CS Senior Capstone Tech Review

November 21st, 2017, Fall 2017

CDK Global: No more touch. No more keyboard. Bring it All Together. Using Technology to Teach Humans.

Prepared for

CDK Global

Prepared by

Group 9

Look Boss, No Hands

Brandon Dring

Signature Date

Abstract

Creating a project that spans across multiple mediums can be a daunting task; technologies used within cloud server work, network communication and virtual reality environments. Which makes choosing the proper technology for each role such a crucial planning step. In this document we will weigh the pros and cons between multiple technologies used in each field, how they compare and why we chose them.

CONTENTS

1 Introduction

1.1 Role

My role in the group is more or less the lead, I communicate most with the client, schedule the meetings, design and plan the project/sprints. I also create all the skeleton stuff to be used, such as the AWS server, development methodology, VR environment, etc. Im one of two members in my group that has a PC powerful enough to even run VR. I try to provide help to my teammates whenever needed as well.

1.2 Goal

Overall, the goal of this project is to incorporate a method for a user to be immersed in data, rather than just seeing it on a computer screen. More specifically, the user shouldnt have to use traditional means of computer input at all. A user should be able to put on a VR headset, and tell a voice assistant what they want to appear in their VR environment. Then, simply have graphs, maps and charts appear in 3D right in front of them. Being in VR, they should then be able to virtually go out and touch and move around the data to refine or examine the data further. Or, if they so choose, continual ask the voice assistant to modify the data and their environment to get to the desired information.

2 Server Side / Alexa scripting

2.1 Overview

The server is the unsung hero of this project, while everyone might be entertained by the voice assistant use, and the VR immersion, the server is what really ties it all together. For any interaction to come from the virtual reality headset, or the voice assistant, the computing and instructions have to be done on the server in the cloud before being sent to a local machine to be played out.

2.2 Criteria

- Server side scripting language should be flexible and robust enough to be used everywhere. From local machines, to cloud use easily with minimal setup.
- Small learning curve.
- Vast ecosystem of modules to install easily, instead of having to code our very own utility modules.
- Support in the AWS Cloud
- Proper documentation and guides regarding any packages used.
- Strong community support to answer any questions or problems encountered.

2.3 Node.JS

Node.JS is a server side scripting language implemented in Javascript. Once reserved for strict web client use, Node.JS has been the pioneer in the JavaScript everywhere mentality. Node.JS is a relatively new framework in JavaScript and has been around for roughly 11 years [1]. Since it was introduced it has spread like wildfire, largely thanks to the speed of the underlying engine V8, the same thing that is used for the Chrome web browser [2]. Node.JS abstracts away the low level details naturally with the Javascript language. In just a few short years, Node.JS has went from a pet project of a single developer to being used to high profile sites like Netflix, Microsoft, and Yahoo![3].

One of the big pluses of using Node.JS, is that it comes with its own package manager, Node Package Manager (NPM), which is incredibly simple to use. There are currently about 500,000 NPM packages[4] that do everything from building a HTTP server, to using single page application frameworks such as ReactJS. Most things in Node.JS are elegantly abstracted away from the developer; instead of worrying about the boilerplate code that needs to be written to create a Http server, you merely call the package itself and start your server. Node.JS is also fantastically scalable, thanks to its non blocking I/O system and its event driven architecture. So under heavy workloads, it doesnt bog down and ruin the end user experience. On top of all that, there is superior documentation for running JavaScript servers on AWS, more specifically the Lambda section in which the Alexa computations rely on.

2.4 Java

Java is a programming language created by Sun Microsystems in 1995, but now owned by the Oracle Corporation. Java is probably one of the most battle tested languages in the world, being used in online banking, e-commerce, and databases. It is also the most popular programming languages in the world[5] for a variety of reason. The main being it is entirely Object Oriented, runs on all machines, and its secure.

However, the Java language due to its Object Oriented methodology, is also very verbose which makes readability difficult. To do simple things, such as access a value in a list, one has to wade through object notation, often a lot of it, producing lengthy code. Getting Java setup on a local machine for the first time is also a hassle and can lead to hours of troubleshooting just to run something simple. It also comes with the overhead of having to compile the language before running it. Java is not the simplest compile process either, as it comes with a complex directory structure, byte-code files, and a virtual machine to actually the code.

2.5 Python

Python is an open source, general purpose programming language used to do anything from server side scripting, statistics, or scientific computing. It was created by Guido van Rossum in 1991 as a fun project and has become one of the top programming languages today[5]. To this day it is still managed by Guido on the side, along with community and foundation development. Python is known for being one of the more english-like programming languages[6] as far as readability is concerned.

The obvious benefit of Python is the small learning curve that it naturally has, meaning that the team could both use it, and learn it relatively quickly. Python is the middle ground between being less Object Oriented like Java is, but more so than NodeJS. There is also a large number of packages for Python, a vibrant community, and is easy to set up. However, Python is also known for being pretty slow in terms of computing speed, due to the fact that it is an interpreted language not being carried by an underlying runtime engine like Node.JS.

2.6 Discussion

Node.JS in comparison to both Python and Java, is a perfect middle ground. It is faster than Python and while marginally slower than Java, more readable than Java, but less so than Python. Again, in contrast to the others, Node.JS has the largest community, best documentation, and number of easy to use packages. While the AWS and Alexa infrastructure is also specifically set up for the JavaScript language.

2.7 Conclusion

For this project, we chose Node.JS due to largely the reasons listed above. Node.JS is a middle ground between seemingly all the advantages of Python and Java. Largely though, we are choosing it because there is abundant documentation and guides on how to use NodeJS inside the AWS/Lambda cloud, such that Amazon even recommends it. Another benefit, is that all of us have rather extensive experience in NodeJS, thus, cutting out the learning curve of trying to pick up a language and implement it. Using Node.JS, we dont have to use one language for our server, then another for trying to program the Alexa, we can just use JavaScript everywhere. As noted above, Node.JS is also highly scalable, so any heavy computing will be handled gracefully and timely by the language in the cloud with low latency. .

3 Network Communications

3.1 Overview

The biggest hurdle of our project is merely getting an interaction from our voice assistant to register to our Virtual Reality (VR) headset and populate an environment. In order to facilitate messaging, we need something to connect from our cloud/server to broadcast down to a local machine with means to deliver data, and signal the headset to load or complete an action.

3.2 Criteria

- Support between programming languages
- Ease of connectivity
- Assurance of message delivery/receive
- Easy to read documentation
- Support in the AWS Cloud

3.3 Polling

Perhaps one of the oldest methods of web communication, polling is nothing more than just a client asking another service if anything has changed. Its so simple that it isnt even particularly a technology, its more of a method on how to fetch data.

The bonus of this is that it is incredibly simply to implement and use. There are no libraries required, or anything external for that matter. You simply set a loop to constantly keep asking the network for an update. However, this is an incredibly inefficient way to do things, both on the local and server networks. Each request made to the server has the potential to combine and overload the server.

3.4 Websockets

Websockets are a method of a TCP connection that is part of the HTTP Protocol. With a Websocket, each client that is connected to a server must go through a handshake phase, which identifies the client to the server[7]. After the handshake, the client then has a specific port on the server which it is listening to, and then can receive messages from the server on events.

Websockets are a great option due to the many packages that are available across a multitude of languages. Websockets also offer the benefit of only sending data from the server to the client when there is a need. Thus we

can programmatically dictate when a local machine will get data from the server. There is also some reliability with websockets, in which as soon as the connection is interrupted, it results to polling and tries to establish the handshake again.

3.5 RabbitMQ

RabbitMQ is a messaging protocol software specifically designed for message brokering. RabbitMQ specializes in the reliability of delivering a message from host to client, which puts messages in a queue should the connection become interrupted. As such, they are also the world leader in use of message brokering at 35,000 products using it[8].

However, RabbitMQ is a little bit too much for our needs with the project. While it might be very secure, extremely scalable, and reliable, it will be overkill for our project.. Using it as a simple message delivery tool to update a local machine when an event occurs, we simply dont need the technical debt that would come with trying to implement it into our app.

3.6 Discussion

Right off the bat, polling is a non starter for our project due to the incredible inefficiency that it naturally has. There was some hope for RabbitMQ, due to its reliability of messages, but with all the extra technical debt that comes with it, it seems unnecessary.

3.7 Conclusion

We will choose Websockets to be used in our project, largely due to the fact that it is the perfect middle ground between the other options. Its more efficient and reliable than polling, but less so than RabbitMQ. There is also better documentation out for websocket packages across the platforms and technologies we are using for the project. The packages that are out for websockets are also easy to implement with little to no setup apart from installing.

4 Graphics Engine

4.1 Overview

The graphics engine is the framework that helps the developer build a 3D video game. With the main job of abstracting away all the math, collision detection, controller input, sounds from the game, etc. The proper graphics engine leads to increased productivity, quick turnaround, and a minimal learning curve.

4.2 Criteria

- Packages/Marketplace
- Learnability
- Low overhead (Memory, CPU, etc.)
- Support for HTC Vive
- Drag and Drop UI

4.3 Unreal

The Unreal Engine is the most successful video game engine used in the industry today[9]. Written in C++, Unreal features a large degree of portability and customizability for games. It was originally used as a game engine for first person shooters, but since has expanded beyond shooters, and is now also used for MMOs or RPG games.

Unreal Engine is cross compatible between operating systems. They also have unique features for game decision logic (eg, drag and drop), and the ability to edit the code and the environment of VR, inside the VR environment itself. Additionally, Unreal has a marketplace of packages that one can install in use in their own projects without the need of creating them themselves. However, Unity is known as a difficult to learn framework, but once the learning curve has passed, it leads to an extremely efficient workflow.

4.4 Unity

Unity is a framework created only 12 years ago, written in C#, with options to use a custom flavor of JavaScript called UnityScript[10]. Unity also is more industry friendly, offering natural support for game consoles and providing an easy to use portability system between platforms and operating systems.

Unity is also seen as easier to learn, due to avoiding the complex nature of C++ used in Unreal. Unity has another perk, being that it has a vast asset store which can be used to download packages to be used within the environment at ease. The HTC Vive headset that we will be using has out of the box support for development within Unity as well. By far the biggest benefit of using it, is that is easy to develop cross platform games.

4.5 CryEngine

CryEngine is a framework that is written in both C++ and C#. CryEngine is another game engines that was originally designed for first person shooters, and has since expanded across all genres. One interesting perk of using CryEngine is that is in the Amazon cloud already, just under the name Amazon Lumberyard[11].

CryEngine, like Unrea,l is known to be difficult to use. However, it is also widely regarded as one of the most powerful engines. CryEngine allows users to create some of the best graphics in the industry. Ultimately, the hallmark piece of the CryEngine is that it allows the developer to jump straight into the design and environment from the text editor.

4.6 Discussion

In comparison to both Unreal and CryEngine, which use C++ for scripting, Unity has their API written in C#. Unity is also naturally supported on the HTC Vive without having to jump through extra hoops to get a non-standard game engine to run through. While the Unreal and CryEngine are known to be more powerful, giving greater flexibility in custom graphics, it does come at a steep learning curve, while with Unity, it is more made for novices and hobbyists.

4.7 Conclusion

Were going to choose Unity because that is what is supported out of the box for the HTC Vive. Unity also has the advantage of being written in C#, which connects with the .NET framework, which supports Websockets as listed above. Utilizing Unity will make the communication aspect between the VR environment and the server that much simpler, not searching for some third party library in C++ to do the same. Unity also has a hallmark package in its marketplace, which helps novices implement VR graphics and interactions abstracting away all the setup and code needed[12].

5 References

1) http://blog.training.com/2016/09/about-nodejs-and-why-you-should-add.html
2) http://mashable.com/2011/03/10/node-js/#1nICIdEHQkqA
3) https://www.quora.com/What-companies-are-using-Node-js-in-production-in-Texas
4) https://www.npmjs.com/
5) https://www.techworm.net/2017/03/top-20-popular-programming-languages-2017.html
6) https://lifehacker.com/five-best-programming-languages-for-first-time-learners-1494256243/1497409477
7) https://www.html5rocks.com/en/tutorials/websockets/basics/
8) http://www.rabbitmq.com/
9) https://www.gamedesigning.org/career/video-game-engines/
10) http://wiki.unity3d.com/index.php/UnityScript_versus_JavaScript
11) https://kotaku.com/amazon-releases-its-own-game-engine-for-free-1757995787
12) https://vrtoolkit.readme.io/