Big Data Research Project

Vehans Ayvazi

MedVoice Biotech Internship

Table of Contents

A. Introduction

1. Use Case

B. Key Terms

C. Lambda Architecture

2. Batch Layer

1.1 Apache Hadoop

1.1.1 HDFS

1.1.2 MapReduce

1.1.3 Ecosystem

- HBase
- Pig
- Hive
- Zookeeper
- Flume
- Sqoop
- Oozie
- Yarn

1.1.4 Hadoop Distributions

- Cloudera
- Hortonworks
- MapR

1.2 Data lake

1.2.1 Azure Data lake

1.3 Data Warehouse

3. Speed Layer

2.1 Event Stream Processing

2.1.1 Azure Event Hubs

2.1.2 Apache Kafka

2.2 Apache Storm

2.3 Apache Spark

4. Serving Layer

3.1 Visualization

3.1.1 Tools

- Power BI
- U-SQL
- Tableau
- Apps
- Excel
- MATLAB

D. Conclusion

E. References

1. Tutorial Recommendations

2. Citations

A. Introduction

This research project was started on March 3, 2016, by Vehans Ayvazi, an intern at MedVoice Biotech. The use case for this project is the MedVoice Biotech health awareness mobile application. It should also be noted that there are other types of Big Data architecture but we will be using an architecture called *Lambda Architecture*.

Keep in mind that a full book can be written on each subsection. This report is meant to give the reader a brief and simplified overview of the ever advancing world of Big Data. The objective of this document is to gather information on the subject of Big Data and make it as easy to digest as possible for the reader.

Complexity is your enemy. Any fool can make something complicated. It is hard to make something simple.

-Richard Branson

1. Use Case

The MedVoice Biotech health awareness application gives users the power to take control of their health by tracking the signals from their body as a part of their lifestyle to improve their overall well-being.

MedVoiceBiotech.com

With varieties of low latency and high latency data needing to be processed, the MedVoice Biotech health awareness application is a great use case for our big data research.

Types of data that is collected by the application include:

- User Login information
- Health history
- Current Health reports
- Streaming User Location

Types of queries that would be performed include:

-

B. Key Terms

This section will define some key terms. It's recommended that the reader have a basic understanding of these terms before moving forward.

Data: Compute data is the information processed and/or stored by the computer.

Query: This is a request for information from a database.

Database: This is a set of data that has a structure. It's organized in such a way that a computer can easily and quickly find the desired data *[linfo.org]*.

SQL: Standing for Structured Query Language, SQL is the a programming language created for managing data held by relational databases.

NoSQL: Referred to as Non-SQL or non-relational, NoSQL works with non- relational databases. NoSQL databases are built to allow the insertion of data without a predefined *[mongodb.com]*.

Structured Data: This type of data is basically the kind than can be put into a table and organized in such a way that it relates to other pieces of data in the table *[smartdatacollective.com]*.

Unstructured Data: This is the opposite of Structured Data, because this type of data can't be organized in a way where the data sets relate to each other. Example: Human voices, Social media posts, and emails *[smartdatacollective.com]*.

Big Data: This a term for a data sets that are so diverse, so large, and are coming in at such a fast rate that traditional data processing applications/techniques are far too inadequate.

Open Source: Open source software is software that can be freely used, changed, and shared (in modified or unmodified form) by anyone *[opensource.org]*.

Machine Learning: This is when we use algorithms to allow a computer to analyze data for the objective of *learning* what action to take when a specific pattern or event occurs *[bigdata-madesimple.com]*.

Server:

Linear Scalability: This is the ability to increase production inputs by a certain percentage and get an equal percentage increase in output of the system (Y = MX).

Scale up:

[tutorials.jenkov.com]

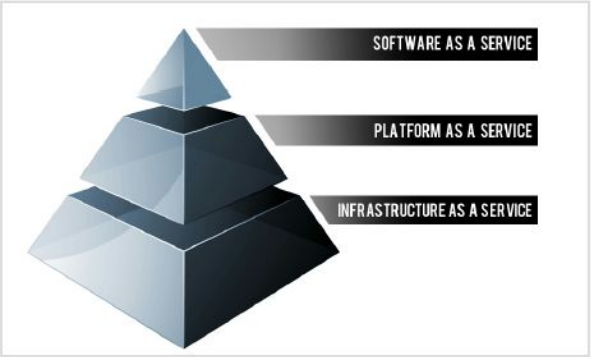
Scale out:

[tutorials.jenkov.com]

Append-only:

On/Off Premises:

Ad hoc: This is used to describe something that has been formed or used for a special and immediate purpose, without previous planning *[dictionary.com]*.



[rackspace.com]

SAAS: Standing for *Software as a Service,* this term is a software distribution method in which applications are hosted by a vendor or service provider and made available to customers over the Internet *[searchcloudcomputing.techtarget.com]*.

PAAS: Standing for *Platform as a Service,* this is a cloud computing model that delivers applications to the user over the internet. The cloud service provider delivers hardware and software tools the it's users as a service. This is a set of tools and services designed to make coding and deploying applications quick and efficient *[rackspace.com].*

IAAS: Standing for *Infrastructure as a Service, this* is the hardware and software that powers the SAAS & PAAS. This includes the servers, storage, networks, and operating systems.

C. Lambda Architecture λ

Created by Nathan Marz, Lambda Architecture is a “generic, scalable and fault-tolerant data processing architecture” *[lambda-architecture.net]*. The Lambda Architecture consists of three layers the **Batch Layer**, the **Speed Layer**, and the **Serving Layer**.

This Architecture gives us a **linearly scalable** system. It’s perfect for a Big Data system because it’s meant to **scale out** rather than **scale up**.

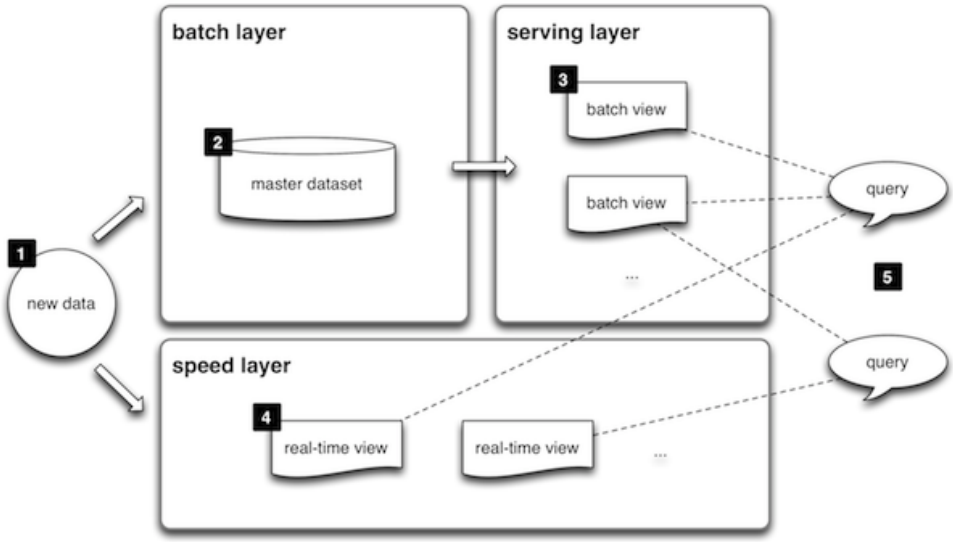


Figure #1: Lambda Architecture
[http://lambda-architecture.net]

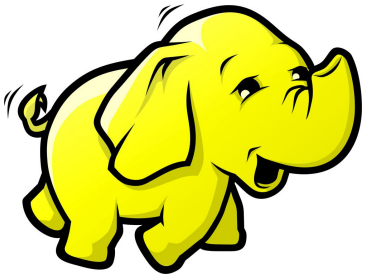
New data is coming in from a single source and goes into the Batch Layer and the Speed Layer (See Figure #1). It’s important to note that this data is not split, the exact same data that goes into the Batch Layer is also going into the Speed Layer. In the following sections we will be diving deeper into what exactly happens in each layer. We will also be exploring all the variety of softwares that live in each layer.

1. Batch Layer

One of the three layers of the Lambda Architecture, the batch layer has two main functions:

- It's first function is it must manage the master dataset. The master dataset is an " immutable, append-only set of raw data". This basically means the master dataset contain ALL raw/unprocessed data.

- It's second function is to pre-compute the batch query views.

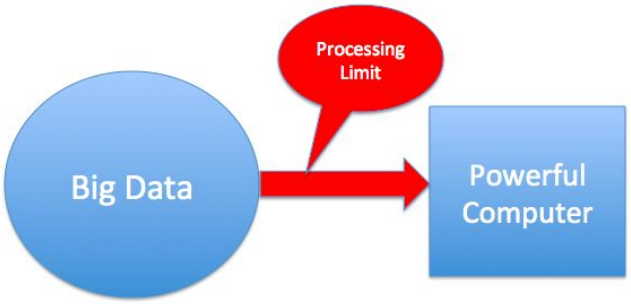


1.1 Apache Hadoop

One of the software that resides in the Batch Layer, and probably the first name that comes to mind when thinking of Big Data, is Hadoop. When referring to Hadoop, one must remember that Hadoop itself is not a software, it's a stack of softwares that combined are called Hadoop. What makes Hadoop extremely popular is that it's completely **open source**.

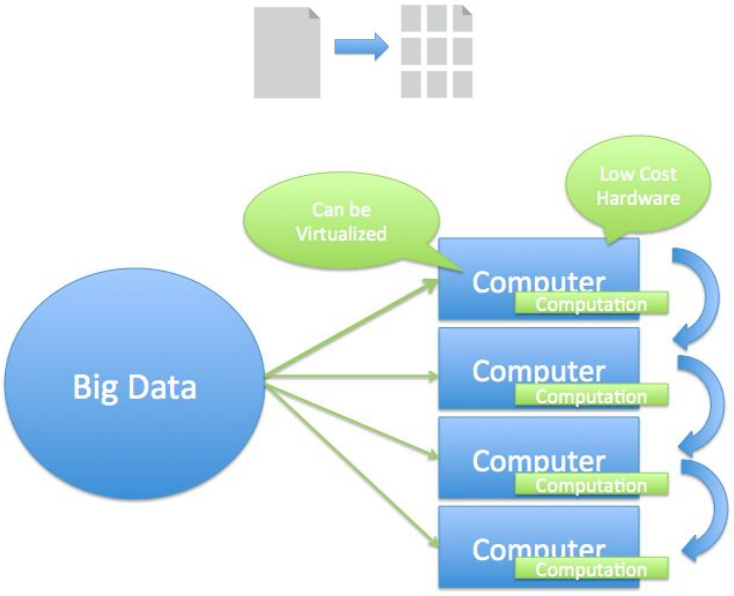
1.1.1 HDFS

What makes Hadoop so revolutionary is that it can solve Big Data problems. Traditional ways of dealing with data has its major limitations when the data is coming in too fast, it's too large, and at different varieties.



One of the ways Hadoop deals with this problem is through the Hadoop File System (HDFS). Hadoop takes each file and breaks it up into pieces and it scatters each piece into low cost commodity hardware called **Nodes**.

This division of labor speeds up processing, makes the system more easily scalable, and cost-efficient. HDFS also replicates each piece between the Nodes, making the system more resilient to failure because if a server (Node) fails the data is never lost and it's always available. HDFS is meant for **high latency** batch processing.



Just to reiterate, the benefits of this system include:

- Easily scalable
- Cost Effective
- Flexible
- Fast
- Resilient to Failure

1.1.2 MapReduce

1.1.3 Ecosystem

- **HBase**

As Hadoop is meant for high latency batch processing HBase is built for low latency queries. HBase is an open source distributed column-oriented database running on top of Hadoop/HDFS. Like MongoDB and Cassandra, **HBase is a NoSQL database**.

- **Pig**

As seen in the previous section, MapReduce is a very hard to understand and get comfortable using. To help make sure Hadoop developers aren't wasting time writing mapper and reducer programs but are using their time wisely when analyzing big data the scripting language Pig was born *[ibm.com]*. Developed at Yahoo, Pig runs on Apache Hadoop YARN and makes use of MapReduce and the HDFS *[ibm.com]*. The language that Pig uses is called PigLatin. **Pig is basically meant for creating MapReduce programs used with Hadoop.**

Why the name, Pig? Like Actual Pigs that eat pretty much everything Pig was developed to handle any kind of data.

- **Hive**

Hive is a open source data warehouse software built on top of Hadoop. It allows you to define a structure for your unstructured Big Data *[Hortonworks.com]*.

Although Pig can be a very powerful and simple language to use, the downside is that it's something new to learn and master for developers *[ibm.com]*. People at Facebook developed a runtime Hadoop support structure that allows anyone who is already fluent with SQL (which is commonplace for relational database developers today) to leverage the Hadoop platform right out of the gate without the learning curve with learning Pig *[ibm.com]*. This language is called Hive Query Language (HQL or HiveQL) which, again, is very similar to SQL.

- **Zookeeper**

The purpose of Zookeeper is cluster management.

- **Flume**

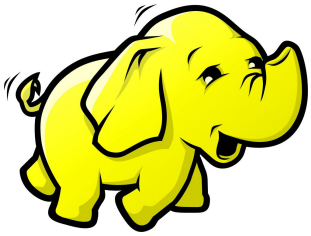
- **Sqoop**

- **Oozie**

Oozie is a open source task scheduler system to help manage Hadoop jobs. Oozie schedules jobs for the tools like MapReduce, Pig, Hive and Sqoop. **This allows develops to perform these jobs at predetermined periods of time.** These “schedules” are written in XML.

- **Yarn**

1.1.4 Hadoop Distributions



- "Vanilla' Version

This is the most basic version. It's the main version of Hadoop. All other distributions are based off of this one.

- Linux Based
- On-Premises

- Cloudera

- Linux Based
- On-Premises
- Components added

- Hortonworks

- Linux & Windows Based
- Azure HDInsight uses this distribution.
- On-Premises
- Easy to install solutions

- MapR

1.2 Data lake

1.2.1 Azure Data lake

1.3 Data Warehouse

2. Speed Layer

The speed layer deals with recent data only. All requests that are subject to low latency requirements is processed here.

2.1 Event Stream Processing

2.1.1 Azure Event Hubs

2.1.2 Apache Kafka

2.2 Apache Storm

2.3 Apache Spark

3. Serving Layer

This layer indexes the batch and speed views so that they can be queried in ad hoc with low latency *[mapr.com]*.

3.1 Visualization

3.1.1 Tools

- **Power BI**

- **U-SQL**

- **Tableau**

- **Excel**

- **MATLAB**

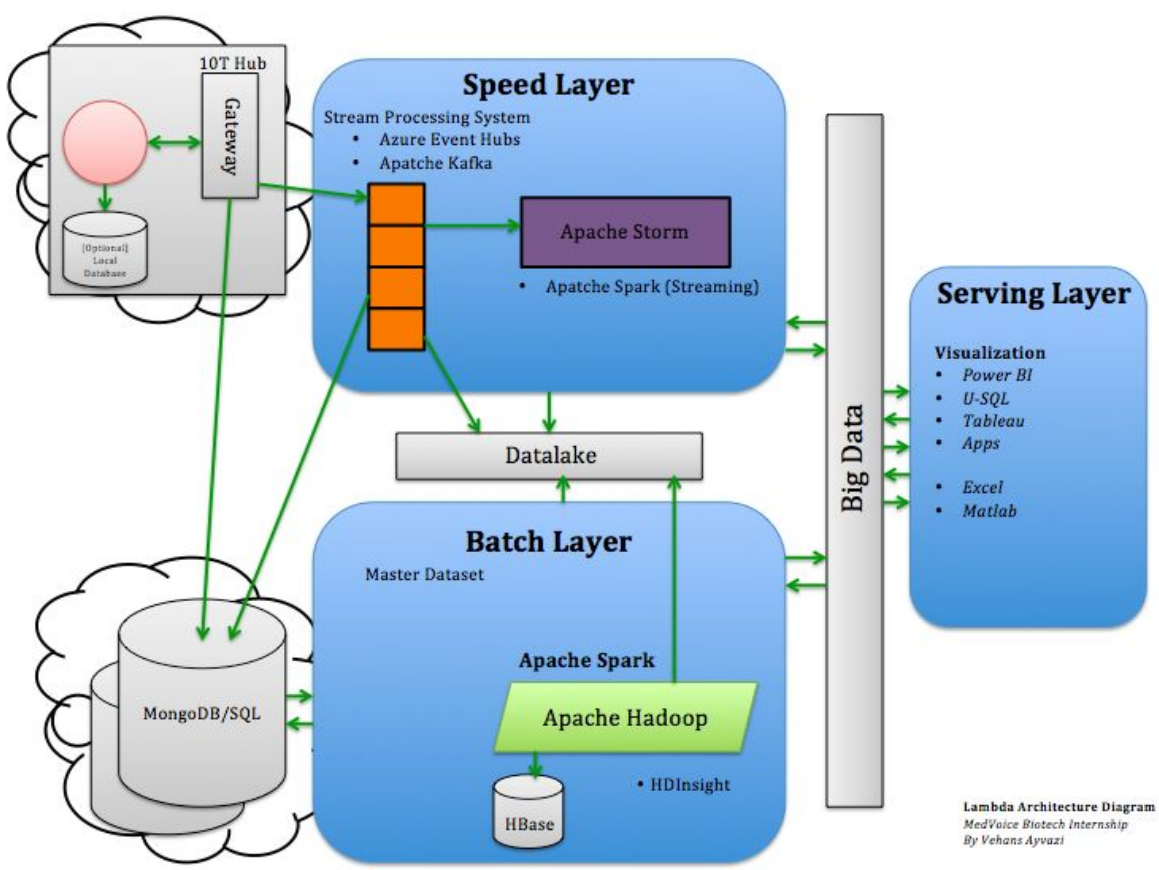


Figure #2:

D. Conclusion

E. References

1. Tutorial Recommendations

Great HBase Tutorial: https://www.youtube.com/watch?v=7AieO1vCUCQ&nohtml5=False

2. Citations

http://www.linfo.org/database.html

http://searchcloudcomputing.techtarget.com/

http://www.smartdatacollective.com/bernardmarr/287086/big-data-22-key-terms-everyone-should-understand

http://bigdata-madesimple.com/big-data-a-to-zz-a-glossary-of-big-data-terminology/

http://lambda-architecture.net/

http://tutorials.jenkov.com/software-architecture/scalable-architectures.html
https://hadoopecosystemtable.github.io/

http://www.blue-granite.com/blog/bid/402596/Top-Five-Differences-between-Data-Lakes-and-Data-Warehouses

https://www.mapr.com/developercentral/lambda-architecture

https://support.rackspace.com/white-paper/understanding-the-cloud-computing-stack-saas-paas-iaas/

What is NoSQL? https://www.mongodb.com/nosql-explained

https://opensource.org/