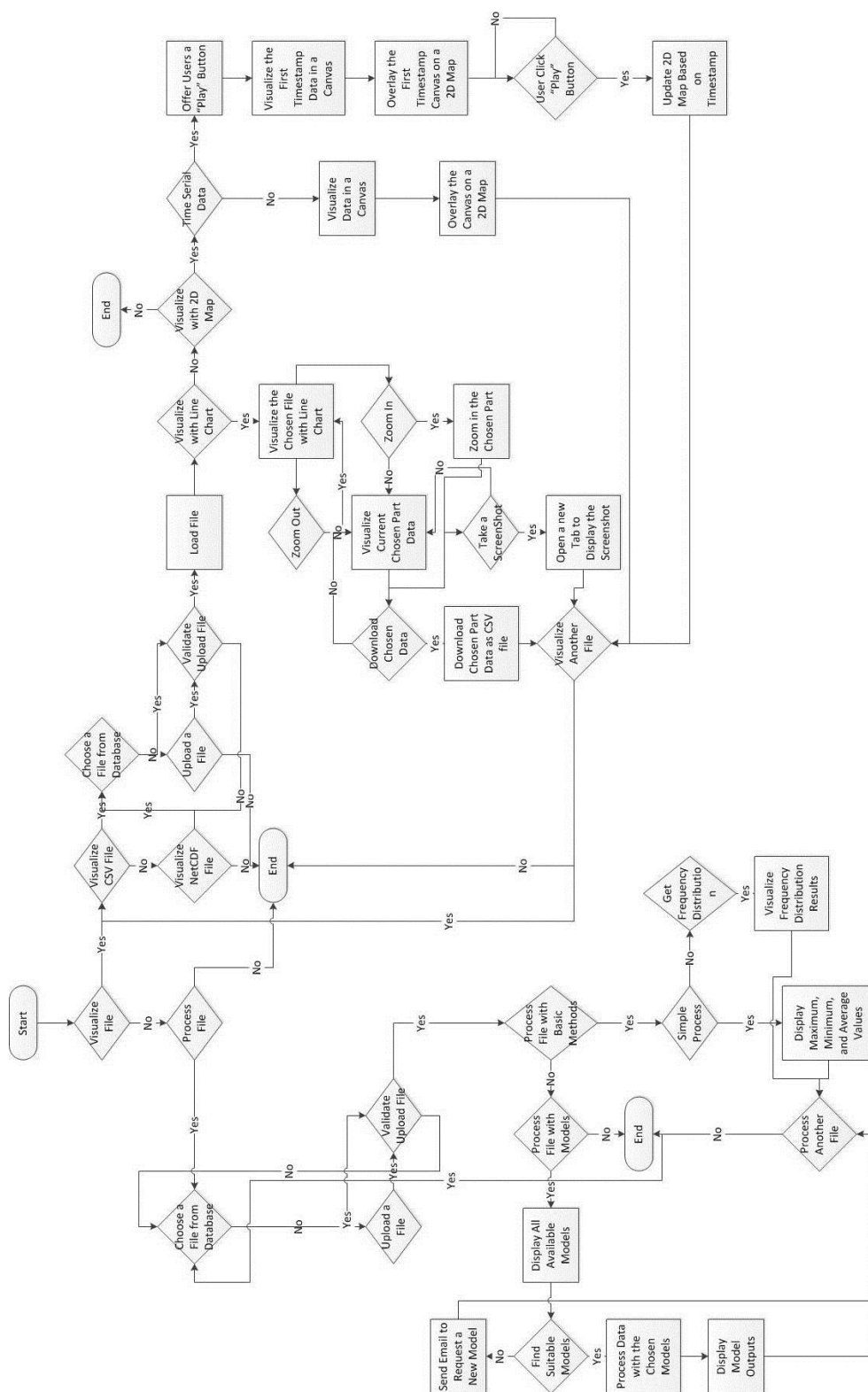


Figure 4.2. ELDP&V Flowchart

4.5 Glossary

This section introduces different terminologies used in this thesis.

Table 4.VI. ELDP&V Glossary

| | |
|-------------------|--|
| NetCDF | NetCDF is short for network common data form, which is designed to create, share and access array oriented data. NetCDF Classic is part of international standard of the Open Geospatial Consortium [OGC 2015]. |
| CSV | CSV is short for comma separated values. It is a simple format for representing a rectangular array (matrix) of numeric and textual values [CSV 2015]. |
| D3 | D3 is short for data-driven documents. This is a JavaScript library which is used to visualize and interact data with fancy methods [D3 2015]. |
| Dygraphs | Dygraphs is a fast, flexible open source Javascript charting library. It allows users to explore and interpret dense data sets [Dygraphs 2015]. |
| Big Data | Big data is a popular term used to describe the exponential growth, availability, and use of information, both structured and unstructured [SAS 2015]. |
| RDBMS | RDBMs is short for relational database management system. RDBMs is based on the relational model created by Dr. Codd [Codd 1972]. Most of the databases are RDBMs, such as Oracle Database and Microsoft SQL Server. |
| No-SQL database | NoSQL means “not only SQL”. No-SQL databases are designed to solve scalability and big data challenges. |
| Dataset discovery | This means the procedure to find datasets. For example, users import datasets from certain repositories or create their own datasets and import them into a database. |
| R | R is a programming language and software environment mainly for statistical computing and graphics. R is very popular with statisticians and data miners for developing statistical software [Fox and Andersen 2005]. |
| NCCP | NCCP [Dascalu et al. 2014] is short for Nevada Climate Change Portal. We can access it from http://sensor.nevada.edu/ . There are a lot of useful climate datasets. The website also provide some tools like VISTED to visualize data. |

| | |
|-------|--|
| JSON | JSON is short for JavaScript Object Notation, which is a kind of data format. JSON is mainly used for data transition between a server and web-application. |
| SciDB | SciDB is a new generation database management system. It is developed by Paradigm4 [Paradigm4 2015], cofounded by Michael Stonebraker. The main characteristic of SciDB is that it uses array to store all the data. |

Chapter 5 ELDP&V PROTOTYPE

This chapter has two parts. The first part presents the main components of EDBP&V. Users can visualize their CSV and NetCDF files directly with EDBP&V. The visualization methods include line charts, bar charts, and 2D maps. The second part explains how our system process data. This part contains basic methods, such as obtaining maximum, minimum, and average values. It also includes some complex methods, such as the display frequency distribution of the chosen dataset, and how to process the chosen data with a given model.

5.1 ELDP&V Main Components

We introduce three main components of ELDP&V: data visualization, processing, and management. Data visualization contains CSV file visualization and NetCDF file visualization. Data processing includes basic data processing methods and models. Data management introduces how ELDP&V manages data.

5.1.1 Data Visualization

5.1.1.1 CSV Visualization

CSV files is widely used in different research fields. There are two workflows in ELDP&V for CSV files visualization: the traditional workflow and the new workflow.

The traditional workflow transfers all the data to the frontend and visualizes the data in the client side. Every column of the CSV files will be visualized in a line chart.

Users can remove or display each column by checking or unchecking the corresponding checkboxes. Figure 5.1 displays the traditional CSV visualization flowchart of ELDP&V.

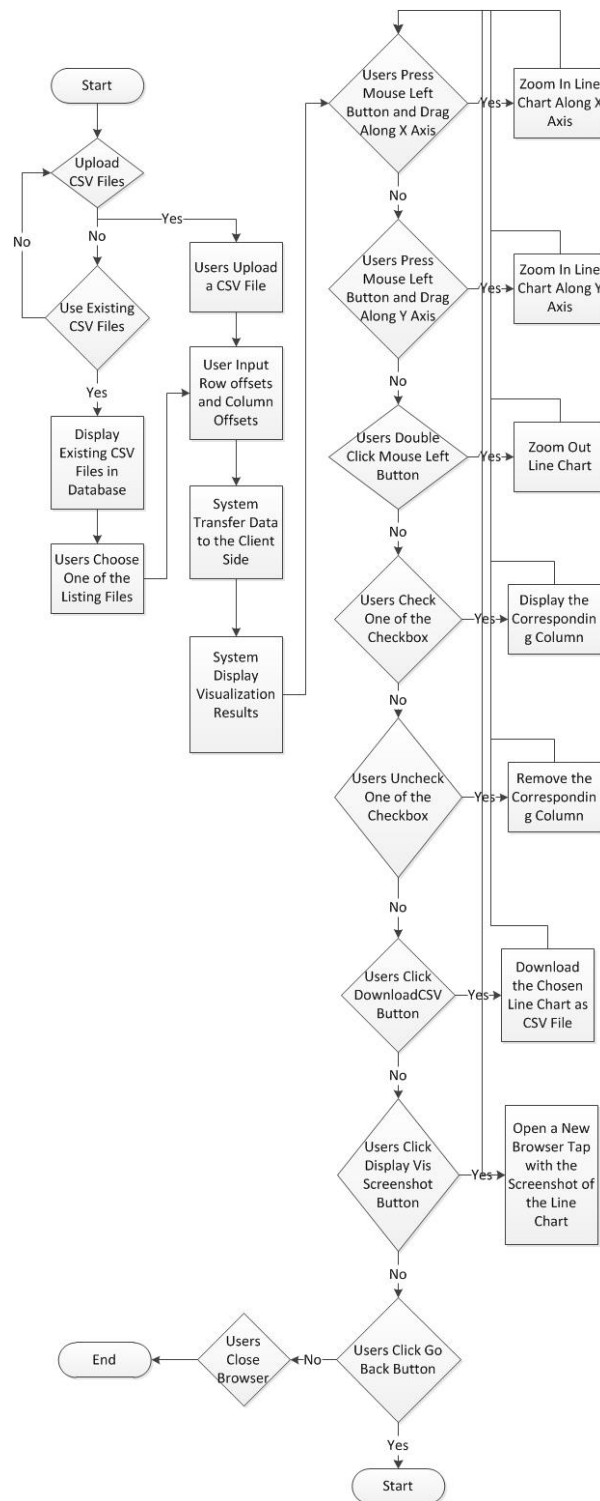


Figure 5.1. Traditional CSV Visualization Workflow

The system offers one method for CSV file visualization--line chart visualization in this version. Figure 5.2 displays a screenshot of CSV file line chart visualization. There are some checkboxes below the line chart part. Each checkbox represents a column in the CSV file. When users check one of the checkboxes, the corresponding column will be visualized in the line chart. When users uncheck one the checkboxes, the corresponding line chart part will be removed. When the cursor hovers over the line chart, the labels of the cursor part data will be shown in the top right corner of the line chart. Users can zoom in on the line chart along x-axis by pressing the mouse left button and dragging the cursor along x-axis. Users can also zoom in along the y-axis by pressing the mouse left button and dragging the cursor along the y-axis. If users want to zoom out, they only need to double click on the line chart. When users click the 'downloadCSV' button, the system will prepare the chosen part dataset (zoomed in part) and offer users a CSV file to download. We do it in this way instead of offer the whole CSV file to download, because some of the cooperators told us that they always receive a big file from sensors. They want to extract data from the big file and do further research on that part of data. When users click 'Display Vis Screenshot', the system will open a new window with the screenshot of the line chart. Users can save the screenshot by right-clicking the screenshot, choosing 'Save image as ...' to save it.

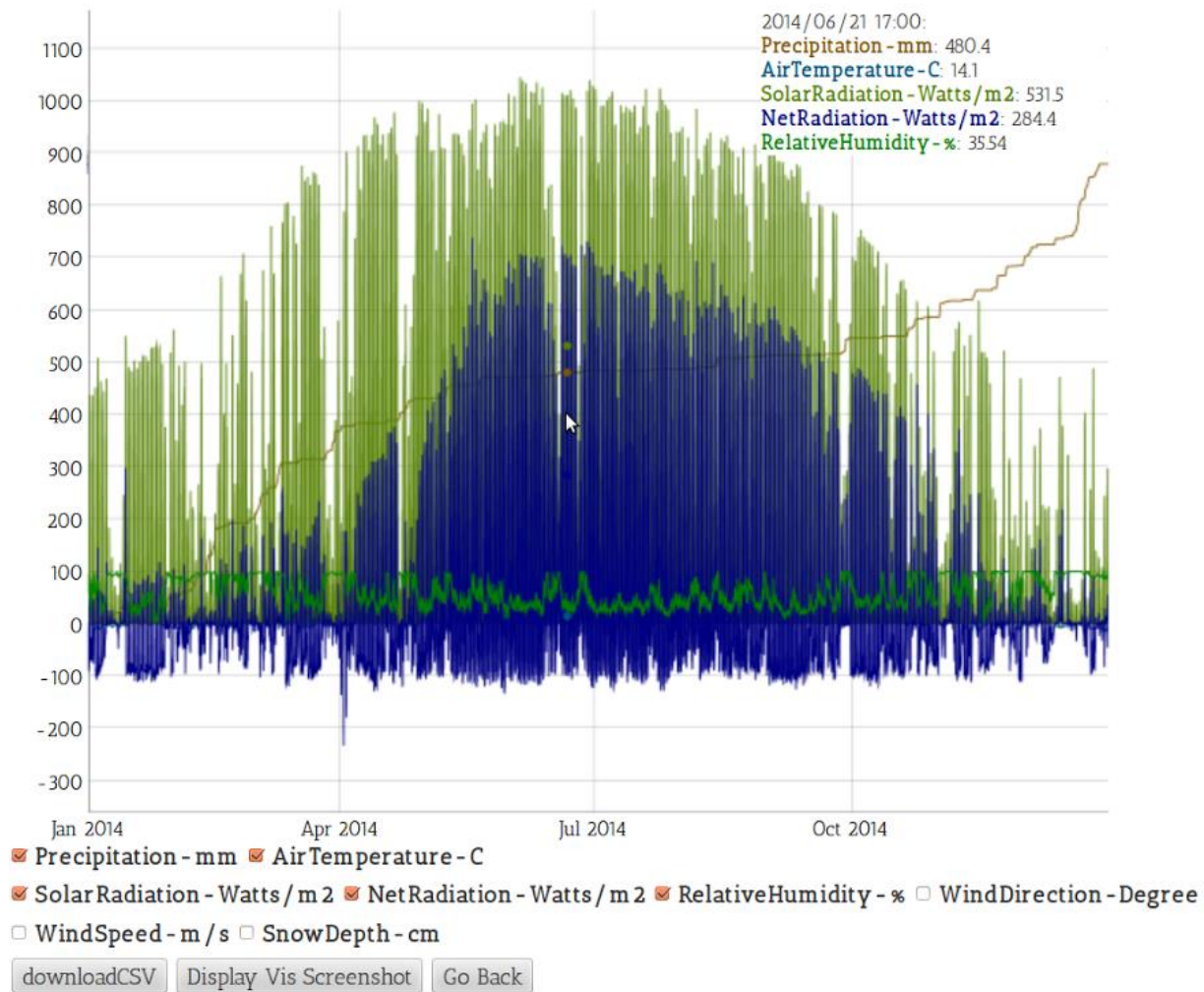


Figure 5.2. Screenshot of CSV Line Chart Visualization

This part is mainly created with Dygraph.js. We extracted data from the uploaded CSV file and transfer a string containing all data information and metadata to the frontend to visualize. The challenging part about this part is how to download the chosen part data as a CSV file. The basic idea is that we used the Dygraph API to locate the start and end point of the data. Then the system will generate a new CSV file for users based on the start and end point.

The new CSV visualization workflow is designed by us. We moved most procedures to the server side. The system will separate the chosen CSV files into N pieces, where N is the processes number. Each process visualizes one CSV file slice. The outputs are the visualization results image of each CSV file slice. We set up our server with an 8-core machine, process a CSV file with eight processes, and the outputs are shown in Figure 5.3. Then, the system combines the outputs into one image, as Figure 5.4 displays.

We used Flask in the server side and it is a python framework. We searched the best parallel programming method for the python platform and we decided to use a library named “multiprocessing”. There are a few reasons that we use processes instead of threads: 1) processes and threads can execute operation independently. Threads need to share the same memory spaces. However, processes run in separate memory spaces. This means there will be no writing memory conflicts if we use processes [Multithreading 2015]; 2) Python has Global Interpreter Lock (GIL). This is a mutex that makes multiple threads parallel execution impossible [GIL-Python 2015]; 3) “Multiprocessing” library is easy to use and powerful. If we run the library with a multi-processor machine, processors are able to use all the processors simultaneously for the assigned tasks [Multiprocessing 2015].

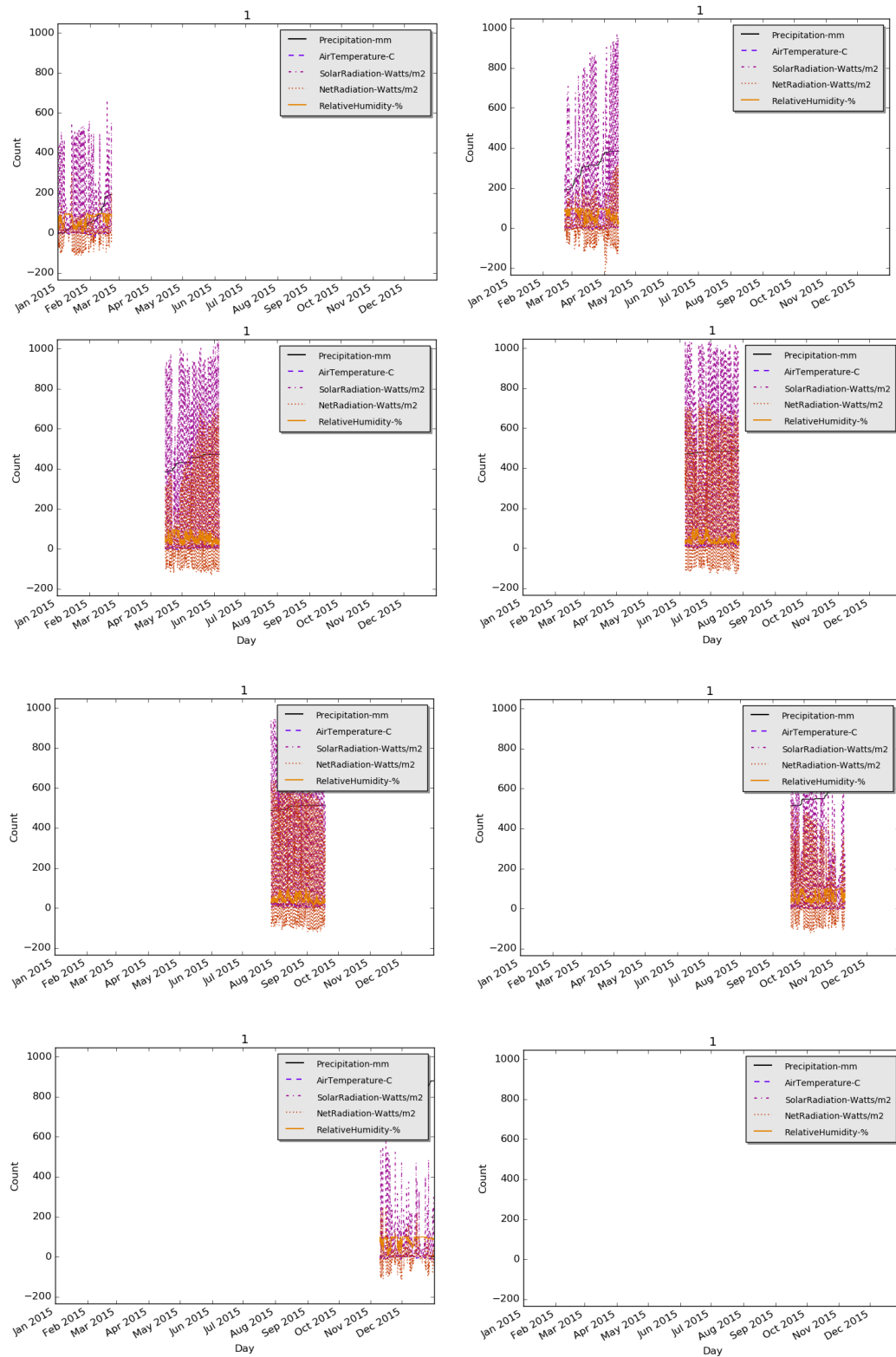


Figure 5.3. Multi-processes CSV Visualization Outputs

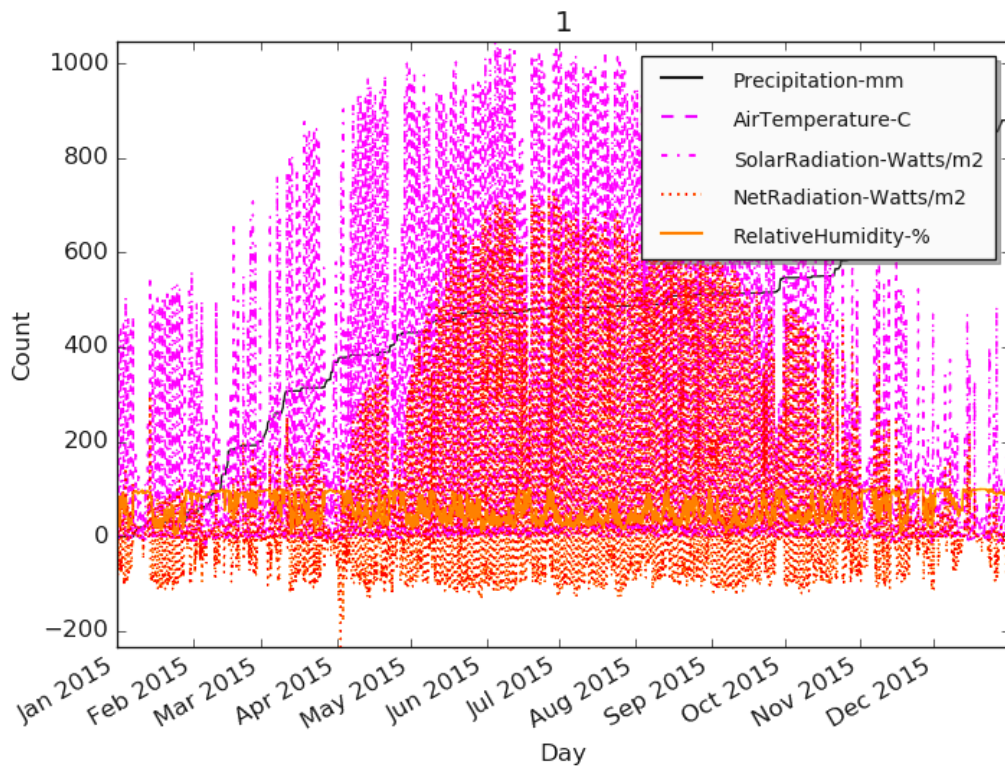


Figure 5.4. Final Visualization Result Image

The system also collect users' interaction information from the frontend and the backend will update the visualization result images based on the interaction information. For the current version, ELDP&V allows users to add a CSV column in the line chart by checking the corresponding checkbox. Similarly, if users want to remove the column, they can uncheck the corresponding checkbox. Figure 5.5 is the activity chart of the new CSV visualization workflow that includes more detailed information.

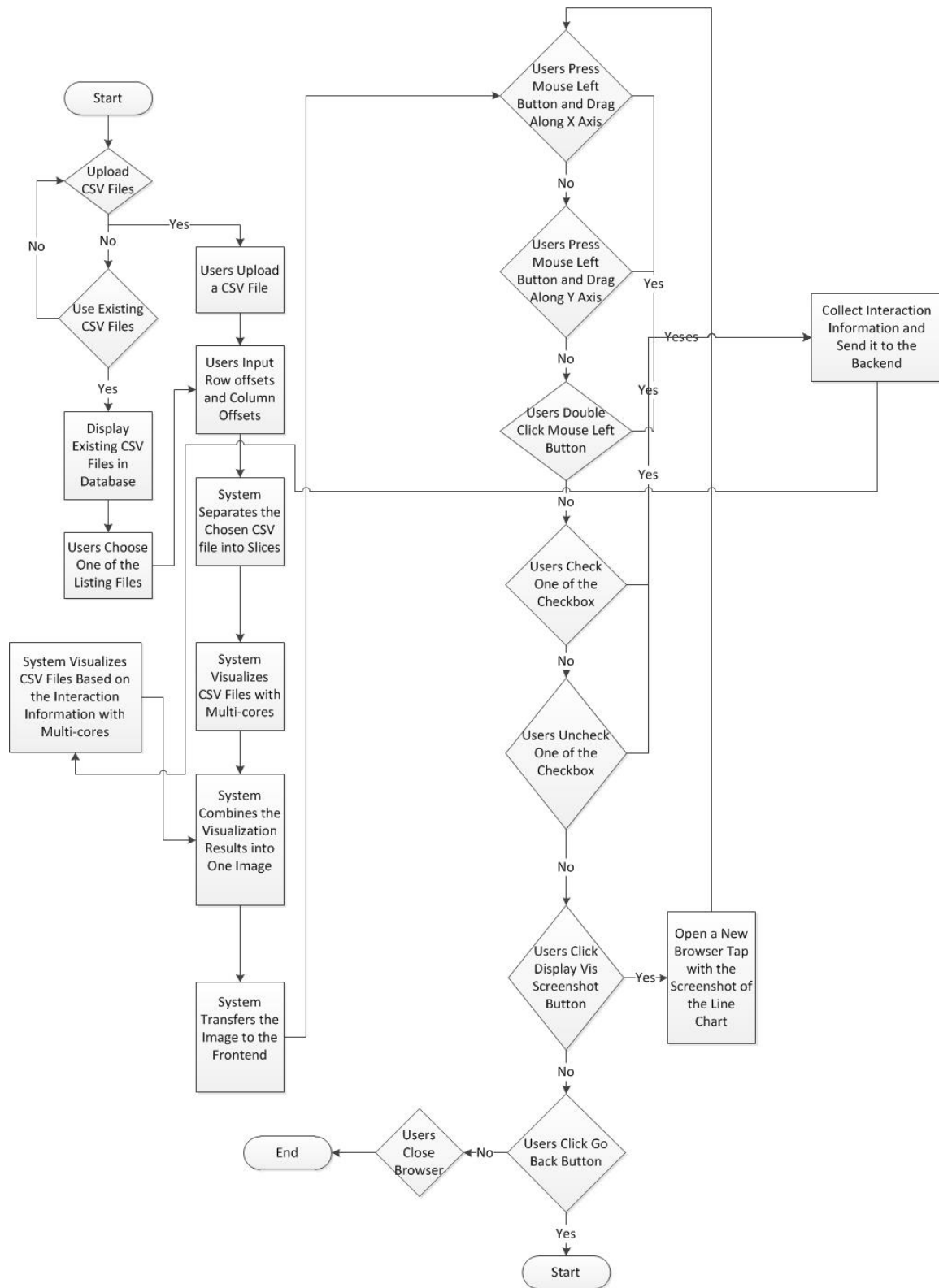


Figure 5.5. New CSV Visualization Workflow

The traditional workflow supports more interaction methods and it offers a more fluent interaction user experience. However, most of prevalent tools and systems using this workflow can only visualize around 100,000 records. The new workflow is designed for large datasets (more than 100,000 records). The dataset size grows larger, our system visualization speed is still fast. However, our current version system has less interaction methods.

We compared these two workflows in the rest part of this section. We run the frontend and backend on the same machine with the following hardware and operating system descriptions:

8 * Intel (R) Core (TM) i7-4770 CPU @ 3.4GHz

12.0 GB DDR3 RAM

Ubuntu 12.04

Figure 5.6 compares the time consumption of two workflows. When there are not many records, traditional workflow is better the new workflow. This is because the transferred data size is smaller than the visualization result image size. Also the new workflow separates CSV files into slices and combines the visualization result images together. These parts mean a large proportion of the total time if the dataset is small.

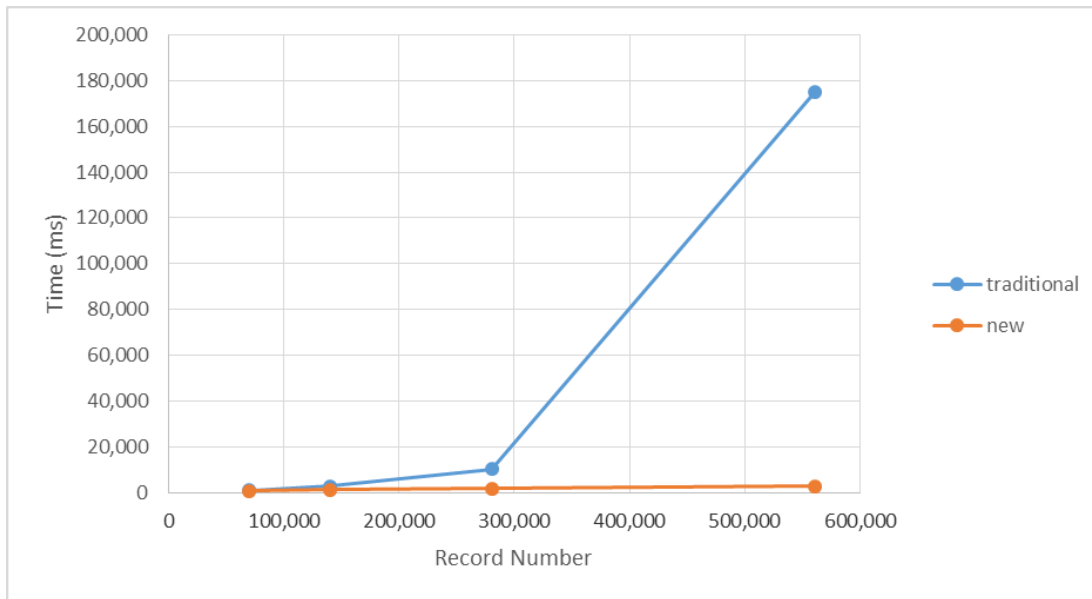


Figure 5.6. Time Consumption Comparison between Traditional and New Workflows

Figure 5.7 shows the new workflow time consumption. The traditional workflow is not in the graph because for most of the traditional workflow tools and libraries stop working or performs slowly when there are more than 100,000 records. When we tested a file with 13,455,368 records with Dygraph.js it ran for more than one hour and then pops up an error.

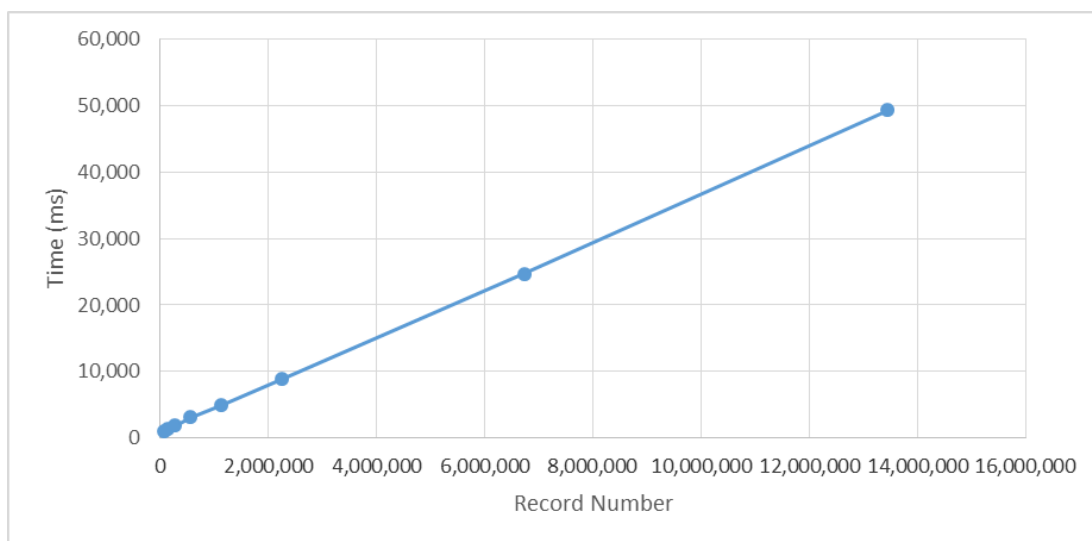


Figure 5.7. New Workflow Time Consumption

We visualize a CSV file multiple times with different number of processes. Figure 5.8 shows that the performance grows better (less time) and then goes down (more time) with the increase of processes.

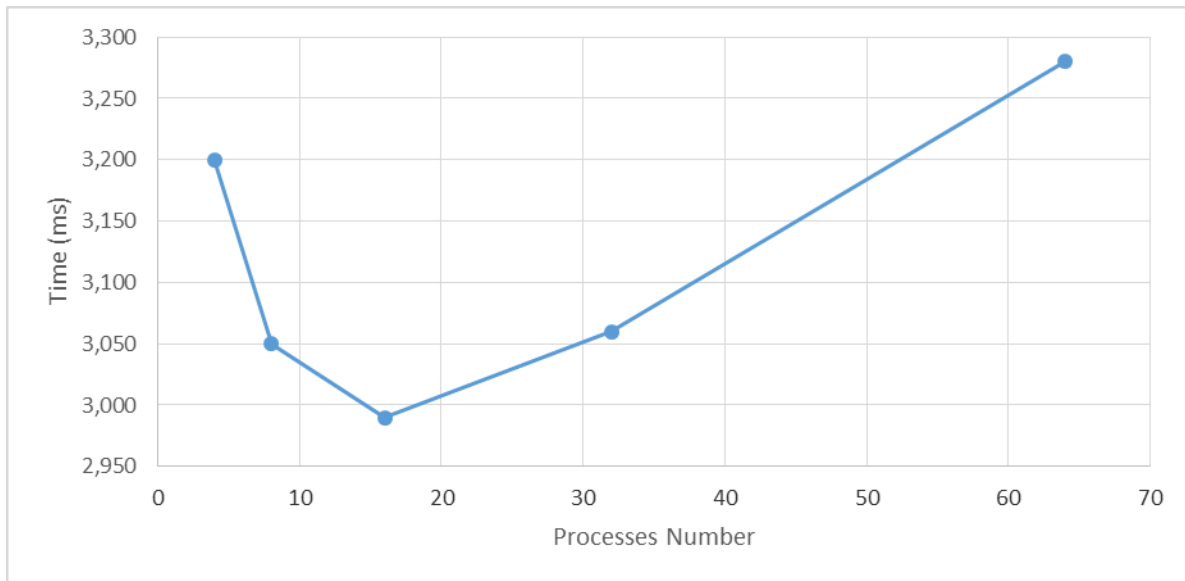


Figure 5.8. Visualize 560640 Records with Different Number Processes, When Process Number Grows, Performance Goes Up and Goes Down

5.1.1.2 NetCDF Visualization

This section introduces how ELDP&V visualizes NetCDF files. The system offers three methods: line chart, bar chart, and 2D map. The general NetCDF file visualization workflow is: Step One: Users choose to upload a NetCDF file; Step Two: Users choose a variable from a selection box to visualize; Step Three: Users choose one of the visualization methods (line charts, bar charts, or 2D maps); Step Four: Users narrow down dimensions based on the chosen visualization method; Step Five: The system displays visualization results. Figure 5.9 presents NetCDF visualization workflow of ELDP&V.

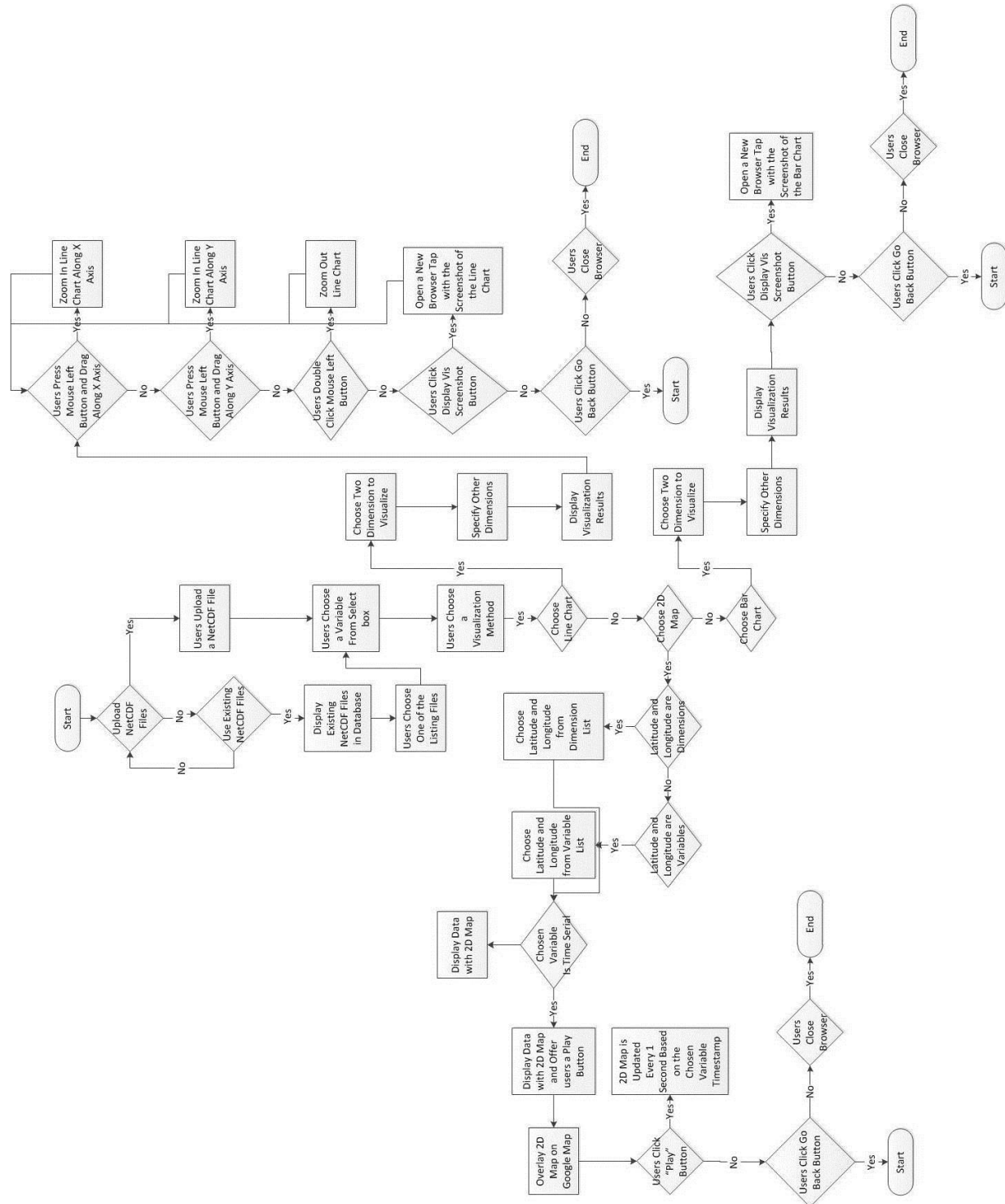


Figure 5.9. NetCDF Visualization Workflow

Line chart is a prevalent data visualization method. It is easy to find the data trend with a line chart. ELDP&V can visualize variables in a NetCDF for users. Here is the basic procedure: If users choose the line chart visualization method, they need to choose two variables from the variable list to visualize with a line chart (first one is for x-axis and second one is for y-axis). This is because of the characteristics of line chart. Each line chart can only contain two dimensions—x-axis and y-axis. The system also require that these two dimensions should not be the same variable and one of the two dimensions should be the chosen variable in Step Two. If the two chosen variables do not fulfil these two requirements, the system will pop up an alert message. After choosing these two variables, users should specify other dimensions. And then, the system will display the visualization results as Figure 5.10 presents. When users click the ‘Display Vis Screenshot’ button, the system will open another window to display the screenshot of the chosen part of visualization results. When users click ‘Go Back’ button, the system will go back to the visualization index page, where users can start another visualization process.

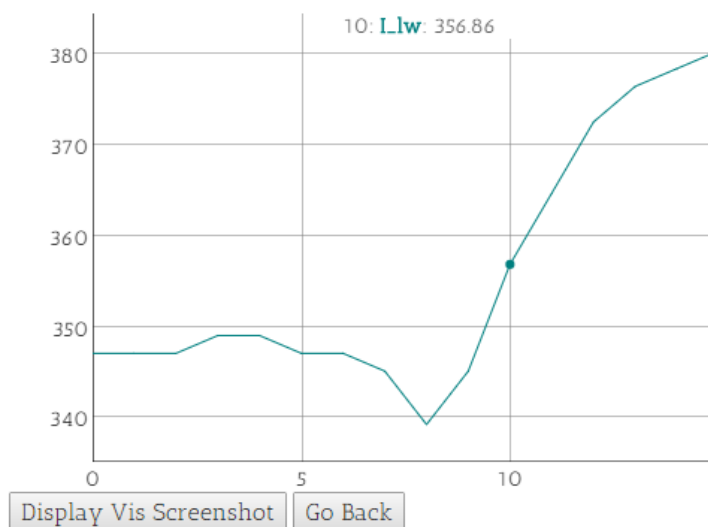


Figure 5.10. Line Chart Visualization Results Screenshot

For this part, we used similar techniques, which are introduced in ELDP&V CSV file Visualization part.

ELDP&V Bar Chart visualization is very similar to Line Chart visualization. It is also a two dimensional visualization method. Figure 5.11 presents a screenshot about this part.

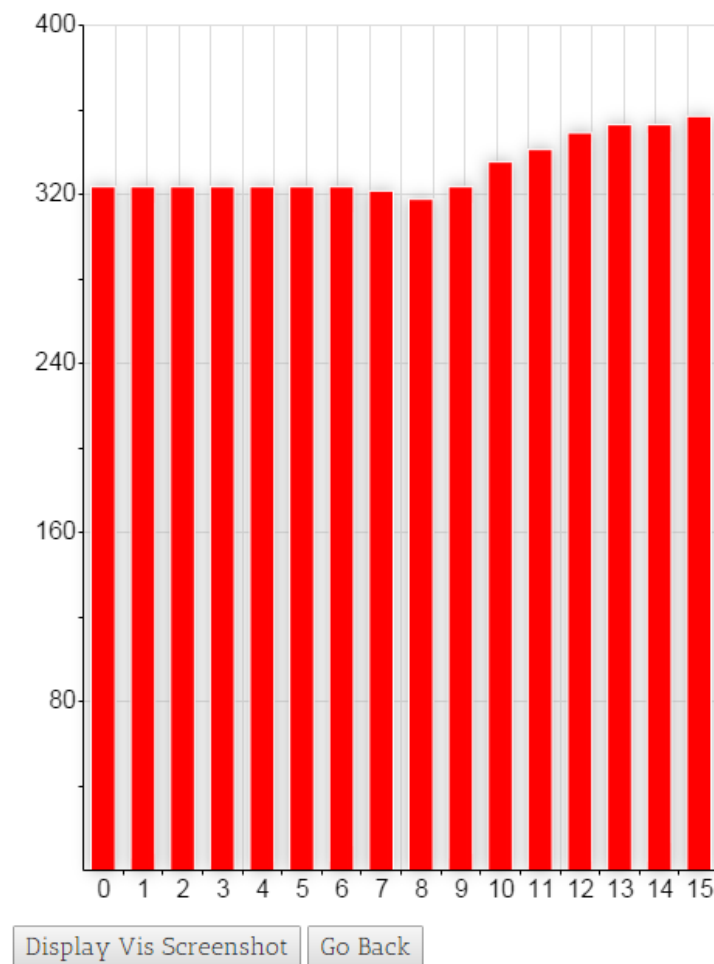


Figure 5.11. Bar Chart Visualization Results

We implemented bar chart visualization with RGraph.

Because some NetCDF files are very large, it will take a long time to load the whole file and transfer the processed JSON file. For example, we have processed some files for one year records generated by a model named “iSnowball” and the file is more than 1GB. Therefore, we designed an alternative method based on “Google Search Engine” principles. When users input some keywords, Google will react very fast and list all the search results. One of the key ideas is that Google does not finish all the task at once. It only finish the one page task each time. For example, if users input keyword “big data”, Google will only list all “big data” links (around 15) for the first page. Similarly, our system will not try to finish all the tasks at once. It will only visualize and process the data that can be viewed by users. Most of the NetCDF files are time based. Therefore, the system will only transfer, visualize and process data frame by frame based on data timestamps. The page loading time will not be decided by the file size by using this method, which means even users choose a large file, our system frontend loading time is almost the same with loading a small file.

2D map visualization is mainly separated into two parts based on the uploaded NetCDF file. Latitude and longitude information is important for 2D map visualization. Some NetCDF files define latitude and longitude information as “dimensions”. Some NetCDF files define latitude and longitude information as “variables”.

If latitude and longitude information is defined as “dimensions”, users need to choose the visualization method “map-lat-lon”. The system will obtain location information directly from “dimensions” information of the uploaded NetCDF file.

If latitude and longitude information is defined as “variables”, users need to choose visualization method “map-other”. This is more difficult than latitude and

longitude information defined as “dimensions”, for two reasons: 1) if something is defined as a “dimension”, it should be a one-dimensional array. If something is defined as “variable”, it can be multi-dimensions. It is clear that accessing data from a one dimensional array is easier than from a multidimensional array; 2) we used “NetCDF4” python library to collect data from NetCDF files. It is easier to get the data from “dimensions” than “variables”. Here is an example to obtain data from NetCDF file:

“lat” is a dimension: `netcdf_aim_file['lat'][count]`

“lat” is a variable, and the chosen file has three dimensions--easting, northing, and time: `netcdf_aim_file['lat'][easting_count][northing_count][time_count]`

Because of the order of dimensions, the python code for location information “variables” can be different too. For example, if the dimensions order is “time, easting, and northing”, then the python code should be: `netcdf_aim_file['lat'][time_count][easting_count][northing_count]`. Therefore, the system also used a function to find the order of each dimensions, if the location information is defined as “variables”.

Regardless of whatever users choose, “map-lat-lon” or “map-other”, there are some similar steps to visualize NetCDF file with a 2D map. After obtaining data from the uploaded NetCDF file, the system will try to separate data into small slices. Each slice is a JSON file with the format:

```
{
  "min_value":xxx,
  "max_value":xxx,
  "x_num":xxx,
```

```

    "y_num":xxx,

    "max_lat":xxx,

    "max_lon":xxx,

    "min_lat":xxx,

    "min_lon":xxx,

    "time":['2014-1-1',...,],

    "location":[{"lat":123,

                  "lon":234,

                  "x_location":xxx,

                  "y_location":xxx,

                  "data":[1,2,3,4]},

                {...}

            ]

    }

```

“x_num” denotes how many elements are in this slice along the x-axis; “y_num” denotes how many elements are in this slice along the y-axis; (x_location,y_location) denotes the coordinate in the 2D map; “min_value” is the minimum value of all locations "data"; “time” should have as many elements as “data” and we can find data timestamp from “time” array; “max_lat”, ”max_lon”, “min_lat”, “min_lon” are used to locate the 2D map in Google map.

Figure 5.12 presents a 2D map visualization of a variable named “I_lw” of the “iSnowball” output. Because “I_lw” has a “time” attribute as timestamp, ELDP&V offers users a “Play” button for users. If the button is clicked, ELDP&V will update the 2D map

visualization part based on the timestamp of “I_lw”. Users can stop the update by clicking “Pause” button. Users can also choose a specific timestamp by sliding the slide bar. The 2D map visualization part contains many small rectangles. Each of the rectangles corresponds to a record of “I_lw”. The position is decided by the location information (latitude and longitude). The color is decided by the value of the chosen variable.

NetCDF Map Visualization

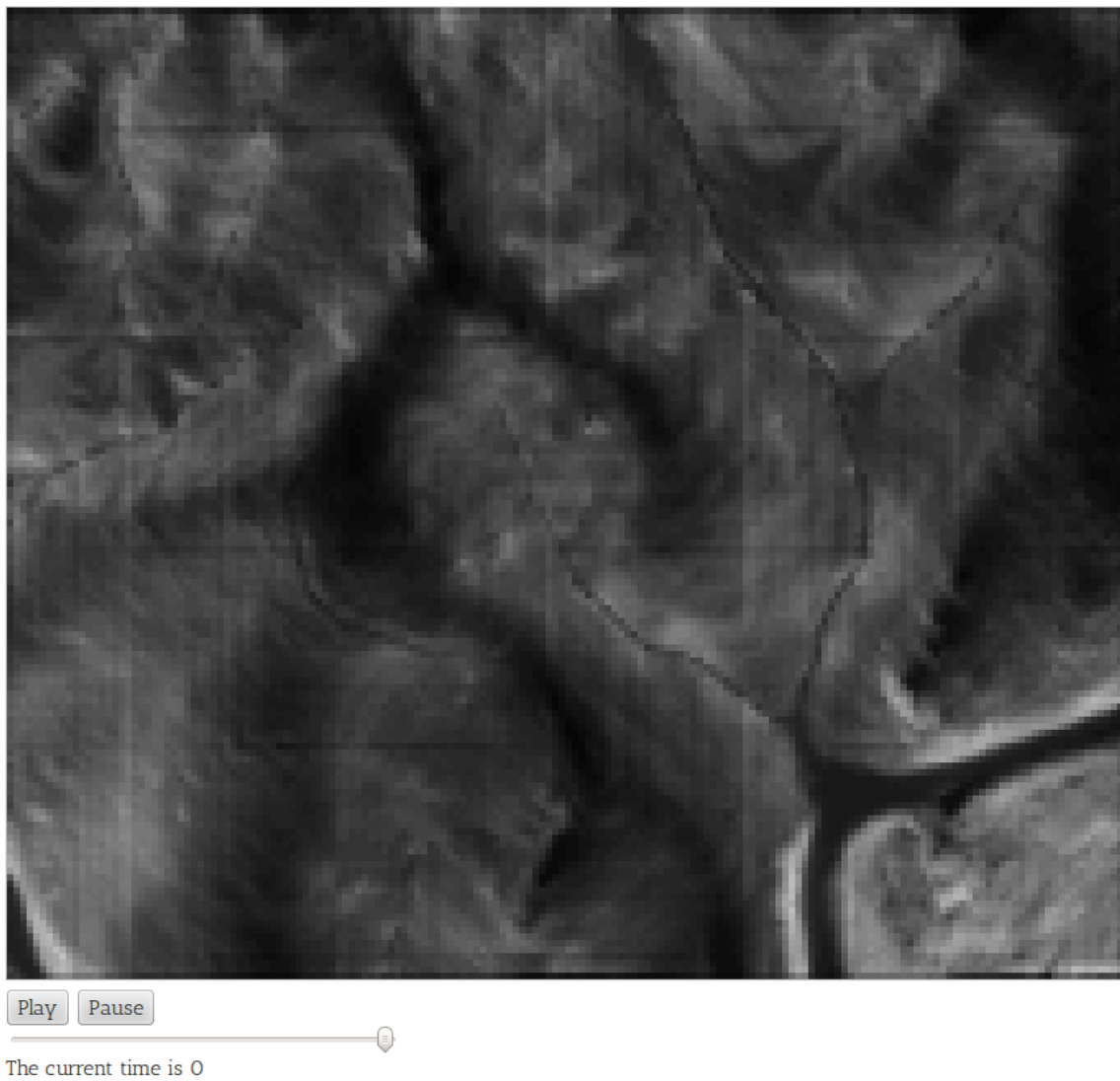


Figure 5.12. NetCDF File 2D Map Visualization

Figure 5.13 displays the ELDP&V overlays 2D map visualization on a Google map. The 2D map visualization is for Dry Creek in Idaho. When users click “Play” button in Figure 5.12, the overlay will be updated too. Users can remove the overlay by clicking “Remove overlay” button. Users can also restore the overlay by clicking “Restore overlay” button. By removing and restoring the overlay, users can check if there is something wrong with the chose variable. If the visualization results do not match the Google map very well, it means there must be errors in the chosen variable. It may be because the sensor goes down or the location information is wrong.

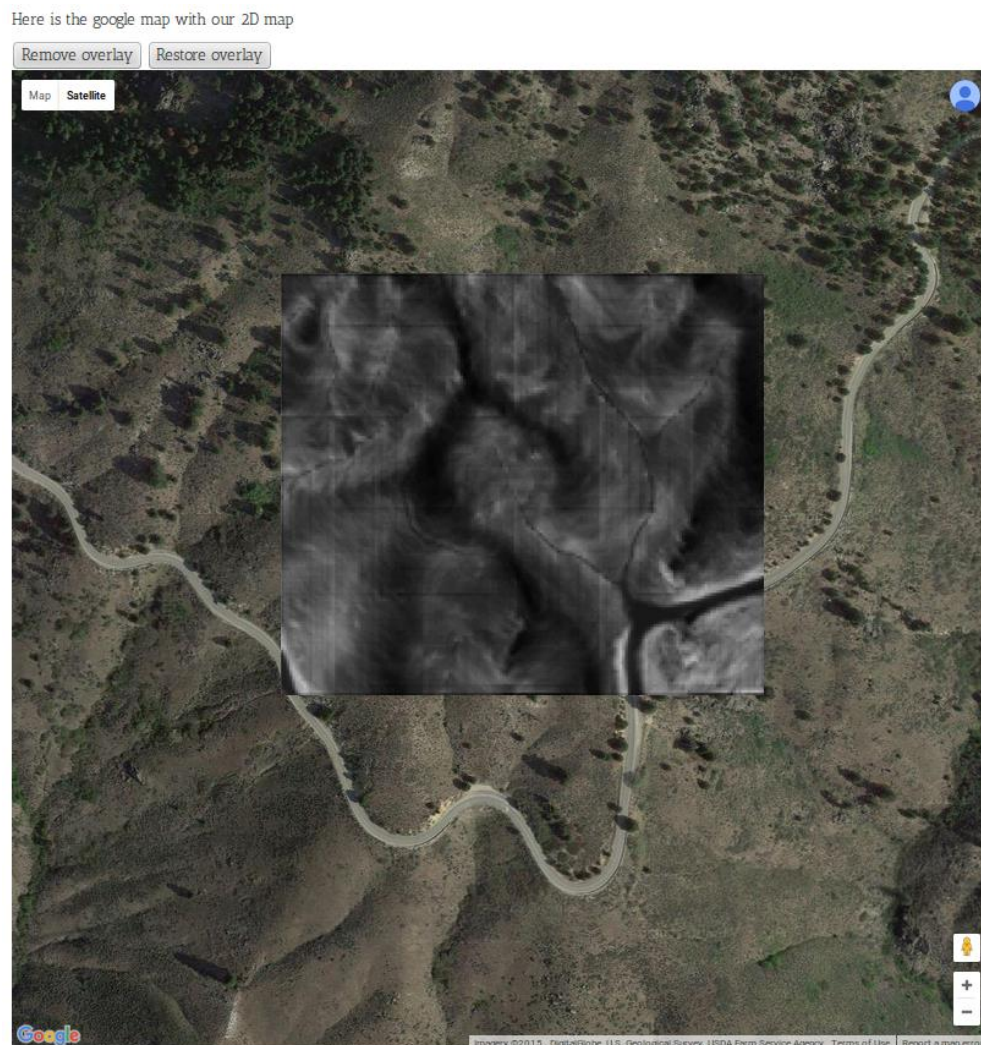


Figure 5.13. 2D Map Visualization Google Map Overlay

5.1.2 Data Processing

5.1.2.1 Basic Method

The basic data processing method means that any data processing methods excludes scientific models in this thesis.

ELDP&V offers a basic data processing method named “Frequency Distribution Histogram”. The basic idea is the system sorts the chosen array, records the elements number of the array in each section, and visualizes the sections number with a bar and line chart.

Here is an example. User A chooses a variable with the value [9,8,7,10,3,1,2,4,5,6] and wants to separate array into three sections. ELDP&V will: 1) sort the array into [1,2,3,4,5,6,7,8,9,10]; 2) separate the sorted array into three sections [1,2,3], [4,5,6], [7,8,9,10]; 3) record the elements number of each section [3,3,4] 4) create tags for each section [“1 to 3”, “4 to 6”, “7 to 10”]; 5) send the section number array and tag array from backend to frontend; 6) draw frequency distribution histogram in frontend.

Here is the JSON file format of each timestamp of the chosen variable:

```
{
  "timestamp":[
    {
      "standard_name":"XXX",
      "units":"YYY",
      "section_results_number":[1,2,3],
      "tab_name_list":["aatobb','bbtooc']
```

```

    },
    {}
  ]
}

```

Standard_name is the chosen variable name. Some variables have units and we record it in the JSON file. If the system cannot find any units information in the chosen NetCDF file, it will leave “units” as an empty string.

Most data processing parts are done in the backend (server side). This is because we want the system execution speed to be mainly decided by the servers, not the client (users’ computers). If most parts of ELDP&V are executed in the client side, the execution time is uncontrollable. For example, if the user’s computer is not good enough and he/she wants to visualize large datasets, it may take an unacceptable amount of time to finish the task.

Figure 5.14 displays a “Frequency Distribution Histogram” example of a NetCDF file that was generated by the “iSnowball” model. The chosen variable is named “donwelling_longwave_flux_in_air”. The x-axis represents a section of the chosen variable. In this example, the first section is from 284.3 to 294.90 Wm⁻². Users can customize the section number. This example has ten sections. The y-axis represents how many elements are included in the corresponding sections. When the cursor hovers over the histogram part, a tag of section information will be shown. Because the chosen variable is time-based, ELDP&V offers three buttons below the histogram: “Display Next Time Distribution”, “Play”, and “Pause”. If users click the “Display Next Time Distribution” button, ELDP&V will update the histogram with the next timestamp values

of the chosen variable. Users can also click the “Play” button and ELDP&V will update the histogram part based on timestamp every one second. If users click “Pause” button, ELDP&V will stop the histogram updating. Users can take a screenshot of the histogram part by clicking “Display Vis Screenshot” button.

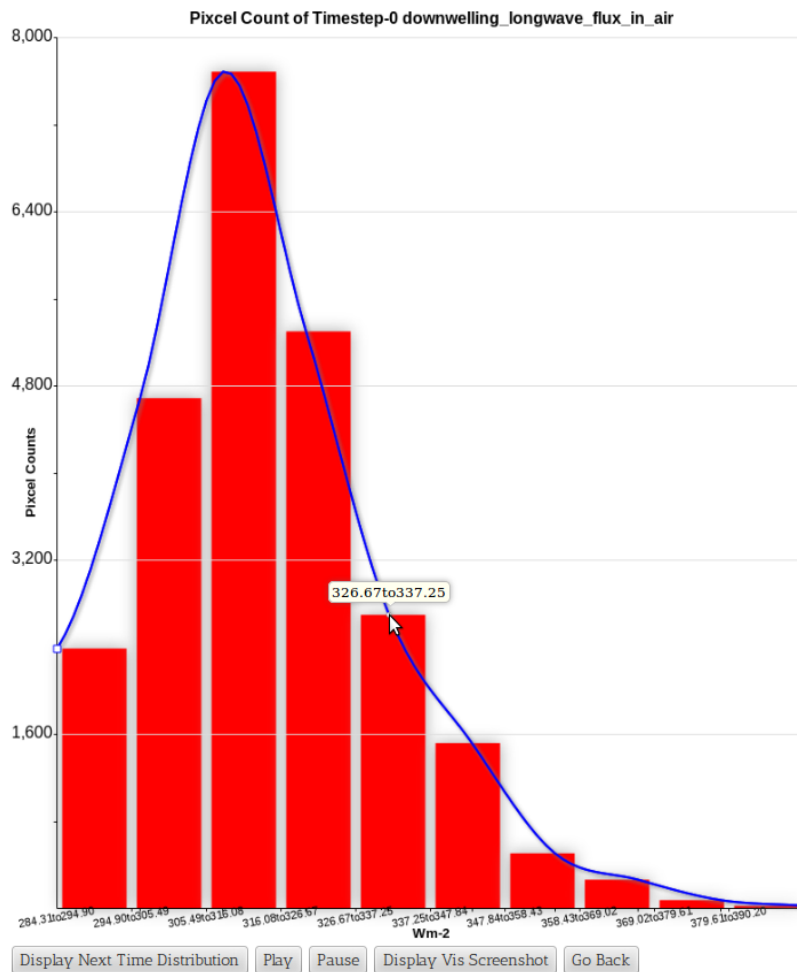


Figure 5.14. Frequency Distribution Histogram

The reason that we add “Frequency Distribution Histogram” into ELDP&V is that our collaborators want to use this method to examine data quality collected from sensors. Sometimes, sensors can collect wrong data or power off. Our collaborators can use

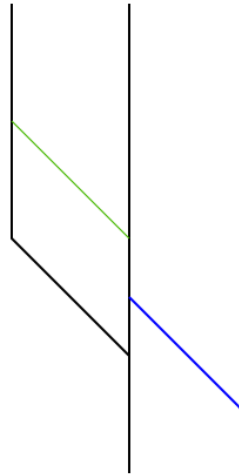
“Frequency Distribution Histogram” to tell if there are severe problems with the collected data.

5.1.2.2 Scientific Models

Most of our collaborators are environmentalists and they prefer to process data with scientific models. “Scientific Models”, in our opinion, are “special programs”. We need to import inputs into models and the models will generate some outputs.

Currently, there is only a fish model in ELDP&V for the current version. The model is created by Scott Fennema, Fawn Brooks, Teresa Campbell, Courtney Cooper, and Savannah Martinez. Users can predict the fish population of a river based on the air temperature, precipitation, elevation, and stream area.

This is our model



- Temp (-50 to 50 centigrades)
- Precip (0 to 10 inches)
- Elevation is 1043
- areastream is 89916

Result

air temp is 22.47, precip is 2.42, fish population is 31155

Figure 5.15. Fish Model [Wu et al. 2015]

Figure 5.15 displays a screenshot of the fish model in ELDP&V. Users input the four parameters and click the “Result” button, and ELDP&V will output the fish population. By using this method, users can run the model once. If users want to run the model multiple times, they can upload a CSV file into the ELDP&V server and click “Result” button. ELDP&V will display the results with a line chart as Figure 5.16 shown.

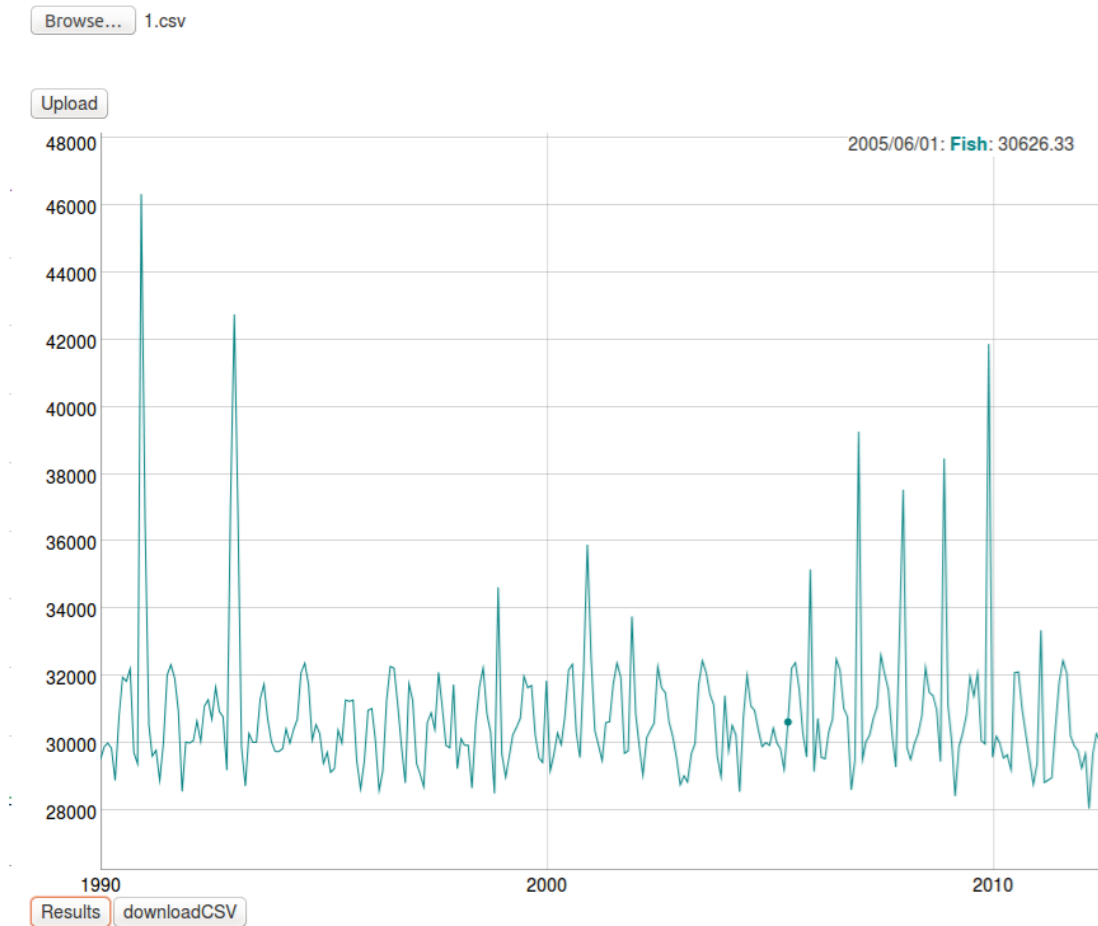


Figure 5.16. Fish Model Multi-runs

Users can zoom in the results line chart along the x-axis by clicking the mouse left button and drag along the x-axis. Users can also zoom in along the y-axis by clicking the mouse left button and drag along the y-axis. When the cursor hovers over the line chart, detailed information about the cursor part data will be displayed in the line chart top right corner. Users can also download the results by clicking the “downloadCSV” button. Users can also download parts of the results by zooming in the wanted parts and then clicking the “downloadCSV” button. ELDP&V will prepare a CSV file of the chosen part data.

5.1.2 Data Management

ELDP&V uses MongoDB and a filesystem to manage data. The basic idea is to store the path of a file into the database and store the file into the filesystem. If ELDP&V needs to access the file, then it will use its name to find the file path in the database and locate the file from the filesystem. The procedure is shown in Figure 5.17.

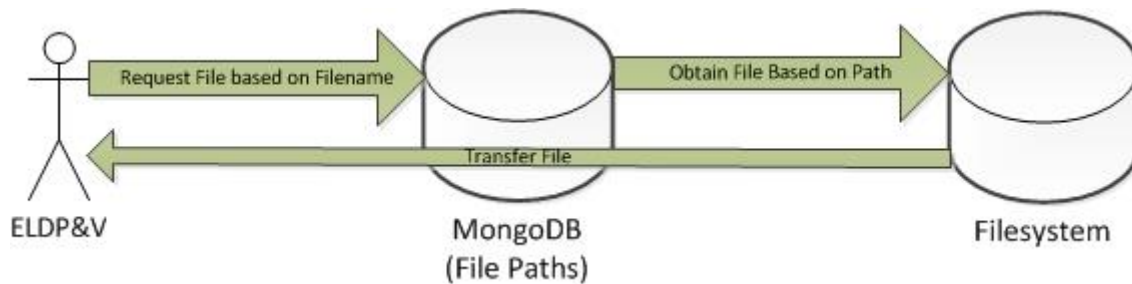


Figure 5.17. Data Management

The storage method is a map between filenames, file paths, and files. We do not store all the files into the database directly. This is because there are too many types of files and it is hard to predict what data types we are going to process in the future. Therefore, it is almost impossible to define a good schema for all the data. By using this method, we can define a unified schema for all types of data that is {filename, file_path}.

5.2 ELDP&V Video Demo

We have uploaded an ELDP&V video demo onto youtube.com. Here is the url for the video:

<https://www.youtube.com/watch?v=WZeDCD3aOp0>

The video demonstrates how ELDP&V visualized, processed CSV and NetCDF files.

Chapter 6 COMPARISON WITH RELATED WORK

In this section, we compare our work with other popular data visualizations and processing software (Microsoft Excel, Matlab, and other web-based application); advantages and disadvantages are listed.

Microsoft Excel [Microsoft 2015] is widely used in different fields. Users can visualize data with a few easy operations. There are different visualization methods, such as line charts, pie charts, 3D surfaces, and radars. Figure 6.1 is a line chart visualization example. Users do not need to program, they only need to input data and choose visualization methods. The icons are intuitive and the steps are easy to remember. Users can process data using Microsoft Excel. The basic methods include getting maximum, minimum, average values, and so on. Users do complex data processing with Microsoft Excel, such as sensitivity analysis [Sensitivity analysis 2015]. Some of our collaborators use Microsoft Excel to test their model ideas. This means they run the inputs with different formulas (model components) and compare the outputs with factual data. If users want to use Microsoft Excel, they need to buy Microsoft Excel licenses and the price is around \$80 [Microsoft Excel Price 2015]. The speed of this software is mainly dictated by the users' machines. If users want to process or visualize big data and they have good machines, then the speed may be acceptable. However, if their computers are not good enough, it may take unacceptable amounts of time to process or visualize the data.

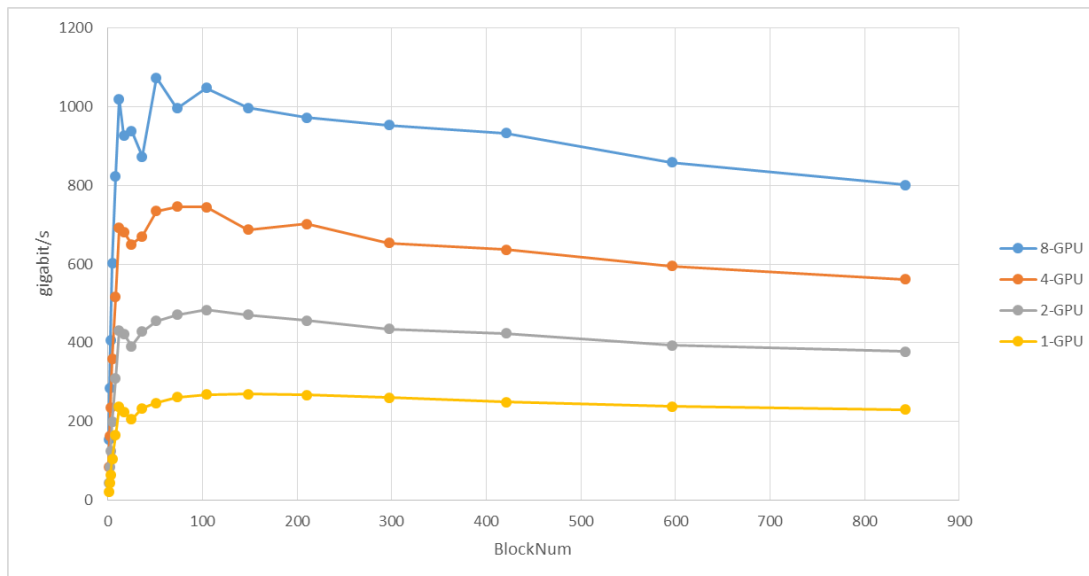


Figure 6.1. Microsoft Excel Line Chart Visualization Example

Matlab is another prevalent data visualization and processing tool. Matlab is very powerful. Users can use it to solve many research problems, such as networking, image processing, and so on. In this section, we only consider data processing and data visualization. Matlab can visualize data in 2D and 3D. Figure 6.2 is the visualization results of the equation:

$$(x^2 + \frac{9}{4}y^2 + z^2 - 1)^3 - x^2z^3 - \frac{9}{80}y^2z^3 = 0$$

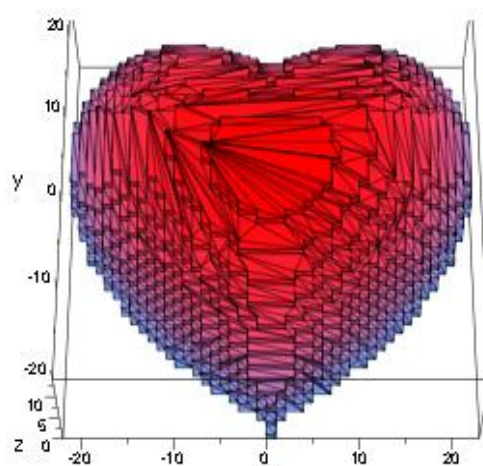


Figure 6.2. Matlab 3D Visualization Example [Wu et al. 2014]

Users can process data with some complex methods and models with Matlab too. Glover and his colleagues introduced how to analyze and process data with numerical models for marine science with Matlab [Glover et al. 2011]. The Matlab license is around \$100 [Matlab Price 2015]. Users have to write Matlab codes to visualize and process data. It can be difficult for researchers and students that have never learned programming. Fortunately, there are many Matlab scripts available to solve different problems on the Internet, and most are free to download and use. Similar to Microsoft Excel, the performance is largely decided by users' computers. If users do not have good machines, the performance may be unacceptable.

One of the solutions to reduce the users' machines effect is to use client/server program architecture [Berson 1992]. We can arrange most of the data processing and visualization workload in the server side.

VISTED [Ravi et al. 2015] is a client/server web application that was implemented by a computer science Ph.D. student named Likhitha Ravi at the University of Nevada, Reno. The software can help climate researchers identify the trends, patterns, and outliers with visualization results generated by VISTED. The tool is easy to use and the visualization results are interactive. However, users can visualize and download datasets generated by the NCAR/WRF climate model [Modeling Output 2015], which is from 1980/01-2009/12 NCEP/NCAR reanalysis and 2040/01-2069/12 for CCSM3 based on the A2 Scenario. Overall, VISTED is a good tool. We obtained a lot of useful suggestions and experience from Likhitha's application, her survey paper [Ravi et al. 2013], and another similar tool [Dittrich et al. 2013] to design ELDP&V.

InstantAtlas is another web application to visualize data with different charts [InstantAtlas 2015]. The basic idea of this tool is that it offers some templates to users and users can then customize these templates in a dashboard. Users need to choose different datasets to process. The InstantAtlas official website offers some datasets based on the chosen area and template. Users can also upload datasets themselves. Based on these datasets, users can add different charts, such as line charts and bar charts. InstantAtlas is a good dataset visualization tool. It is not complicated to use and there are some tutorials on the official website. However, this tool is not free, and only when users upload the datasets that fit the chosen template will it work.

Table 6.I concludes the comparisons between these tools and systems.

Table 6.I. Comparisons between Different Tools and Systems

| Software Name Features | Microsoft Excel | Matlab | VISTED | InstantAtlas | ELDP&V |
|---------------------------------------|--------------------------|------------------------|--------|--------------|--------|
| Need Users to Program | No | Yes | No | No | No |
| Offer Data Management Service | No | No | No | Yes | Yes |
| Performance Decided by Users' Machine | Yes | Yes | No | No | No |
| Interactive Visualization Results | Yes | Yes | Yes | Yes | Yes |
| Support NetCDF | Yes (third party add-in) | Yes | Yes | No | Yes |
| Support CSV | Yes | Yes | Yes | Yes | Yes |
| Free to Use | No | No | Yes | No | Yes |
| Data Processing with Models | Yes (complex operations) | Yes | Yes | No | Yes |
| Offer Data Display Templates | Yes | Yes | Yes | Yes | Yes |
| Display Results Less than Five steps | Yes (for most datasets) | No (for most datasets) | No | Yes | Yes |

Most of these prevalent software do not consider data management, thus users need to combine these software with other data management tools. Also, some prevalent tools do not support NetCDF files and some of them need to install third party add-ins. This is an issue. Because NetCDF file is widely used in scientific research.

Chapter 7 FUTURE WORK

In the future, we want to add some new features into ELDP&V. First, ELDP&V will support more data formats. ELDP&V supports CSV and NetCDF formats, which are the two formats widely used in scientific researches for the current version. ELDP&V should support more formats for the general uses. We also plan to enable users to define their own file formats to visualize.

Second, we plan to process and visualize real big data (more than 1TB). Currently, ELDP&V can visualize and process 1GB data, which is not real big data. We need to reprogram some parts of ELDP&V to make it ready for real big data. As we discussed in Chapter 5, we designed a new data visualization workflow that visualize data in the server side and only transfer result images to the client side. The new workflow implemented in our current version system is not perfect. Because we only use one computer with multi-processes in the server side and our system loads data into memory first and then visualizes data, the result is that ELDP&V can only visualize the maximum server machine memory size 12GB data. To improve the workflow, we plan to use distributed systems on the server side, as presented in Figure 7.1. We will create functions to allocate a data slice to each node and combine the results from different nodes. We also plan to use Graphics Processing Units (GPUs) in each node to further increase the performance. The improved new workflow will generate many images. We have an idea to combine them in a parallel way as Figure 7.2 presented. There is another limitation for this new workflow. Each time users interact with the visualization results, the system will redraw the image again from the start (obtain data, visualize data, and

send result image to the client side); this is slower than the traditional workflow that does not need to send any requests to the server side after loading all the data in the client side. We plan to improve the new workflow by reducing communication with the server side. For the visualization part, we also want to add more interaction methods in the future. For the current version, ELDP&V only allow users to add or remove CSV file columns from the result images using check boxes. The future possible interaction methods may include zoom in and zoom out of result image. We can collect users' mouse information (coordinates, click, and release). Based on the information, we can locate the zoom in part data, redraw image, and update the result image in the client part. Also, when users double click on the result image, the system will redraw and update the image with the zoom out part data.

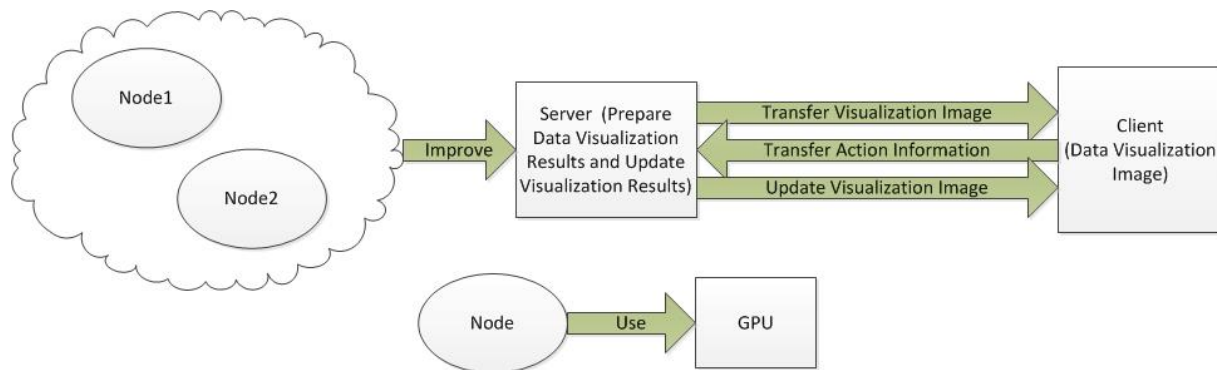


Figure 7.1. New Workflow Improvement

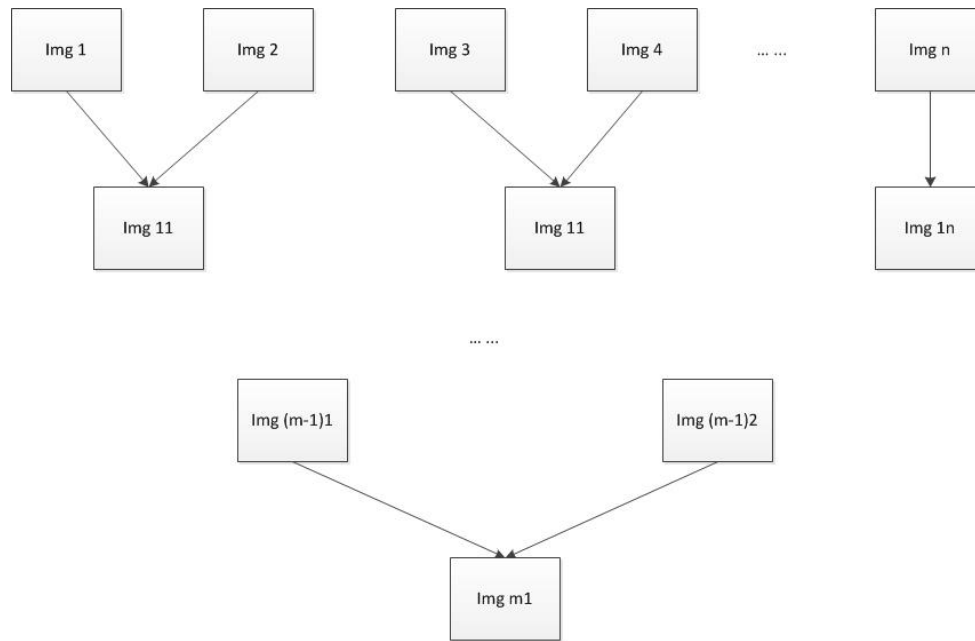


Figure 7.2. Parallel Image Combination Method

Third, we are going to add more visualization methods. ELDP&V has mainly three visualization methods: line charts, bar charts, and 2D maps. Different problems have their own best visualization methods. For example, if users want to view the ratio of each section, the best visualization method may be the pie chart. ELDP&V needs more visualization methods for general use.

Last but not least, ELDP&V needs a more effective method to import models. We require users to email us the pseudocode and we will implement the model by ourselves for the current version. This is not effective. Sometimes the model is too complex and requires a lot of time to implement. We plan to develop a new method to import models, such as a general method to create a dynamic link library for a model.

Chapter 8 CONCLUSIONS

In this thesis, we introduced our system, ELDP&V, which is a client/server web-based application. The system, focused on dealing with environmental data, is used to handle big data from three aspects: data visualization, data processing, and data management. ELDP&V supports CSV and NetCDF files. These two file formats are widely used in scientific research.

Visualization is a significant method to find the “gold nuggets” hidden in big data. ELDP&V provides three methods to visualize data: line charts, bar charts, and 2D maps. We believe interaction is necessary for visualization, because users do not need to repeat many operations if they can interact with visualization results. For example, users visualize temperature records of Reno from 10/01/2014 to 10/01/2015 with a line chart, and they want to visualize temperature records of Reno from 05/01/2015 to 10/01/2015 with a line chart for further study. They do not need to repeat operations to obtain the data. All they need to do is to zoom in on the previous step of visualization results. Most of ELDP&V visualization results are interactive. ELDP&V offers two workflows for CSV visualization. The traditional workflow transfers all the data to the client side and visualize data with Dygraph, which is a JavaScript library. This workflow has more interaction methods, such as zoom in, zoom out, download the chosen part data, and display line chart screenshot. If the data size is small (under 100,000 records) the interaction response is fast and the data transfer time is short. However, this workflow requires the client side’s computers to load data to the memory. Therefore, if the data size is bigger than the client side’s machine memory size, this workflow will not work. Also,

if the data size is very large, the data transfer time can be unacceptable. Another workflow is designed by us and is different from most of other web-based client/server application visualization workflows. It visualizes data in the server side and only transfers resulting image to the client side. This workflow is faster than the traditional workflow if users want to visualize large size datasets. This new workflow also has some limitations. The interaction response is slower than the traditional workflow for small datasets because the system will redraw all visualization result images from the start (obtain data from the chosen file, draw images on the server side, and transfer updated images to the frontend). In conclusion, users should choose the traditional workflow if they want to visualize small datasets and they should choose the new workflow if they have large datasets or a slow-speed network.

Scientists use different methods to process data and discover rules behind data. ELDP&V can process data with basic methods and models. Frequency distribution histograms are one of the basic data processing methods in ELDP&V. Our collaborators use this method to identify the errors hidden in the chosen datasets. ELDP&V has only a fish model in its the current version. Users can predict the fish population density of a river, and they can run the model multiple times with a CSV input file.

Data management is important and fundamental for the whole system. We used MongoDB and a filesystem to manage our data files. The file paths are stored in MongoDB with filenames and IDs. Based on the paths, the system can find the file locations in the filesystem. We designed the system in this way because there are too many file formats and it is hard to create some fixed schemas for all the files.

Our main contributions are: 1) Our system can visualize and process NetCDF files

in an intuitive way. Users can visualize their NetCDF files in less than four steps and there are detailed explanations for each step. Most web-based applications do not support the NetCDF file format; 2) Users can efficiently interact with our visualization results. Some interactions can save users a lot of time. For example, if users want to extract data from a huge file, they can zoom in the wanted part and click the “download” button. 3) Because we used some new methods and a new workflow, we can visualize and process large size datasets very fast. For example, in our system the current version, ELDP&V has visualized 13,455,368 records in 50 seconds with eight CPU-cores. The performance can be improved with a distributed system, which has been introduced in Future Work Chapter. Most popular web-based tools and libraries stop working or are very slow when there are more than 100,000 records.

REFERENCES

- [Azemi 1996] Azemi, A. 1996. Utilizing MATLAB in undergraduate electric circuits courses. In *Proceedings of 26th Annual Conference Frontiers in Education Conference, FIE'96*, Vol. 2, pp. 599-602.
- [Berson 1992] Berson, A. (1992). Client-server architecture (No. IEEE-802). McGraw-Hill.
- [Big Data Explained 2015] Big Data Explained | MongoDB, (last accessed 07/27/2015),
<https://www.mongodb.com/blog/post/building-inexpensive-petabyte-database-mongodb-and-amazon-web-services-part-1>
- [Box and Draper 1987] Box, G. E. and Draper, N. R. 1987. Empirical model-building and response surfaces (Vol. 424). New York: Wiley.
- [Codd 1972] Relational completeness of data base sublanguages, IBM Corporation, pp. 65-98.
- [CSV 2015] CSV, Comma Separated Values (RFC 4180), (last accessed 04/06/2015),
<http://www.digitalpreservation.gov/formats/fdd/fdd000323.shtml>
- [Cudré-Mauroux et al. 2009] Cudré-Mauroux, P., Kimura, H., Lim, K. T., Rogers, J., Simakov, R., Soroush, E., ... and Zdonik, S. 2009. A demonstration of SciDB: a science-oriented DBMS. In *Proceedings of the Very Large Data Bases Endowment*, 2(2), 1534-1537.
- [D3 2015] D3.js-Data-Driven-Documents, (last accessed 07/27/2015),
<http://d3js.org/>
- [Dascalu et al. 2014] Dascalu, S., Harris, F.C., Jr., McMahon, M., Jr., Fritzinger, E., Strachan, S. , and Kelley, R. (2014). An Overview of the Nevada Climate Change Portal. In *Proceedings of the 7th Intl. Congress on Environ. Modelling & Software (iEMSS-2014)*, San Diego, CA, June 2014, vol. 1, pp. 75-82.
- [Dittrich et al. 2013] Dittrich, A., Dascalu, S., and Gunes M. (2013). ATMOS: A

- Data Collection and Presentation Toolkit for the Nevada Climate Change Portal. In *Proceedings of the International Conf. on Software Eng. and Applications (ICSOFT-EA 2013)*, Reykjavik, Iceland, July 2013, pp. 206-213.
- [Dygraphs 2015] Dygraphs, (last accessed 07/27/2015), <http://dygraphs.com/>
- [Facebook 2015] Scaling the Facebook data warehouse to 300 PB (last accessed 05/04/2015), <https://code.facebook.com/posts/229861827208629/scaling-the-facebook-data-warehouse-to-300-pb/>
- [Flask 2015] Welcome | Flask (A Python Microframework), (last accessed 09/28/2015), <http://flask.pocoo.org/>
- [Fox and Andersen 2005] Fox, J. and Andersen, R. (2005). Using the R Statistical Computing Environment to Teach Social Statistics Courses. Department of Sociology, McMaster University.
- [GIL-Python 2015] GloablInterpreterLock-Python Wiki, (last accessed 07/27/2015), <https://wiki.python.org/moin/GlobalInterpreterLock>
- [Glover et al. 2011] Glover, D. M., Jenkins, W. J., and Doney, S. C. (2011). Modeling methods for marine science. Cambridge University Press.
- [Han et al. 2011] Han, J., Haihong, E., Le, G., and Du, J. (2011). Survey on NoSQL database. In *Proceedings of the 6th International Conference on Pervasive Computing and Applications (ICPCA), 2011 IEEE*, pp. 363-366.
- [Hao 2011] Hao, Y. 2011. Application Research on deformation monitoring of SMW method H steel pile using BOTDA. In *Proceedings of 2011 International Conference on Multimedia Technology (ICMT)*, pp. 3862-3865.
- [Heer et al. 2005] Heer, J., Card, S. K., and Landay, J. A. 2005. Prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems ACM*, pp. 421-430.
- [IBM 2015] IBM big data platform – Bringing big data to the Enterprise, (last accessed 10/26/2015), <http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html>

- [InstantAtlas 2015] Interactive mapping software | InstantAtlas (last accessed 5/12/2015), <http://www.instantatlas.com/>
- [Janne 2014] Why WebGL just isn't enough | MyCadbox Blog, (last accessed 09/04/2015), <http://blog.mycadbox.com/?p=4211>
- [Kelton and Law 2000] Kelton, W. D. and Law, A. M. (2000). Simulation modeling and analysis. Boston: McGraw Hill, pp 9-12.
- [Laney 2001] Laney, D. 2001. 3D data management: Controlling data volume, velocity and variety. META Group Research Note 6: 70.
- [Leavitt 2010] Leavitt, N. (2010). Will NoSQL databases live up to their promise? Computer, 43(2), 12-14.
- [Lee et al. 2014] Lee, S., Jo, J. Y., and Kim, Y. (2014). Performance testing of web-based data visualization. In *Proceedings of 2014 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 1648-1653.
- [Ma et al. 2004] Ma, X., Winslett, M., Norris, J., Jiao, X., and Fiedler, R. 2004. Godiva: Lightweight data management for scientific visualization applications. In *Proceedings of 20th International Conference on Data Engineering, 2004*. pp. 732-743.
- [Matlab Price 2015] Amazon.com: Matlab and Simulink Student Suite R2015b, (last accessed 10/21/2015), http://www.amazon.com/MATLAB-Simulink-Student-Suite-R2015b/dp/0989614026/ref=sr_1_3?ie=UTF8&qid=1445456675&sr=8-3&keywords=matlab
- [Microsoft 2015] Spreadsheet Software Program | Excel Free Trial, (last accessed 10/20/2015), <https://products.office.com/en-us/excel>
- [Microsoft Excel Price 2015] Amazon.com: Microsoft Excel, (last accessed 10/20/2015), http://www.amazon.com/s/ref=nb_sb_ss_c_0_15?url=search-alias%3Daps&field-keywords=microsoft+excel&prefix=microsoft+excel%2C%2C%2C164
- [Microsoft SQL 2015] Structured Query language (SQL), (last accessed 09/22/2015), <https://msdn.microsoft.com/en->

- [gb/library/windows/desktop/ms714670\(v=vs.85\).aspx](http://gb/library/windows/desktop/ms714670(v=vs.85).aspx)
- [Modeling Output 2015] Modeling Output (last accessed 5/12/2015), <http://sensor.nevada.edu/NCCP/Downloads/Modeling%20Output.aspx>
- [MongoDB 2015] MongoDB, (last accessed 07/27/2015), <https://www.mongodb.org/>
- [MongoDB Users 2015] Out Customers | MongoDB, (last accessed 09/23/2015), <https://www.mongodb.com/who-uses-mongodb>.
- [Multithreading 2015] Multithreading-What is the difference between a process and a thread-Stack Overflow, (last accesses 11/18/2015), <http://stackoverflow.com/questions/200469/what-is-the-difference-between-a-process-and-a-thread>
- [Multiprocessing 2015] GIL Python & Multiprocessing Performance, (last accesses 11/18/2015), <https://www.quora.com/On-multicore-machines-will-Python-processes-achieve-the-same-level-of-performance-as-threads-multiprocessing-Does-the-GIL-limit-Python-performance-on-multi-core-machine-to-that-of-a-single-core-even-if-threading-is-used>
- [Ncview 2015] Ncview, (last accessed 07/27/2015), http://meteora.ucsd.edu/~pierce/ncview_home_page.html
- [OGC 2015] OGC network Common Data Form (NetCDF) standards suite, (last accessed 04/06/2015), <http://www.opengeospatial.org/standards/netcdf>
- [Paradigm4 2015] Home, (last accessed 04/04/2015), <http://www.paradigm4.com/>
- [Ramachandran 2015] Ramachandran, P., and Varoquaux, G. 2011. Mayavi: 3D visualization of scientific data. *Computing in Science & Engineering*, 13(2), 40-51.
- [Ravi et al. 2015] Ravi, L., Dascalu, S., Harris, F.C., Jr., Mejia, J., and Belkhatir, B., (2015). VISTED: A Visualization Toolset for Enviromental Data, In *Proceedings of the 2015 Intl. Conf. on Computers and Their Application (CATA-2015)*, March 2015, Honolulu, HI, pp. 335-342.
- [Ravi et al. 2013] Ravi, L., Yan, Q., Dascalu, S. M., and Harris, F.C., Jr. (2013). A Survey of Visualization Techniques and Tools

- for Environmental Data. In *Proceedings of the 2013 Intl. Conference on Computers and Their Applications (CATA 2013), March 2013*, Honolulu, Hawaii.
- [RGraph 2015] RGraph: Free HTML5 charts that get you noticed, (last accessed 09/21/2015), <http://www.rgraph.net/>
- [SAS 2015] What is Big Data? | SAS (last accessed 5/4/2015), <http://www.sas.com/big-data/>
- [Savory 1995] Savory, P. A. 1995. Using Mathematica to aid simulation analysis. In *Proceedings of the 27th conference on Winter simulation*, pp. 1324-1328.
- [Sensitivity analysis 2015] Sensitivity Analysis Using Excel, (last accessed 10/20/2015), <http://treeplan.com/chapters/sensitivity-analysis-using-excel.pdf>
- [SQL RDBMS Concept 2015] SQL_RDBMS Concepts, (last assessed 09/22/2015), <http://www.tutorialspoint.com/sql/sql-rdbms-concepts.htm>
- [Stonebraker 2010] Stonebraker, M. (2010). SQL databases v. NoSQL databases. *Communications of the ACM*, 53(4), 10-11.
- [Stonebraker et al. 2009] Stonebraker, M., Becla, J., DeWitt, D. J., Lim, K. T., Maier, D., Ratzesberger, O., and Zdonik, S. B. 2009. Requirements for Science Data Bases and SciDB. In *Proceedings of Fourth Biennial Conference on Innovative Data Systems*, Vol. 7, pp. 173-184.
- [Taft et al. 2014] Taft, R., Vartak, M., Satish, N. R., Sundaram, N., Madden, S., and Stonebraker, M. 2014. Genbase: a complex analytics genomics benchmark. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data* (pp. 177-188). ACM.
- [Tavares 2012] WebGL Fundamentals – HTML5 Rocks, (last accessed 09/04/2015), http://www.html5rocks.com/en/tutorials/webgl/webgl_fundamentals/
- [Voulgaropoulou 2012] Voulgaropoulou, S., Spanos, G., and Angelis, L. 2012. Analyzing measurements of the R statistical open source software. In *35th Annual IEEE Software Engineering Workshop (SEW)*, pp. 1-10.

- [Wu et al. 2015] Wu, R., Dascalu, S., and Harris, F.C. (2015). Environment for Datasets Processing and Visualization Using SciDB. In *Proceedings of the 2015 International Conference on Software Engineering and Data Engineering (SEDE-2015)*, San Diego, CA, October 2015.
- [Wu et al. 2014] Wu, R., Palathingal, L., Redei, A., and Dascalu, S. (2014). Waldo3D: Printing 3D Models from 2D Pictures, In *Proceedings of the 2014 International Conference on Software Engineering and Data Engineering (SEDE-2014)*, New Orleans, LA, October 2014, pp. 125-130.
- [YouTube 2015] Statistic—YouTube (last accessed 05/04/2015), <https://www.youtube.com/yt/press/statistics.html>