DATA WAREHOUSE AND BIG DATA INTEGRATION

Sonia Ordoñez Salinas and Alba Consuelo Nieto Lemus

Faculty of Engineering, Distrial F.J.C University, Bogotá, Colombia

ABSTRACT

Big Data triggered furthered an influx of research and prospective on concepts and processes pertaining previously to the Data Warehouse field. Some conclude that Data Warehouse as such will disappear; others present Big Data as the natural Data Warehouse evolution (perhaps without identifying a clear division between the two); and finally, some others pose a future of convergence, partially exploring the possible integration of both. In this paper, we revise the underlying technological features of Big Data and Data Warehouse, highlighting their differences and areas of convergence. Even when some differences exist, both technologies could (and should) be integrated because they both aim at the same purpose: data exploration and decision making support. We explore some convergence strategies, based on the common elements in both technologies. We present a revision of the state-of-the-art in integration proposals from the point of view of the purpose, methodology, architecture and underlying technology, highlighting the common elements that support both technologies that may serve as a starting point for full integration and we propose a proposal of integration between the two technologies.

KEYWORDS

Big Data, Data Warehouse, Integration, Hadoop, NoSql, MapReduce, 7V's, 3C's, M&G

1. INTRODUCTION

Information is one of the most valuable resources of an institution, and adequate use to support decision making has become a challenge of ever increasing complexity. Enterprises invest in solutions that allow them to use big data in the best possible way, to generate new business strategies, improve customer service or develop public policies, among many other uses.
Nowadays the data volume required to be processed within an enterprise can reach the order of the Exabyte [1]. This poses storage and processing challenges that require new technological solutions that allow not only storage, but also updating, efficient exploitation and that have into account data requirements. This is sometimes referred as the seven V´s [1]: Volume, Variety, Velocity, Veracity, Value, Variability and Viability and the three C´s [1]: Cost, Complexity and Consistency.

Given the limitations of traditional techniques used so far and the new data requirements, enterprises face several challenges in managing large volumes of data. The concepts of Data Warehouse and Big Data tend to blend and it is not easy to find a divide between them. While Data Warehouse is a mature management paradigm supported by widespread and well-established methodologies [2] [3] [4], Big Data is still a field under development, which seeks to address individual aspects of the problem but still lacks an integral solution. As a result of the art state review, we can conclude that some articles present Big Data as the Data Warehouse replacement, others as Data Warehouse evolution [5], some propose the extension of Data Warehouse to support some Big Data characteristics and others partially explore the possibility of integrating the two.

This work presents a critical review of the elements that characterize the two technologies and that could allow their convergence in an architectural model that considers the processes of ingestion, pre-processing, validation, storage and analysis of the different data types and data sources that organizations are currently facing. The result of the analysis leads to the conclusion that integration is possible only if the different types of data, their life cycle and treatment are explicitly considered. This integration is materialized in the proposal of a multi-layered architecture model that provides a systematic solution, recurrent in time and not, in an isolated way.

This article has been divided into the following sections: section 2 reviews the purpose and scope of the two technologies; section 3 is a review of the methodologies used for the development of Data Warehouse (DW) and Big Data (BD); section 4 reviews architectural models for DW and BD from the point of view of the sources, ETL processes, storage, processing and associated technologies; in section 5, are discussed the characteristics describing one and the other; section 6 refers to the Multilayer Staggered Architecture Model for Big Data, and finally section 7, conclusions.

2. PURPOSE AND SCOPE OF BIG DATA AND DATA WAREHOUSE

Data Warehouse emerged in the 80's as an alternative for storing and organizing data in a consolidated and integrated manner, allowing users to perform statistical analysis and business intelligence. The term Big Data was coined in 1997 by Michael Cox and David Ellsworth [6], NASA researchers who had to work with generally very big data sets, which overloaded the principal memory, local disc and remote disc capacity. They called this the Big Data problem.
Despite being so widely referenced today; Big Data doesn't have a rigorous and agreed upon definition. It is usually associated with the treatment of massive data, extracted from different sources and without predefined structures. For some authors, Big Data is nothing more than a data set which size is beyond the typical databases tools to capture, store, manage and analyze. Unlike Data Warehouse, Big Data goes beyond information consolidation because it is used mainly for the storage and processing of any type and volume of data with a volume that potentially grows exponentially. Nevertheless, what is concluded in this paper is that both Data Warehouse and Big Data have a common ultimate, goal: data exploration with the purpose of describing situations, behaviours, look for patterns, relationships and inferences.

Data Warehouse has as a principle the integration and consolidation of the information in a rigid multidimensional structure. One example is the snowflake model [2] [3], used to do Online Analytical Processing (OLAP) [7] applying Business Intelligence (BI). On the other hand, Big Data does not have as a principle the consolidation and integration under predefined structures, it is more about the storage and management of large volumes of raw data (of types, sources and heterogeneous arrival speeds [69]), for which a distributed infrastructure and a set of specialized hardware and software is required. The processing and data analysis use advanced techniques of data science, in which consolidation is irrelevant, as this depends on the nature of the data and the particular problem.

3. METHODOLOGY

A fact that motivates this analysis must do with how projects associated with Data Warehouse and Data Big develop. While there are methodologies for the development of projects with DW that are widely used, such as the life cycle of data warehouse of Kimball [2], the point methodology of Todman [3] and the flow model enterprise reference of Inmon [4], these have fallen short when predicting exponential growth and the changing nature of the data because great efforts are required to modify or include new requirements. Some less known methodologies propose to include heterogeneous data types, such as streams and geo-referenced data for the multidimensional modelling [9], but they still do not cover the entire life cycle of DW.

Data Warehouse is considered a mature technology, widely supported in the research field and with proved results at the organizational level in multiple business contexts. BD does not yet have a standardized and widespread terminology [10]; this problem is being addressed by the standardization group NIST Big Data Working Group (NBD-WG) [11] whose results have not yet been published.

Big Data is newer than DW and there are not still standardized proposals for its development. It is presumed it will be necessary, besides resolving the same problems and challenges present in Data Warehouse building methodologies, to consider the development life cycle, non-structured data, heterogenic data sources and no transactional data in general, as well as a fast adaptation to change.

Currently the projects seeking to extract added value from the data must consider the V's and C's characteristics mentioned before. They can result to be complex and it is therefore necessary to adopt management strategies for their development, maintenance and production support. Governance policies should be established to reach agreements, create communication mechanisms between different actors (internal and external) and include adaptation to change, management standards, control restrictions and adoption of best practices throughout the life cycle of a Big Data project and general management metadata. Additional to the V`s and the C`s, Management and Governance (M&G) characteristics should be considered (see figure 1).

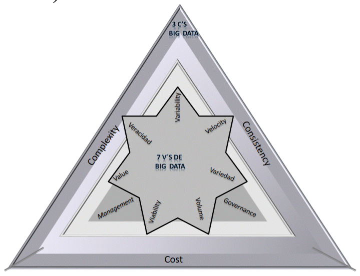

Figure 1. Big Data characteristics (Source: Author)

C's and V's characteristics are explicitly made evident for Big Data, even when no methodologies for its development and no integrated framework capable of systematically solving any requirement exist (independently of the knowledge and expertise of the user). For the traditional DWs, V's and C's characteristics are not yet explicitly evident, and even when are already considered in software suites for technological development (tools), the methodologies do not contemplate the role they play in the development life cycle. There are certainly incentives for the integration of Big Data and Data Warehouse in one unique solution, but so far the definition of new technologies capable of handling the architecture, processing and data analysis is required.

4. ARCHITECTURE

From the point of view of the logical abstraction of architecture, both DW and BD have the same components: Data Sources, Extraction, Transformation and Loading processes (ETL), storage, processing and analysis. Because of this, an overview of the architecture in terms of these components is presented below.

4.1. Sources and data types

4GL technologies (fourth generation languages) facilitated the development of transactional applications that allowed the automation algorithms on repetitive structured data. Structured data (SD) is characterized for being well defined, predictable and soundly handled by an elaborated infrastructure [12].

Technological developments, digitization, hyperconnected devices, and social networks, among other enablers, brought unstructured information to the scope of enterprises. This includes information in digital documents, data coming from autonomous devices (sensors, cameras, scanners, etc.), and semi-structured data from web sites, social media, emails, etc. Unstructured data (USD) don't have a predictable and computer recognizable structure, and may be divided into repetitive and non-repetitive data [12]. Unstructured repetitive data (US-RD) are data that occur in many occasions in time, may have a similar structure, are generally massive, and not always have a value for analysis. Samples or portions of these data can be utilized. Because of its repetitive nature, processing algorithms are susceptible of repetition and reutilization. A typical example of this category is data from sensors, where the objective is the analysis of the signal and for which specific algorithms are defined.

Unstructured unrepetitive data (US-URD) have varying data structures, which implies that the algorithms are not reusable (and the task of predicting or describing its structure is already a complex one). Inmon places elements of textual nature (that require techniques from Natural Language Processing and computational linguistics) inside this category [12]. From our perspective, besides free-form text, imagery, video and audio also pertain to this category.
Traditional DWs were born with the purpose of integrating structured data coming from transactional sources and to count with historical information that is supported by OLAP-based analysis. With the upcoming of new data types, some authors propose DW to adapt its architecture and processes, as suggested in Inmon with DW2.0 [13] and Kimball in The Evolving Role of the Enterprise Data Warehouse in the Era of Big Data Analytics [14].

4.2. Extract-Transform-Load processes (ETL)

Construction of Data Warehouse requires Extraction, Transformation and Loading processes (ETL). These must consider several data quality related issues, as for instance duplicated data, possible data inconsistency, high risk in data quality, garbage data, creation of new variables using transformations, etc. That raises the need of specific processes to extract enough and necessary information from the sources and implementing processes for cleansing, transformation, aggregation, classification and estimation tasks. All these, besides the utilization of different tools for the different ETL processes, can result in fragmented metadata, inconsistent results, rigid models of relational or multidimensional data, and thus lack of flexibility to perform generic analysis and changes [15].

Thus, the need of more flexible ETL processes and improved performance gave birth to proposals such as real time loading instead of batch loading [16]. Middleware, for instance the engine of flow analysis, was also introduced. This engine makes a detailed exploration of incoming data (identifies atypical patterns and outliers) before it can be integrated into the cellar. On the same line is the Operational Data Storage (ODS) [17], that proposes a volatile temporal storage to integrate data from different sources before storing it in the cellar. The work presented in [18], unlike the traditional architectures, creates a ETL subsystem in real time and a periodic ETL process. Periodic ETL refers to the periodic importation in batch from the data sources and the ETL in real time. Using Change Data Capture (CDC) tools, changes in the data sources are automatically detected and loaded inside the area in real time. When the system identifies that certain conditions are met, data are loaded in batch into the cellar. The stored part can be then divided in real time and static area. Specialized queries for sophisticated analyses are made about storage in real time. Static data are equivalent to DW and historical queries are thus handled in the traditional way. It`s worth to mention that for DW some changes have been observed, including temporal processing areas and individualized processes according to the data access opportunity.

4.2.1. ETL requirements for new data

With the need to manage unstructured repetitive data (US-RD) and unstructured unrepetitive data (US-URD) coming from diverse sources (like the previously mentioned) new requirements [73] were raised, among which we may count the following:

- Managing exponential data growth. DW is characterized for using transactional databases of the organization as the main source, eventually flat files, and legacy systems. Although this data volume grows, it does so at a manageable pace. The new DW and BD provide solutions to the management of large data collection by the MapReduce programming model [19], which allows to parallelize the process for then gather the partial results; all this supported in a distributed file system like Hadoop Distributed File System (HDFS) [36].
- The frequency of arrival. This can range from periodic updates to bursts of information. Traditional DWs does not face this problem, since it always focuses on data that can be extracted or loaded in a periodic and programmed way. Since BD was designed to receive all the coming information at any moment in time, it must use any required amount of memory, storage and processing.
- Longevity, frequency and opportunity of use. Statistically, the most recently generated dataset will be used more frequently and in real time. As datasets become old, new data are added and thus the frequency of use may decrease. But old data cannot be ruled out because it can be always used for historical analysis [12] [20].
- Integration of data. While the traditional DW was intended to integrate data across a multi-dimensional model, the appearance of unstructured repetitive data (US-RD) raised problems related to find adequate ways to group the data under a context independent of the data type. For example, to group pictures with dialogues, even within the same type of data (to determine the context of an image by the same image) to find the structure that best represents all the data and do algorithms to integrate, transform and represent such data[70]. With unstructured unrepetitive data, in addition to identifying the context and structure, an algorithm for each dataset may be required, which prevents reuse and increases complexity [71]. The integration of heterogeneous datasets may be the main difference between DW and BD. In DW, the underlying purpose of integration is to have a global and uniform vision of the organization, while in BD, integration is not the ultimate goal. For BD, some unstructured datasets not amenable to integration should be kept in raw format, allowing the possibility of further uses that may be not foreseeable now.

4.2.2. ETL for new data life cycle

Seeking response to the characteristics of the new data, Inmon´s proposal in DW2.0[13] defines the life cycle of the data for BD and proposes three storage and processing sectors. First is the interactive sector where most of new data resides, the update is done online and a high response time in performance is required. Second, the integration sector where the interactive sector data is integrated and transformed. In this sector, data can remain longer depending on the needs of the organization. And third, the archive sector, which maintains historical data and has a lower access probability. Similarly, Kimball [20] presents the data highway, consisting of 5 caches arranged per the frequency and longevity of data: a) raw and immediate use data; b) real-time data and frequency of use in seconds; c) data for business monitoring and frequency of use in minutes; d) data to generate reports for business decisions and frequency of use in hours and e) aggregated data to support historical analysis and frequency of daily, monthly and annual use. Mayer`s proposal [21] defines a reference architecture based in components that allow handling of all kind of data: acquisition, cleaning, integration, identification, analysis and management of data quality. It also includes transversal components for data storage, metadata, lifecycle and security handling.

As for products, already in the market, it`s worth to mention the following. Oracle [22] implements a global proposal including both structured and unstructured data and defines different storage areas where the lifecycle of the data is done in a similar way as proposed by Inmon and Kimball. The proposal relies in a set of tools that permit data gathering, organization, analysis and visualization. SAP Data Warehousing offers a solution that integrates features of BD and DW in real time allowing the analysis and identification of patterns in complex structures of both structured and unstructured data. SAP supports ETL processes in the SAP NetWeaver tool, which allows to integrate data from different sources [23]. Pentaho as free software platform includes the component that allows to do the typical ETL processes for DW and, through Hadoop, supports ETL for BD [24]. The mentioned solutions consider the life cycle of the data, but are focused more on the technology handling than on integration architectures and the methodology itself.

4.3. Storing, processing and analysis

Data Warehouse systems have traditionally been supported by predefined multidimensional models (star [2] [3] and snowflake [3] [4]) to support Business Intelligence (BI) and decision making. These models are generally implemented on relational databases (Relational Online Analytical Processing ROLAP) and managed through Structured Query Language (SQL) [8]. Less frequently, implementations under multidimensional schemes (Multidimensional Online Analytical Processing MOLAP) are also found. Although the traditional DW manage large amounts of information, its architecture is supported on client-server models that can only be scaled in a vertical way, implying massive technological and economic efforts for both its development and maintenance. Contrary to this, BD and the new DW generation neither have predefined analytical models [72], nor rely on client-server architectures and must support the horizontal scaling. The answer to the new needs is the use of extensive memory, data distribution and processing parallelization, which in one way or another are included in Hadoop, MapReduce, NoSQL databases, storage and processing in memory and technologies complementary to these.

4.3.1. Hadoop and MapReduce

To support parallel and distributed processing of large volumes of data, most solutions involve Hadoop and the MapReduce algorithm [25] [26] [27] [28] [29]. Hadoop is a framework [30] based on distributed processing of large data volumes across multiple clustered systems. This distribution is based on a file system, Hadoop Distributed File System (HDFS) that provides high performance access to data, is scalable, and offers high availability and tolerance to failures by replication [31]. To ensure parallel processing, most solutions suggest the use of MapReduce, which with the function Map transforms a dataset in hash pairs to distribute the data segments in different nodes of a cluster, and in this way, parallelize processing. After processing, the segments are combined into a single result using the reducer function. The algorithm was initially implemented by Google to solve PageRank processing [32], but the most referenced implementation is Apache Hadoop [26] [25].

Hadoop has already been incorporated into both commercial and free suites. Oracle uses Hadoop to extract data from an Oracle RDBMS database, process it, transform it and load it in a HDFS

[33] system; Pentaho incudes Hadoop for the integration of data and business intelligence [24]; Microsoft has developed a connector that allows data transfer between Hadoop and Sqlserver [34].

Under the same philosophy of data distribution and parallel processing, there are alternatives to Apache Hadoop like GreenHDFS that implement a strategy that divides the servers in the Hadoop cluster in hot and cold regions to distribute the data of lower activity to cold areas. Tests with traces of Yahoo, resulted in energy savings close to 24% [35]. Spark [36] implements an extended version of MapReduce known as Map-Shuffle-Reduce, that primarily works in memory, and through a cyclical directed graph allows to execute sequences of Map -Shuffle - Reduce - Shuffle - Reduce type, improving the performance of the programing model MapReduce [37] [38].

4.3.2. NoSql databases

The term NoSql refers to a set of management systems based on non-relational structures that facilitate horizontal scaling and unstructured data management. To improve the performance these databases, transactional ACID compliance properties (Atomicity, Consistency, Isolation, Durability) are not granted, adhering instead to the BASE design principle (basic availability, soft state, eventually consistence). The CAP theorem states that a distributed system can only simultaneously provide two properties between consistency (C), availability (A) and tolerance to partition (P) [39]. ACID compliance retains consistency and availability, thus giving up the possibility of parallel-distributed implementations. BASE, instead, adopts tolerance to partition, and therefore either gives up immediate availability (CP) or immediate consistency (AP), depending on the kind of requirements in the application layer [40].

NoSQL databases can be coarsely grouped into documental, column-family, key-value, or graph databases. In document-oriented databases, each record is stored as a document [40] that is encapsulated and encoded in a semi-structured standard format such as XML, JSON, BSON [41]. The most popular document-oriented databases are MongoDB and CouchDB. The column-family oriented databases are characterized by the aggregation of data columns and family columns within data containers (keyspace) [40] [41]. Some implementations under this scheme are Google BigTable, Cassandra, Apache HBase, Hypertable and Cloudata. The Key-Value oriented databases are characterized for storing data as key-value pairs. This works efficiently in memory using map structures, associative arrays or hash tables, and it also manages persistent storage [40] [41]. Apache Accumulo, CouchDB, Amazon Dynamo, and Redis, among others, are the most used key-value databases. Finally, the graph oriented databases are characterized by a different storage organization, in which each node represents an object that may be related to other objects via one or more directed edges (which, also, may be objects). Within this category we can mention Neo4J, AllegroGraph and FlockDB. Graph oriented databases may sometimes be configured to comply with the ACID principle.

The first DW solutions were supported in relational databases, but as that the data volume and the need to change the ACID for BASE properties increased, NoSQL databases were integrated into the DW [49] proposals and in the BD. Some BD examples are Cassandra used by eBay, Github, Instagram, Netflix [42], Hbase used by Facebook [43] and Twitter [44], Dynamo Amazon [45], among others. For DW, some research results like [46] present a dimensional model under MongoDB, where query efficiency is analyzed against a SQL Server. In [47] an aggregation operator for OLAP cubes using HBase is proposed and the computational efficiency of the operator is shown. In [48] an API that allows to manage administrative tasks of the database oriented to Neo4j graphs through SQL commands is developed. In [49] an extension to star pattern considering the NoSQL databases HBase and MongoDB to exploit the horizontal scalability is proposed.

4.3.3. In-memory storage and processing

In-memory storage and processing improves the performance on data access and analysis. Several proposals are based on memory-resident tabular or columnar structures, among which we may count SAP HANA, which is a memory-resident database that achieves high analysis performance by means of a structured database oriented to columns and without requiring independent indices [50]. In [51] it is proposed to store large amounts of data in memory by the partition of an extensible multidimensional array without partitions or arrangement to exceed the amount of available memory. In [52], the authors present an optimization query model on a columnar memory database, built by analyzing the computational cost of dependent factors of the data (size of data, different values, data distribution and ordering) and of independent data factors (aggregation functions, number of aggregations, hash implementations, etc.).

For memory-resident performance analysis, in [15] a tool to organize, analyze and store data in tabular form as if it were a spreadsheet is described. Data are represented as cells of an organized arrangement of rows and columns in memory. In [53] and [54], a tool for data analysis that allows to upload and store data in memory with the Associative Query Logic model (AQL), to do joins and real-time calculations. In [22] a layer managed architecture in main memory (data repository, exploration laboratories and pattern discovery) for ETL processes and analysis is proposed.

4.3.4. Technologies complementary to Hadoop and MapReduce

Along with the Hadoop file system and the MapReduce programming model, a set of complementary tools that allow distribution, analysis and search have been developed. Within this group we can mention Yarn (Yet Another Resource Negotiator) [55], which is a tool to plan and monitor tasks and infrastructure resources, allowing different types of MapReduce applications to be run in cluster. Pig [56] is a compiler that manages parallel MapReduce program sequences through PigLatin language. Drill [57] is a distributed system for interactive analysis of large datasets, distributing data (structured, semi-structured and nested) on multiple servers to respond to ad-hoc queries with low latency and high speed. Apache Storm [58] is a system that allows integration with databases and communication with processes through Remote Procedure Call (RPC) to perform for instance real-time analyses and automatic online learning. Apache Hive [59] manages data storage through HDFS files and bases as Hbase, and allows to do queries on Hadoop through HiveQL language. Apache Sqoop [60] is an application that habilitates data transfer through relational bases and Hadoop, supporting incremental loads either from a table or a SQL query. Cloudera Impala [61] is a query engine for the Massive Parallel Processing (MPP) with Hadoop. It executes queries with low latency and under the MapReduce framework, without the necessity of transformation or migration of data stored in HDFS or Hbase. Apache Thrift [62] is a framework for sharing services between languages, such as C ++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C #, Cacao, JavaScript, Node.js, Smalltalk, OCaml , Delphi, among others. Finally, ZooKeeper [63], it is a centralized system for distributed applications that maintains configuration information and service name.

Around Hadoop and MapReduce there have arisen several technological solutions and BD projects that integrate some of the tools listed above. In [64], Hadoop, Hbase and Hive are integrated to manage messages on Facebook. In [65] a processing distributed system that dynamically adjusts the cluster size is built and analyzed, combining MapReduce programming model with the query language Pig. In [66], a methodology of incremental integration of different sources is presented, based on MapReduce, Hadoop and Pig. In [67] Hadoop Yarn is used to evaluate the performance of the co-location (location of data as close as possible to the processing) under different configurations and workloads. For DW, however, fewer studies integrate some of these tools, specifically those that do it explicitly refer to BD.

5. DISCUSSION

Result of the review and analysis, the characterization of each of the different types of data is done from the point of view of the domain, processing, storage and the end user to which they are directed (see Table 1).

Table 1. Characterization of the different data types. (Source: Author)

Data type	Characteristics	Scope	Processing	Store	User
Structured data (SD)	They are capable of being represented by predefined structures (vectors, graphs, tables, among others). The structure can be generalized	This data belongs to the domain of traditional database systems and data warehouses.	They can be stored by data structures such as tables or arrays and managed through widely distributed languages such as Sql.	They are usually stored and managed through relational databases.	Business analysts
Unstructured repetitive data (US-RD)	They do not have predefined structure; Are recurrent in time; They are generally massive; Not all have a value for the analyses so you can use samples or portions of these.	They come from electronic sensors whose objective is the analog analysis of the signal as: vital signs, seismic movements, positioning, biological and chemical processes, among others.	Generally, there are defined algorithms for the treatment of this type of data, like Fourier analysis for the signals, among others; Are susceptible to repetition and reuse.	They are stored raw and free of context; This is done using NoSql databases (document-oriented, key-value, among others) and flat files.	Data mining experts applied to different domains
Unstructured unrepetitive data (US-URD)	They do not have a single structure	It includes textual information, image analysis, dialog analysis, video content analysis and	The algorithms for processing this type of data are not reusable and the mere fact of predicting its structure is already a	They are stored raw and context-free in NoSql databases and flat files.	Data Science Experts

		string analysis.	complex task. Different processing is required depending on the type of data, such as Natural Language Processing and Computational Linguistics techniques for text-type data.		

The Table 2 summarizes the characteristics of BD and DW, from the point of view of the purpose, data sources, data types, scope, architecture, technology, methodology, technology, actors, persistence and processing.

Table 2. Characteristics of DW and BD. (Source: Author)

Characteristics	**Data Warehouse**	**Big Data**
Purpose	Treatment of data collections oriented to a specific business area or context; Integrated, non-volatile and time-varying. Supports decision-making, formulation of strategies and public policies.	Data processing, structured, semi-structured, unstructured, repetitive and non-repetitive, from diverse sources and whose volume of data exceeds the ability of traditional tools to capture, store, manage and analyse data.
Data source	Usually transactional databases, in finance, marketing, health, communications, among others.	Various sources and data types (social networks, sensors, e-mail, control systems, video, audio) in a wide range of applications and business contexts.
Data type	Traditional Data Warehouse considers DS, but these have evolved to include US-RD.	Big Data focuses particularly on US-URD, but does not exclude US-RD and DS
Scope	Structured data integration to support Business Intelligence (BI) and OLAP (Online Analytical Processing).	Capture, analyse and discover knowledge from large volumes of data characterized by the 7 v's, 3c's and M&G. The integration of the data is not relevant and it is privileged the analysis of the raw data through the science of the data.

Architecture	Oriented to processes of extraction, storage, processing and analysis of data. It is based on rigid multidimensional models (star, snowflake). Even though the literature presents proposals oriented to the storage in memory and under structured oriented to columns; They do not yet solve the problems of change management and horizontal scaling.	Despite the large amount of research in the area, there is still no reference architecture or standardized terminology that considers the functional viewpoint and the coexistence of data types different. They are proprietary and product-oriented architectures, proposals reduced to the solution of a company and focused to the technology. Even though ETL processes are related in the literature, the complexity that is generated around each type of data is not explicitly or systematically addressed.
Methodology	Constructed under rigid structures with difficulty to handle the change in the requirements and data sources. Well-known authors: Kimball, Todman, Inmon, with top-down, bottom-up, or hybrid proposals.	It is yet to be defined, a methodology that allows managing the project life cycle of Big Data, flexible to changes and consider the 7 v's, 3 c's and the new M&G
Technology	Mature and tested tools in large amount applications, both free and licensed software: Pentaho Mondrian, Oracle BI, SAP, among others.	Ecosystem consisting of a set of tools that generally includes Hadoop.
Actors	Business analysts act as end users who do not require specific knowledge of technologies or data exploration.	Data scientists or data analysts with knowledge in technologies, algorithms, mathematics and statistics
Persistency	They generally use relational databases of client-server type and with disk storage. Although there are proposals that use databases in main memory, it is not a generalized tendency	Depending on the needs, NoSql databases, database in memory or distributed file systems are used.

Processing	They generally under a Client-Server architecture	Generally, under a distributed cluster architecture

6. Multilayer Staggered Architecture Model For Big Data (MSAM-BD)

The proposed model (see Figure 2) consists of three layers: a) Data upload; b) Data processing and storage; and c) Data analysis. It explicitly considers the characteristics of the different types of data in the proposed layers and a staggered process for the management of the data life cycle. In addition, some transverse components are required.

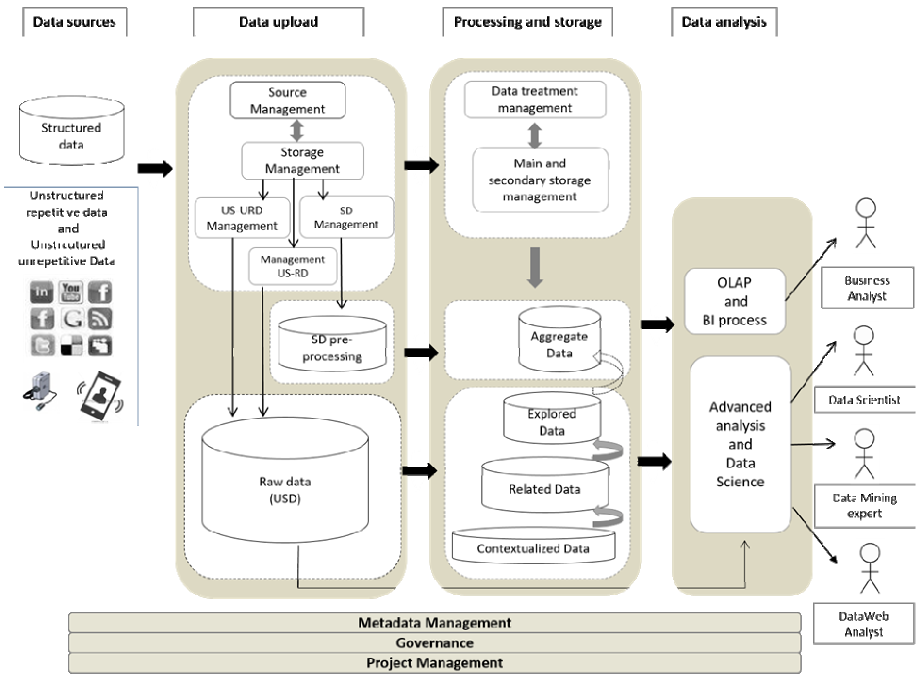

Figure 2. Multilayer Staggered Architecture Model for Big Data (MSAM-BD). (Source: Author)

6.1 Data upload

This layer specializes in storage according to the data type, whether it is structured (SD), unstructured repetitive data (US-RD) and unstructured unrepetitive data (US-URD). The SD are susceptible to pre-processing and storage under structures and standard algorithms, for example, on relational databases. US-RD and US-URD should be stored raw and free of context; for this purpose, distributed NoSql databases can be used.

6.2. Processing and storage

Structured data are aggregated through a predefined model. While the US-RD and the US-URD require a categorization and filtering process to store in the "Contextualized Data", the others will remain in the "Raw Data" area. Through a process of searching for relationships or patterns, the data in the "Contextualized Data" area will scale to the "Related Data" area without involving its deletion from the "Contextualized Data" area, as it may be used in the future with another vision. The "Related Data" that are already processed and capable to be adapted to predefined structures or semi-structures, and using some type of indexation that allows to systematize the queries, will be able to scale to "Explored Data" area. Finally, it is possible that some of the “Explored Data” can be integrated into the "Aggregate Data" area to be analyzed using OLAP techniques and business intelligence.

6.3. Data analysis

The data staggered process through different storage areas will not only manage the data life cycle, but also the access opportunity for decision making: in the "Contextualized Data" area, the most recent data that support immediate analyzes are located; in the "Related Data" area are located data of average longevity that have already been processed; in the "Data Exploited" area are present historical data that support statistical analysis, and in the "Data Aggregates" area the historical data are located to support OLAP analysis and Business Intelligence.

The architecture proposed specializes in several types of users: Business Analysts, who are end users of OLAP solutions and Business Intelligence, will work in the "Aggregate Data" area. Data and Semantic Web Analysts, Data Mining experts and Data Scientists will work with data from another area.

6.4. Transverse Components

The proposed architecture model – MSAM-BD- will require the existence of a transverse component that management the metadata in all processes, and that provides cohesion and semantics to the data using specialized ontologies or databases.

A project of this nature is complex because of the number of considerations that must be considered, and therefore, it will be necessary to adopt Management strategies for its development and maintenance, as well as Governance policies to define agreements and mechanisms of communication between the different actors. Such Governance should consider the change management, the standards management, the control of constraints, and the adoption of best practices for development. In consequence, besides the 7'Vs and 3'Cs, in this paper we propose to include the new features: M&G (Management and Governance).

7. CONCLUSION

With the raise of Big Data, it could be speculated that Data Warehouse has already met the apex of its life cycle. Nevertheless, as it was exposed in this work, DW and BD are complementary and could be integrated to share not only data, but also storage and processing computational resources. Any integration proposal should consider the characteristics described in Table 1. About the review made in this paper it can be concluded that during the last years the Data Warehouse developments are evolving to support some of the characteristics associated with Big Data, incorporating technologies like Hadoop, MapReduce and NoSql databases in the core of their design. Nevertheless, it is clear that the lack of standards has generated proprietary solutions that prevent a seamless integration of other DW and BD associated tools. Unfortunately, the demand of fast solutions and the versatility of some tools have led the DW projects to develop without the required conceptual, methodological and architectural framework. After costly investments in time and resources, businesses are trapped in "solutions" that do not meet the initial expectations, are not flexible to change, are difficult to maintain and to scale [2]. Particularly for Big Data, there is a whole ecosystem of technological proposals, not necessarily integrated into a single platform, which can increase the complexity of developing a project.

As for the purpose, the goal of both Data Warehouse and Big Data is the same, i.e., data exploration to identify patterns, support decision making, generate statistics and performance indicators. What is different are the limits, the nature of the data, the user to whom it is addressed and the procedures and tools for acquisition, storage and analysis. Big Data aims at the exploration of raw data such as unstructured unrepetitive data (US-URD), which are not susceptible to aggregation or systematic treatments. That is why BD users are more sophisticated (technology and data science experts), who with the use of novel tools, techniques and special algorithms could identify patterns and arrive into valuable conclusions about the data. Data Warehouse is focused on structured data (SD) and unstructured repetitive data (US-RD), which require pre-processing before information can be given to the final users (who may lack data mining or other kind of specialized knowledge), so they can make their own analysis, independently of the data sources, storage type, architecture, tools and algorithms used to reach the result. Information should be presented with an adequate aggregation and format, which means that the exploration performed by the final user is not on the raw data, but on previously processed data.

Because of what was exposed above, it is necessary to assume the integration of the two technologies in a framework of architecture as we have proposed in this work (MSAM-BD), considering the particularities widely discussed. For the implementation of such model, the integration of technologies that support the MapReduce, Relational Data Bases and NoSql programming model is required.

We hope that this work will be useful, especially for software architects, who must formulate and manage BD or DW projects at the enterprise level, since it is clear that the proposed solutions must be supported in a scalable architecture model that allows to handle the new sources and data types in a systematic way.

ACKNOWLEDGEMENTS

This project was funded by the Universidad Distrital F.J.C, Bogotá, Colombia

REFERENCES

[1] P. Bedi, V. Jindal, and A. Gautam, “Beginning with Big Data Simplified,” 2014.
[2] R. Kimball, M. Ross, W. Thorthwaite, B. Becker, and M. J, The Data Warehouse Lifecycle Toolkit, 2nd Edition. 2008.
[3] C. Todman, Designing A Data Warehouse: Supporting Customer Relationship Management. 2001.
[4] W. H. Inmon, Building the Data Warehouse, 4th Edition. 2005.
[5] “Oracle Database 12c for Data Warehousing and Big Data .” [Online]. Available: http://www.oracle.com/technetwork/database/bi-datawarehousing/data-warehousing-wp-12c-1896097.pdf. [Accessed: 09-Sep-2015].
[6] M. Cox and D. Ellsworth, “Application-Controlled Demand Paging for Out-of-Core Visualization,” 1997. [Online]. Available: http://www.nas.nasa.gov/assets/pdf/techreports/1997/nas-97-010.pdf. [Accessed: 09-Apr-2015].
[7] S. Chaudhuri and U. Dayal, “An overview of data warehousing and OLAP technology,” ACM SIGMOD Rec., vol. 26, no. 1, pp. 65–74, 1997.
[8] T. Maiorescu, “General Information on Business Intelligence,” pp. 294–297, 2010.
[9] “Data Warehouses and OLAP: Concepts, Architectures and Solutions: 9781599043647: Library and Information Science Books | IGI Global.” .
[10] Y. Demchenko, C. De Laat, and P. Membrey, “Defining Architecture Components of the Big Data Ecosystem,” Collab. Technol. Syst. (CTS), 2014 Int. Conf., pp. 104–112, 2014.
[11] G. NBD-PWG, “ISO/IEC JTC 1 Study Group on Big Data,” 2013. [Online]. Available: http://bigdatawg.nist.gov/cochairs.php. [Accessed: 24-Oct-2015].

[12] D. L. W.H. Inmon, Data Architecture: A Primer for the Data Scientist: Big Data, Data Warehouse and Data Vault. Amsterdam,Boston: Elsevier, 2014.

[13] G. N. W.H. Inmon, Derek Strauss, DW 2.0: The Architecture for the Next Generation of Data Warehousing (Morgan Kaufman Series in Data Management Systems) (): : Books. Burlington, USA: Morgan Kaufmann Publishers Inc., 2008.

[14] R. Kimball, "The Evolving Role of the Enterprise Data Warehouse in the Era of Big Data Analytics," Kimball Gr., 2011.

[15] M. Muntean and T. Surcel, "Agile BI - The Future of BI," Inform. Econ., vol. 17, no. 3, pp. 114–124, 2013.

[16] D. Agrawal, "The Reality of Real-Time Business Intelligence," in Business Intelligence for the Real-Time Enterprise, vol. 27 , M. Castellanos, U. Dayal, and T. Sellis, Eds. Springer Berlin Heidelberg , 2009, pp. 75–88.

[17] R. Castillo, J. Morata, and L. del Arbol, "Operational Data Store (ODS) - 933.pdf," Actas del III taller nacional de minería de datos y aprendizaje, pp. 359–365, 2005.

[18] S. YiChuan and X. Yao, "Research of Real-time Data Warehouse Storage Strategy Based on Multi-level Caches," Phys. Procedia, vol. 25, no. 0, pp. 2315–2321, 2012.

[19] A. Ma. P. Díaz-zorita, "Evaluación de la herramienta de código libre Apache Hadoop," Universidad Carlos III de Madrid Escuela Politécnica Superior, 2011.

[20] R. Kimball, "Newly Emerging Best Practices for Big Data," Kimball Group, p. 14, 2012.

[21] M. Maier, "Towards a Big Data Reference Architecture," no. October, pp. 1–144, 2013.

[22] O. Corporation, "ORACLE ENTERPRISE ARCHITECTURE WHITE PAPER. An Enterprise Architect ' s Guide to Big Data," no. February, 2015.

[23] F. Kramer, H. Muller, and K. Turowski, "Acceleration of Single Inserts for Columnar Databases -- An Experiment on Data Import Performance Using SAP HANA," in Signal-Image Technology and Internet-Based Systems (SITIS), 2014 Tenth International Conference on, 2014, pp. 672–676.

[24] M. R. Patil and F. Thia, Pentaho for Big Data Analytics, vol. 2013. PACKT PUBLISHING, 2013.

[25] S. G. Manikandan and S. Ravi, "Big Data Analysis Using Apache Hadoop," in IT Convergence and Security (ICITCS), 2014 International Conference on , 2014, pp. 1–4.

[26] J. Nandimath, E. Banerjee, A. Patil, P. Kakade, and S. Vaidya, "Big data analysis using Apache Hadoop," 2013 IEEE 14th Int. Conf. Inf. Reuse Integr., pp. 700–703, 2013.

[27] A. Katal, M. Wazid, and R. H. Goudar, "Big data: Issues, challenges, tools and Good practices," in Contemporary Computing (IC3), 2013 Sixth International Conference on , 2013, pp. 404–409.

[28] A. Pal and S. Agrawal, "An Experimental Approach Towards Big Data for Analyzing Memory Utilization on a Hadoop cluster using HDFS and MapReduce .," pp. 442–447, 2014.

[29] R. Zhang, D. Hildebrand, and R. Tewari, "In unity there is strength: Showcasing a unified big data platform with MapReduce Over both object and file storage," in Big Data (Big Data), 2014 IEEE International Conference on , 2014, pp. 960–966.

[30] "Welcome to ApacheTM Hadoop®!" [Online]. Available: https://hadoop.apache.org/. [Accessed: 26-Mar-2015].

[31] "HDFS Architecture Guide." [Online]. Available: http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html. [Accessed: 26-Mar-2015].

[32] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," in Computer Networks and ISDN Systems, 1998, pp. 107–117.

[33] D. Garlasu, V. Sandulescu, I. Halcu, G. Neculoiu, O. Grigoriu, M. Marinescu, and V. Marinescu, "A big data implementation based on Grid computing," in Roedunet International Conference (RoEduNet), 2013 11th, 2013, pp. 1–4.

[34] A. Jorgensen, C. Price, B. Mitchell, and J. Rowlan, Microsoft Big Data Solutions. John Wiley & Sons, Inc., 2014.

[35] R. T. Kaushik, M. Bhandarkar, and K. Nahrstedt, "Evaluation and Analysis of GreenHDFS: A Self-Adaptive, Energy-Conserving Variant of the Hadoop Distributed File System," in Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, 2010, pp. 274–287.

[36] J. G. Shanahan and L. Dai, "Large Scale Distributed Data Science Using Apache Spark," in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 2323–2324.

[37] R. S. Xin, J. Rosen, M. Zaharia, M. J. Franklin, S. Shenker, and I. Stoica, "Shark: SQL and Rich Analytics at Scale," in Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, 2013, pp. 13–24.

[38] J. Li, J. Wu, X. Yang, and S. Zhong, "Optimizing MapReduce Based on Locality of K-V Pairs and Overlap between Shuffle and Local Reduce," in Parallel Processing (ICPP), 2015 44th International Conference on, 2015, pp. 939–948.
[39] E. Brewer, "CAP Twelve Years Later: How the 'Rules' Have Changed," InfoQ, 2012. [Online]. Available: http://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed. [Accessed: 26-Mar-2015].
[40] G. Vaish, Getting started with NoSQL. 2013.
[41] V. N. Gudivada, D. Rao, and V. V. Raghavan, "NoSQL Systems for Big Data Management," 2014 IEEE World Congr. Serv., pp. 190–197, 2014.
[42] Cassandra, "The Apache Cassandra Project," httpcassandraapacheorg, 2010. [Online]. Available: http://cassandra.apache.org/.
[43] D. Borthakur, "Petabyte Scale Databases and Storage Systems at Facebook," in Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, 2013, pp. 1267–1268.
[44] J. Huang, X. Ouyang, J. Jose, M. Wasi-ur-Rahman, H. Wang, M. Luo, H. Subramoni, C. Murthy, and D. K. Panda, "High-Performance Design of HBase with RDMA over InfiniBand," in Parallel Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International, 2012, pp. 774–785.
[45] G. Weintraub, "Dynamo and BigTable - Review and comparison," Electr. Electron. Eng. Isr. (IEEEI), 2014 IEEE 28th Conv., pp. 1–5, 2014.
[46] D. Pereira, P. Oliveira, and F. Rodrigues, "Data warehouses in MongoDB vs SQL Server: A comparative analysis of the querie performance," in Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on, 2015, pp. 1–7.
[47] K. Dehdouh, F. Bentayeb, O. Boussaid, and N. Kabachi, "Columnar NoSQL CUBE: Agregation operator for columnar NoSQL data warehouse," in Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on, 2014, pp. 3828–3833.
[48] Y. Liu and T. M. Vitolo, "Graph Data Warehouse: Steps to Integrating Graph Databases Into the Traditional Conceptual Structure of a Data Warehouse," in Big Data (BigData Congress), 2013 IEEE International Congress on, 2013, pp. 433–434.
[49] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, "Benchmark for OLAP on NoSQL technologies comparing NoSQL multidimensional data warehousing solutions," in Research Challenges in Information Science (RCIS), 2015 IEEE 9th International Conference on, 2015, pp. 480–485.
[50] F. Färber, S. K. Cha, J. Primsch, C. Bornhövd, S. Sigg, and W. Lehner, "SAP HANA Database: Data Management for Modern Business Applications," SIGMOD Rec., vol. 40, no. 4, pp. 45–51, 2012.
[51] K. M. A. Hasan, M. T. Omar, S. M. M. Ahsan, and N. Nahar, "Chunking implementation of extendible array to handle address space overflow for large multidimensional data sets," in Electrical Information and Communication Technology (EICT), 2013 International Conference on, 2014, pp. 1–6.
[52] S. Müller and H. Plattner, "An In-depth Analysis of Data Aggregation Cost Factors in a Columnar In-memory Database," in Proceedings of the Fifteenth International Workshop on Data Warehousing and OLAP, 2012, pp. 65–72.
[53] H. Plattner, "A Common Database Approach for OLTP and OLAP Using an In-memory Column Database," in Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, 2009, pp. 1–2.
[54] J. Schaffner, A. Bog, J. Krüger, and A. Zeier, "A Hybrid Row-Column OLTP Database Architecture for Operational Reporting," in Business Intelligence for the Real-Time Enterprise SE - 5, vol. 27, M. Castellanos, U. Dayal, and T. Sellis, Eds. Springer Berlin Heidelberg, 2009, pp. 61–74.
[55] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, and E. Baldeschwieler, "Apache Hadoop YARN: Yet Another Resource Negotiator," in Proceedings of the 4th Annual Symposium on Cloud Computing, 2013, pp. 5:1–5:16.
[56] "Apache Pig Philosophy." [Online]. Available: http://pig.apache.org/philosophy.html. [Accessed: 26-Mar-2015].
[57] "Architecture - Apache Drill." [Online]. Available: http://drill.apache.org/architecture/. [Accessed: 26-Mar-2015].
[58] "Storm, distributed and fault-tolerant realtime computation." [Online]. Available: https://storm.apache.org/. [Accessed: 26-Mar-2015].
[59] "Apache Hive TM." [Online]. Available: https://hive.apache.org/. [Accessed: 26-Mar-2015].
[60] "Sqoop -." [Online]. Available: http://sqoop.apache.org/. [Accessed: 26-Mar-2015].

[61] “Impala.” [Online]. Available: http://www.cloudera.com/content/cloudera/en/products-and-services/cdh/impala.html. [Accessed: 26-Mar-2015].
[62] “Apache Thrift - Home.” [Online]. Available: https://thrift.apache.org/. [Accessed: 26-Mar-2015].
[63] “Apache ZooKeeper - Home.” [Online]. Available: https://zookeeper.apache.org/. [Accessed: 26-Mar-2015].
[64] D. Borthakur, J. Gray, J. Sen Sarma, K. Muthukkaruppan, N. Spiegelberg, H. Kuang, K. Ranganathan, D. Molkov, A. Menon, S. Rash, R. Schmidt, and A. Aiyer, “Apache Hadoop Goes Realtime at Facebook,” in Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, 2011, pp. 1071–1080.
[65] B. Ghit, A. Iosup, and D. Epema, “Towards an Optimized Big Data Processing System,” in Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on, 2013, pp. 83–86.
[66] P. Agarwal, G. Shroff, and P. Malhotra, “Approximate Incremental Big-Data Harmonization,” in Big Data (BigData Congress), 2013 IEEE International Congress on, 2013, pp. 118–125.
[67] Y. Elshater, P. Martin, D. Rope, M. McRoberts, and C. Statchuk, “A Study of Data Locality in YARN,” 2015 IEEE Int. Congr. Big Data, pp. 174–181, 2015.
[68] A. H. B. James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, “Big data: The next frontier for innovation, competition, and productivity,” McKinsey Glob. Inst., no. June, p. 156, 2011.
[69] J. S. Marron, “Big Data in context and robustness against heterogeneity,” Econom. Stat., vol. 2, pp. 73–80, 2017.
[70] L. Kugler, “What Happens When Big Data Blunders?,” Commun. ACM, vol. 59, no. 6, pp. 15–16, 2016.
[71] S. Sagiroglu, R. Terzi, Y. Canbay, and I. Colak, “Big data issues in smart grid systems,” in 2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA), 2016, pp. 1007–1012.
[72] A. Gandomi and M. Haider, “Beyond the hype: Big data concepts, methods, and analytics,” Int. J. Inf. Manage., vol. 35, no. 2, pp. 137–144, 2015.
[73] Jameela Al-Jaroodi, Brandon Hollein, Nader Mohamed, "Applying software engineering processes for big data analytics applications development", *Computing and Communication Workshop and Conference (CCWC) 2017 IEEE 7th Annual*, pp. 1-7, 2017