Wits University

School of Electrical and Information Engineering

ELEN7046 - Software Technologies and Techniques

Big Data Visualization using Commodity Hardware and Open Source Software

Authors:	*Student Number:*
Gareth Stephenson	778919
Matsobane Khwinana	779053
Sidwell Mokhemisa	1229756
Dave Cloete	1573016
Kyle Trehaeven	0602877N

Abstract

The project aims to provide a low cost solution to big data processing problems, enabling a commercially viable option to individuals, small businesses and academia using commodity hardware. This report explores the use of Open Source technologies Apache Spark and Scala as well as the low cost and scalable Raspberry Pi's. After building a prototype system to source, process and visualize data from twitter, it was concluded that

commodity hardware is a viable small-scale solution to working with Big Data.

June 30, 2016

Declaration of Originality

Contents

1 Introduction

This report presents the work done by Group 2 in response to the project brief for ELEN7046: Software Technologies and Techniques.

The report will broadly focus on the following topics:

- The Methodology followed to execute the project;
- The Architecture of the solution developed for the project; and
- The different technologies used to deliver the solution.

1.1 Problem Statement

There is an abundance of big data available to individuals and companies with very limited capacity to refine this data into meaningful information, particularly for small scale endeavours. Big Data processing is often locked behind high cost barriers to entry, and individuals, start ups and academics may find it difficult to be involved in Big Data processing.

1.2 Solution Summary

Commodity hardware is available to provide a means by which the barrier to entry for Big Data projects can be overcome. Simple, low cost components can be leveraged to address each of the parts of a big data processing solution, whether it be data sourcing, transormation, or visualization; and can be scaled according to needs or as required.

1.3 Approach

2 Literature Review

2.1 Data Sourcing

This project was intended to deliver a system or solution for the visualization of Big Data. For this reason it is important that we start by defining what Big Data

is.
According to Maden[1], Big Data can be defined as ”data that is too big, too fast, or too hard for existing tools to process.”
The above definition further supports our group’s decision to focus on Twitter as the source of data for the project. Statistics have shown the following average figures with regards to the amount of data that one can get from Twitter[2]:

- 6000 per second;
- 350 000 per minute;
- 500 million per day; and
- 200 billion in a year.

2.2 Big Data Processing Algorithm

The team sought to have all the processing of the Big Data received from Twitter done on multiple nodes following the principles of MapReduce.

Apache Spark was used for this project together with MapReduce which in the main delivers the MapReduce functionality based on the algorithm that is broken down into the Mapper class and the Reducer class[3].

3 Lifecycle Methodology

In order for the team to successfully deliver this project, a development methodology based on IBM Rational Unified Process (RUP) was followed albeit tailored to cater for the specific needs of this project.

The diagram below depicts the IBM RUP model:

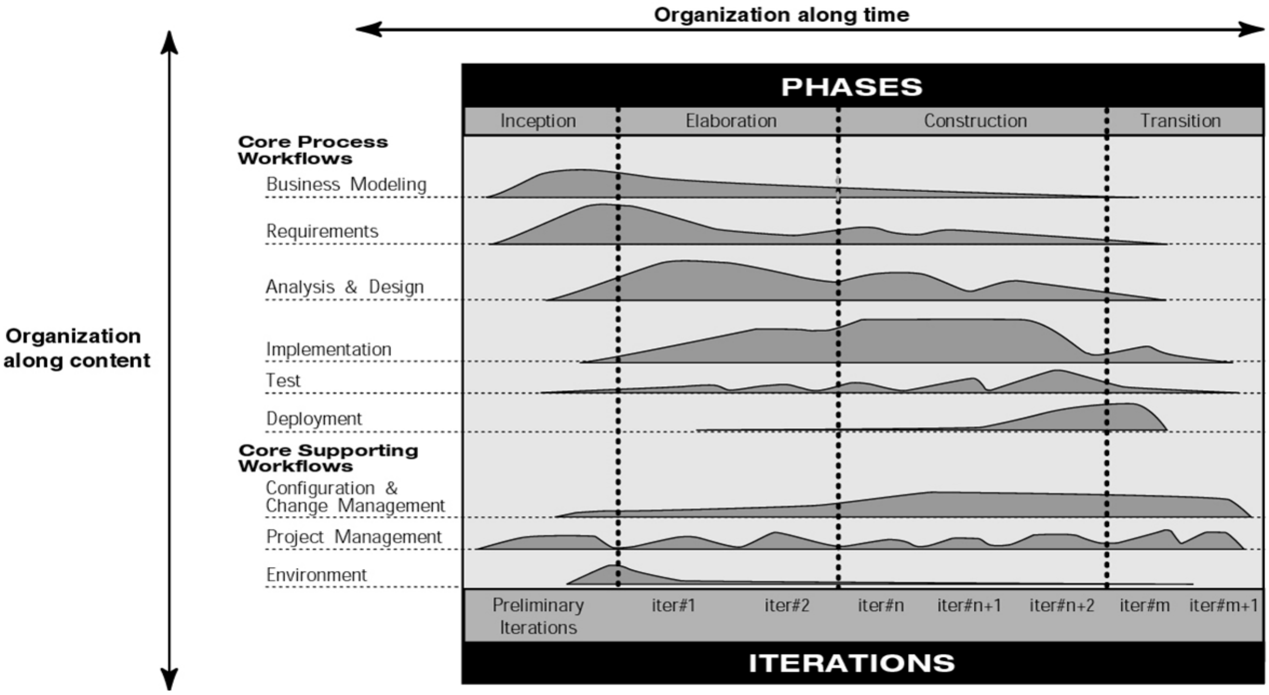


Figure 1: IBM Rational Unified Process (Source: RUP, Best Practices for Software Development Teams)

4 Assumptions and Constraints

4.1 Tweet Locations

The group encountered tweet location issues.

4.2 Pros and Cons determinations

Rudimentary algorithm for determining Twitter statements (tweets) that are against or for a particular candidate was adopted...

5 Design Decisions

The table below details all the key design decision made in the delivery of the solution:

- **History Data:** History data/ batch interface to Twitter was designed to provide past data subscribed to based on date range.
- **Streaming Data:** This interface provides for all subscribed data in real-time starting from now and going into the future.
- **GoogleMaps:** Tweet location was derived from the co-ordinates found in a tweet where the person tweeting enabled location services on their device to provide current location information. Where privacy settings were not enabled for current location, an interface to GoogleMaps was developed to read the location information of the person tweeting from their Twitter profile and resolve this to actual co-ordinates.
- **Security Directive(s):** It was decided that where security is concerned, only the Twitter and GoogleMaps security requirements be adhered to where integration is concerned. All data acquired from Twitter is data that is already in the public domain, therefore no effort was required to secure the data while in transit, hence the use of only FTP instead of a lot more secure transport mechanism.

6 Success Criteria

- Build a system that uses commodity hardware to solve for big data problems;
- Provide a solution that covers the data sourcing, transformation and presentation of social media data from Twitter relating to the United States and South African elections. The end result must be visualization that provides insight to the sentiment of election candidates on Twitter.

7 Solution Design

7.1 High Level Design: Component Architecture

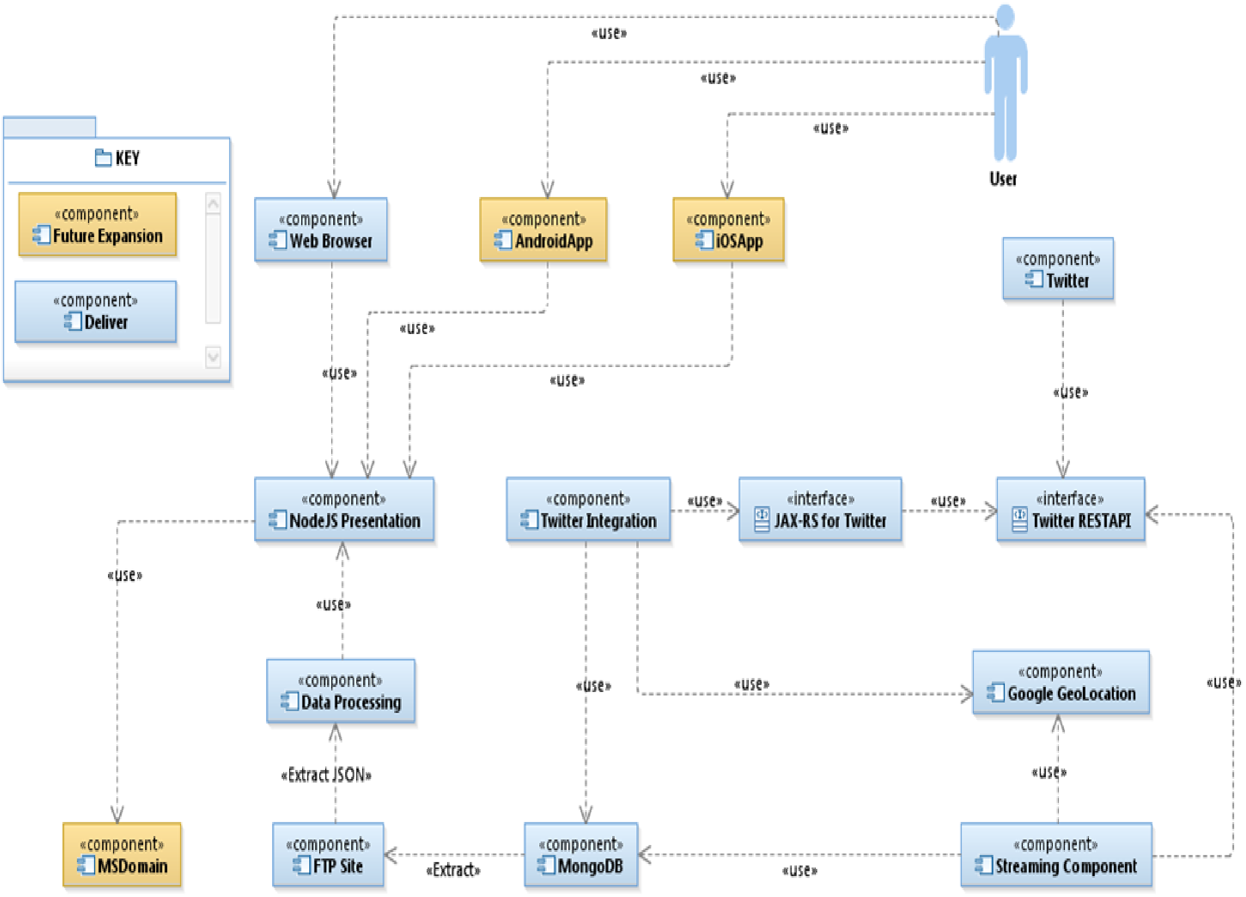
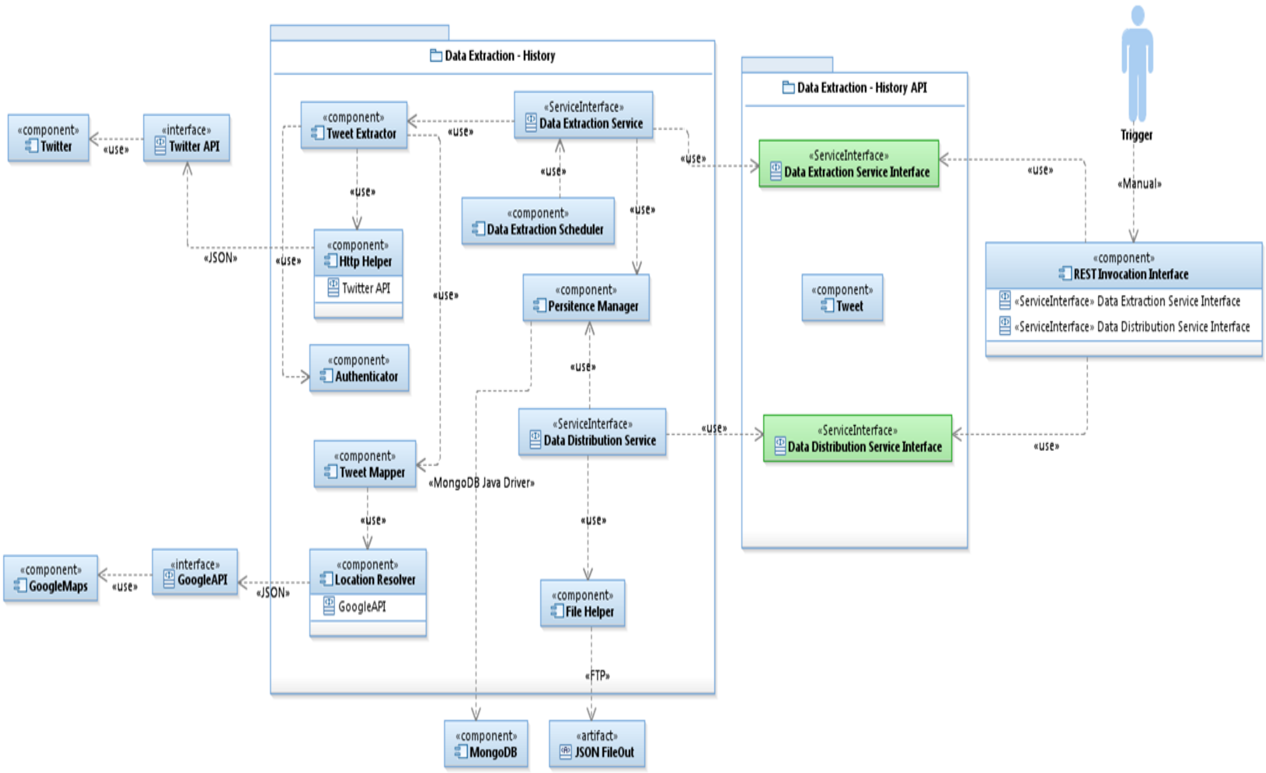


Figure 2: High Level Component Model.

7.2 Detailed Designs

7.2.1 Data Acquisition (Batch)



Data Extraction - History
Data Extraction - History API
«component» Twitter
«use»
«interface» Twitter API
«JSON»
«component» Tweet Extractor
«use»
«ServiceInterface» Data Extraction Service
«use»
«use»
«component» Http Helper
Twitter API
«use»
«component» Data Extraction Scheduler
«use»
«component» Authenticator
«use»
«component» Persitence Manager
«use»
«ServiceInterface» Data Distribution Service
«use»
«component» Tweet Mapper
«MongoDB Java Driver»
«use»
«use»
«component» GoogleMaps
«use»
«interface» GoogleAPI
«JSON»
«component» Location Resolver
GoogleAPI
«component» File Helper
«FTP»
«component» MongoDB
«artifact» JSON FileOut
«ServiceInterface» Data Extraction Service Interface
«component» Tweet
«ServiceInterface» Data Distribution Service Interface
Trigger
«Manual»
«use»
«use»
«component» REST Invocation Interface
«ServiceInterface» Data Extraction Service Interface
«ServiceInterface» Data Distribution Service Interface

Figure 3: Component Model: Streaming Module

7.2.2 Data Acquisition (Streaming)

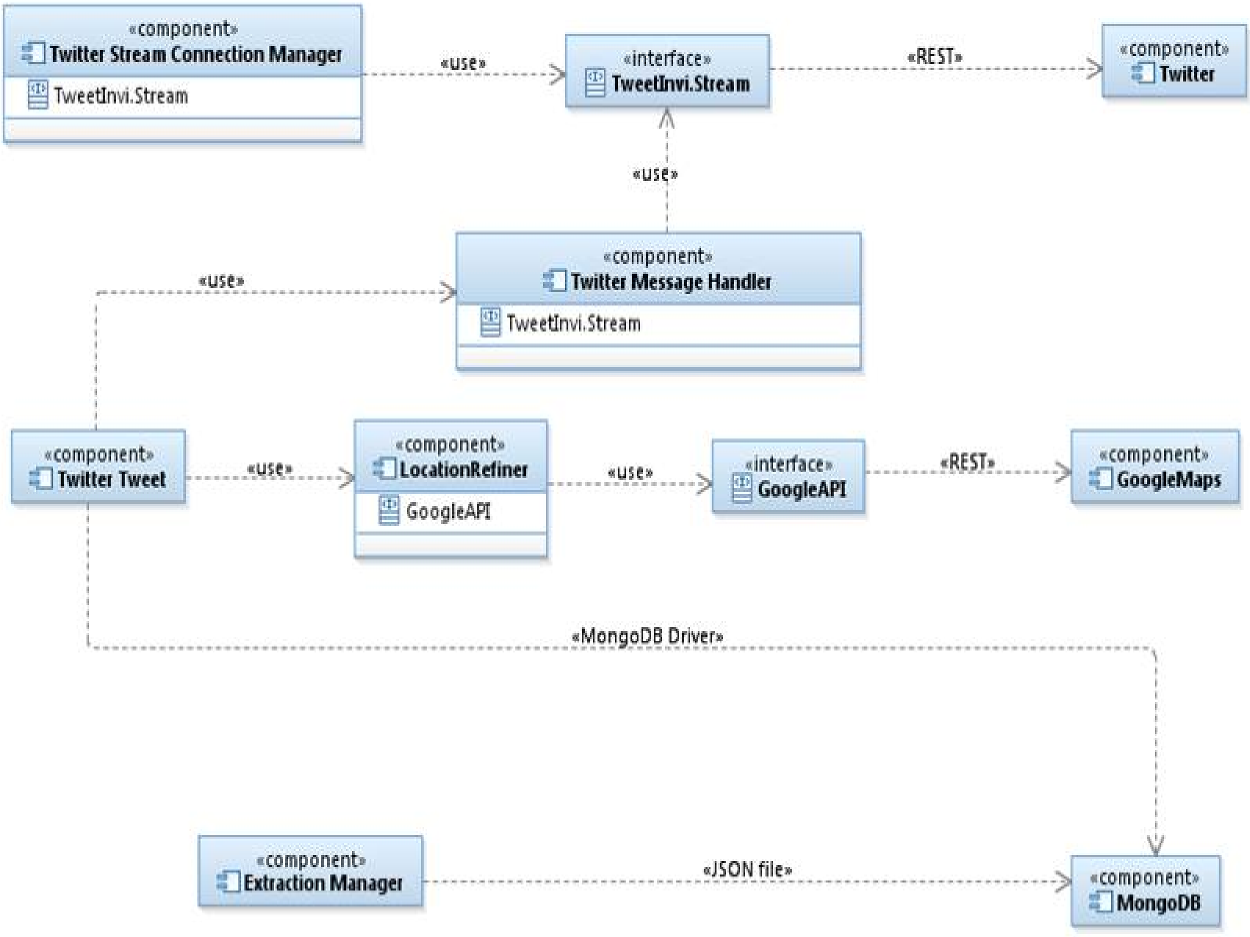


Figure 4: Component Model: Streaming Module

7.2.3 Data Processing

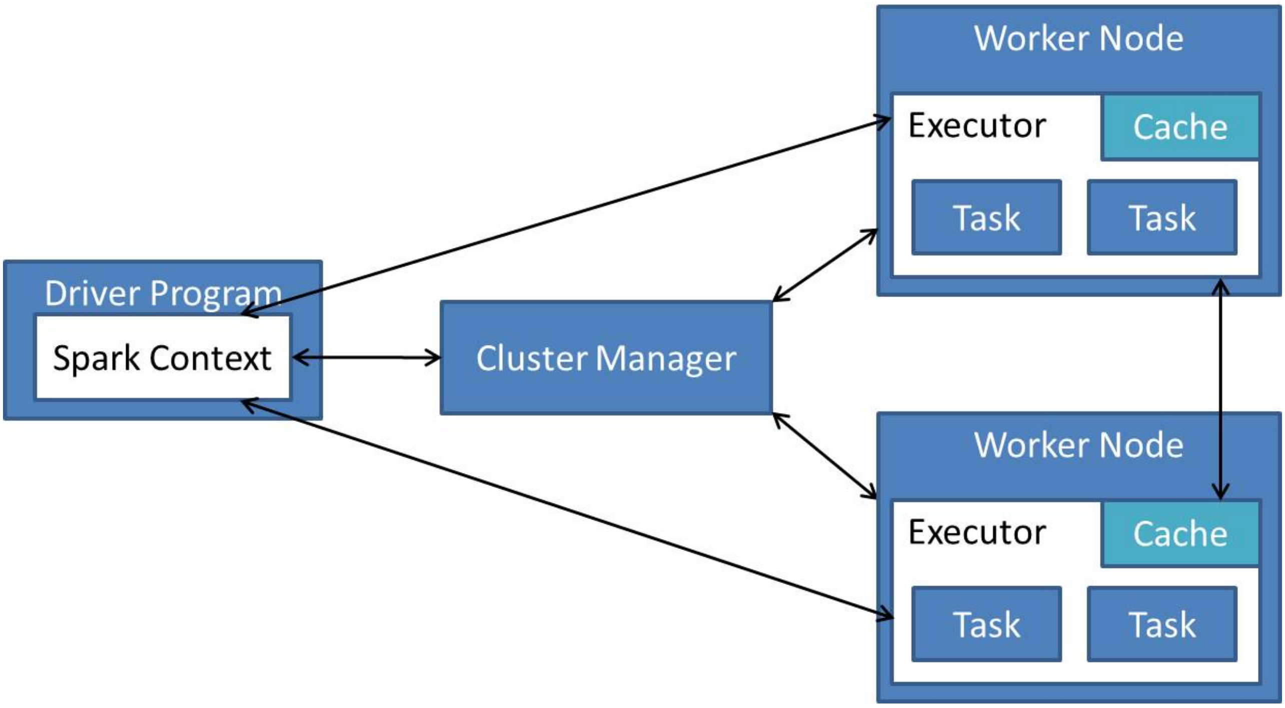


Figure 5: Context Diagram: Data Processing

7.2.4 Data Visualization

7.3 Operational Model: Infrastructure Design

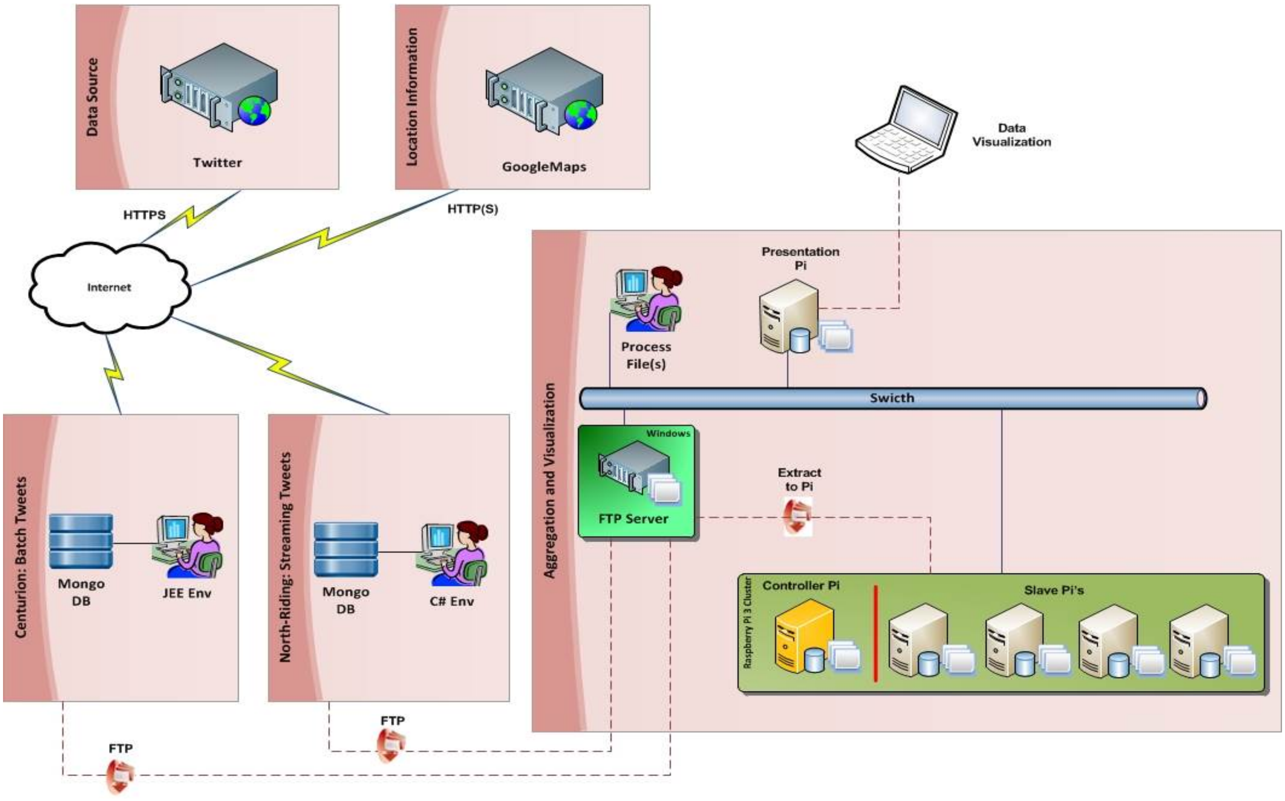


Figure 6: Operational Model: Phyical

7.4 Possible Extensions

- **Personnel Shortfall:** Inexperience in the management team is a potential risk, due to the possible oversight and inaccuracies . Leach does not have sufficient IS Management experience*(pg 502 paragraph 4)*, the project may suffer if leach continues at his current position

8 Conclusion

All this hardware and software is available to anybody interested in Big Data processing.

The hardware is cheap and the software is free.

The learning curve in the beginning can be quite steep but is ultimately very rewarding in terms of what can be achieved with so little financial investment.

References

[1] . S. Madam. From Databases to Big Data. IEEE Computer Society, 2012.

[2] . V. Kumar, R. Yuvaraj, C. Anusha. Effective Distribution of Large Scale Datasets Clustering Based on MapReduce. 2016.

9 Appendices