CS Capstone Technology Review

December 3, 2017

CoverageJSON Response Handler for OPeNDAP

Prepared for

NASA Jet Propulsion Laboratory

Lewis John McGibbney

Prepared by

Group55

River Hendriksen

Abstract

This document reviews the possible technologies choices that can be used for the CoverageJSON Response Handler for OPeNDAP project.

1 Technology: Server

1.1 Overview

1.1.1 AWS

1.1.2 Windows Server

1.1.3 Local Hosting

1.2 Discussion

1.3 Conclusion

2 Technology: Hosts

2.1 Overview

2.1.1 Hyrax Data Server

2.1.2 Apache HTTP Server

2.1.3 ASP .NET

2.2 Discussion

2.3 Conclusion

3 Technology: Coding Languages

3.1 Overview

3.1.1 Java

3.1.2 Python

3.1.3 C++

3.2 Discussion

3.3 Conclusion

1 Technology: Server

1.1 Overview

Servers are something that is integral to how the Internet works, however it is often overlooked on a hardware level. A server defined is defined as any hardware that allows for other applications, or programs, to be accessed by clients [8]. Clients in the context of this project will be users with an interest in the data hosted by NASA JPL. This is not to be confused with a host, or web server, which is the software on the server that handles HTTP requests. Hosts will be discussed in length in section three. Servers easily accessed whether that means virtually or physically, for this reason there is only three choices, Windows Server, AWS, and a local server.

1.1.1 AWS

There are many options when discussing cloud hosting for servers however none is complete without mentioning AWS, or Amazon Web Services. AWS is the most widely used web services provider in the world, hosting such platforms as Mozilla, HULU, etc. AWS is versatile in its implementation allowing for any type of service to be stored on their platform, this can mean data server to red hat server. Think of AWS as a cloud computing platform rather than a specific server, since the applications allowed are so wide. [1]

1.1.2 Windows Server

Windows server is not a physical hardware, it is an Operating System that allows for web frameworks to be applied to. Since all other services use RedHat for their preferred operating system it is important to then discuss the option of Windows Server. For a general user the differences between Windows and Linux may not be obvious, both functionally do the same thing. However from an IT management side they are far from similar. windows has the benefit of being a non-open source Operating System, meaning that one has to purchase the license to use Windows Server from Microsoft. This is beneficial because Microsoft then supports the software allowing easy debugging unlike Linux. On the flip-side RedHat is free and easy to implement on any hardware source. RedHat allows for much customization of its services since IT would have direct control of the packages being installed to it. [9]

1.1.3 Local Hosting

Local hosting is less of a specific server and more an option for where to host. Having a home server, or one physically accessible, allows for customization of hardware and software. This is useful for quick debugs of problems and not having to worry about loss of data if the service company loses the server. However both cost upfront and down the line cost with business class Internet licenses make this the worse choice of the three.

1.2 discussion

There are differences and similarities between all these options, so as to give them all fair discussion we will discuss the pros and cons of each. AWS has the benefit of being one of the most popular web server service in the world this would help with scalability and accessibility to users across the globe. Cons of AWS are that if this project were to go into production (which it is planned too) then the cost of hosting would increase to the point that could not be afforded on this projects non-existent budget. As mentioned before Windows has the benefit of being a supported system, so many of the network design decisions would be not have to be considered. On the other hand there is no place to remotely host a Windows Server that has a free tier, and Windows would create difficulties with the hosts that are planned to be used.

Local hosting has the most obvious pros and cons, pros being that being closer to the server allows for customization and easy debugging of hardware and software. However there are many cons, ISP's generally do not take kindly to business applications being implemented on domestic subscriptions, the system would not be scalable, and we would have to implement our own handler requests.

1.3 Conclusion

The preferred choice for server is still not obvious as of writing this technically document. As it currently stands AWS will be used for the testing and creation of the code since it has a free tier and allows all parties in this project to have access easily. After the end of this project the server is not defined, though it will likely be integrated into the current server that OPeNDAP uses.

2 Technology: Hosts

2.1 Overview

Hosts are one of the most important parts to consider when implementing a tool that is expected to be usable by users across the web. Hosts, defined in this context as web servers, will hold the OPeNDAP protocol code that will be called when a user makes a request for the coverage JSON. For this reason the server must be accessible, easy to use, and be able to handle many requests. The hosts we will talk about are Hyrax servers, ASP .NET, and local hosting.

2.1.1 Hyrax data Server

The Hyrax data server architecture is a design that is currently implemented with OPeNDAP. The purpose of Hyrax is specifically to allow for the OPenNDAP protocol and the handling of the requests from a web daemon to DAP software services. Hyrax offers both a client interface (see the NASA JPL website for an example of this), and web daemon back end for users to make requests. This is done via Java servlets, which are used to take in requests. The front end OPeNDAP Lightweight Front end Server(OLFS) will take in requests and push it to the second server, this can be the same server, called the Back End Server (BES). [5]

2.1.2 Apache HTTP Server

Apache HTTP Server is an open source Web service handler that can take in requests via HTTP. Apache is beneficial in that is already a standard in many servers and is considered easy to implement on any hardware. Apache also has the benefit of having MultiProcessing Modules (MPMs) that allow for scalability on demand depending on the current load. [2]

2.1.3 ASP .NET

ASP .NET is Microsoft's implementation of MVC framework. MVC, Model View Controller, is a standard method for a user interacting with a web service on a page. MVC will not be discussed in detail since it is not the point of this paper, but know that almost all popular websites use it. However that does not mean all websites ASP .NET, which is generally used in internal business applications. .NET has the benefit of being a a Microsoft owned language and as such has easy to access documentation and is fully supported by the Visual Studio IDE (also a Microsoft product). [3]

2.2 discussion

There are differences and similarities between all these options, so as to give them all fair discussion we will discuss the pros and cons of each. Hyrax servers have the obvious pro of being already implemented with the OPeNDAP protocol, allowing less time being allocated to creation of a request processor. The con's of Hyrax are that the current implementation is complicated and a lot of time would be spent having to understand the current systems. Apache has the benefit of being a widely adopted server type, and as such there is plenty of documentation on how to use it correctly. The con of Apache being that it is also difficult, and if done so incorrectly can create security flaws. ASP .NET has the pro of already being usable an in IDE which makes it easier to debug and test. But this comes with the problem of ASP .NET being only usable on Windows Servers, which from the previous section was decided to not be used.

2.3 Conclusion

The obvious benefits of Hyrax make it the clear winner. There will have to be resources dedicated to understanding the architecture that OPeNDAP created, but this is far less time then making a different handler. Not to mention that if the CovJSON handler had a different handler than the request of other data formats that are already implemented it would fragment CovJSON and make it less likely to be adopted.

3 Technology: Coding Languages

3.1 Overview

It is well known among programmers that all coding languages have their own benefits and quirks. Languages are robust but some are specifically better for some things than others, for example JavaScript is almost universally used for web design but would not be used to manipulate data in a server, although it is used as a web server on sometime services. For the case of OPeNDAP there are three options for what language to use, those are Java, Python, and C++.

3.1.1 Java

Java is an object oriented coding language that is robust in the sense it can be implemented on any platform. Java is popular in client-server connections generally on the server side. In the case of OPeNDAP it is used within the Hyrax framework. The popularity of Java can be seen in its use in the Android SDK, the most widely used mobile operating system in the world. [6]

3.1.2 Python

Python is an interpreted coding language that is useful in teaching and implementing non-time dependent code. That is not to say that python cannot handle time, more so the interpretation of the code can cause slow run times. Python is useful in that it is easy to share libraries and modules between users. This means that if someone has written a function that you need you can look for it and implement it quickly, rather than having to rewrite it yourself.[7]

3.1.3 C++

C++ is similar to Java in that it is an object oriented coding language that works on most hardware. C++ is an extension of the C coding language which makes up most all operating systems today (besides windows). C++ is a standard in many systems and is ideal for data server manipulation, it makes it easy to interact with the SQL data coding language. [4]

3.2 Discussion

Again we will go into a discussion of the pros and cons of each language. Java has the pro of being what is currently implemented within the Hyrax on the OLFS, and being object oriented it is easier to work with request information. The Con of Java is that it doesn't work well when interacting with data sources. Python has the benefit of the CovJSON interpreter already being implemented within it, however python is slow and doesn't work well as a server. C++ has the pro's of being fast, object oriented, and manipulating data from a source. However it requires low level computer science knowledge which can make it difficult to code in.

3.3 conclusion

C++ is the best choice in this case, as it is the required language based on the prerequisites for the project. C++ has many advantages, namely its ability to manipulate and call SQL. This project is based around the manipulation of data, so this feature is imperative.

REFERENCES

[1] Amazon ec2, 2017.

[2] Apache http server, 2017.

[3] Asp .net mvc, 2017.

[4] C++, 2017.

[5] Hyrax data server, 2017.

[6] Java, 2017.

[7] Python, 2017.

[8] Server (computing), 2017.

[9] Adam Cady. Linux servers vs. microsoft windows servers, Jan 2017.