Groundstation: Technology Review

Team #25
High-Altitude Rocketry Challenge
Natasha Anisimova
Terrance Lee
Albert Morgan

Abstract

The *Groundstation* software package will collect telemetry from a rocket while is in flight and graphically display the telemetry in real-time. Groundstation is made up several different components regarding the data received: collection, storage, interpolation, and display. This document will examine nine different components of the system. For each of these components, three different technologies will be described and evaluated for use in this component. Finally, a recommendation will be made about which technology should be used.

CONTENTS

I. Web Server

This section will describe the various options for the web server. Author: Albert Morgan.

A. Apache

Apache is the most popular web server option, with over 50% of the market share [1]. Apache uses a powerful and flexible module system that allows for a variety of different behaviors [2]. Each connection to an Apache server spawns a new thread or process, depending on which modules are used. This feature has advantages and disadvantages. Because each connection spawns a new thread, if the thread serving one connection crashes, it will not affect the other connections. However, because each connection spawns a new thread, the overhead for that thread must be paid for each new connection.

B. NGINX

NGINX is 2nd most popular web server on the Internet [3]. Web servers such as Apache that use a one-thread-per-connection model can have problems with large number of simultaneous connections, an issue known as the C10k problem [2] [4]. NGINX solves this issue by having one thread serve many connections. This approach reduces the overhead caused by spawning so many threads, and makes NGINX very lightweight and scalable [5].

C. Lighttpd

Lighttpd, pronounced “lighty”, is a web server that is optimized for low memory footprints and fast response. It is optimized for serving a large number of concurrent keep-alive connections, such as serving high-volume AJAX driven web sites [6]. Many web sites use lighttpd for static content delivery while using another web server, such as Apache, for dynamic content [7].

D. Conclusion

All three web servers listed above have advantages for this project. Apache’s one-thread-per-connection approach increases stability. NGINX and Lighttpd are both lightweight, which is important for running an application on a low-power system like a Raspberry Pi. However, Groundstation does not have to support more than 20 concurrent users, so the memory and processor savings for choosing a lightweight option will be minimal. Additionally, stability is a primary concern of the High-Altitude Rocketry Challenge. For these reasons, we recommend Apache.

II. Web Backend

This section will describe the options for the web backend. Author: Albert Morgan.

A. Node

Node allows web backends to be written in JavaScript using V8, the same JavaScript engine that powers Google Chrome [8]. The client side of the web application will certainly use JavaScript, and writing the backend in JavaScript as well will reduce the overhead of learning and programming in new languages. Additionally, because JavaScript would be used on the front-end and backend, it may be possible to reuse code between these two systems.

Node uses an event driven architecture, and is built to work with JavaScript Object Notation (JSON) data. With Node, is it easy to create a web application that response to small events and sends small bits of data to the client [9]. Because Node uses JavaScript, it integrates naturally with JSON.

Benchmarks show that Node is over 5 times as fast as PHP, even when using just-in-time compilation [10].

B. PHP

PHP has been around the web since 1994 and is used in over 80% of web sites today [11] [12]. Given its maturity and wide adoption, there are a lot of packages, documentation, and example available for developers. PHP’s primary advantage is the ease in which is can be incorporated into an static webpage. Using PHP tags, HTML and PHP code can be intertwined seamlessly.

C. Ruby

Ruby is an open source programming language designed for simplicity and productivity [13]. It is well-known on the web as part of the framework Ruby on Rails, which allows for embedded Ruby to be inserted directly into HTML [14]. However, compared to PHP, Ruby on Rails has a steep learning curve. Ruby’s syntax can be hard for a beginner to read, and it draws some concepts from functional programming, which can be difficult for a new user [15].

D. Conclusion

Node is fast, which is important for the Groundstation server, which will run on a Raspberry Pi. Node also uses an event driven architecture, which is perfect for our software. Data will be received at regular intervals and need to be pushed out the clients quickly. For these reasons, we recommend using Node.

III. Logging

This section will describe how the telemetry will be stored. Author: Albert Morgan.

A. Raw data

The simplest logging method is to save the raw data stream to a file as it comes in. This method would be very low cost, in terms of development time, processing, and memory usage. The data would simply be read from the serial port and saved to a file, where it could be reconstructed at a later time. Saving the data at this point, before any processing is done, will reduce the chance of any corruption.

B. JSON

JSON is a data format that is easy for both machines and humans read and write. The telemetry may be stored after it is transformed from its raw form from the serial port to JSON. The server will already be converting the raw data to JSON in preparation to send it to the client, so there will be no additional overhead in the conversion. However, it is likely that the JSON data will take up more space than the raw data due to additional formatting, so it will take up more space in the storage media [16].

C. Database

Storing the telemetry in a database as it comes in will allow for fast random-access reading and writing of the data. This would allow the server to easily grab a certain subset of the data. For example, the server could send the client only data that was generated in the last 2 minutes, or data that comes off of one particular sensor. Data could easily be retrieved using queries in a language such as SQL.

D. Conclusion

Because the system only needs to store data from a single launch, the size of the data set will be relatively small. The overhead of sending all of the data would be minimal, so the benefits of using a queryable database would be small. In the event that the system goes down, it would need to start back up again and begin operation very fast. Storing the data in JSON format would allow the data to be quickly loaded back into memory without significant processing. Additionally, if the data is stored in JSON format, the server would not need to keep the entirety of it in memory to serve it to a connecting client, freeing up resources on the Raspberry Pi. For these reasons, we recommend that the data be logged after it is converted into JSON format.

IV. Data Visualization

This section will describe the various options for the data visualization. Author: Natasha Anisimova.

A. 2D-Representation

2-dimensional representations of maps are usually equal in the x and y dimensions [17]. Graphs and charts are representations that most users are comfortable with, but they can also be a little dull. For displaying the location of a land-bound vehicle or person, a 2D graph is useful [18]. The ground can be viewed as a 2D plane [19]. With a rocket the altitude, latitude, and longitude has to be taken into account and displayed. The most important aspects of the software project are to find the rocket's location in the end and to be able to tell if it reached the team's one-hundred-thousand-foot goal.

B. 3D-Representation

2-dimensional topographic maps are hard to read if one has not had a lot of experience with them. Considering that most of the team will be mechanical engineer,s and no earth science majors will be participating, a 3-dimensional map makes more sense. Cartographers in the past decade have shifted to creating 3D perspective maps. These types of maps are more expensive and time intensive to produce. The extra time that is put into the visualization makes it more appealing to the readers [20]. The 3D representation is easy to visualize and understand [21].

C. Numerical Display

A numerical display is often used when looking for the longitude and latitude of a specific point on a map. It would also be useful when displaying the highest altitude of the rocket [22]. The numerical display is a number that would constantly be reviewed and needed. A graph can display the data, but it would still take a second to get that information. Having a table will show all the most crucial aspects of the project with the values would make the webpage much more usable.

D. Conclusion

3D visualization for the location of the rocket as well as a small numerical display is the best way to go. Professor Mike Bailey has created software that can create and extract the topology of a certain area, making it much more manageable. The High-Altitude Rocketry team will only need the topology data of Lucerne Dry Lake area of the Mojave Desert. The images for the area can be pulled from NASA's website, and from there they can be texture mapped to the topology. The numerical display will be used to give a more accurate and direct way of giving the data to the users.

V. User Interface Interaction Model

This section will describe the various options for the user interface interaction model. Author: Natasha Anisimova.

A. Unity

Unity 3D is an engine, not so much an application. Groundstation software is going to be used by about thirty to forty people, if not more, and most of them would not have the Unity3D Web-Player installed. Unity is a game development tool, and since this project is focusing on just creating a useful visualization of the data that is collected, Unity is not suitable for this particular case [23].

B. OpenGL

OpenGL has one of the fastest languages to develop with, C++. Double buffering also solves any problems with the program being in the middle of plotting data and the user getting a black screen. Instead it shows the old frame until the new one is ready to be displayed. Unfortunately, OpenGL does not run on the web [24].

C. WebGL

WebGL is the Javascript API that allows you to create 3D graphics in a browser. Three.js is a tool used on top of WebGL, making it easier to create those 3D graphics. Three.js is used when creating more complicated 3D scenes (textures, models, and rendering). Several different types of browsers support it as well, from Desktop Opera to Firefox and Chrome. WebGL is also the easiest 3D application to use, as well as the best tested and documented [25]. Performance-wise it is also very fast even though programs are written in JavaScript.

D. Conclusion

WebGL is the most fitting choice to go with for this project. OpenGL is not supported by web browsers, while Unity is not as widely readily supported by web browsers as WebGL. The WebGL application can be used while still having the comfortable environment of the web development environment that HTML and JavaScript have to offer [26]. While there is a high learning curve, there is plenty of support for it at OSU as well as information on the web. The only major downside is that Internet Explorer, before 2013, does not support it, which is used by few and far in between [27]. WebGL is supported by mobile devices if the software is expanded to them in the future.

VI. User Interface Organization

This section will describe the various options for the user interface organization. Author: Natasha Anisimova.

A. Paper Prototypes and User Interviews

In order to understand what are users are most comfortable with, it would be most cost effective and time efficient to create paper prototypes [28]. A small sample of the High-Altitude Rocketry Team could be taken through a user study where we would try and figure what would work. Feedback would be taken then and there. In some cases the feedback could be implemented quickly into the paper prototypes so that the user could see how it would affect the layout.

B. Flight Simulator Displays

A lot of flight simulator games efficiently display the information of the planes' or rockets' location, altitude, velocity, and status. Simulators are often able to do this with various dials on either side of the screen, much like a dashboard of a car. Having a recognizable display for the users would greatly reduce any misunderstandings that could occur [29]. Units and dials would be labeled for clarity. Colors would be used to help emphasize important goals and perhaps how close they are to being reached.

C. Finnish Astronautical Society

The Finnish Astronautical Society has launch several different rockets since 2011 and successfully displayed the data that they have received from the rockets while they have been in space. As mentioned on their website, most of the data is streamed to a ground station in real-time using a radio link and then from there onto an Internet server [30]. Their solution to the problem of keeping track of their rocket is very similar to what we have planned, except that they have a few different sensors and equipment. They have considerably more information about all their launches on their website, making it a bit cluttered, but the content of their website is something that would be useful to follow.

D. Conclusion

For the organization of the user interface it would be useful to pull inspiration from websites or implementations that already exist but also have the freedom of easily changing it and being creative. The paper prototypes would be useful for making sure that the software is easily tailored in the early stages of the web page. We have currently been pulling user stories from the team in order to know what would be most important for them to see on the web page. Continuing with that trend, we can continue to create the webpage with their feedback. This would also ensure that they are not shocked or bewildered when seeing the final product before the launch date or on site.

VII. Retrieval

This section will compare programming languages for retrieving telemetry from the serial port. Author: Terrance Lee.

A. C

The C language has an extensive library which has many choices we can use. The main one for the sensors is the extensive math.h library. It allows us to do any math functions. Plus we can do error conditions in our code if needed. When it comes to references, C has many. Since C has had a couple of major programs, for example, Unix was rewritten in C in 1973 many people and companies have used C [31]. This has allowed many references including the Internet, books, and instructors. C's disadvantage is that it is a very low level language and can be security flawed.

B. C++

The C++ library is just as extensive as the C library except that it has made it so that it can be supported in an object oriented programming language. For example, it has to bring out a cmath library so it works with C++. C++ is not as old as C. For example, in 1998 an ANSI/ISO standard for C++ was adopted [32]. It is a favorite to many because of its object oriented programming (OOP). Also, since it is OOP it is easier to write than C, but you have a lot more to learn. That is its disadvantage: no developer can use its entire set of code that C++ provides. It would take an extremely long time. Just like C, there are many resources for C++ out there if we run into a problem.

C. Python

The Python library is very extensive. The new Python version 3.6 has about 37 main libraries which is sub-sectioned so that when you program you can be more specific [33]. The main libraries that we need are the data types, numeric, and the mathematical modules. When it comes to references, it seems that Python users understand that many schools do not have a class for it. For example, when I was researching the libraries it had a reference page for the library. Especially when it comes to the Internet, there are a lot of references to help if you run into issues with coding in Python. Python's disadvantage is that it is not popular. Not many companies have given it a chance yet, and it has not had a big breakthrough.

D. Conclusion

All of these options are great. They all have extensive libraries and have great references. If we run into any issues we have plenty of references online, as well as books and actual people we know that we can connect with and speak to about it. We did some extra research and spoke to instructors and people that have used data collecting software in their career field before. We asked them which language they preferred. We got mixed results when it came to which one was the best language to use. We believe that was because everyone has their favorite language that they prefer.

We decided that the best language to use for the retrieval of data would be C. It meets all of the criteria and beyond. It has all of the libraries that we would need to retrieve the data from the sensors on the rocket. If we lose GPS from a sensor we can still calculate the position of the rocket with those libraries as well. With the amount of resources and references that are available to C, if we have an issue we should be able to find the solution to our problem quickly so that we do not get stalled on it for long.

VIII. Interaction Modes

Support needs to be given for an interaction mode, PC or mobile, so that the High-Altitude Rocketry team can see the data. Author: Terrance Lee.

A. PC

The PC interaction mode is the most available to the teams. It has the largest screen size and it has the most flexibility when coding. The disadvantage with the PC is that it is larger than mobile so the team will have to carry it around. It can use up more power than the mobile too, but the battery can be more robust.

B. Mobile

The mobile mode is also available to most of the people on the team as long as they have a smart phone. The mobile option will give them a lighter option to carry around and will not waste as much power. The disadvantage is that it will have a smaller screen and we have less flexibility when coding. We will not have cell service when we are out at location.

C. PC and Mobile

The third option is using both the PC and mobile. With this option we will have all the advantages of both. We will still have a disadvantage which is that we will have to program in multiple languages and formats.

D. Conclusion

With the PC, once the team gets the information they will be able to save it right there on their team computers and move it easily through USB or other devices. With mobile, our biggest issue is not the programming but the lack of cell service. Without cell service, if the team wants to look at any data they are reliant on the Raspberry PI Wi-Fi only. That is not going to be on all the time, either. If we use both, it has to be as a stretch goal because of the time constraints.

The best option for this series that we would recommend would be to use the PC for the interaction mode. It is reliable, most available, flexible when coding, and has a large screen size. If multiple people are looking at the screen, they are pushing each other out of the way to see it. When it comes to carrying it around, most of us have a computer bag and can carry it around that way. That is not that big of a drawback. With the power issue, we plan on bringing a power generator with us to the launch site, which the power plug for most computers are compatible with. That should not be that big of an issue because we only need to have the computers on for launch and test launch.

IX. Toolkit to generate the User Interface (UI)

For the toolkits to generate the UI, the options are Prototyping toolkits, jQuery UI, and the Dojo toolkit. These are all part of the top ten JavaScript Web UI Libraries, Frameworks and Toolkits [34]. Author: Terrance Lee.

A. Prototype UI

Our first option is the Prototype UI toolkits. Their advantages are that they are reliable and have the functionality for our purpose. The Prototype UI toolkits will allow us to generate the UI we want and has great documentation. The disadvantage is that they cost money. The cheaper they are the less functionality, and the more expensive the more functionality. The best one we found was called Mockplus, but its biggest disadvantage besides money is that it has a lack of responsive design features [35]. One big advantage is that ease of use due to its drag and drop feature.

B. jQuery

The second option is jQuery. jQuery is free to use, easy to use, and has great documentation. One disadvantage with jQuery is that it has limited functionality in some cases [36].

C. Dojo

Our last option is the Dojo toolkit. The Dojo toolkit is under the academic free license version 2.1 so the license is free [37]. Its other advantage is that it is easy to use. Its disadvantages are that it does not have great documentation and functionality for our purpose. Dojo has a great framework, but what we need is something with a good library.

D. Conclusion

They seem to all have good advantages. Prototype toolkits are for more professional work. If you were doing a project and you could afford one of the high-end toolkits they seem like the way to go. For this type of project, jQuery and Dojo have the best advantages functionality-wise for this scale of this project. For the criteria we felt that jQuery was the best way to go, even though Dojo has great framework. We need is a really good library for this project. Also, the team is very familiar with it. The limited functionality is a disadvantage only in some cases. We believe that we should not run into that issue because we just need it for some of its basic features.

X. Web Server Revision

This section will describe revisions to the web server technology. Author: Albert Morgan.

A. Node

Node [8] has a built in web server. Using this web server instead of a standalone server such as Apache or NGINX will allow us to use one single technology stack. Keeping technologies in Node will have several benefits. First, because the rest of the application is written in Node, using a Node-based web server will allow for better integration. Second, keeping the development in Node will allow the development team to work faster. Using a separate technology would mean having to learn how to configure and use that technology, but if everything uses Node, then configuring and integrating it with the rest of the system will be much easier. Finally, using Node instead of another technology means less hassle with installation. Using Node instead of a separate web server technology means that there will be less components for our end-users to install.

B. Revised Conclusion

Because we don't need the flexibility of Apache or the scalability of NGINX, we have decided to use Node for our web server. Groundstation is designed to be installed and deployed by other engineers in an environment which would not have access to the Internet. This means that if anything goes wrong, support may be hard to get. Keeping everything in one technology means easier installation with less steps. If everything is part of Node, installation of our software may becomes as easy as "Install node, download the repository, and type 'npm start'." For these reasons, we have decided to use Node for the web server.

XI. Logging Revision

This section will describe revisions to the logging system. Author: Albert Morgan

A. Comma-Separated Values

Comma-separated values, or CSV, is a simple file format in which a row of data is stored as individual datums separated by columns. This format has many benefits. First, it is easy to read. CSV files can easily be opened, read, and modified by a human with a simple text editor. Second, CSV files are easy to import into applications such as Microsoft Excel. Keeping the logged data in a file format that is easy to work with will make post-launch analysis of the data much easier.

XII. Revised Conclusion

We initially chose JSON to log the launch data. JSON would make it very easy to import the data into a Node object, making it much easier for the web server to work with the data. However, as we thought about the problem more, there were clear problems with JSON. The biggest problem is delimiters. Every JSON object must have a closing curly brace. Without the brace, it is not a valid JSON format. One of the reasons we are logging the data is to recover in the event of an unscheduled program halt. However, if the program halts unexpectedly, it will not have had the opportunity to write the proper closing braces to the JSON file. A CSV does not suffer from this drawback. Considering the ease of working it CSV and the ability to use applications such as Microsoft Excel to work with the data, we have decided to use a CSV file for the log.

REVISION HISTORY

Name	Section Title	Date
Albert Morgan	Web Server Revision	2016-02-14
Albert Morgan	Logging Revision	2016-02-14

INDEX

REFERENCES

[1] (2016) Usage statistics and market share of apache for websites. [Online]. Available: https://w3techs.com/technologies/details/ws-apache/all/all
[2] J. Ellingwood. (2015) Apache vs nginx: Practical considerations. [Online]. Available: https://www.digitalocean.com/community/tutorials/apache-vs-nginx-practical-considerations
[3] (2016) Usage statistics and market share of nginx for websites. [Online]. Available: https://w3techs.com/technologies/details/ws-nginx/all/all
[4] D. Kegel. The c10k problem. [Online]. Available: http://www.kegel.com/c10k.html
[5] O. Garrett. (2015) Nginx vs. apache: Our view of a decade-old question. [Online]. Available: https://www.nginx.com/blog/nginx-vs-apache-our-view/
[6] Lighttpd. [Online]. Available: https://www.lighttpd.net/
[7] Powered by lighttpd. [Online]. Available: http://redmine.lighttpd.net/projects/1/wiki/poweredbylighttpd
[8] Node.js. [Online]. Available: https://nodejs.org/en/
[9] G. Woodfine. event driven architecture node.js. [Online]. Available: http://garywoodfine.com/event-driven-architecture-node-js/
[10] R. Sanchez. Comparing node.js vs php performance. [Online]. Available: http://www.hostingadvice.com/blog/comparing-node-js-vs-php-performance/
[11] History of php. [Online]. Available: http://php.net/manual/en/history.php.php
[12] Usage of server-side programming languages for websites. [Online]. Available: https://w3techs.com/technologies/overview/programming_language/all
[13] Ruby. [Online]. Available: https://www.ruby-lang.org/en/
[14] Getting started with rails. [Online]. Available: http://guides.rubyonrails.org/getting_started.html
[15] L. Teo. (2016) Ruby on rails vs php - the good, the bad. [Online]. Available: http://www.leonardteo.com/2012/07/ruby-on-rails-vs-php-the-good-the-bad/
[16] Introducing json. [Online]. Available: http://www.json.org/
[17] 2d is better than 3d. [Online]. Available: https://www.nngroup.com/articles/2d-is-better-than-3d/
[18] 2d's company, 3d's a crowd. [Online]. Available: http://www.scribblelive.com/blog/2011/12/13/2ds-company-3ds-a-crowd/
[19] 2d and 3d presentation of spatial data: A systematic review. [Online]. Available: https://www.researchgate.net/profile/Matthias_Trapp/publication/268496521_2D_and_3D_Presentation_of_Spatial_Data_A_Systematic_Review/links/546c88d50cf2c4819f217143.pdf
[20] Evaluating the effectiveness of 2d vs. 3d trailhead maps. [Online]. Available: http://www.mountaincartography.org/publications/papers/papers_lenk_08/schobesberger.pdf
[21] 3d represenation for software visualization. [Online]. Available: http://www.cs.kent.edu/~jmaletic/papers/softvis03.pdf
[22] R. Korsara. Data display vs data visualization. [Online]. Available: https://eagereyes.org/criticism/data-display-vs-data-visualization
[23] A. Lian. Enhanced visuals, better performance, and more: The unity 5.4 public beta is ready for you to download. [Online]. Available: https://blogs.unity3d.com/2016/03/15/enhanced-visuals-better-performance-and-more-the-unity-5-4-public-beta-is-ready/
[24] Why you should use webgl. [Online]. Available: http://codeflow.org/entries/2013/feb/02/why-you-should-use-webgl/
[25] Webgl best practices. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API/WebGL_best_practices
[26] P. Cozzi. (2016) Why use webgl for graphics research? [Online]. Available: http://www.realtimerendering.com/blog/why-use-webgl-for-graphics-research/
[27] Webgl - 3d canvas graphics. [Online]. Available: http://caniuse.com/#feat=webgl
[28] What is prototyping. [Online]. Available: http://www.umsl.edu/~sauterv/analysis/prototyping/proto.html
[29] The impact of flight simulation in aerospace. [Online]. Available: http://www.aerosociety.com/Assets/Docs/Publications/DiscussionPapers/The_impact_of_flight_simulation_in_aerospace.pdf
[30] Real-time rocket telementry experiment. [Online]. Available: http://haisunaata.avaruuteen.fi/realtime/2011/
[31] R. Hauben. History of unix. [Online]. Available: http://minnie.tuhs.org//TUHS/Mirror/Hauben/unix-Part_I.html
[32] H. Schildt, *C/C++ Programmer's Reference, Literature*. McGraw-Hill, 2000.
[33] T. P. S. Foundation. The python standard library. [Online]. Available: https://docs.python.org/3.6/library/
[34] S. Deering. (2012) 10 javascript web ui libraries, frameworks, and toolkits. [Online]. Available: https://www.sitepoint.com/10-javascript-web-ui-libraries-frameworks-toolkits
[35] G. Jia. Pros and cons of four prototyping tools. [Online]. Available: https://medium.com/@gracejia/pros-and-cons-of-four-prototyping-tools-dda37d3b21bf\#.w9xtvrzb2
[36] Jquery: Advantages and disadvantages. [Online]. Available: http://www.jscripters.com/jquery-disadvantages-and-advantages
[37] Dojo toolkit license. [Online]. Available: https://dojotoolkit.org/license.html