**NIST Big Data Public Working Group (NBD-PWG)**

**NBD-PWD-2015/6a-DW-w-HBase-abbreviated**

**Source: NBD-PWG**

**Status: Draft**

**Title: Big Data Use Case #6 Implementation, using NBDRA**

**Author: Russell Reinsch (CFGIO), Afzal Godil (NIST), Wo Chang (NIST), Shazri Shahrir**

To support Version 2 development, this Big Data use case (with publicly available datasets and analytic algorithms) has been drafted as a partial implementation scenario using the NIST Big Data Reference Architecture (NBDRA) as an underlying foundation. In an attempt to highlight essential dataflow possibilities and architecture constraints, as well as the explicit interactions between NBDRA key components relevant to this use case, this document makes some general assumptions. Certain functions and qualities will need to be validated in detail before the undertaking any proof of concept. In this use case, Data governance is assumed. Security is not addressed. Feedback regarding the integration of the disruptive technologies described is welcomed.

## **Use Case #6, part a): Data Warehousing**

### Introduction

# ​Big data is defined as high-volume, high-velocity, and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision-making. Both data mining and data warehousing are applied to big data and are business intelligence tools that are used to turn data into high value and useful information.

# The important differences between the two tools are the methods and processes each uses to achieve these goals. The data warehouse (DW) is a system used for reporting and data analysis. Data mining (also known as knowledge discovery) is the process of mining and analyzing massive sets of data and then extracting the meaning of the data. Data mining tools predict actions and future trends and allow businesses to make practical, knowledge-driven decisions. Data mining tools can answer questions that traditionally were too time consuming to be computed before.

### Given Dataset

2010 Census Data Products: United States (<http://www.census.gov/population/www/cen2010/glance/>)

Census data tools <http://www.census.gov/data/data-tools.html>

### Given Algorithms

1. Upon upload of the datasets to an HBase database, Hive and Pig could be used for reporting and data analysis.
2. Machine learning (ML) libraries in Hadoop and Spark could be used for data mining. The data mining tasks are: 1) Association rules and patterns, 2) Classification and prediction, 3) Regression, 3) Clustering, 4) Outlier Detection, 5) Time series analysis, 6) Statistical summarization, 7) Text mining, and 8) Data visualization.

**Specific Questions to Answer:**

#1. What zip code has the highest population density increase in the last 5 years? And

#2. How is this correlated to unemployment rate in the area?

### Possible Development Tools Given for Initial Consideration

**Big-Data:** Apache Hadoop, Apache Spark, Apache HBase, MongoDB, Hive, Pig, Apache Mahout, Apache Lucene/Solr, MLlib Machine Learning Library.

**Visualization:** D3 Visualization, Tableau visualization.

**Languages:** Java, Python, Scala, Javascript, JQuery.

General components of a data warehousing project

Acquisition, collection and storage of data

Organization of the data

Type of workload

Processing

Analysis of the data

Query

1. **Assumptions**

Data integrity levels, preconditions and minimal guarantees need to be addressed for each of the following components: OS, file system, DB, virtual machines, middleware and applications.

1. **Generalized abstract of the data warehousing in this use case**

Acquire data 🡪 organize and store 🡪 processing 🡪 query and analysis

1. **Flow scenarios**

Main flow scenario for Data Warehousing with HBase, mapped to the NIST BDRA M0437 Activity and Functional Component View diagrams:

|  |  |  |
| --- | --- | --- |
| Role | Activity | Event detail; numeric values; specific technology |
| Application Provider | Collection | Transferring data into DB |
| Infrastructure | Storage | HBase is wide-column |
| Platforms | Index | HBase is organized in a four dimension non-relational format. |
|  | Processing | Hive: Minimal transformation. |
| Analysis Processing | Batch and Interactive | Hive: in memory reservoir or in memory warehouse. |
| Application | Access | Query: aggregate analysis may be MapReduce batch or individual record. |
| Platforms | Read and Update | Read and write workloads |

|  |  |  |
| --- | --- | --- |
| Role | Functional Component | Event detail; numeric values; specific technology |
| Infrastructure | Storage | HBase is wide-column. |
| Platforms | Distributed file system | Method of data organization |
| Platforms | Batch and Interactive | Read and write workloads |
| Generalizing highlight or specific challenge |  |  |
| Reference information |  |  |

Typographic conventions for narrative mapping of the use case to the M0437 diagrams: [below]

Use case Activities that match the BDRA M0437 Activities View diagram are noted in red type.

Use case Functions that match the BDRA M0437 Functional Components View diagram are highlighted in yellow and noted in red type.

Primary roles and actors

1. Business Management Specialist. The role of System Orchestrator in the M0437 diagram may be filled by this actor. Should have previous experience with governance and compliance, and scientific method.
2. General Data Scientist. The role of System Orchestrator in the M0437 diagram may be filled by this actor. Should have experience with various types of data, NLP, ML, systems admin, DB admin, front end programming, optimization, data mgmt., data mining, and scientific method.
3. Developer / Engineer. Should have experience with product design, various data types, distributed data, systems admin, DB admin, cloud mgmt., back end programming, front end programming, math, algorithms, and data management.
4. Research Scientist or Statistician. This role is likely very similar to or identical to the Data Consumer or Data Provider represented in the M0437 diagrams. . Should have experience with machine learning, optimization, math, graphical models, algorithms, Bayesian statistics, data mining, statistical modeling, and scientific method.

**Main Flow Scenario**

Gathering and acquisition of data:

One of the first things to address in this use case is identification of the data itself.

A) the actor serving in the Data Provider role will sometimes need to identify what type of data is in the given dataset (JSON, XML?) that can be acquired from the and stored as the data type is not always identified by the Census Bureau.

B] the Data Provider must also consider the expected total size of the data captured from the Census Bureau.

Sqoop can be used as a connector for moving the data from the Census Bureau to the storage reservoir [in this use case, HBase]. Sqoop and another ingest technology Flume both have integrated governance capability. While Flume can be advantageous in streaming data cases requiring aggregation and web log collection, Sqoop is especially well suited for batch and external bulk transfer. The Data Provider will be responsible for the coding required to implement the Sqoop connection. The data transfer activity discussed here matches with the Big Data Application container, Collection oval in the BDRA Activities View diagram.

Storage: As a Bigtable that scales horizontally and capable of hosting billions of rows, HBase is appropriate for economically persisting the dataset from the Census Bureau [may in fact be too big for this use case]. Please see the Organization section below. The storage activity discussed here matches with the Infrastructure container, Store oval in the BDRA Activities View diagram. The storage function discussed here also matches with the Infrastructure container, Storage box in the BDRA Functional Components View diagram.

Hardware requirements: a cluster of five servers is required for storage.

Organization of the data:

HBase uses a four dimension non-relational format consisting of a row key, a column family, a column identifier and a version. The organization activity discussed here matches with the Platforms container, Index oval in the BDRAAV diagram. The organization function discussed here also matches with the Platforms container, Distributed File System box in the BDRA Functional Components View diagram.

Type of workload:

HBase is suited for high I/O, random read and write workloads. The read access is the primary activity of interest for this use case. HBase supports two types of processing and access. Batch processing and aggregate analysis can be performed with MapReduce; and interactive processing and random access to a single row or rows [a table scan] can be performed through row keys. HBase 0.98.9 and 0.98.12 are compatible with MapReduce version 2.7. The MapR Ecosytem Support Matrix can be consulted for compatibility between earlier versions of these components. The read [and write] activities discussed here match with the Platforms container, Read [and Update] oval[s] in the BDRA Activity View diagram.

The read [and write] functions discussed here match with the Platforms container, Batch and Interactive box[es] in the BDRA Functional Components View diagram.

Architecture maturity: (level of unification between a raw data reservoir, and DW). In this use case, the HBase reservoir does not need to be combined with an information store or formalized DW to form an optimized data management system.2

Processing:

ETL vs. direct access to the source. With Hive, there should be no need to transform the data or transfer it prior to sandboxing and data warehousing. See the Query section for more on Hive. The processing activity discussed here matches with the Processing container in the BDRA Activities View diagram.

Hardware requirements: a high speed network is not required here for data transport in this use case [because of Hive].

In cases where more comprehensive ETL functionality would be required [or should it be required here], some of the additional ETL functionality may be provided by the access layer, which could also be Hive, or MapReduce, or potentially another application other than the MapReduce processing. MapReduce covers a wide range of stack functions, capable of serving as a storage layer computing engine, a transformation layer, or a technique for accessing the data.

Hive versions 0.13, 1.0 and 1.2.1 are compatible with HBase versions 0.98.9 and 0.98.12.

The Data Application Provider will qualify whether or not the data needs to be modeled prior to analysis. If so, whether minor transformations can be added in reservoir; or whether during ETL to the DW, or in the DW itself.

Analysis:

With Hive, analysis can occur in place whether in reservoir or warehouse. In this use case, Hive would serve as an application (data warehouse infrastructure/framework) that creates MapReduce jobs on both the staging area [reservoir] and the discovery sandbox [DW], providing query functions; and if necessary the data could be transferred after initial analysis, unmodeled, to a discovery grade data visualization technology next layer up [Qlik, Spotfire, Tableau, etc.] which would be able to complete additional descriptive analysis without ever modeling the data. The analysis activity in Hive matches with the Processing container in the BDRA Activities View diagram, both Batch and Interactive ovals.

Hardware requirements: in memory h/w is required for this sandbox function.

Performance [Processing and Access layers]: Assumption: the data will be at rest.

Query:

HBase allows users to perform aggregate analysis [through a MapReduce batch] or individual record queries [through the row key]. The query activity discussed here matches with the Big Data Application container, Access oval in the BDRA Activity View diagram.

A second option for ad hoc query, is to go through a “distribution” (Enhanced HBase SQL query almost always involves going through a distribution). A distribution would be an option if the agency piloting the POC lacks personnel with an open source skill set qualified for the analysis in this use case; distributions greatly reduce the complexity. Distributions are covered in more detail in the last section. A third option involves bypassing MapReduce and Hive.

Alternatives to SQL, also referred to as SQL Layer Interfaces for Hadoop: OS technologies Apache Hive and Pig have developed higher layer scripting languages that sit on top of MapReduce and translate queries into Jobs used to pull data from HDFS.

Background on Hive: HiveQL is a declarative query language originally spinoff from Yahoo then developed by Facebook for analytic modeling in Hadoop. Purpose built for petabyte scale SQL interactive and batch processing, Hive is usually used for ETL or combining Hadoop with structured DB functions equivalent to SQL on relational DWs.

HDFS, Hive, MapReduce and Pig make up the core components of a basic Hadoop environment. Each has satisfactory batch and interactive functionality, but by no means create a panacea for all things big data. Pig and HiveQL are not good for more than basic ad-hoc analysis. HiveQL will execute NoSQL queries on HBase via MapReduce, however HBase’s own documentation [FAQ] describes MapReduce as slow. Hive does not make advancement on the limitations of batch processing.

State: consistency: ACID is probably not a requirement for the processing in this use case but in the event project planners were considering types of applications to deploy, support for row level ACID is built in to HBase, however batch level ACID support must be handled by the reporting and analysis application. Zookeeper is an option for coordinating failure recovery on distributed applications.

Subflows: Beyond the data warehouse: further analysis; visualization and search applications.

The first part of this use case (a) has discussed basic descriptive analysis. For advanced analysis tasks and data mining, including Apache Mahout, MapReduce enabled analytical tools Radoop and Weka; and scripting languages R and Python as well as Python packages Matplotlib, Numpy and Scipy, please refer to companion documents for this use case, (part b), Data Mining. These advanced analytics activities will match with Big Data Application container, Analytics oval in the BDRAAV. These advanced analysis functions match with the Big Data Application container, Algorithms box in the BDRAFCV.

Visualization activities match with Big Data Application container; search matches with the Infrastructure container, Retrieve oval in the BDRAAV diagram. The Visualization function matches with Big Data App container Visualization box in BDRAFCV diagram.

1. **Background on HBase:**

HBase is an Apache open source (OS) wide column DB written in Java. HBase runs NoSQL programs directly on top of the Hadoop Distributed File System (HDFS) to provide analysis capabilities not available in HDFS. A Java API facilitates programming interactions. Linux / Unix are the suggested operating system environments.

Well suited for OLTP and scanning tables with rows numbering in the billions, HBase works perfectly with HDFS and allows for fast performance of transactional applications, however it can have latency issues due to its memory and CPU intensive nature.

Assumption: that connection speed [latency] between components is not an issue is not an issue in this use case.

TCO concerns for this use case: HBase can be memory and CPU intensive. Hadoop and MapReduce implementations can easily turn into complex projects requiring expensive and scarce human resources with advanced technical expertise to write the code required for implementation.

Skills required: The human resources with the skill sets for managing such technical projects are in short supply and therefore their cost is high; which can offset the advantage of using the lower cost OS technology in the first place. Hiring demand for HBase programmers is second only to Cassandra; indicating acute scarcity. MapReduce requires significant technical know-how to create and write Jobs; for small projects that use MapReduce, the TCO may not necessarily be lower.

Alternative technologies and strategies that could be appropriate substitutes in this use case:

Accumulo: the closest substitute to HBase. Developed by the National Security Agency (NSA), Accumulo is now an open source distributed database solution for HDFS. Thanks to its origin, this technology takes a serious approach to security, going down to the cell level. Written in Java and C++. Any or all key value pairs in the database can be tagged with labels which can then be filtered during information retrieval, and the overall performance is not affected.

Although the user API is not an interface that can be considered easy to connect to other applications, the Apache Thrift protocol expands Accumulo’s basic capability to integrate other popular application building programming languages, and Zookeeper for MapReduce jobs. Sqrrl expands Accumulo’s basic capabilities in security, information retrieval and ease of use.

Sqrrl is a NoSQL DB that uses Accumulo as its core, and also utilizes other columnar, document and graph NoSQL technologies as well as HDFS and MapReduce. Real time full text search and graph search are complimented by extensible automated secondary indexing techniques. Sqrrl Enterprise boasts strong security and an environment that simplifies the development of applications for distributed computing, reducing the need for high end engineering skills. App for Hunk. Cambridge, MA.

Cassandra is another OS wide column option. Cassandra is a Java based technology originally developed by Facebook to work with their own data. [Facebook eventually replaced Cassandra with HBase and Hadoop, and released the Cassandra code to open source]. Modeled after Google BigTable, the technology combines well with enterprise search, security, and analytics applications, and scales like crazy, making it a good choice for use cases where the data is massive. Cassandra has its own query language, CQL; users will need to predefine queries, and shape indexes accordingly. The interface is user friendly. Short and long term demand for programmers with Cassandra skills is very high, on par with MongoDB4.

Hypertable: written in C++, licensed under GPL 2.0, sponsored by Baidu. Less popular, but faster than HBase.

Hadoop Distributions with Analytic Software Platforms:

Cloudera, Hortonworks and MapR are known as the big three independent platforms for on-premise analytics preformed on distributed file based systems. The use of the term independent in this case means a commercial bundle of software and supporting applications in versions that have been tested for compatibility, plus support services to go along with the software bundle. Typically referred to as a “Distribution,” these bundles may provide all the resources required to make a Hadoop or other distributed system useful. Other vendors in this space include EMC Greenplum, IBM (IDAH), Amazon Elastic MapReduce (EMR) and HStreaming.

Among the big three, MapR Enterprise DB has the strongest operations of KV solutions, and one of the best if not the best integration with Hadoop and HBase, including an HDFS data loading connector. MapR Technologies also provides a SQL Layer Solution for MapReduce as discussed in the Query section.

The MapR solution is not entirely OS but some balance of proprietary and OS technology which provides some advantages, in the form of readymade capabilities that are potentially lacking in Hortonworks and Cloudera. These include an optimized metadata management feature with strong distributed performance and protection from single point of failure; full support for random write processing; and a stable, node based job management system. Write processing is not applicable in this use case.

Areas not yet addressed in this, part of use case #6:

Background on integration technologies, Hcatalog for Metadata;

Required monitoring and security and privacy functions;

Concerns driving the use case.

Types of analytic libraries that could be used to do the data mining can be found in part b.

Related documentation and references:

1. <http://docs.media.bitpipe.com/io_10x/io_108315/item_639580/07.29.13FINAL%20Hadoop%20Platforms.pdf>
2. Oracle: Information management and big data – a reference architecture.pdf
3. Documentation on programming HBase with Java and an HBase data analysis with MapReduce (Haines): <http://www.informit.com/authors/bio/d07f5092-99eb-4de8-97a4-a876a60b3724>
4. The HBase data model: <http://internetmemory.org/en/index.php/synapse/understanding_the_hbase_data_model>
5. Apache configuration documentation: <http://hbase.apache.org/book.html#hbase.secure.simpleconfiguration>
6. NoSQL skills demand: Diana: <http://regulargeek.com/2012/02/23/nosql-job-trends-february-2012/>