Groundstation: Technology Review

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

Abstract

Abstract goes here

Here's how to cite something [1]. Check the LaTeXsource. The actual sources are in groundstation.bib, add new sources as necessary. We can re-use this file so we don't have to rewrite it again in the future, so give your sources useful names. Note that if you don't cite a source from the .bib file, it doesn't show up in the references, which is why we can re-use that file.

Pro-tip: put a tilde in between the cite command and the word before it to prevent LaTeXfrom putting the citation on a new line.

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 won't 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, I 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 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 page. 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, I 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 be converting the raw data to JSON already in preparation to send it to the client, so there will be no additional overhead in the conversion. However, itt 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 server it to a connecting client, freeing up resources on the Raspberry Pi. For these reasons, I recommend that the data be logged after it is converted into JSON format

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] 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] The c10k problem. [Online]. Available: http://www.kegel.com/c10k.html
[5] 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] event driven architecture node.js. [Online]. Available: http://garywoodfine.com/event-driven-architecture-node-js/
[10] 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] 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/