Capstone Assessment

Project Introduction

Currently, the focus of this IoT Analytics project will be building a data pipeline template that is capable of collecting atmospheric data from wireless embedded devices. To synchronize the stream of data being collected, the server (hosted from a laptop) will prompt the devices for data every few minutes. The atmospheric data will be sent in real time from the devices to processes running on the server. One process will aggregate and process the data in batches for later viewing. Another process will host a web server that serves a dashboard for viewing the data in batches and in real time. Once this pipeline is created, documentation covering the development process will be made. This documentation will help outline how the pipeline was constructed, what pitfalls were encountered, and what technological alternatives exist.

Project Correlation to Past Coursework and Co-ops/Internships

Though most of the skills being applied to this project were learned on the job, my coursework helped grow and refine these skills. Data Structures (CS2028C) taught me how to implement important data structures such as linked lists, stacks, and queues in C++. Additionally, this class taught me to be comfortable with pointers and dynamic memory allocation. These concepts are very useful in the systems programming and IoT field, where low level languages are typically used due to their low overhead. On the other hand, Python Programming (CS2021) and Database Design & Development (CS4092) prepared me for working with data processing pipelines. The Python Programming class briefly covered data processing packages such as Pandas, Matplotlib, and NumPy. The Database Design & Development class covered database structures and SQL. The skills from both of these classes would greatly help me at my data postprocessing co-ops at Kinetic Vision.

At Kinetic Vision, I mainly worked on data postprocessing scripts written in Python. These scripts would use an object relational mapper (ORM) to find and download HDF data files. These files would then be read into a pandas dataframe. From there, key performance metrics and data plots would be created from the dataframes and uploaded to the cloud. I could use these skills to create Python backend that loads batches of data into HDF files, processes these files, and stores the results on the server. Asides from data postprocessing, I also did some UI development using React JS. I could use my knowledge of React JS to create a web dashboard that displays processed data plots and metrics via HTTP requests to the backend. From there, I could use WebSockets to send real-time data to the dashboard for live viewing.

To collect the data being used by such a pipeline, I could use the knowledge I gained from my internships at the Air Force Research Laboratory (AFRL). During these internships, I was tasked with creating a model plane. The plane's interface used a Raspberry Pi touch screen display to control the various features the model had. The Raspberry Pi would then communicate the input to two Arduinos on the plane. One Arduino controlled the plane's flaps, engine, and weapons. The other Arduino controlled electroluminescent wire, which visualized power running the various aircraft components. Additionally, the Arduinos used MOSFET circuits to control some of the components (such as the weapons). Using my experience with devices such as a Raspberry PI and Arduinos, combined with my knowledge of basic control circuitry and my familiarity with using ZeroMQ for wireless communication, I could to create my own data collection devices. If this proves to be too big of a task, I could then understand how the devices I choose will work at a fundamental level.

Project Motivation

The main motivation of this project is to showcase the benefits of creating a data pipelines in-house. A notable amount of organizations and companies use data acquisition hardware they bought externally from vendors such as National Instruments. However, this hardware is typically closed-source and can be difficult to extend and modify as a result. Data processing software solutions can also have the same issue with extensibility. This is especially true for Microsoft Power Apps, which are rather limited in capability from my experience. Hopefully, this project will show that pipelines made in-house don't have extensibility issues when implemented properly. Completing this project quickly and cost effectively would also help strengthen the benefits of pipelines made in-house.

Another motivation of this project is to showcase new technology that can be used to improve the safety and efficiency of data pipelines. For example, newer programming languages such as Rust to improve the security and efficiency of the software running in the pipeline. The world of embedded electronics has also evolved since I last worked with such devices. Newer and more capable devices could lead to solutions being created with less time and effort.