Florida Department of TRANSPORTATION

Big Data - Phase I
Action Plan

TWO 021
Contract No.C-9C62

FDOT Office
District Five

Author

Date of Publication
August 18, 2016

Table of Contents

Introduction

The Florida Department of Transportation – District Five (5) desired the creation of an action plan document to comprehensively determine the various applications and benefits of utilizing big data sets for transportation planning and other transportation data stakeholders. This action plan will outline the process and procedures needed to be in place in order to transfer the HERE data to the District Five servers. For the transfer to be completed the District will set up a big data store that will host the HERE datasets as well as other big data sets.

In order to create and set up a big data store, several steps must be taken into consideration. The creation of the FDOT District 5 ITS Big Data Store Action Plan includes the following:

- Background information on the main big data store platforms
- A comparison of these identified big data store platforms
- A recommendation of which big data store platform should be implemented
- Identification of the minimum hardware, software, and architecture required for the recommended big data store platform solution for both the Testing and Production environment
- An outline process for the transfer of Historic HERE.com data collected by the consultant
- Outline of the process needed for continued HERE.com data collection
- LOS Methodology for the delivery of a proof of concept for combining HERE.com data with the LOS_ALL

Once this action plan is approved by the District, the HERE data set will then be transferred.

Project Use Case Goal

The primary project use case of this action plan is to develop a test case for centralized planning data and documents to be contained within a big data platform. Access to this data and documents will be available through the big data platform and a series of application program interfaces Application Program Interface (API).

Applications and Benefits of using Big Data for Transportation Planning

The FDOT TSM&O Data Report documented extensively the challenges of utilizing relational databases to perform analysis dynamically on large transportation related data sets. That report also provided a comprehensive documentation outlining the benefits of sharing multiple platforms currently available in the technology industry to handle big data sets. With big data applications, the District will have the ability to perform analysis across a range of historic information enabling complex data relationships. The use of big data technologies offers an additional benefit in the reduction of data silos and can minimize the amount of data duplication by allowing individual departments to leverage these datasets for their own needs from within the big data store no longer requiring each cost center to maintain an independent base of information. The big data store will serve this purpose, allowing multiple users with different data needs to dynamically access the common data for different purposes minimizing data silos in a consistent way.

The use of a big data store to represent a centralized source, which is different from the database structure currently in use in the FDOT environment. The big data store allows for the introduction of APIs, which are capable of directly and efficiently interact with the big data store.

Advantages of utilizing APIs include but are not limited to:

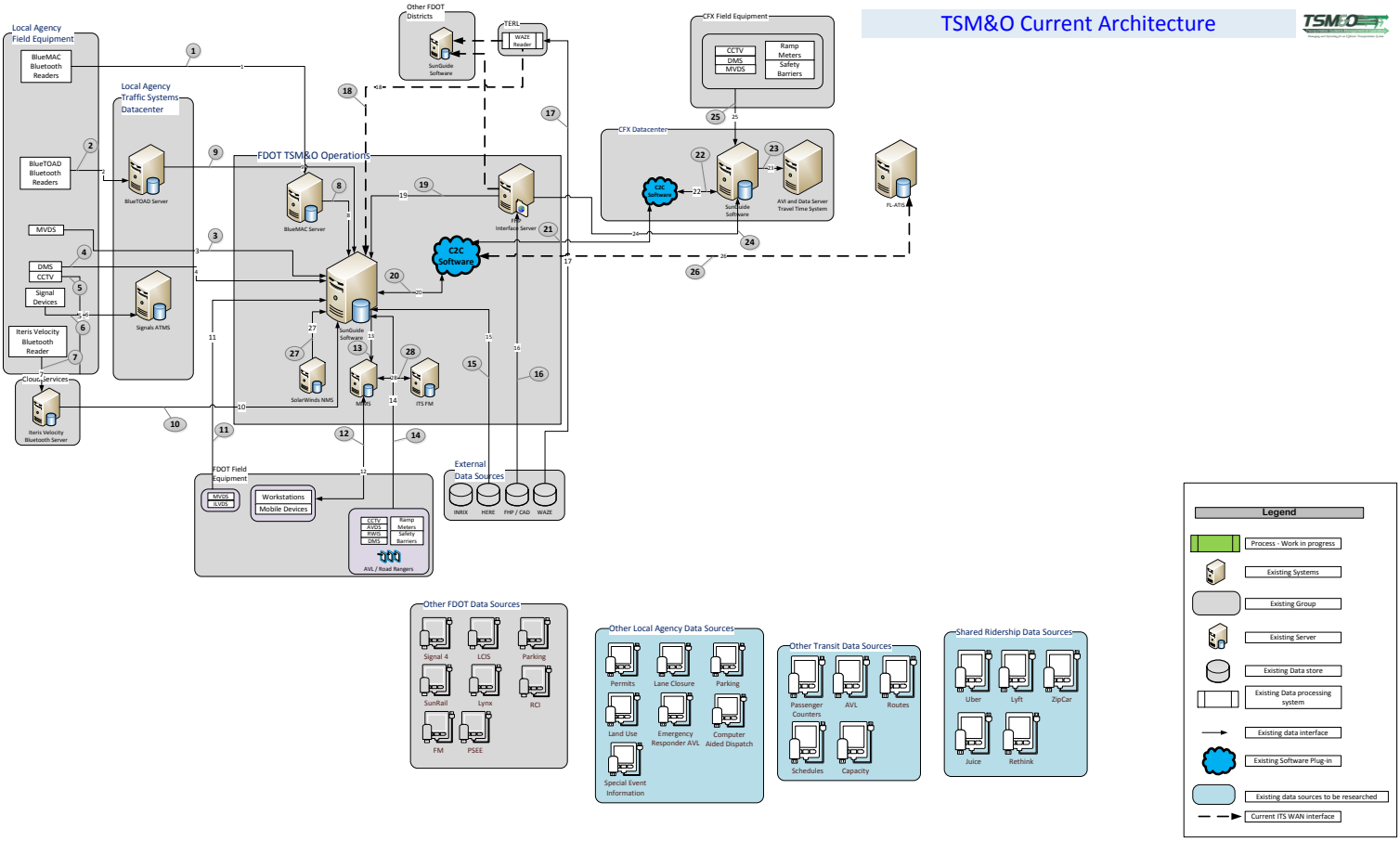
- Centralizing methods on how to access the information in the Data Fusion Center.
- Allows other applications and devices to utilize information later through a single set of methods.
- Allows for security on the API methods to utilize a role system to authenticate users to certain sets of data.
- Modularize the design methodology to allow the various layers of the application and its data to operate independently of one another. This allows for easier upgrading and repair of the applications and the data layer since they are not dependent on one another.

The use of these APIs may provide the District the opportunity to reutilize developed API code from cost center to cost center leveraging investments and resources across multiple departments, provide a consistent use, and feel for applications, which utilize the information in the Data Fusion Center.

Understanding the hardware and software infrastructure currently in place at the District is the first step required for the development of a recommendation for a big data store environment. Understanding the current infrastructure environment at the District provides an opportunity to leverage existing hardware and software components into the recommendation for the big data store architecture.

District 5 Current Hardware and Software Infrastructure

A data and systems inventory of FDOT District 5 was completed in August of 2016 as part of the FDOT District 5 TSM&O Data and Systems Inventory. The current District TSM&O architecture documented in that inventory is represented as per the below diagram.



Proposed District 5 Big Data Store Architecture for data workflow

Big Data Platform Comparison Matrix

There are many types of platforms that can be utilized to establish a big data store. After a comprehensive analysis of the many platform types available, the current and future software and hardware available at the District, the District needs for integration between multiple databases and datasets including data in text and spatial formats, two main platforms were selected to be compared as they would attend and suit the District needs and could potentially compose the architecture of the upcoming Districtwide big data store. The platforms that have been selected for comparison are MongoDB and Elasticsearch (ES). Before discussing the comparison between these two platforms, it is important to note that the District has indicated there is not an anticipation to utilize videos, pictures or sound data within the big data store in the upcoming years. Integration of adequate platforms to handle these data types were taken into consideration when designing the recommended Big Data Store Architecture which will be presented and discussed in the upcoming sections of this action plan.

A detailed matrix was prepared to discuss comparable information between Mongo DB and Elasticsearch (ES). The comparison includes platform description, database model, URL to access the website of the corresponding platform, links for technical documentation, general licensing information, list of supported implementation language, compatible server operating systems, data schema, support for SQL Query Language, API access method, list of supported languages, and others.

The next section will discuss the similarities and findings of each platform and how they would compose the architecture for the District Data Fusion Center.

Big Data Store Architecture Diagrams

Two primary big data platforms were considered for the development of the recommended FDOT big data store. They are Elasticsearch with Hadoop and MongoDB. These platforms were selected for the analysis since they can provide the best alternative scenarios for a big data store architecture, which ultimately could support the FDOT District needs. Both systems share similarities and demonstrate strengths and weaknesses. The platforms comparison matrix containing more in depth information is available at the Appendix B of this document.

After comparing these two primary big data platforms, the next step was to outline big data store architecture scenarios considering components such as Virtualization, Clustering, Processing, Storage, and others. The information below illustrates the exercise of outlining the ideal system architecture for the District big data store considering these two main platforms. This information will be presented through a series of pros and cons considerations that guided the project team to comprehensively discuss the challenges and benefits of each configuration while taking into the account the vision of the District for the big Data Fusion Center.

Elasticsearch (ES) and MongoDB Diagram Approaches

Technology used:

- Elasticsearch with Hadoop Distributed File System (HDFS)
- MongoDB
- Established San Storage
- Virtualization Software (VMWare)
- Hardware / Physical Commodity Servers

Elasticsearch Approach (Pros and Cons):

**Each of the referenced diagrams are available at the Appendix A of this document.*

Diagrams 1 – 4 utilize a mixture of Big Data platforms. Elasticsearch will work in unison with Hadoop (HDFS) for the purpose of allowing each platform to best work to its strength while relying on the other to complete the balance. For example: Elasticsearch shines with the ability to index and search data quickly however this platform lacks the ability to manage non-text data. Data such as files and images will be handled by Hadoop (HDFS). Hadoop's ability to manage large volumes of data (including non-text data) and this platforms map reduce ability makes Hadoop (HDFS) a definite strength in this architecture combination. Hadoop is better suited for larger data sets and longer running queries while Elasticsearch is designed for speed and response, in this configuration Hadoop will be responsible for storage and select queries while Elasticsearch will be utilized to perform faster queries and returns.

Diagram 1:

This diagram outlines the best practiced architecture for a minimal Elasticsearch setup. This setup utilizes a **mixture of virtualized and hardware (commodity) servers**. One of Big Data's strengths is that it can utilize what are known as commodity servers. A commodity server in essence is a lower cost server (usually a physical hardware server) that can be used in a Big Data cluster. One benefit of this configuration of servers is that if a particular server becomes defective, it can be removed from the cluster and replaced with a new commodity server and it will configure itself alongside the rest of the cluster. This setup will utilize physical hardware servers in the Data Lake, all other servers in the setup can be virtualized.

- **It is recommended that the District utilize the Diagram 1 as its architecture to compose the big data store of the Data Fusion Center. Additional details on the recommendations can be found in the recommendation section**

Diagram 2:

This diagram outlines the exact same setup as Diagram 1 with the exception that the Data Lake will be made up of **virtualized servers** in place of hardware physical servers. The benefit here is that one can reuse the current virtual server resources that are already in place. The downside to this approach is that the server hosting the virtual servers (also known as a host server) becomes the single point of failure in this setup. If the host server fails then all virtual servers within that host server will fail as well. Another downside of this approach is that using virtual servers removes one of Big Data's main features, the ability to quickly and cost effectively expand horizontally with cheaper commodity servers.

Horizontal scaling is still possible but must be completed by adding more virtual server nodes to the cluster and is limited by the resource pool of the host server itself.

Diagram 3 and 4:

These diagrams outline and follow closely diagrams 1 and 2 with the exception that the setup starts **with only one server** (virtual or physical hardware). This configuration and setup is more ideal for a development environment but is not recommended for staging or production.

- **Diagram 4 represents the minimum requirements for set up a big data store. This diagram illustrates the Data Fusion Center server containing the big data store as well as serving as the testing environment for the District. This server will be duplicative of the production Data Fusion Center.**

Mongo DB Approach (Pros and Cons):

Diagrams 5 – 8 utilize Mongo DB as the Big Data platform. Mongo DB has many advantages that makes this platform to be a great candidate for a Big Data solution. The ability to handle any kind of data natively (text and file data) allows Mongo to fit many Big Data solutions. Mongo has the ability to index and search data like Elasticsearch (not as quickly) and has the ability to store data natively using a native technology known as GridFS. Another benefit of the Mongo platform is the ease of setup and use. Mongo can be configured to store data across multiple mongo instances (also known as Sharding) and has the ability to horizontally expand. Mongo DB is also one the largest and fastest growing Big Data platforms in the market today.

Diagram 5:

Similar to diagram 1, this setup shares a close design with the exception that the Elasticsearch and Hadoop portions have been replace with Mongo DB. In this diagram, you will see the best practiced approach for a minimal Mongo setup utilizing **both hardware / physical and virtual servers**. This allows the data lake layer to utilize physical hardware commodity servers to scale the data lake horizontally when needed.

Diagram 6:

Like diagram 5, this setup is similar in design with the Elasticsearch and Hadoop layers having been replaced with Mongo DB. This diagram outlines the best practiced approach for a minimal Mongo setup utilizing all **virtual** servers. The main consideration to this solution is that the host server becomes the single point of failure in the event the server itself goes down. The host server's resource ability also becomes a drawback as scaling is limited by the host server's resource pool and ability.

Diagram 7 and 8:

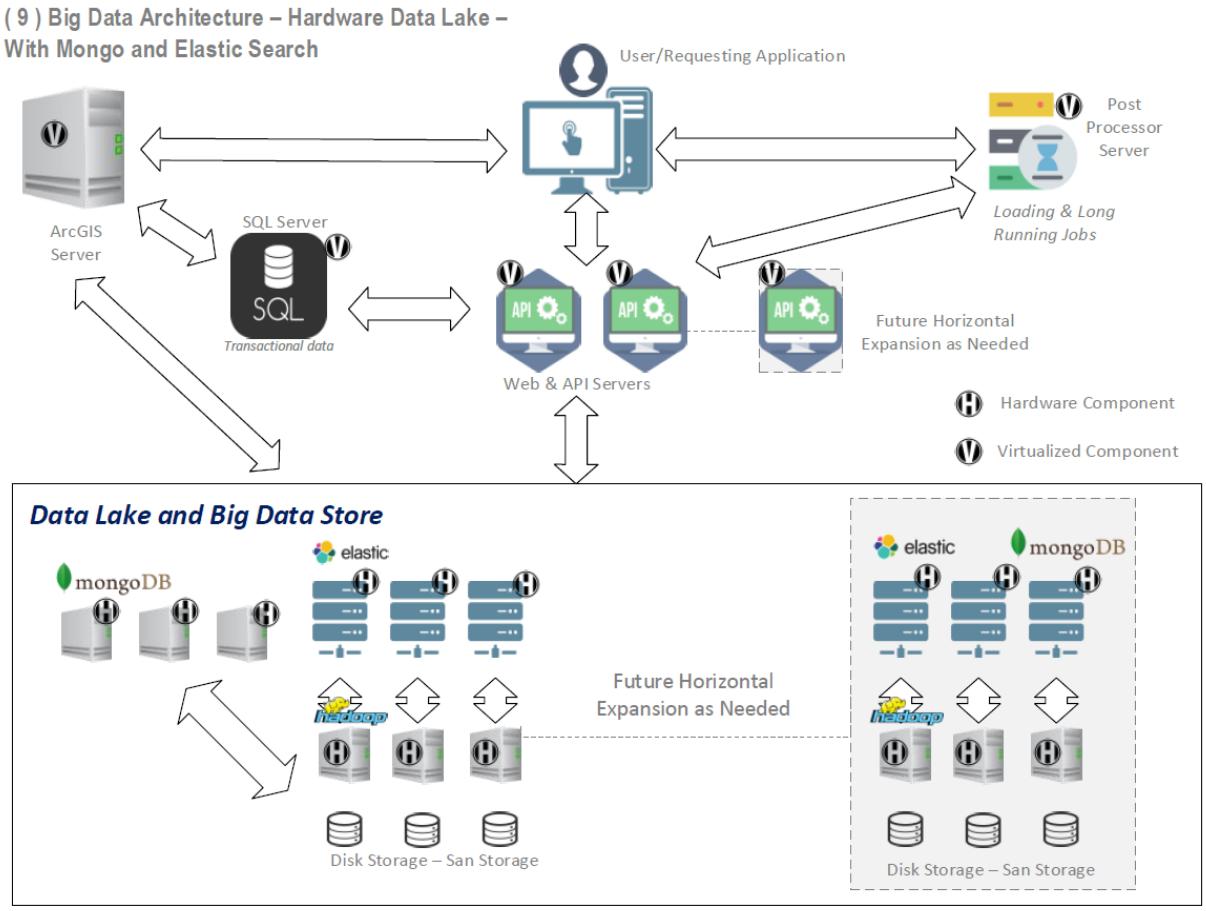
This diagram outlines the same setup as diagrams 5 and 6 with the exception that the setup starts with **only one server** (virtual or physical hardware). This setup is more ideal for a development environment and isn't recommended for staging and production.

Diagram 9:

This diagram utilizes an architectural design that mimics **diagram 1** with the addition of adding a Mongo DB cluster. This design isn't really recommended unless a use case arise where Mongo DB has a feature that can't be achieved with Hadoop. Mongo and Hadoop share many features that are alike. The main difference is that Hadoop can utilize the similar features better. So having an environment where two platforms share so many duplicate features wouldn't utilize the capabilities of the Big Data platform to its fullest.

Important Consideration: While the Production and Staging Environments can utilize a combination of physical or virtualized server approaches, it is important that both environments mimic each other as close as possible. This will assist in ensuring that tools, data, and configurations are better able to deploy between the environments.

- **The below Diagram outlines how MongoDB and Elasticsearch plus HDFS can be configured in the future if the District decides to incorporate this platform to specifically handle videos, pictures and sound.**



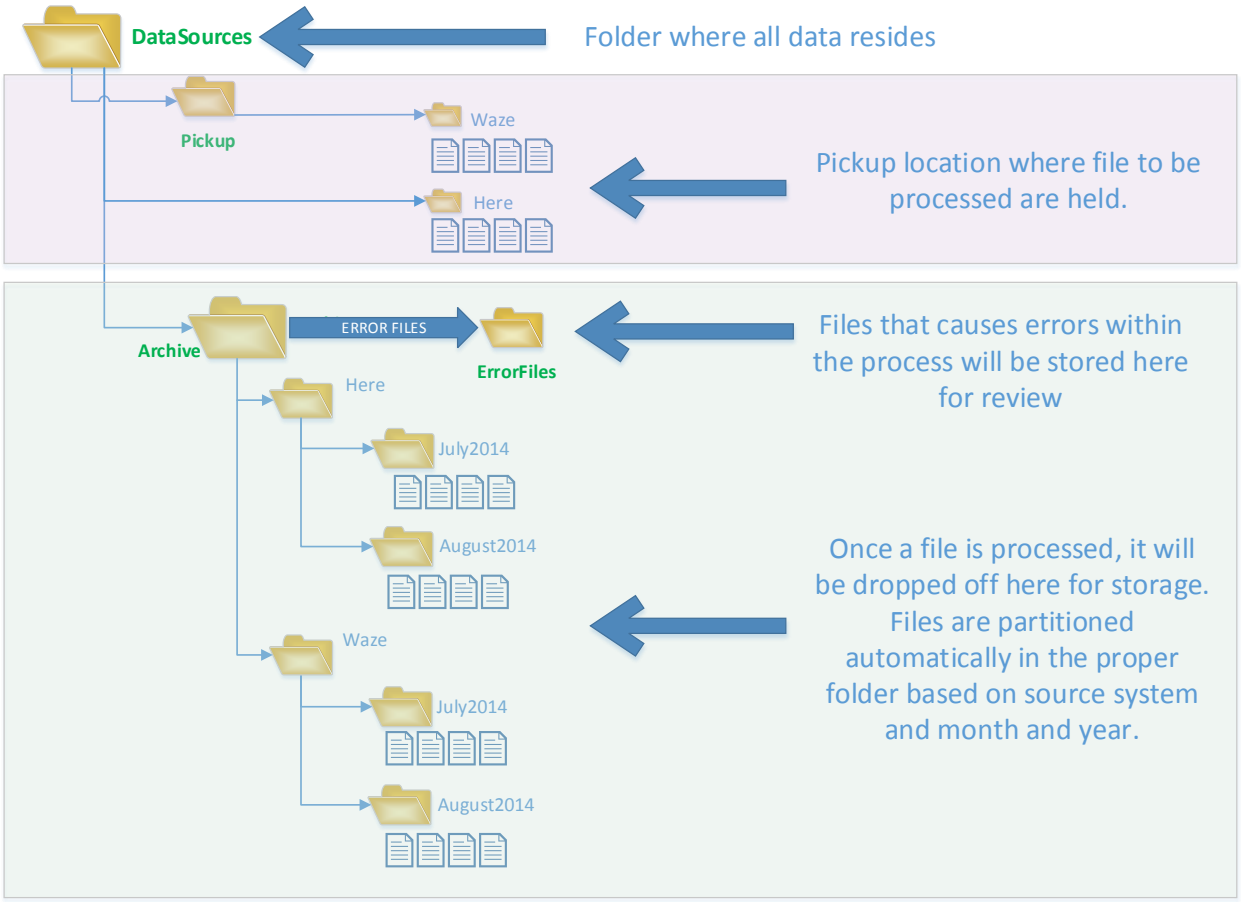
As part of the recommended big data store architecture, a workflow containing the standardization of naming conventions, folder structure, and a dynamic API will be used to provide a continuing structure and organization for the information in the big data store. Each of these items are further defined below:

FDOT District 5 Folder and Naming Convention and Folder Structure

The recommended folder structure and naming convention is as per following:

DataSources - Master folder containing data sources to be processed into the big data store and historical record keeping of processed files.

Raw Data Directory & File Structure



Dataset Name - This dataset name is to be a clear and concise representation of the data source. This name will serve as the data source folder name and will represent the name of the API dynamic variable that will be used in the API calls in the Big Data Store. Spaces and other illegal characters should be avoided in the naming of this dataset folder.

- **Pickup** - Automated processes will review the contents of this folder on a regular interval to identify as new files are placed in this location for processing. Files in this folder will be

processed for inclusion into the big data store. These files are processed and programmatically moved to the archive folder for the dataset. Files which are currently stored in a different location but which are accessible to the big data store processes will be referenced from those locations and may not be required to be placed in the "Pickup" folder (For instance, an existing SQL or Oracle Database.)

- **Archive** – The archive folder is the location for processed files that have been successfully included in the big data store to represent a historical review of processed data from the "Pickup" folder. The archived data in this folder can be set to automatically move after a set time to deep storage, be deleted, or permanently stored in this location.

    - Within the archive folder, an automated process will create a folder for the current month in which the data is being processed by the big data store. This automated process will run at the start of each month, ensuring the folder following the structure of "MonthYear" where the Month is fully spelled out and the Year will be represented by a four-digit value is available.

        *Note that historical data will be placed in the folder at the time of processing and not in the folder which corresponds to the date/time values in the file itself.

Summary:
The "DataSource" folder will serve as the primary folder for all raw data files to reside grouped by their data source name (Example: HERE, WAZE, TWITTER). Within the "DataSource" folder will be folders named 'Pickup' and 'Archive'. The pickup folder will serve as the point for processing of new raw data and files into the Big Data Source. Once successfully integrated into the big data store, these files will be moved to the "Archive" folder. The "Archive" folder will be organized by data source, with subfolders organizing the files by date processed into the big data store.

FDOT District 5 Dynamic API (data mart) Methodology

Dynamic API will be used to assist in the organization and retain a connection between datasets and the API which is used to reference that information. The name of the data source which is used to create the folder in the above reference "Pickup" and "Archive" folder will serve as the dynamic API name for the data contained that partition of the big data store. The use of this dynamic API will allow developers to plan accordingly for how the data can be retrieved once digested into the big data platform. The APIs will be organized comprehensively in a form of an API Library / Catalog containing compressive metadata and data metadata to facilitate the identification its data source and purpose of utilization.

Preliminary Implementation of FDOT District 5 Big Data Store – Methodology

HERE.com Data

Historic HERE.com data

As part of the deliverable with this task the Historic HERE.com data has been transferred onto the FDOT domain and is currently located on the VHB Development Server (FDOT Domain).

The historic data has been organized following the naming and folder structure identified in the above section of this action plan document. The historic HERE.com data is stored in .xlsx format (Microsoft Excel). This data will undergo a transformation to be included in the Data Fusion Center (Testing) once that platform architecture is fully built. The historic raw format of the data, once processed, will continue to be preserved as raw files for historical record keep as well as be available from the big data store.

HERE.com Data collection process

Below is the outline of the three step HERE Data ingestion process. A graphic view of this flow can be viewed on the next page.

STEP 1:

The data initially starts in the HERE.COM cloud. At a predefined interval, our local processing server will request from HERE.COM the latest updated data. This data will be downloaded to the local processing server. When it arrives it the updated HERE.COM data will be an .xml file that in enclosed within a .zip file. Once the download has been verified to be complete, the local processing server will initiate step two of the ingestion process.

STEP 2:

In this step, the zip file is programmatically opened, and the xml HERE.COM data is extracted. Once the .xml file is extracted, the file is consumed into the local processing server's memory. Once in memory, the data is examined. Any ETL, Validation, or Data Enrichment processes will be applied at this time. Once all processes are applied to the data, a new .JSON file is created and the final step in the ingestion process is then started.

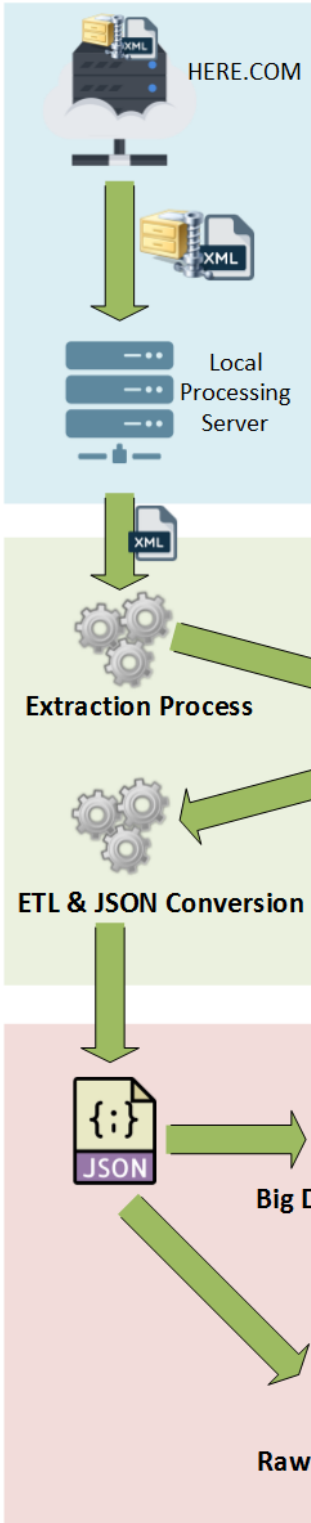
STEP 3:

In this step, the newly create JSON file is then marked as "HERE" data and is then inserted into the proper location within the "HERE" partition of the Big Data platform. The file is placed within the platform by its predefined indexes for simple retrieval later. A copy of the .JSON file is also sent to its predefined Raw File Archive folder for storage.

De diagram below illustrated how the HERE.com data ingestion process occurs.

If an error occurs within any of these steps, the file will be sent to a partitioned error location and an error log is recorded. This process will allow the system to set aside erroneous file and continue its regular process without completely stopping. Once an error file is found, this file can be examined to identify the reasons the failure occurred and what further actions must be taken.

HERE Data Ingestion Diagram
Version 1.0.0



STEP 1

Download HERE File From HERE.COM

At a pre-define interval, the local processing resource downloads a zip file from 'HERE.COM' containing the data needed in a xml file.

STEP 2

Extract XML and Convert To JSON

Once the .xml file is retrieved, a python script is kicked off that extracts the .xml file from the .zip file. Next, the xml file is consumed into temporary memory and any validation and / or etl processes needed will be applied. Finally, any additional meta-data needed to the file will be applied. Once completed, the data in memory will be saved out as a JSON file.

STEP 3

Insert JSON file into Big Data Store and Raw Data Archive

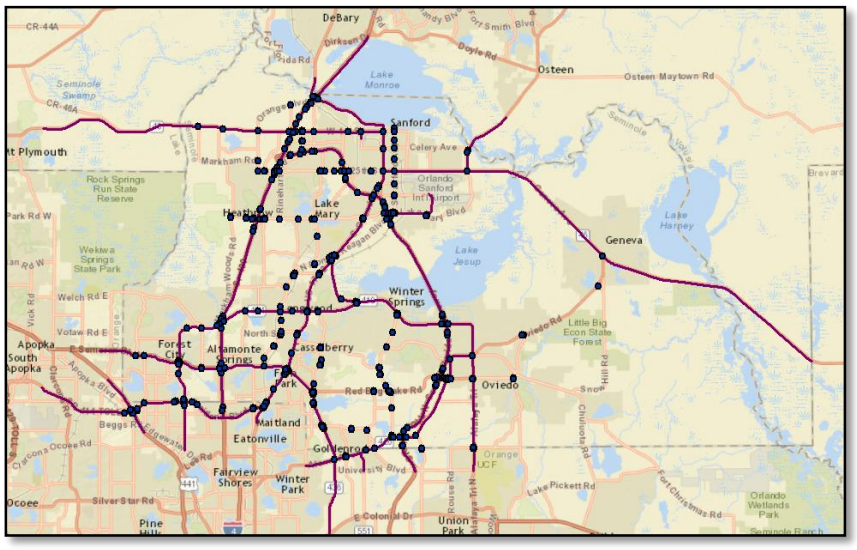
The newly create JSON file is then inserted into its proper partition within the Big Data Platform.

A copy of the JSON file is also sent to its proper archived location for safe keeping and historical purposes.

LOS - Congestion Management

As part of this action plan, a proof of concept is being developed for the combination of the LOS_ALL segments with the HERE.com data and information. This proof of concept will provide an ability to visualize the capacity analysis of the LOS with the average speed review of the HERE.com traffic information.

To successfully integrate the LOS and HERE.com segments (below) Seminole County, FL was chosen to serve as the county in which the proof of concept would be conducted.

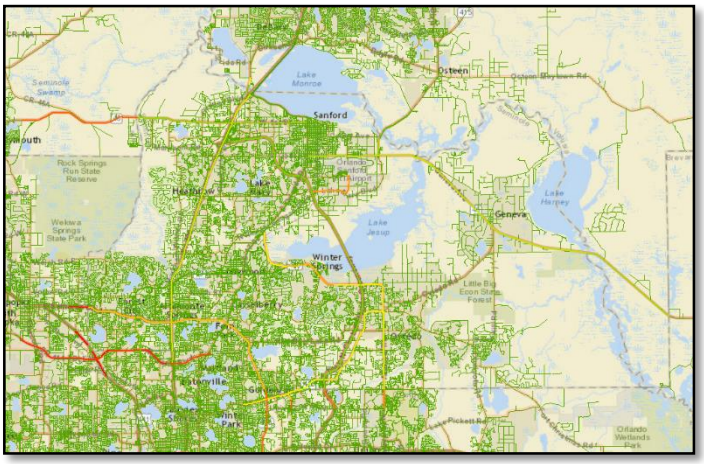


To bring these datasets together for use in this proof of concept a base dataset was required.

Base Data creation of the LOS

As part of combining datasets a practice of conflation is required in order to bring together datasets which do not share matching segments. As part of the LOS proof of concept the NavTeq streets feature layer was chosen as base. The selection of this dataset as base was based on the following:

- Most complete network including directionality and lanes
- A detailed network which could continue to serve as a base in the integration of future datasets
- A regional network which could be symbolized to reflect various datasets



A methodology was developed in which through a combination of ArcGIS processing and manual QA/QC the LOS segments were successfully applied to the base layer. This provided a base dataset of all roads in the District with LOS values able to be applied to segments for which that data was available. Road segments which did not contain LOS values were left intact to continue serving as a complete network which could be reutilized for future data incorporation.

Methodology for Integration with HERE Real-Time data

Upon completion of the base data with the LOS information, a second methodology was conducted to successfully conflate the base layer containing the LOS information with HERE.com segment information.

This process involved the HERE data being successfully added through a conflation process to the same base dataset. This process was achieved through an automated and manual procedure to ensure that the HERE segments were properly aligned on the base data.

At the completion of this conflation, the HERE.com segment information was placed in the base feature class along with the LOS information. Once these datasets were successfully placed in the same data table a crosswalk was generated detailing the HERE.com segments which corresponded with the LOS sections.

Through the use of scripts and ArcGIS Server Manager the real time HERE.com data feed is used to populate the created base layer with traffic information on a live and continuing basis. An additional script has been developed which through the use of the crosswalk table performs an aggregation by LOS segment to produce a LOS segmented file with HERE.com averages.

Methodology for Inclusion with Big Data Store Platform and Historic HERE.com data

To provide access to historic HERE.com and complete the proof of concept. A subset of HERE.com data was used to test the proof of concept. The crosswalk table discussed above has been added for the LOS proof of concept big data store as a mechanism to verify a successful API call which can return these values.

Recommendations

Big Data Store Recommendation

It is recommended that the Data Fusion Center big data store be composed by a platform infrastructure that has a mix of hardware components and virtualized components as found on Appendix A – Diagram 1. With our recommended approach, the platform will be able to deliver capacity on the present immediate needs, as well as scale out capacity to when needed. High availability, fault tolerance, scalability and speed are the prioritized factors that were given consideration when making our recommended choice. The recommended architecture for the Data Fusion Center Big Data store can be found on Diagram 1 in the Appendix section of this action plan. The Big Data platform technologies used in this recommendation are Elasticsearch, Hadoop and the HDFS file structure (alongside an installation of either Cloudera or Hortonworks). Cloudera and Hortonworks are enterprise packages of Hadoop that allows for the management, setup, and maintenance of the Hadoop and the HDFS layers. Hadoop in its native structure is a collection of projects and services that work together to form what is known as the Hadoop and HDFS platform. Cloudera and Hortonworks are platforms that manages and enhances the Hadoop and HDFS platform.

Elasticsearch is a technology that will allow for fast searching of the various types of "text type" data that resides in the Big Data solution such as json, xml, tables, etc. Elasticsearch was also chosen to be part of the Big Data solution because of its known native functionality to communicate with ESRI's Big Data Store features. This will allow FDOT to utilize and leverage their already established ESRI geospatial platform toolsets and subscriptions integrating, consuming and drilling into the data lake.

Hadoop is another component of this recommendation. Hadoop balances the weaknesses found in Elasticsearch. Hadoop has the ability to store and manage any type of data while Elasticsearch can only handle 'text' type data. With Hadoop's ability to work with non-text data, the recommended platform architecture will allow to deliver regardless of the data type. Hadoop also brings a large set of projects that can be later added into the Big Data platform that expands the functionality and features found natively within Hadoop. Projects like 'HIVE' that will allow the District to utilize the SQL language within its own environment or various analytic tools to dive deeper into large sets of data will prove valuable as the Data Fusion Center store platform grows.

This recommendation contains a mixture of virtualized components as well as physical hardware components. The virtualized components can utilize virtualized technologies that FDOT currently has available. These components can be quickly expanded and deployed, utilizing standard images of servers to install the required dependencies and requirements onto the additional servers. The hardware components should be (servers / nodes) found within the Data Fusion Center. It is recommended that these hardware components be physical commodity servers. By utilizing commodity servers, the methods to scale out the Big Data platform can be utilized without limitations found within virtualized environments (such as reaching the processing and resource limits of the virtual server host itself).

This recommendation outlines the utilization of the following servers.

ARCGIS Server: This server will handle the geo data that is used within the Data Fusion Center. Its ability to utilize ESRI services, ESRI big data store and execute complex analysis, produce maps and map data will prove viable in the platform.

SQL Server: This server will be used to hold many datasets that ARCGIS will use. This server will also hold many datasets that are transactional in nature or datasets that ARCGIS regularly use to build its maps and data from.

Post Processing Server: This server will act as a separate application server. It will be used to house processing job applications, loading applications, and application / services that pulls in data from the Big Data Lake to later run longer running ETL processes against.

API / Web Servers: These servers will house the various web applications and the Big Data platform API's. These API's are used to make calls to the Data Fusion Center and send the returned data back to the original request caller. These API's will be the methods by which outside users will have to use to validate themselves and retrieve data from the Big Data platform.

Elasticsearch Nodes / Servers: These servers will house the Elasticsearch framework. These servers will be used in parallel to process request to the Big Data platform and return the requested data to the caller. These servers will handle most of the quick and ad-hoc query request sent to the platform.

*It is recommended that these servers are commodity servers.

Hardware Specs per Node:

- Multi-Core Modern CPU (Intel i5 or better)
- 32gb – 64gb Ram
- Multi TB Local Hard Drives (SSD or preferred or server grade spindle drives)

Hadoop Node / Servers: These servers will house the various Hadoop tools needed to maintain the HDFS environment. Depending on the final decision, either Hortonworks or Cloudera will be install in this environment to maintain, update and grow the HDFS layer as needed. These servers can easily scale out as needed. As this cluster grows and more demanding features are needed, this layer can also have other Apache projects (such as 'HIVE' and various data analytics tools) install to scale out the native features of the Big Data platform.

*These servers SHOULD be hardware commodity servers.

Hardware Specs per commodity server:

- Multi-Core Modern CPU (Intel i5 or better)
- 32gb – 64gb Ram
- Multi TB Local Hard Drives (SSD or preferred or server grade spindle drives)

SAN Storage / Network Storage: This layer will be used in junction with the commodity servers as a replication and backup point for the data. Utilizing the SAN Storage at this capacity is better practice. This allows the commodity servers to grow and have their part of the overall data set and then at predefine intervals, replicate the data as need to the SAN for historical and archiving purposes.

The high level diagram representing the architecture recommendation can be found in the Appendix section for this Action plan under the Diagram 1.

Big Data Store Server Requirements – Software and Framework Requirements & Dependencies

Below you will find a list of software and framework requirements needed for the defined servers within the Big Data solution documented in the above referenced graphic.

ArcGIS Server:

- ArcGIS Server
- ArcGIS Desktop (Advanced License)
- Windows Server 2012 R2 64bit or Higher
- Geoprocessing Tools for Hadoop
- GeoEvent processor
- ArcGIS for Portal
- ArcGIS Data Store

Hardware Specs:

- Intel CPU i5 or Greater
- 16gb Ram
- 1tb Hard Drive

SQL Server: *(Minimum specifications will updated upon data review)

- SQL Server 2012 or Higher
- Windows Server 2012 R2 64bit or Higher
- Latest .NET Frameworks and Windows Updates

Hardware Specs:

- Intel CPU i5 or Greater (Multicore – 4 or 8)
- 16gb Ram
- 1tb Hard Drive

Web / API Server (Per Server):

- Windows Server 2012 R2 64bit or Higher
- Latest .NET Frameworks and Windows Updates (*Verify That The .NET Frameworks Are Available For Resource Pool Allocation and ISAPI registration)
- NODEJS
- Python 2
- Python 3
- JAVA (Latest JRE)
- Scala

Hardware Specs:

- Intel CPU i5 or Greater (Multicore – 4 or 8)
- 16gb Ram
- 1tb Hard Drive

Post Processing Server:

- Windows Server 2012 R2 64bit or Higher
- Latest .NET Frameworks and Windows Updates

Hardware Specs:

- Intel CPU i5 or Greater (Multicore – 4 or 8)
- 32gb – 64gb Ram
- 1tb Hard Drive – Minimal (Can Grow / Scale As Needed)

Elasticsearch Node / Server (Per Node):

- Windows Server 2012 R2 64bit or Higher OR Ubuntu Server 16.04+
- JAVA
- Spatial Framework for Hadoop (GIS Tools for Hadoop)
- ESRI Geometry API for Java
- Apache Hive (or equivalent)

Hardware Specs:

- Intel CPU i5 or Greater (Multicore – 4 or 8)
- 32gb – 64gb Ram
- Multi-Terabyte Hard Drives (SSD Preferred, or Server Grade Fast Spinning Drives)

Hadoop (HDFS) (Per Node):

- Windows Server 2012 R2 64bit or Higher OR Ubuntu Server 16.04+
- JAVA
- TBD – Depends on the deployed package chosen. (Hortonworks or Cloudera)
- Hadoop Tools**

Hardware Specs:

- Intel CPU i5 or Greater (Multicore – 4 or 8)
- 32gb – 64gb Ram
- Multi-Terabyte Hard Drives (SSD Preferred, or Server Grade Fast Spinning Drives)

MongoDB (Per Node) – *Future Implementation If Needed*:

- Windows Server 2012 R2 64bit or Higher OR Ubuntu Server 16.04+

- Python 2
- Python 3
- JAVA (Latest JRE)
- Mongo framework

Hardware Specs:

- Intel CPU i5 or Greater (Multicore – 4 or 8)
- 32gb – 64gb Ram
- Multi-Terabyte Hard Drives (SSD Preferred, or Server Grade Fast Spinning Drives)

(*) Additional frameworks may be required depending on the final agreed upon Big Data Platform

(**) Additional may be required depending on the final agreed upon Big Data Platform and installation package used. (Example: Hortonworks or Cloudera).

(***) Hard Drives: Prefer SSD or Server Grade High Spinning Drives

ESRI ArcGIS GIS Platform License Requirements

Below you will find a list of ESRI GIS license and software requirements needed for the Data Fusion Center (Production and Testing Environment) servers within the Big Data store.

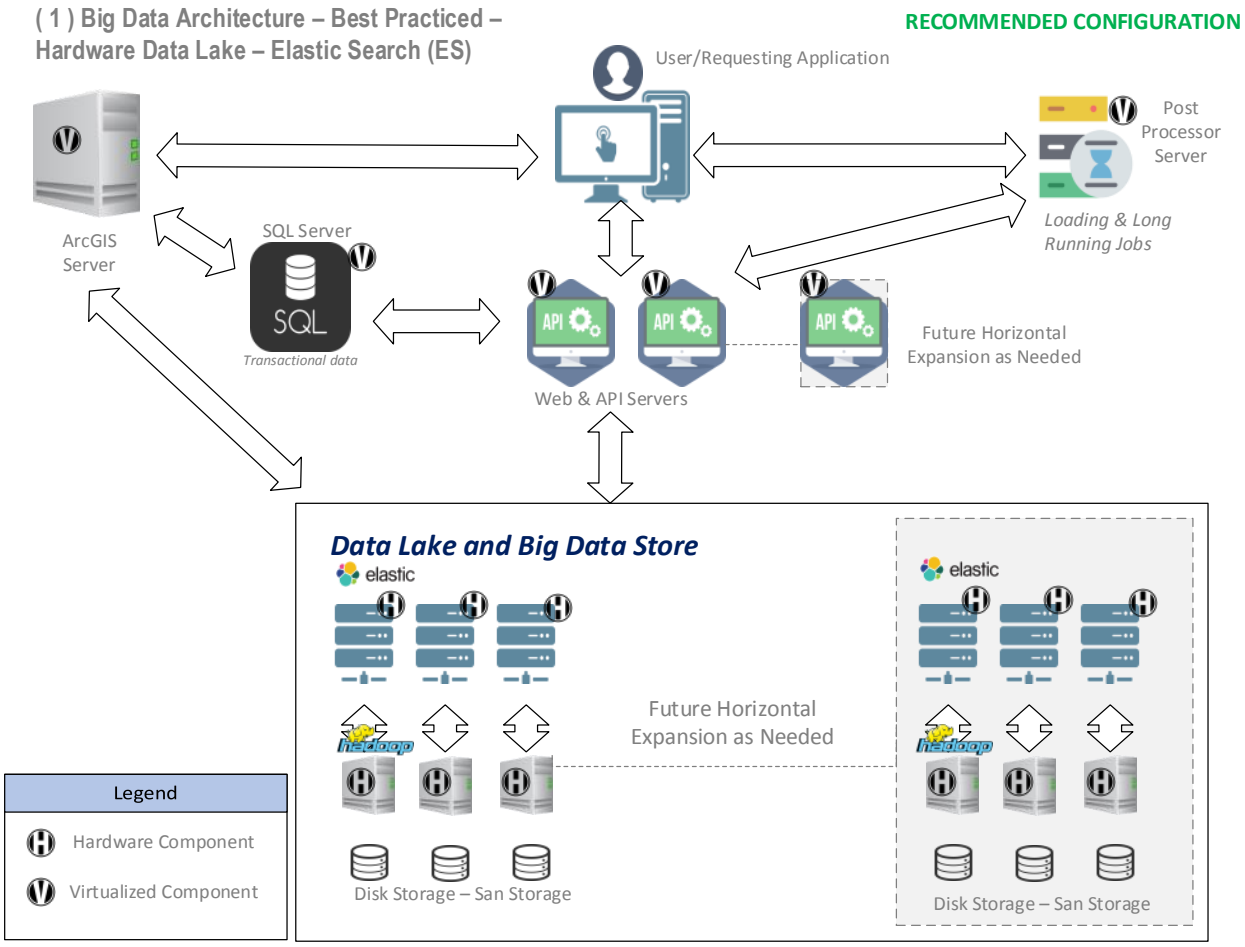
Data Fusion Center – Production Environment

- ArcGIS Server Standard– 1 instance (license required)
    - Included with Standard:
        - ArcGIS for Portal
        - ArcGIS Data Store extension
- GeoEvent processor extension (license required)
- ArcGIS Desktop (Advanced License) – 1 instance
- License Not Required/Open Source
    - Geoprocessing Tools for Hadoop

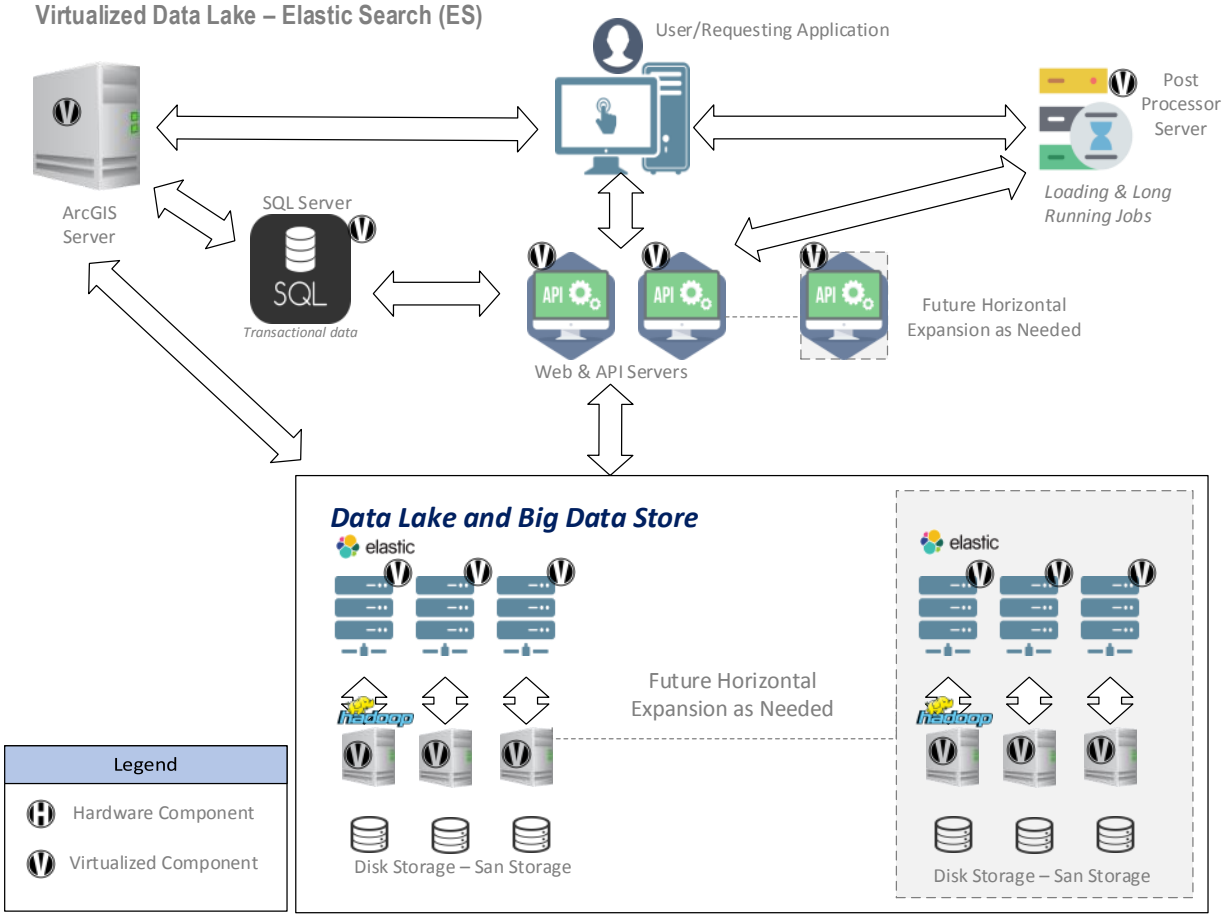
Data Fusion Center – Testing Environment

- EDN license that includes ArcGIS Server
    - Included with EDN:
        - ArcGIS for Portal
        - ArcGIS Data Store extension
- GeoEvent processor extension (license required)
- ArcGIS Desktop (Advanced License) – 1 instance
- License Not Required/Open Source
    - Geoprocessing Tools for Hadoop

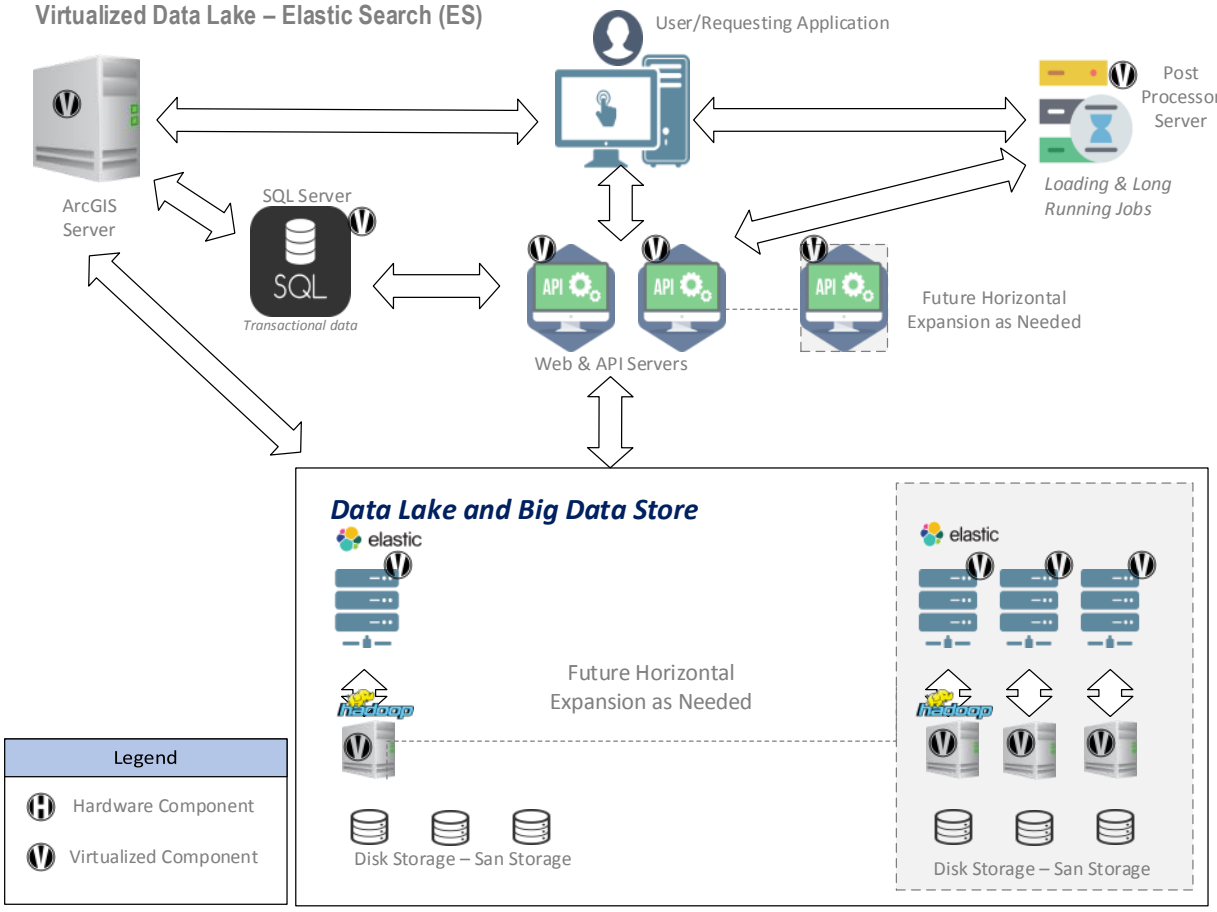
Appendix A: Big Data Architecture Diagrams



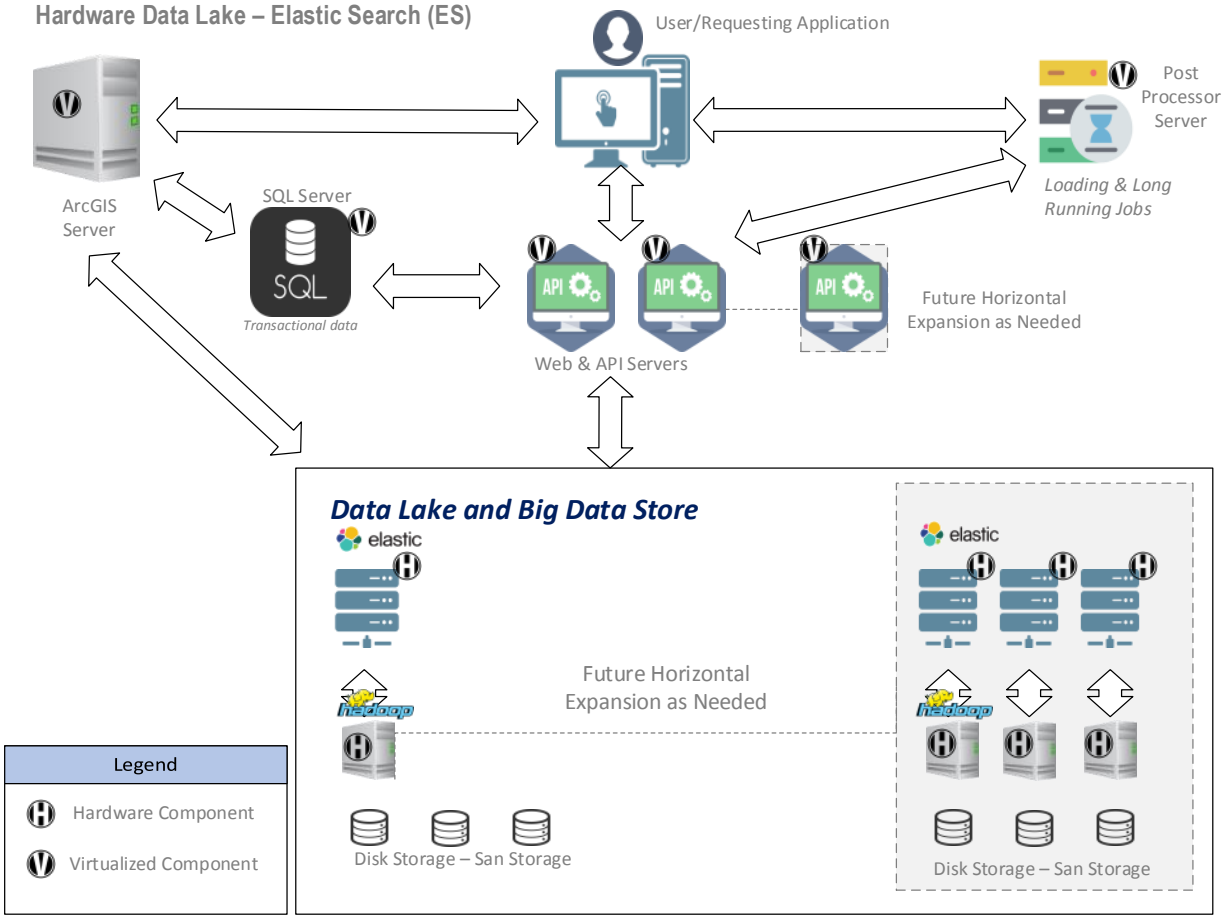
(2) Big Data Architecture – Best Practiced -
Virtualized Data Lake – Elastic Search (ES)



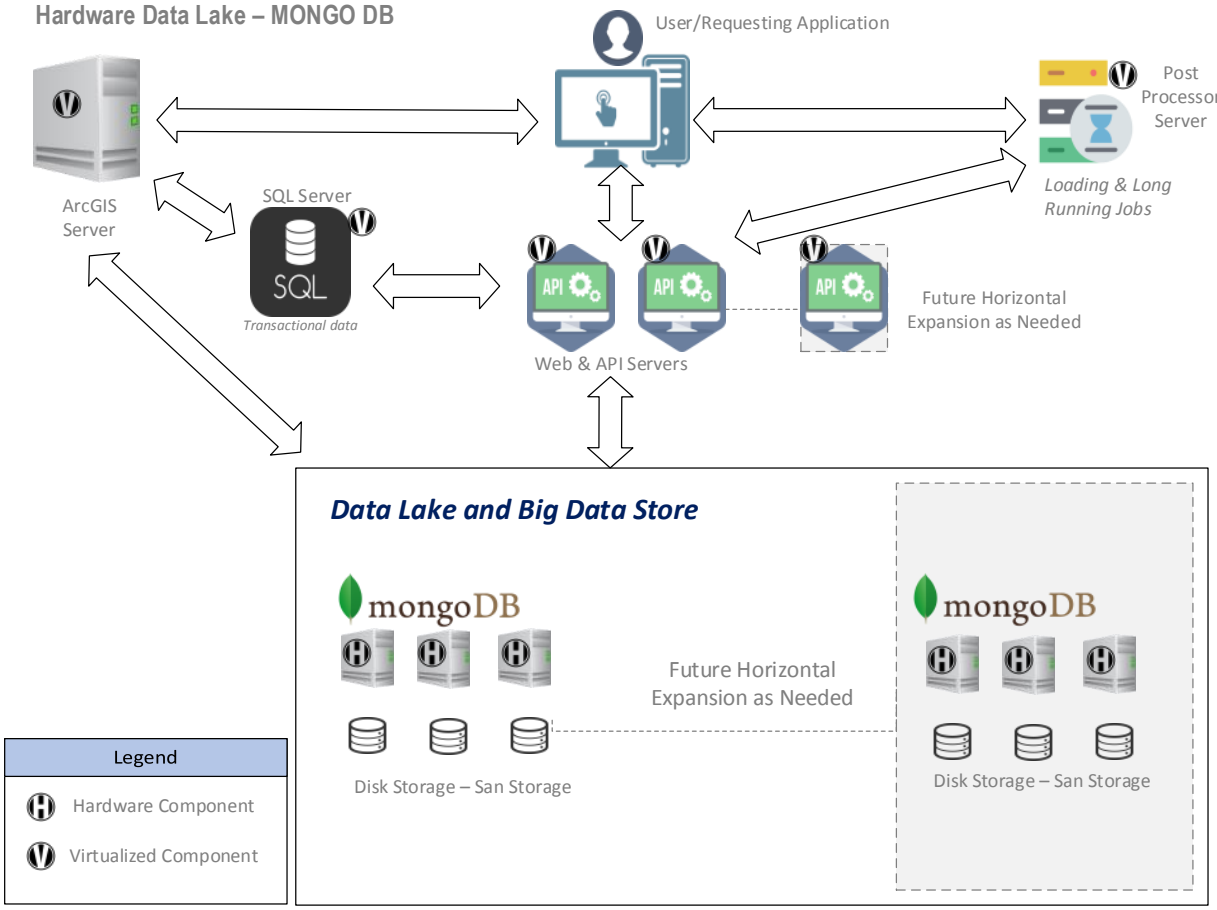
(3) Big Data Architecture – Single Server Setup – Virtualized Data Lake – Elastic Search (ES)



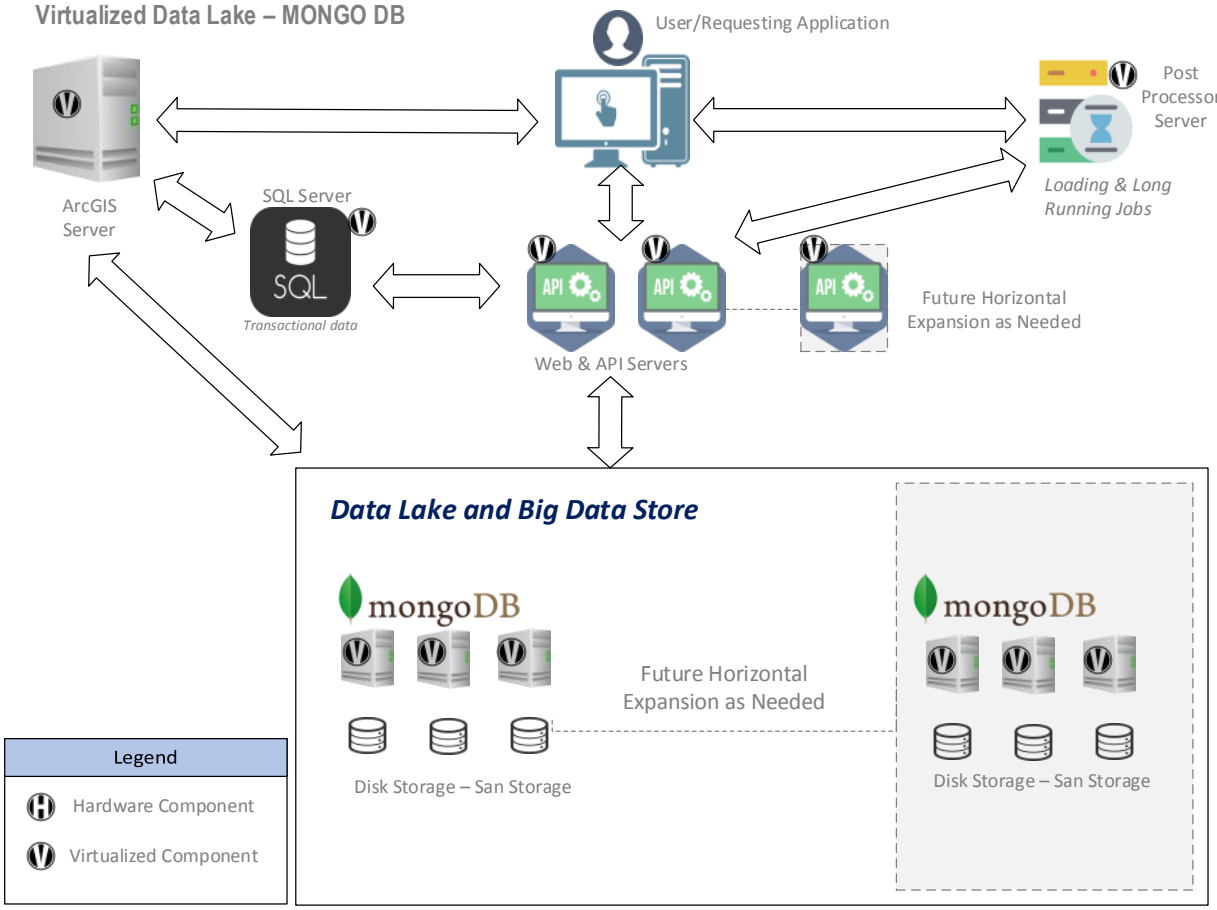
(4) Big Data Architecture – Single Server Setup – Hardware Data Lake – Elastic Search (ES)



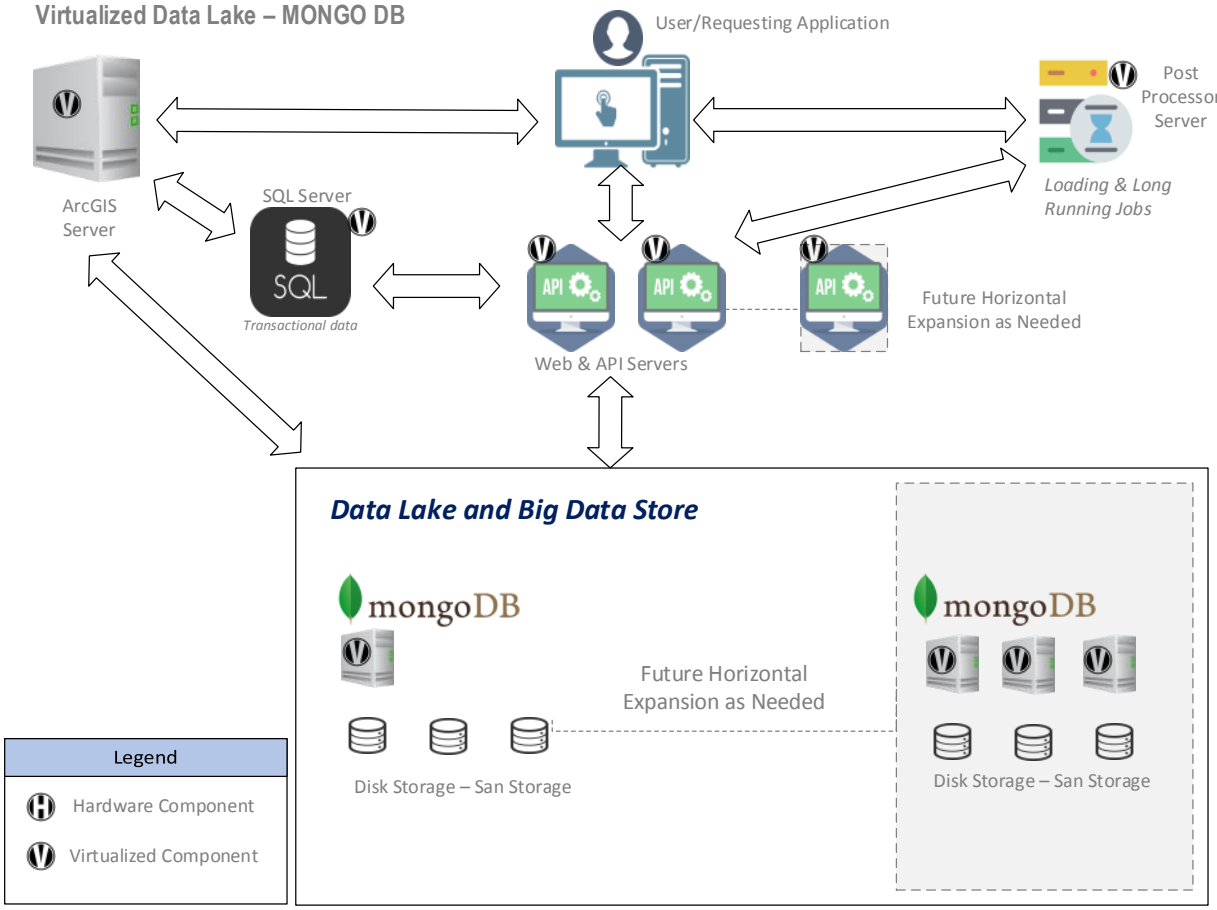
(5) Big Data Architecture – Best Practiced – Hardware Data Lake – MONGO DB



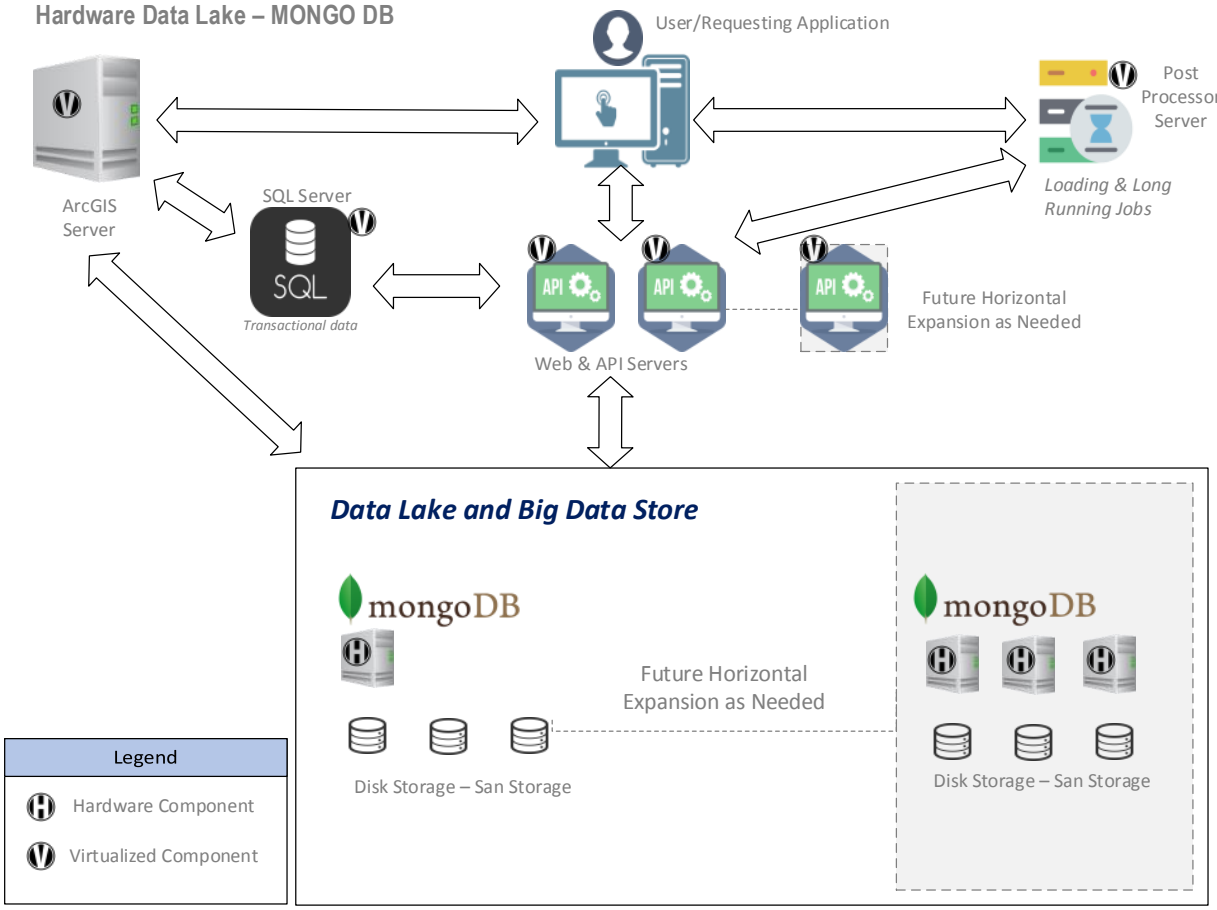
(6) Big Data Architecture – Best Practiced – Virtualized Data Lake – MONGO DB



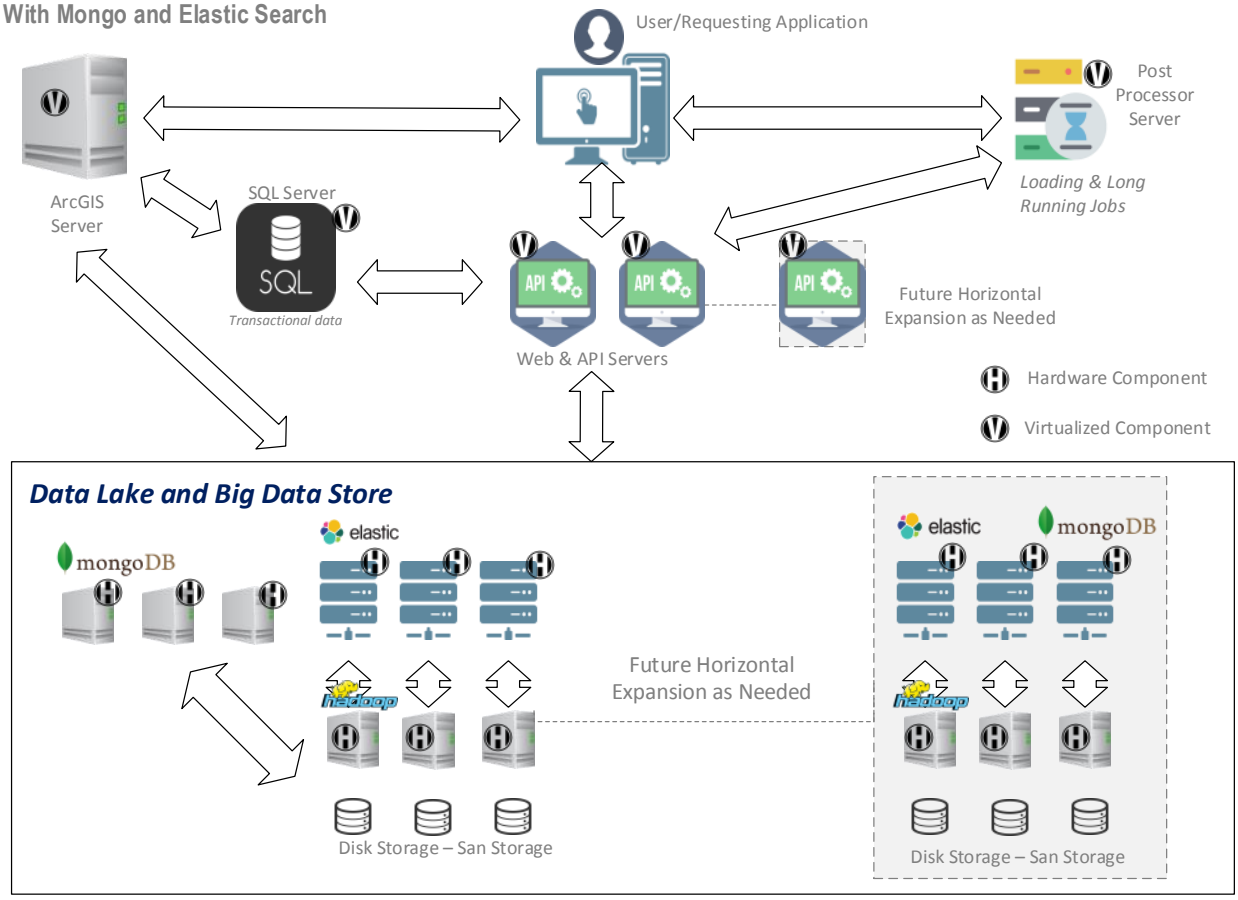
(7) Big Data Architecture – Single Server – Virtualized Data Lake – MONGO DB



(8) Big Data Architecture – Single Server – Hardware Data Lake – MONGO DB



(9) Big Data Architecture – Hardware Data Lake – With Mongo and Elastic Search



Appendix B: Big Data Platforms Comparison Matrix

Big Data Platform Comparsion (MongoDB vs Elasticsearch(ES) vs Hapoop

Prepared By: Claudia Paskauskas, Cedric Gaines and Keith Smith

	MongoDB	Elasticsearch (ES)	Hadoop	Row Definition
Description:	One of the most popular document stores today. A true document store system that allows for ease of use and support from many development languages.	A modern search and analytics engine based on Apache Lucene (a free and open-source information retrieval software library)	Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware.	Quick description of the platform
Database Model:	Document Store	Search Engine	Apache Hadoop is not a single program, tool or application but a set of projects with a common goal integrated under one umbrella / term Hadoop.	Type of database the platform is
Website:	www.mongodb.org	www.elastic.co	hadoop.apache.org	Vendor's Website
Technical Docs:	docs.mongodb.org/manual	www.elastic.co/guide	https://hadoop.apache.org/docs/r2.7.2/	Links to documentation
License:	Open Source (can purchase a support contract)	Open Source (can purchase subscription service that offers more tools and functionalities)	Open Source	Type of license and if support contract are available
Implementation Language:	C++	JAVA	JAVA	Language the platform was written in
Compatible Server Operating Systems (OS):	Linux, OSX, Solaris, Windows	All OS with a Java VM (OS That can run JAVA).	Linux and Windows. (Windows requires Cygwin)	What are the compatible operating systems.
Data Schema:	Schema-free	Schema-free	Schema-free	Must the structure of the data be defined first, or is it free of defined schemas.
Support For SQL Query Language:	No	NO	Yes – Via The HIVE Project	Is the SQL language supported for queries
API & Access Method:	Proprietary protocol using JSON	Java API	Proprietary protocol, RESTful Services and SQL. (Depending on additional projects install on top of the Hadoop instance).	Method and Ways to access the platform's data and query against it.
Supported Languages:	Actionscript	.Net	Java	Development languages that are supported for the platform.
	C	Clojure	C	
	C#	Erlang	C++	
	C++	Go	#	
	Clojure	Groovy	Python	
	ColdFusion	Haskell	Php	
	D	Java	Ruby	
	Dart	JavaScript		
	Delphi	Lua		
	Erlang	Perl		
	Go	PHP		
	Groovy	Python		
	Haskell	Ruby		
	Java	Scala		
	JavaScript			
	Lisp			
	Lua			
	MatLab			

Big Data Platform Comparsion (MongoDB vs Elasticsearch(ES) vs Hapoop

Prepared By: Claudia Paskauskas, Cedric Gaines and Keith Smith

	MongoDB	Elasticsearch (ES)	Hadoop	Row Definition
	Perl			
	PHP			
	PowerShell			
	Prolog			
	Python			
	R			
	Ruby			
	Scala			
	Smalltalk			
Triggers:	No	Yes	Yes, Via Additional Frameworks	Ability to know when things happen within the system data wise. (example: record inserted, record deleted)
Replication:	Yes	Yes	Yes	Is data copied / synced to other nodes within the platform's cluster to ensure data availability and consistency
MapReduce:	Yes	No	Yes	A programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.
Consistency Concept:	Immediate Consistency	Eventual Consistency – May be a small delay.	Eventual Consistency – May be a small delay.	A measure as to when data will be available and if it's partitioned.
User Access Control	Yes	No	Yes	Does the platform understand what a user and group is and can it assign privileges to that user or group.
Ease Of Scalability:	Easy	Medium	High	How easy is it to add nodes to the cluster to allow for horizontal scalability
Ease Of Use / Maintenance:	Easy	Medium	High	How easy is it to maintain the platform cluster.
Need For Multiple Servers:	Yes – For Sharing Purpose (allowing you to split data across multiple servers)	Yes – For Sharing Purpose (allowing you to split data across multiple servers)	Yes, For Replication and Parallel Processing	Are more than 1 machine needed to build platform cluster.
Parallel Processing:	Yes	Yes	Yes	Does the platform allow for multiple nodes within the cluster to process data at the same time (for faster response).
Data Types	All data types (Blob Data via GridFS)	Text Data	All data types	What types of data can the platform manage natively.