Gartner

Big Data Means Big Changes for Business Intelligence

Analyst(s): *Nancy McQuillen*

Summary

Big data analytics based on Hadoop-MapReduce technology has well-documented potential to reveal new insights and provide significant business value. However, the technologies, programming styles, and analytic methods of big data analytics depart drastically from traditional data warehouse methods and tools. Enterprise data warehouse (EDW) teams and business intelligence (BI) competency centers will need to evolve their team structure, skill set, and methods to successfully harness this relatively young programming framework.

Summary of Findings

Bottom Line: Business intelligence (BI) derived from big data is crucial to business competition and will depend on significant evolution of BI competencies and data warehouse (DW) architectures. Big data BI is not just traditional BI performed on bigger datasets. It also tackles data variety, velocity and complexity. The open source Hadoop-MapReduce projects serve as the technical basis for many big data projects. In addition to parallel computing clusters, they utilize different programming styles, consume diverse data types, provide new data transformation methods, and enable advanced analytic methods. These attributes add up to improved organizational capabilities to find patterns and derive insights from data.

Context: In recent years, the explosion in data volumes and data formats has caught the attention of CIOs as both a leading problem and a leading business opportunity. Big data technologies are maturing rapidly, fostered by a vibrant open source community around the Apache Hadoop and MapReduce projects. At the same time, parallel advances in text analytics and natural-language processing (NLP) are making it easier to mine Web data, documents, and human communications for enriched insight. [1] However, most enterprise data warehouse (EDW) and BI teams currently lack a clear understanding of big data technologies, potential application areas, and why "big data BI" contrasts with traditional BI tools. These teams are increasingly asking the question, "How can we use big data to deliver new insights?"

Take-Aways: Big data BI is not "BI business as usual." It differs dramatically from traditional BI in terms of both capabilities — the business questions and analytics that can be performed at reasonable cost — and in the technologies used to achieve those capability breakthroughs.

Big data BI can be used to answer a multitude of questions. The following seven questions reflect common patterns of analysis, where "they" and "them" often refer to customers, system users, employees or other human entities, but can refer to any entity of interest to the enterprise:

1. What are they (e.g., customers, users, or employees) saying?
2. What are they doing?
3. How can we characterize and group entities?
4. What are they likely to do?
5. What could we change to influence them?
6. What is changing in our environment?
7. What artifacts and events (e.g., prior cases or documents) are most relevant to our search requests?

The Hadoop-MapReduce technologies differ from the traditional BI paradigm in several important ways, including 1) data contrasts, 2) programming and processing contrasts, and 3) modeling and metadata contrasts.

- Data contrasts:
  - **Fundamental data structure:** Big data is expressed in "key-value pairs"[2] that can be arbitrarily structured and used to contain almost any type of data content, including text, emails, social media messages, documents, and images. Relational data and highly structured tables such as online transaction processing (OLTP) records can be included in big data, but represent only one type of data and are not accessed or processed using the traditional BI languages and methods (e.g., SQL, online analytical processing [OLAP], and pivot tables).
  - **Common data size and index scheme:** As its name implies, big data is indeed characteristically bigger (e.g., routinely petabytes vs. megabytes or terabytes). Many performance-driven practices of traditional DW preparation do not apply, including prebuilding indexes on the relational database, denormalization into star schema designs, and preloading of multidimensional online analytical processing (MOLAP) cubes. Big data technology depends instead on parallel processing plus optional dynamic indexing for its performance boost.
- Programming and processing contrasts:
  - **Fundamental programming paradigm:** Big data programming in MapReduce is Java-based but also incorporates functional programming techniques, custom processing pipelines (i.e., MapReduce sequences and data flows), user-defined functions (UDFs), and other mechanisms that are in stark contrast to SQL's declarative style and to relational database processing.
  - **Extract, transform, and load (ETL) development and processing:** Hadoop can be used as an ETL engine, and this is becoming increasingly common in big data applications.
  - **Program execution:** The functional programming foundation in MapReduce enables the automatic parallelism of programs. This enables processing of massive datasets with reasonable performance.
- Modeling and metadata contrasts:
  - **Fundamental data modeling:** The big data paradigm is agile and exploratory so there is little time for upfront data modeling. Schemas are optional and may be evolved because they are not tightly bound with the basic key-value data model, in contrast with the required schemas in relational database architectures.
  - **Modeling styles and usage:** Big data "modeling" primarily refers to *discovery* of models within the data, for example, discovering "prototypical" credit card customers via clustering techniques on huge transaction sets, or observing patterns of behavior per customer category. These models are then used to predict future behaviors and business outcomes.

    The following recommendations are offered to organizations that are interested in leveraging big data for BI and are currently in the starting phases of planning for the effort:

- Start planning for enterprise big data BI infrastructure as a long-term and equal analytic partner to traditional EDW and BI.
- Develop a big data BI "skills matrix" and staffing plan by gradual investment in BI staff training, by partnering with organizational experts, and by adding staff or consultants as needed. Include the following prerequisite skills:
  - Both Java and functional programming skills
  - Knowledge of data mining algorithms and statistical methods
  - Hadoop computing server configuration and cluster management
  - Custom ETL design exploiting MapReduce methods
  - Exploratory style of data analysis and business question sequencing
  - Open source code management
  - Commercial tools selection and application. An ecosystem of tools is emerging in the marketplace to simplify, package, or assist with Hadoop and MapReduce data management.
- Manage the risk of initial big data BI projects by selecting projects that present only one or two of the extreme data challenges — volume, variety, velocity, or complexity.

  Conclusion: The business requirement to conquer and mine big data is here to stay for most enterprises. The potential value is huge, but the technologies and skill sets to manage big data BI depart drastically from traditional data warehousing, dimensional modeling, BI, and OLAP. Despite extra challenges for BI teams embarking on Hadoop-MapReduce projects, the strengths of the technology framework are assessed to outweigh the weaknesses. EDW teams and BI competency centers should learn to identify big data opportunities and evolve their team structure, skill set and methods to successfully harness this new technology paradigm.

Analysis

Traditional EDWs are becoming massive, but their style, purpose and foundational technologies are drastically different than the emerging "big data" paradigm. To remain competitive, most organizations will soon need to exploit both "traditional BI" and "big data BI," using them synergistically within planned enterprise architectures.

In January 2011, Gartner published an in-depth document titled "Hadoop and MapReduce: Big Data Analytics" that reviews the technologies and the business potential of big data. This analysis does not repeat the information presented in that foundational report, but rather builds upon it to discuss big data more specifically in the context of EDW and BI planning. As big data continues to gain attention and momentum in industry, IT professionals who are responsible for BI are asking Gartner, "How can we leverage big data to improve business intelligence and deliver new insights?"

This analysis begins by defining big data and contrasting big data technology with traditional data warehousing and BI, focusing specifically on analytic data warehousing and dimensional modeling. The dimensional DW paradigm has matured and is a strong influence in the BI tools industry and in the skills requirements of EDW and BI teams. Given the characteristics of big data technology, what will these teams need to learn and do in order to take advantage of big data?

Scope of the Analysis

"Traditional BI" in this analysis refers to BI derived from traditional architected DWs (e.g., integrated online transactions processing [OLTP] data sources) that often include dimensional models. The data sources traditionally combine data from multiple enterprise applications or ERP systems (e.g., manufacturing or production, sales, marketing, human resources, and finance applications). The DWs and dimensional models are commonly implemented using some combination of relational databases, star schemas, and OLAP cubes.

This analysis intentionally excludes newer (i.e., nontraditional) BI paradigms suited for personal desktop or isolated departmental analyses, such as those that are commonly beginning to leverage in-memory datasets for rapid and ad hoc data discovery and visualization. [3]

The "big data" technologies in scope for this analysis are restricted to solutions based on the open source Apache Hadoop framework and projects, [4] and in particular this subset — selected for relevance to DW applications:

- Apache Hadoop projects:
  - **Hadoop Common :** The common utilities that support the other Hadoop subprojects. (Note: Hadoop "CORE" was renamed to "Common" in July 2009.)
  - **Hadoop Distributed File System (HDFS) :** A distributed file system that provides high-throughput access to application data
  - **Hadoop MapReduce :** A software framework for distributed processing of large datasets on compute clusters
- Other Hadoop-related projects at Apache include:
  - **HBase :** A scalable, distributed database that supports structured data storage for large tables
  - **Hive :** A DW infrastructure that provides data summarization and ad hoc querying
  - **Pig :** A high-level data flow language and execution framework for parallel computation

Commercial tools based upon the Apache Hadoop projects are available in the marketplace, and more are emerging due to the popularity of the projects. These commercial tools are also relevant to this analysis because they build on the Apache Hadoop technical foundation. Several of these tools are discussed in prior Gartner documents (see "Hadoop and MapReduce: Big Data Analytics"); however, to further identify and assess specific vendor tools is beyond the scope of this analysis.

Recognize that Apache Hadoop is not the only approach to big data technology and big data management. Other technology platforms, paradigms and vendors support big data, and some may define big data differently than the Gartner definition provided in this assessment. Gartner does not exclude these alternate technologies as being inferior or less important than the Apache Hadoop project, but Hadoop has a prevalent and growing open source community and is thus selected as a convenient starting point for the discussion of big data in BI. Examples of other big data technologies include the high-performance compute cluster (HPCC) from LexisNexis Risk Solutions [5] and the MarkLogic Server from MarkLogic. [6]

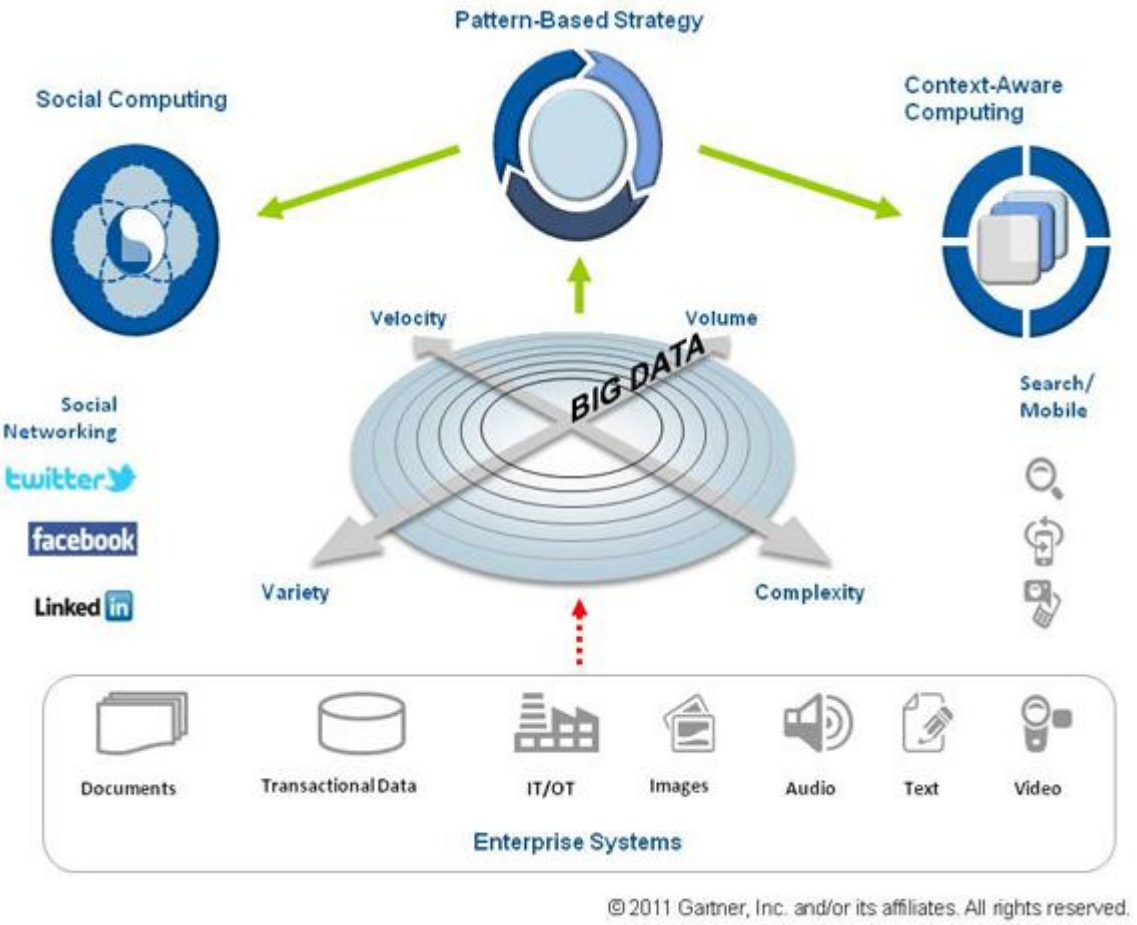
Big Data as Synonym for Extreme Data (More Than Just Big)

When business leaders or data management professionals talk about big data, they often emphasize volume, but recognize that big data is about much more than data volumes. The additional dimensions of velocity, variety, and complexity and more contribute to "extreme information," a broader term sometimes used by Gartner to encompass *all* of these aspects and to avoid overemphasis on "big." [7] Although big data does not have a single succinct definition, a description of these four areas provides an overview of many important aspects:

- **Volume** of big data normally refers to many terabytes or petabytes of information — in many cases being collected daily, weekly or monthly in the course of enterprise operations. Big data volume is not an absolute concept, because "big" depends on an organization's ability to handle and process the volume. Hundreds of terabytes might overwhelm the capacity of some organizations, whereas others routinely manage petabytes per day. In either case, these volumes exceed human capacity for reading and comprehension, creating demand for automated or highly assisted computer techniques for data exploration, navigation, and discovery.
- **Velocity** means how fast the data arrives, how fast it is available, and how fast it can be evaluated and processed to meet the intended purpose, such as decision support or business process support. These multiple stages of latency should be considered when planning the data flow. It may involve streams of text data, massive amounts of in-coming sensor data, or structured records created from high-volume transaction processing systems. Velocity adds to the challenges of meeting data access and delivery requirements.
- **Variety** refers to the myriad data types and formats that may be included in the big data sources. These include tabular data (e.g., spreadsheets or relational database tables), hierarchical data, documents, email, social data, free text, metering data, video, image, audio, stock ticker data, financial transactions and more. Big data *tends* to include a high percentage of text data in comparison to traditional databases, but many successful big data applications exist only to process OLTP or structured database data in new ways or at new volumes.
- **Complexity** is a multifaceted notion and can arise from any complicating factor beyond the sheer size, speed, and variety of the data. It refers to the different data structures, standards, domain rules and storage formats that can exist with each asset type. For example, an information management system for media may have multiple video formats, and a data mining suite may need to process data that initially arrives in various forms such as arrays, nested lists, or proprietary formats that can be parsed only with metadata from the supplier. Complexity may also refer to the need to link, preprocess or relate multiple data streams and data types in preparation for analysis.
Note that "extreme information" challenges can exist when just one or more of these dimensions are present. It does not require all four, and the challenges increase when more of the dimensions are involved.

As shown in Figure 1, Gartner research emphasizes all four of these aspects of big data quantification, which appear as four dimensions on the central "Big Data" platter.

Figure 1. Big Data for Information Initiatives



Source: Gartner (December 2011)

Note: OT = operational technology

As indicated in Figure 1, big data plays an important role in multiple data management initiatives by enabling a range of improved capabilities, including:

- **Social computing:** Ability to mine text conversations, infer soft concepts such as mood and sentiment, plus graph social networks and relationships to determine patterns of influence
- **Pattern-Based Strategy:** Ability to seek diverse patterns due to flexible computing framework that enables incorporation of advanced data mining algorithms, plus ability to analyze larger datasets, avoid sampling error, and improve statistical precision
- **Context-aware computing:** Ability to manage huge data volumes generated by "always-on" monitoring equipment and location-aware devices, and respond appropriately by recommending best next action in the context of the precise situation or location

See "Pattern-Based Strategy: Getting Value From Big Data" for additional information on these initiatives and their use of big data.

Extreme "MAD" Data

Gartner is not alone in using the word "extreme" to describe the new world of data and information management. In a 2009 journal report on very large database(s) (VLDB), the authors characterize and define the emerging practice of "Magnetic, Agile, Deep (MAD) data analysis" as a radical departure from traditional EDWs and BI. [8] They emphasize that "mad" has also become a pop cultural term meaning extreme. The report was published jointly by researchers at University of California (UC) Berkeley and three commercial companies (Greenplum, Fox Audience Network, and Evergreen Technologies). The following are their descriptions of the three aspects of MAD:

- **M agnetic:** Traditional EDW approaches "repel" new data sources, discouraging their incorporation until they are carefully cleansed and integrated. Given the ubiquity of data in modern organizations, a DW can keep pace today only by being "magnetic": attracting all the data sources that crop up within an organization regardless of data quality niceties.
- **A gile:** Data warehousing orthodoxy is based on long-range, careful design and planning. Given growing numbers of data sources and increasingly sophisticated and mission-critical data analyses, a modern warehouse must instead allow analysts to easily ingest, digest, produce and adapt data at a rapid pace. This requires a database whose physical and logical contents can be in continuous rapid evolution.
- **D eep:** Modern data analyses involve increasingly sophisticated statistical methods that go well beyond the rollups and drilldowns of traditional BI. Moreover, analysts often need to see both the forest and the trees in running these algorithms — they want to study enormous datasets without resorting to samples and extracts. The modern DW should serve both as a deep data repository and as a sophisticated algorithmic runtime engine.

These three aspects of extreme or "MAD" data provide evidence of upcoming changes and major implications for EDW and BI teams as they adapt their practices and skills in order to exploit big data.

What Is MapReduce?

MapReduce has little in common with SQL — the current predominant language for DWs and BI tool developers. In this section, Gartner provides a short historical review of MapReduce in order to emphasize its procedural, functional nature — an important difference for DW and BI developers that are practiced in the declarative style of SQL. MapReduce operations are driven and sequenced from a procedural program shell written in the Java language, but the essence and the important part of the analytic process occurs in the execution of the inner "map" and "reduce" steps. Therefore, it is important for big data developers to master both Java and MapReduce data flows and function planning. Many technical references on MapReduce mention that it is often difficult for programmers to adapt to the MapReduce style of functional programming and data flow design.

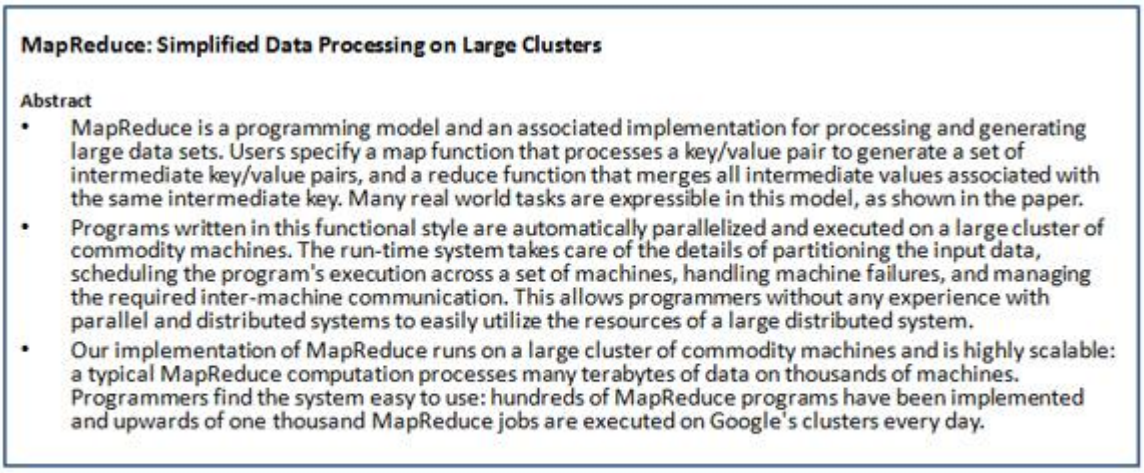
Roots in the Web Search Business

MapReduce is a programming model first developed by Google for the processing of large datasets. It was developed to meet challenges of performing Internet-scale searches on huge Web content datasets, using a large number of machines and parallel processing techniques. Yahoo also has been a major contributor and user of Hadoop and MapReduce open source projects. [9]

With roots in the Web search business, MapReduce is clearly talented at counting words. However, the general framework can be used for much more. The sidebar in Figure 2 is an abstract from a Google research paper describing MapReduce. [10] It emphasizes several important and fundamental programming model features, including these five characteristics:

- MapReduce is optimized to operate on *large datasets* .
- The technique utilizes a *key-value pair* data structure and enables transformation of the pairs into new key-value pairs.
- Analysis execution is *automatically parallelized* .
- Analysis can be executed on a *large cluster of commodity machines* .
- Programmers can use the tool *without any experience in parallel and distributed systems* .

Figure 2. Google Description of MapReduce



MapReduce: Simplified Data Processing on Large Clusters

Abstract

- MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown in the paper.
- Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system.
- Our implementation of MapReduce runs on a large cluster of commodity machines and is highly scalable: a typical MapReduce computation processes many terabytes of data on thousands of machines. Programmers find the system easy to use: hundreds of MapReduce programs have been implemented and upwards of one thousand MapReduce jobs are executed on Google's clusters every day.

Source: http://labs.google.com/papers/mapreduce.html
Authors: Google Research Scientists: Jeffrey Dean and Sanjay Ghemawat

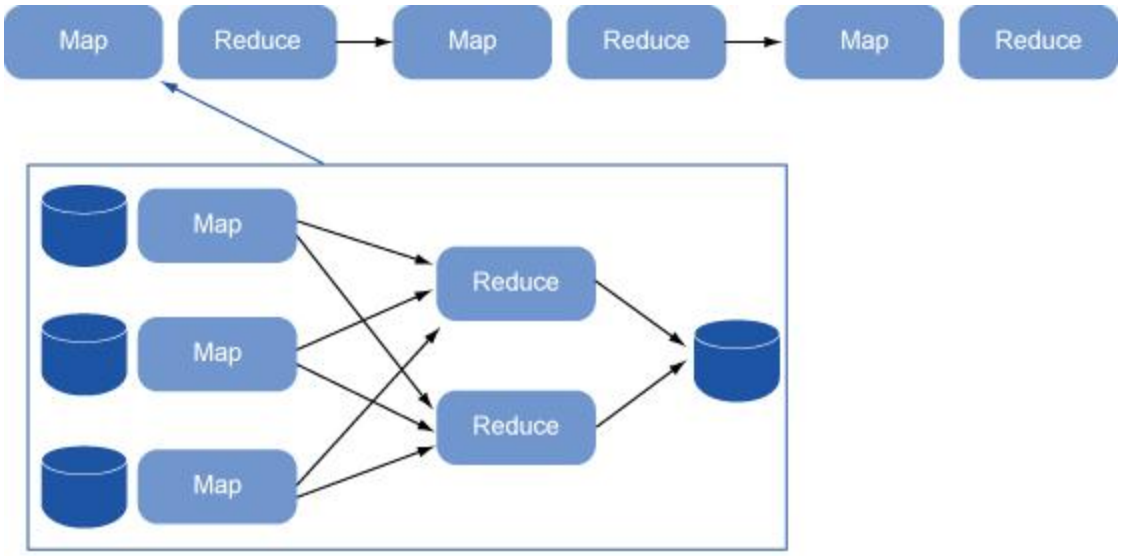
Source: Gartner (December 2011)

The name MapReduce was inspired by the "map" and "reduce" combinators from a functional language like LISP — the " **LIS** t **P** rocessing" language that has been heavily used in artificial intelligence applications for decades. Disciplined functional programming is the secret to simple parallelism. Because disciplined functions have no side effects — that is, each function has inputs and outputs but does not reach beyond its boundaries to affect other processing or create any mischief — data processing is entirely independent and therefore can be easily and safely dispatched to a separate parallel processing thread. Google's online MapReduce tutorials provide additional information on the MapReduce programming model, including this brief explanation of the workings of the map and reduce steps:

> In Lisp, a "map" takes as input a function and a sequence of values. It then applies the function to each value in the sequence. A "reduce" combines all the elements of a sequence using a binary operation. For example, it can use "+" to add up all the elements in the sequence. [11]

Most big data analytic data flows contain many successive MapReduce steps, with each automatically parallelized, as portrayed in Figure 3, which shows the main flow at the top and a parallel breakout of one map and reduce step in the box below. Several successive transformed versions of the dataset may be saved along that way — that is, one version per map-reduce step within the pipeline. This provides flexibility to stage data in arbitrary and incremental ways and to potentially reuse intermediate stages as input to follow-on analyses. For example, various levels of data aggregation may be retained from successive steps.

Figure 3. MapReduce Workflow



Source: Gartner (December 2011)

MapReduce Recognized as a Computing Pattern

As evidence of the recognized utility and probable ongoing importance of MapReduce, researchers at UC Berkeley have elected it as one of 13 essential "dwarf" computing patterns (named so because there were originally seven dwarfs identified by Phillip Colella [12]). The dwarfs present a method for capturing the common requirements of classes of applications while being reasonably divorced from individual implementations. A dwarf is an algorithmic method that captures a pattern of computation and communication. The Berkeley researchers were inspired by the work of Phil Colella, who identified seven numerical methods that he believed will be important for science and engineering for at least the next decade. They examined more applications in search of basic patterns and added extensions, resulting in a final list of 13 dwarfs as follows [13]:

- Dense Linear Algebra
- Sparse Linear Algebra
- Spectral Methods
- N-Body Methods
- Structured Grids
- Unstructured Grids
- MapReduce
- Combinational Logic
- Graph Traversal
- Dynamic Programming
- Backtrack and Branch-and-Bound
- Graphical Models
- Finite State Machines

The Berkeley researchers explain the generalization of the "Monte Carlo" technique to become the MapReduce dwarf (Number 7 in the list above). The MapReduce dwarf was originally called "Monte Carlo," after the technique of using statistical methods based on repeated random trials. The patterns defined by the programming model MapReduce are a more general version of the same idea: repeated independent execution of a function, with results aggregated at the end. Nearly no communication is required between processes.

In summary, the fundamentals and programming primitives of MapReduce offer powerful data processing capabilities, but MapReduce style is far removed from the style and skill set of SQL programming for relational databases.

Use Cases for Big Data BI

Big data will be big in terms of business value derived from ability to find patterns in huge fuzzy and dynamic *unstructured* data resources. Less sampling is required, meaning more accurate and subtle patterns can be detected. Big data also adds new capabilities for analyzing *structured* resources in new ways. This section reviews some common patterns of usage that are emerging, followed by examples of business questions and applications.

Common Patterns of Analysis

MapReduce can be used for advanced mathematics, statistics and data mining. However, it is frequently reported in applications of simple operations on huge datasets, including these generic patterns:

- Counting "things" (e.g., words, objects, or entities)
- Counting the references from one thing to another thing (explicit relationships)
- Graphing the references between things (i.e., unraveling the "network" of things)
- Indexing things
- Sorting things
- Discovering common relationships and correlations between things
- Applying a wide variety of machine learning algorithms for pattern discovery

Indexing and sorting in particular can be very slow, resource intensive and problematic in traditional data processing without the assistance of parallel techniques. MapReduce offers breakthroughs in the size and speed at which huge collections can be indexed and sorted.

Counting is a very general and widely applicable pattern of use. Due to the roots of Hadoop and MapReduce in the word-intensive Web search business, counting has often been applied to words. Most textbooks and tutorials on MapReduce seem to feature a word counting exercise as the first example of how the technology works — for example, determining the frequency of each unique word appearing in a document or collection of documents. However, counting can be applied to many kinds of "things," including objects or entities. Imagine the possibilities for counting and summarizing the prevalence of these other more complex data types, for example, document types, image types, instances of values (e.g., descriptors, categorizing variables, and symptoms), named entities (e.g., customers, products, and companies), and identical data structures such as common paired entities. These counts could be transferred to a traditional EDW, for example, to become the values of cells within pivot tables or OLAP cubes.

Business Questions and Applications

Big data clearly has business value (see use cases and case studies in "CEO Advisory: 'Big Data' Equals Big Opportunity," "Hadoop and MapReduce: Big Data Analytics," and "Pattern-Based Strategy: Getting Value From Big Data"). Many examples are also listed at the Apache website, [14] and the list is growing as the technology is adopted by more and more organizations. However, it is safe to assume that many organizations are *not* eager to publish their application stories or the technical details of successful projects due to a desire to maintain competitive advantage. Therefore, organizations need to understand the basic capabilities of the technology and the common patterns of application, and use creativity in applying these patterns to their own extreme data problems.

The following seven generic business questions provide a starting framework for identifying potential big data analytic applications in your enterprise. They are intentionally expressed simply and informally, using the terms "they" and "them" to refer to a broad range of possible important organizational entities (i.e., parties or "players") such as customers, system users, employees, and service clients. It will be necessary to replace those generic words with the entities or nouns that matter within your business context in order to brainstorm and explore for big data analytic applications. Each question is accompanied by a few examples synthesized from the prior Gartner research on big data:

1. What are they (e.g., customers, users, or employees) saying — and feeling?
    - Sentiment analysis
    - Product feedback; product rankings
    - Analysis of customer complaints
2. Who are they and what are they doing?
    - Analysis of individual behaviors; user tracking and log analysis
    - Fraud detection; anonymous behavior identification

- Cross-system identity resolution and tracking; behavior profile composition
- Web clickstream analysis; advertising response analysis
- Patient behaviors and healthcare treatment regimens in relation to outcomes
- Analysis of behavioral patterns; group behaviors correlated to common outcomes

3. How can we characterize and group entities?

- Market segmentation
- Financial risk categories, loan or instrument ratings, and customer ratings
- Customer category discovery; cluster analysis

4. What are they likely to do?

- Customer churn analysis
- Loan default prediction
- Security monitoring; inferred models of malware and malicious activity
- Prediction of actions based on group assignment

5. What could we change to influence them?

- Buyer behavior prediction
- Service utilization monitoring and correction
- Factor analysis
- Compliance influencers; medical compliance
- Group feature identification and weighting

6. What is changing in our environment?

- Image comparisons; geographical state monitoring
- Medical image comparisons
- Financial profile monitoring; organizational risk identification
- Business state monitoring and anomaly detection; exceeded threshold alerts

7. What artifacts and events (e.g., prior cases or documents) are most relevant to our search requests?

- Document retrieval; enterprise search across data types; relevance ranking
- Similar case retrieval; customer support applications
- Filtering OLTP transaction sets for reasonable matches to patterns
- Web content retrieval on basis of keywords or concept sets

Traditional BI has been used for many years to address several of the applications indicated in the examples shown with these seven questions. However, big data analytics has several unique capacities:

- The ability to process *massive volumes* of familiar data types (e.g., OLTP transaction records) *more quickly*
- Special talents for *mining text* due to the free-form key-value data structures and the freedom to incorporate a broad range of text analytic techniques, including natural language processing (NLP) and semantic search techniques
- The abilities to program very *specialized data parsing and preparation* (e.g., transformations performed within the MapReduce pipeline) and to incorporate *highly specialized data mining and statistical methods*

These contrasts with traditional BI are further described in the following section of this document.

In addition to these general contrasts, it is worth calling out Question 2 above — "Who are they and what are they doing?" — as an especially unique and important capability. Big data BI can assist in verifying or inferring user identity *across systems* for security purposes and in composing patterns of activity *across data types* , for example, by correlating explicit transaction behavior such as account activity, sales or returns with other forms of evidence such as document searches, Web searches, or other related activity gleaned from log files of related applications. Because Hadoop-MapReduce was developed in the Internet search arena, success stories often emphasize word search capabilities, including Web content and social media, overlooking structured data analytics. However, ability to track these types of broad behavior patterns in OLTP systems is an important pattern of application that extends EDW and BI capabilities in the areas of security, identity verification and threat detection. [15] For an in-depth look at behavioral analysis methods that may be applied to big transaction data, see "Behavioral Analytics: Detect and Assess Activity, Good and Bad, in Transaction Data."

Contrasts Between Traditional BI and Big Data BI

Big data BI fills a much-needed and very different use case than traditional BI and OLAP by supporting interactive data discovery and pattern discovery based on very little upfront understanding or modeling of data. Experimentation is enabled by the low-cost parallel processing achieved with clusters of commodity servers. Analysts can "afford" to do explorations on huge datasets that previously would have been impossible or prohibitively expensive — possibly requiring high-cost database servers or supercomputers.

These are different but complementary worlds. Big data BI is not a replacement for traditional BI. Slicing and dicing of targeted metrics will continue to be needed. Traditional BI is governed, suitable for compliance and for producing predefined metrics, precise values and answers to known questions based on carefully structured data. In contrast, big data BI is more open to unknown data types and data sources, ungoverned, exploratory and suitable for interactive mining of data, with the goal of discovering patterns and deriving models that are stable enough to be used for predictive analysis.

The following sections elaborate further on the differences between traditional and big data processing, separating the contrasts into three areas: 1) data contrasts, 2) programming and processing contrasts, and 3) modeling and metadata contrasts.

Data Contrasts

Big data can be bigger and more diverse than traditional EDW data, but it also differs in more subtle ways, including the nature of the data structures and the type of indexes that are commonly applied. Table 1 presents some of the primary contrasts in these two areas.

Table 1. Data Contrasts

Traditional DW processing	Big data processing	Comparison, impact
Fundamental data structure		
Relational tables: Predefined structured rows and columns (i.e., relational tuples, with column order imposed by the physical relational schema) Schema is formalized, and can be reverse engineered or published (i.e., as relational Data Definition Language [DDL]).	Key-value pairs, lists of pairs: Arbitrary data structures may be represented in the values, including complex and diverse data types (e.g., text, images, XML documents, arrays, graphical objects, maps, and so on). Schemas are optional and do not follow one formal style. Explicit schema may not exist, except in program code or programmer's mind.	Fundamental data structures of big data allow more freedom and diversity, but may be more difficult to manage due to factors such as complex parsing schemes and lack of formalized metadata standards to enable automated discovery and management.
Common data size and index scheme		
Gigabytes or terabytes: Pre-indexed, often multi-indexed Access selectively using indexes to select a small set of relevant records. May be modeled (e.g., star schemas) or preloaded (e.g., multidimensional cubes)	Terabytes or petabytes: Non-indexed, or dynamically indexed with possibility for an "index everything" approach rather than selected columns Access an entire dataset, using automatic parallel partitions for performance (i.e., the goal is not normally quick access to a small	Big data technologies enable high-performance processing of massive datasets without pre-indexing. Because indexes often require extensive disk space in traditional databases, the size difference between big data and traditional data may be moderated or less than expected.

Table 1. Data Contrasts

Traditional DW processing	Big data processing	Comparison, impact
for performance	set of records)	

Source: Gartner (December 2011)

As summarized in Table 1, traditional data to support BI (i.e., relational data) contrasts starkly with big data in terms of:

- **Fundamental data structure:** The relational table structure of traditional databases is more regular and formal in comparison with the more free-form "key-value pairs" of big data. The value portion of the key-value pair commonly carries the "content" of the pair and can be used for the many diverse data formats and date types that are becoming prevalent and important in big data analytics. Note that the "key" portion of the pair also provides a great deal of flexibility and could be considered a type of arbitrary and dynamic metadata tag applied to the value portion of the pair. Both the key portion and the value portion can be arbitrarily structured, and both can be transformed and restructured through successive MapReduce operations, but only the key portion is used to collect the values back together. The "key" in the pair is thus different than the notion of a unique key or a surrogate key in relational databases. The big data key is an important aid to aggregating values during the reduce step, so the key might be manipulated (e.g., successively structured or transformed) to aid in a particular data reduction step.
- **Common data size and index scheme:** Traditional methods are more dependent on some combination of pre-indexing, modeling, and physically reorganizing data in order to achieve high performance or reasonable performance for information retrieval in BI applications. In relational DWs, these techniques may include dimensional models that use a combination of extensive indexing (i.e., foreign keys from fact tables to numerous dimension tables) plus de-normalization of the dimension tables to prepare the data for extensive querying of pre-aggregated cells and flexible drilldowns along the numerous dimensions of analysis. This extensive preplanning, modeling, and preloading of star schemas and cubes is not relevant in the big data paradigm due to the parallel processing capability boost.

In comparison with the relational data that has traditionally been used in data warehousing as the foundation for BI, big data also frequently exhibits these characteristics:

- The data contains a high percentage of **unstructured** data objects (e.g., text documents, paragraphs, emails, images, or video). The values of the key-value pairs can contain these objects. This offers great analytic challenges but also important new opportunities to derive richer context, deeper understanding and enhanced insight into behavior of people and processes. Deriving this insight often depends on the application of text mining and natural language processing (NLP) techniques.
- The data may include a **vast number of attributes** , for example, thousands of columns, parsed or collected from multiple sources, in an effort to collect all forms of evidence without the discrimination, quality control and screening that is more common to relational database and DW planning.
- The relevant data may be spread among **many data sources, and new datasets are added frequently and dynamically** . This dynamic data may not be formally modeled, cleansed, or integrated as is customary for traditional DW environments. Datasets may be analyzed individually in specialized and possibly one-time analysis, particularly because they may be of very different formats and without formal models, making it more difficult to integrate them in meaningful ways. However, in the liberal spirit of big data acceptance and exploration, they are accepted into the big data "ocean," where they may be used as novel forms of supporting or confirming evidence. This is in contrast to the EDW tradition of striving for one consistent, governed "version of the truth." The big data paradigm is more exploratory and inclined to search for a multitude of weak signals, which taken together reveal theories and patterns to be considered and further evaluated.
- The data may be of **unknown quality** due to the lack of control of data sources and the inclination to accept new datasets liberally.

Taken together, these unique characteristics of big data mean big changes in terms of data management. New methods, tools, and attitudes are required to leverage big data for BI.

Programming and Processing Contrasts

Big data provides a high degree of computational freedom, freedom for arbitrary and custom data structures, flexibility, and programmer control. Table 2 presents some of the primary contrasts with traditional DW programming.

Table 2. Programming and Processing Contrasts

Traditional DW processing	Big data processing	Comparison, impact
Fundamental programming paradigm		
Declarative, set-oriented, using SQL Moderate capability to include functions and statistical algorithms Mature commercial tools market exists for formal models (e.g., relational and dimensional) accompanied by standardized access methods and BI tools (e.g., Open Database Connectivity [ODBC], Multidimensional Expressions [MDX], XML for Analysis [XMLA], and SQL).	Procedural, functional, using Java (or other Java Virtual Machine [JVM]-based language) to express MapReduce processing pipelines. However, higher-level languages and tools that extend MapReduce (e.g., Pig and Hive) can assist in simplified scripting or SQL-like development. Advanced capability to include predefined functions, statistical algorithms, and UDFs Open source software tools	Big data BI is a more open programming paradigm, supporting the ability to include diverse types of simple or advanced analytics, performed on a diverse array of data types. The general nature of the programming paradigm, plus the dependence on open source software, means that advanced development skills may be required.
ETL development and processing		
ETL is developed and run as a separate preprocess to prepare the DW, often using ETL tools separate from the BI reporting and delivery tools (e.g., pivot tables, cubes, and other standard BI delivery and presentation tools). ETL developers tend to be separate from BI developers, with distinct tool specializations. ETL tools tend to include connectors to many disparate databases and data formats, for general use as translators between multiple systems.	ETL-like transformations can be developed as an integrated subset of the overall pipeline processing of MapReduce steps for big data analytics. Transformations are developed directly in MapReduce as an integral part of the data flow and analytic process. MapReduce reads and writes data to HDFS, but is not intended as an ETL tool in the sense of providing extensive connectors to multiple other databases and data formats.	Big data "ETL" can potentially be run on massive datasets, facilitated by the automatic distribution of processing across the computer cluster. This is in contrast to the more limited parallel ETL that can be achieved with manual parallel ETL data flows built into traditional ETL mappings, but the traditional tools may be easier to use due to extensive transformation function libraries and visual interfaces for designing these ETL mappings. Some ETL vendors are adding big data capabilities to their general framework by adding Hadoop connectors plus enhanced UDF capabilities.
Program execution		
Program processing is not automatically distributed	Program processing is automatically distributed across a	It is easier to develop programs that can process huge datasets without

Table 2. Programming and Processing Contrasts

Traditional DW processing	Big data processing	Comparison, impact
across a cluster of computers.	cluster of computers.	subdivision or sampling, due to the automated assistance in distributing the workload across the computing cluster.

Source: Gartner (December 2011)

As summarized in Table 2, the programming and processing of traditional EDW data preparation and BI data delivery contrast starkly with big data in three areas:

- **Fundamental programming paradigm:** Traditional BI tools tend to incorporate only basic descriptive statistical methods, so it is often necessary to export data to separate tools and experts for advanced inferential statistics and predictive modeling. In comparison, big data programming offers better support for directly incorporating advanced statistical algorithms without exporting the data to powerful external statistical packages such as Statistical Product and Service Solutions (SPSS), Statistical Analysis System (SAS) and R. In addition, the statistical algorithms may be able to run on entire massive datasets, without the requirement for sampling. This provides exciting new possibilities for accuracy. However, it should be noted that big data statistics may require more technical prowess and programming skill than use of statistics within either traditional BI tools or specialized statistical packages. This is because MapReduce is a generic framework and statistics is only one of the data processing operations that can be performed.
- **ETL development and processing:** Hadoop can be used as an "ETL-like" engine to read data from a Hadoop Distributed File System (HDFS) source, apply arbitrary transformations, and write it back to an HDFS target. The data transformation, data reduction and analysis capabilities of the MapReduce framework can also be exploited to prepare data for loading and further analysis in a relational EDW or analysis in a Hadoop Hive DW (see description of HDFS and Hive in "Hadoop and MapReduce: Big Data Analytics").
- **Program execution:** The functional programming foundation in MapReduce enables the automatic parallelism of programs. This enables processing of massive datasets with reasonable performance — a primary reason for the success of the Hadoop projects. Programmers are isolated from having to worry about details of parallel processing, and the processing environment adapts to the number of available processors. The programs are largely isolated from changes in the computing cluster configuration.

Modeling and Metadata Contrasts

The big data paradigm is intended to be agile and exploratory so there is little time for upfront data modeling and strict data governance. New data sources are accepted more freely, without the goal of formal integration into planned EDW data models, so data modeling is optional and may be applied within the processing pipeline only as needed. Table 3 presents some of the primary contrasts with traditional EDW modeling and metadata:

Table 3. Modeling and Metadata Contrasts

Traditional DW modeling	Big data modeling	Comparison, impact
Fundamental data modeling		
Schemas required and designed in advance. Physical schemas are tightly coupled to implementations such as relational tables and	Schema-less, or schemas added as needed during programming, or after data exploration; schemas may evolve or change.	It may be possible to impose multiple different models on the data, for example, multiple hierarchies or category schemes, or possibly different modeling styles to support different target application needs. This is due

Table 3. Modeling and Metadata Contrasts

Traditional DW modeling	Big data modeling	Comparison, impact
dimensional cubes. The target application is presumed to use relational data structures and SQL, hence modeling style matches (e.g., entity-relationship diagrams).	Schemas are not tightly bound to the key-value pair data model. The target application style is not presumed, and therefore the modeling approach is not dictated or singular style.	to the loose notion of schemas, but may bring the danger of misinterpretation and lack of reuse — due to lack of explicit formal, standardized metadata.
Modeling styles and usage		
Top-down descriptive modeling by humans of fundamental data shape (e.g., expressed as entity-relationship models, dimensional models, and other types of explicit schemas and metadata) Predictive modeling is not normally emphasized.	Bottom-up model discovery through computerized data exploration, mining, and relationship analysis, plus optional top-down modeling Derived models of both*shape* (e.g., schema discovery) and *data content* (i.e., prototypical pattern discovery and dataset characterization, for use in follow-on prediction of behaviors or outcomes)	Big data modeling involves computerized support for finding prevalent entities and relationships in the data (e.g., particularly in unstructured or semistructured datasets). This discovery approach and data mining orientation provides the ability to find surprises and unexpected patterns, clusters, or data characterizations.

Source: Gartner (December 2011)

As summarized in Table 3, traditional *data* modeling contrasts starkly with big data *modeling* in two areas:

- **Fundamental data modeling:** The relatively "schema-less" style of big data processing includes pros and cons. Traditional data modeling takes time and requires knowledge of the data, so it is not necessarily applicable, possible, or desirable in a programming environment that is designed to be agile, exploratory, and adaptive. In the big data paradigm, schemas are optional and may be evolved or replaced because they are not tightly bound with the basic key-value data model, in contrast with the Data Definition Language (DDL) that is formally bound with tables in relational database architectures. However, the lack of explicit metadata in standardized form may create problems. How can the meaning of the data be reliably expressed to others if schemas are only implicit, and "buried in the code" — or in the mind of the programmer that designed the MapReduce data structures and processing pipeline? Because no clear separation of the application and data layer exists in this paradigm, the metadata may become intermingled with the code. Fortunately, some of the programming tools frameworks that extend MapReduce (e.g., Pig) enable the addition of explicit schemas where needed.
- **Modeling styles and usage:** The whole notion of "modeling" is quite different in the big data paradigm compared with traditional data modeling. An expert in data mining or statistics may automatically think of modeling as being *discovery* of models within the data, for example, discovering "prototypical" credit card customers via clustering techniques on huge transaction sets, or observing patterns of behavior per customer category. These models are then used to predict future behaviors and business outcomes. This difference is emphasized here partly because most college curriculums in computer science and information systems do *not* train IT professionals sufficiently in quantitative modeling skills and data discovery skills. Hence, BI teams may lack these skills. Recently, a great deal of discussion in the industry and academic press has emphasized the need for improved curriculums and training programs for data analysts and "data explorers." In the meantime, building cross-disciplinary business teams of both

statisticians and IT professionals may be necessary in order to handle the complete task of big data "modeling" for successful projects.

Strengths

In comparison to traditional EDW and BI, using a Hadoop-MapReduce open software foundation offers the following BI advantages:

- Ability to handle **extreme data size** , with reasonable speed. Auto-parallelism exploits the computer clusters, enabling processing of huge datasets in reasonable time frames.
- Scalability at **reasonable cost** , due to low-cost commodity servers used in the clusters, in comparison to the high-cost specialized database machines and licenses that may be required in traditional EDW infrastructure.
- Ability to process **many data types** , including unstructured text, images, audio, XML, structured tables, and so on. This is a critical advantage because relational databases contain only a small percentage of organizational data assets in most enterprises.
- Ability to detect **behavioral analysis** patterns, such as specific user actions and outcomes over time; for example, in clickstream analytics for Web shopping behavior, and in security and fraud detection applications that depend on cross-system verification and tracking of individual user behavior.
- Ability to perform free-form and **complex computing** , in controlled procedural style, including use of UDFs, statistical analyses, and data mining algorithms as needed in successive stages of data transformation, aggregation and pattern detection. This processing diversity offers types of analytics that are not practical or feasible in BI based on SQL and relational databases alone.
- **Reliability of operational environment** , due to automatic redundancy and failover features that are built into the cluster management scheme of Hadoop.

Weaknesses

In comparison to traditional EDW and BI, the use of Hadoop-MapReduce as BI infrastructure introduces extra challenges for technical teams. The contrasts in skills requirements and implications for the team profile are very significant and should be a major consideration in deciding how and when to tap into big data with Hadoop and MapReduce.

The analysis identified the following two interrelated areas of challenge, including the prerequisite skill set for big data BI application development, plus technical sophistication to manage the relatively young Hadoop-MapReduce open source software tool environment:

- Skills requirements for BI application and report designers and developers:
  - **Both Java and functional programming skills:** MapReduce depends on Java and related Java-oriented tools and APIs from the Apache Hadoop project, plus ability to design MapReduce steps and data flows — a unique functional programming skill beyond more traditional object-oriented and procedural styles. Therefore, required programming skills are both object-oriented and procedural (i.e., using Java) and functional (i.e., designing the "map" and "reduce" operations which have their roots in LISP and functional programming). Is your BI team sufficiently skilled in Java coding? Does the team have experience with functional programming languages, including development of specialized UDFs? Several higher level language tools are available to hide the complexity of MapReduce and Java programming. These include Apache projects (e.g., Pig and Hive) plus commercial tools such as graphical development environments and user-oriented desktop clients for data integration and analytics. Some of these tools add a SQL-like "veneer" to MapReduce. However, the SQL programming skills of traditional DW and BI professional teams will likely not be completely adequate for managing significant big data BI projects.
  - **Knowledge of data mining algorithms and advanced statistical methods:** In addition to the prerequisite Java and functional programming foundations, BI developers will often need to incorporate specialized algorithms and analytic methods within the big data processing pipeline. This implies the need for BI team members to understand, compare and contrast these methods, or to work closely with data mining and statistical experts who can provide this expertise to the BI team.
  - **Computer server and cluster management:** Although the parallelism of MapReduce programs is primarily automated by the Hadoop framework, new data center skills are required to install the servers and manage the computing clusters optimally. Thus, hardware experts are required and will need to communicate with software experts to determine requirements for the size and configuration of clusters to support the BI portfolio.
  - **Custom ETL-like design exploiting MapReduce:** Big data BI processing can optionally *include*transformation steps, rather than having ETL performed as a separate preprocess as is common in traditional EDW preparation.

MapReduce for ETL is particularly useful when data volumes demand parallel processing, or when extreme challenges are present due to data diversity or data complexity. The art of developing ETL jobs in MapReduce is much different than developing traditional ETL mapping using mature commercial ETL tools with their established transformation libraries and GUI tools to support data flow designs. Although traditional ETL tools skills help, it will take time and practice to adapt them to ETL development in MapReduce steps due to the unique nature of MapReduce programming. Ability to craft UDFs is helpful in MapReduce programming, including within data transformation steps of the pipeline.

- **Exploratory style of data analysis:** The art of big data mining contrasts with the art of governed DW and data mart design, where BI metrics and queries are more often predicted in advance, and the data is more carefully modeled and prepared in advance according to best practices. Big data exploration has no established "cookbook." Big data mining tends to require a multifaceted "tolerance for the unknown," including unknown data quality, unfamiliar and ever-changing data sources, diverse algorithms and methods to grapple with the ever-changing data sources, and evolving business questions. The big data BI analyst is ideally an interactive data explorer, adding value by applying background knowledge and instincts in the business topic to pose business questions in a probing and exploratory sequence. Business analysts with programming and quantitative skills may be well suited and adaptable to the needs of big data BI development. Some traditional BI team developers who have been pure deep technical specialists (e.g., in selected ETL, OLAP, BI, or relational database management system [RDBMS] reporting tools) may have trouble adapting to the fluid role of the big data explorer. Others may be highly capable of making this transition but simply prefer to continue to apply their important skills within the traditional BI realm.

- Technical sophistication required to manage Hadoop-MapReduce open source software and hardware infrastructure, plus related commercial tools:
  - **Open source code management:** Access to source code comes with the added burden and responsibility to understand and manage code, local customizations, versions, and configurations. The Hadoop and MapReduce projects are also relatively immature in comparison to other open source technology areas.
  - **Commercial tools selection and application:** An ecosystem of tools is emerging in the marketplace to simplify, package, or assist with Hadoop and MapReduce data management. In addition, several Apache projects related to Hadoop and MapReduce are still evolving. [16] The tools market is relatively young and changing rapidly, so product selection alone can be complex, and investments should be made with care and foresight. Some of the tools introduce new languages and procedures layered on top of Hadoop-MapReduce, possibly to great advantage but also accompanied by learning new syntax or style that can result in dependence or lock-in to a tool vendor or Apache project.

Despite these extra challenges in the area of big data BI skills and competencies, the business requirement to conquer and mine big data is here to stay — or coming soon — for most enterprises. At the same time, the Hadoop-MapReduce open source framework is available as an early entrant and solution to the big data problem. It is maturing rapidly, is successfully deployed in a significant number of production environments, and is also spawning an array of supporting tools and experts in the marketplace. Thus, the strengths of this solution paradigm are assessed to outweigh the challenges. Consistent with this assessment, the following section offers recommendations for embarking on early applications of big data BI.

Recommendations

Big data BI is a different and necessary beast, and traditional BI is not going to magically morph into big data BI anytime soon. Big data BI fills a much-needed niche in the parsing and analysis of unstructured data (e.g., text and documents) as well as enabling complex and creative analytics on larger collections of structured data (e.g., massive transaction sets from OLTP and ERP systems). The traditional BI and ETL tools markets are evolving rapidly in many important ways, including becoming more virtualized, higher capacity, more self-service, and more real time. However, none of those changes will make it capable and suitable for doing this completely new Hadoop-MapReduce type of exploratory, massive-scale data culling, pattern finding, and predictive modeling.

The affordable cluster computing foundation of Hadoop-MapReduce is proving to be a disruptive technology, enabling performance breakthroughs for many organizations. This performance advantage is coupled with the drastically different open computing style that enables creative application and mixing of data mining methods, statistics, and complex data structure parsing. These factors taken together indicate that big data will be an important resource and source of competitive advantage for most organizations. The following recommendations are offered to

organizations that are interested in leveraging big data for BI, and are currently in the starting phases of planning for the effort:

Start planning for enterprise big data BI infrastructure as a long-term and equal analytic partner to traditional EDW and BI.

For most enterprises, both traditional BI and big data BI will be required in the future to maintain competitive advantage. Big data technology has already proven to be valuable to many organizations, and additional case studies are emerging rapidly. Organizations should begin planning for big data analytic capabilities as a permanent enterprise service, taking small steps if necessary according to BI budget limitations and capacity for new technology experimentation. Isolated prototypes may provide a logical starting point, but approach the overall planning task with an enterprise service center mentality. Look ahead, plan for growth, and consider ways to manage the trajectory of the adoption. Spread the effort between central IT and departments or business units, sharing both expertise and technical infrastructure whenever possible. Duplicating the effort to set up infrastructure in each department would be inefficient and costly.

Big data BI is not a temporary sideshow or departmental experiment, but rather an equal partner and capability for the enterprise. Begin the evaluation and selection process to find the big data BI tools and architecture most suited to enterprise needs. The Hadoop-MapReduce open source project is well-established and likely to persist for some time due to the proven value of the computing pattern (see theMapReduce Recognized as a Computing Pattern section) plus the growing open source community of developers, users, and related Apache projects and commercial tools. Organizations that currently have the technical sophistication and capacity to manage open source should consider Hadoop-MapReduce as a viable candidate for big data infrastructure in the short term. Other organizations with less sophistication for open source management or less bandwidth for new technology introduction will need to wait, watch and proceed more carefully as commercial tools mature — both around Hadoop-MapReduce and around competing paradigms. Consider BI team skill sets and preferences as important criteria in tool selections, but also look to expand those skill sets (see the following recommendation regarding skills development).

Big data BI is not a replacement for traditional BI; for the foreseeable future, they should be considered complementary technologies. Use Hadoop-MapReduce (or other cluster computing technology alternatives) along with traditional BI and DW architecture and tools to meet specific challenges of extreme data. Build new big data BI competencies gradually but deliberately by tackling application areas that would be impossible without the new technology support. Do not abandon current methods in the short term. Do not plan to migrate existing dimensional models, OLAP, and DW reporting onto the big data technology, even if it is feasible to replicate the functionality of those applications. Big data analytics is not intended or optimized for dimensional modeling and OLAP, and it lacks the accompanying standards (e.g., ODBC and MDX) and presentation tools such as pivot tables, drill down reports, scorecards, and dashboards.

Develop a big data BI "skills matrix" and staffing plan by gradual investment in BI staff training, by partnering with organizational experts, and by adding staff or consultants as needed.

Big data BI differs from traditional BI in many ways, including fundamental data structures and formats, programming languages and styles, processing environment, and data modeling approach (see theContrasts Between Traditional BI and Big Data BI section for further discussion of these areas of difference). The total skill set to support Hadoop-MapReduce big data BI projects is multifaceted and specialized, with demands for expertise in several areas. It is unlikely that one individual designated staff member will bring all of these forms of expertise, so an isolated small-scale prototype may not work as effectively as a team approach. To achieve success even in initial and exploratory projects, it may be necessary to "cover all the bases" of Hadoop-MapReduce skills, including the following:

- Java programming (or a JVM-based programming language)
- Functional programming and UDF design
- ETL data flows and data reduction design
- Business domain knowledge
- Statistics and data mining algorithms
- Hadoop computing cluster management

For further discussion of each of these skills in the context of big data BI, see the Weaknesses section, which asserts that the demand for new skills over and above traditional BI team skills presents a significant major challenge for many BI teams embarking on Hadoop-MapReduce projects.

The optimal big data BI team may be best formed as a cooperative of multiple departments, with a matrixed project reporting style and with participants contributing just a portion of their time and attention to big data BI. Partnering with departmental business analysts and experts in statistics and data mining may be the best strategy in order to find the right mix of prerequisite business and data knowledge, along with the proper mind-set for business-oriented exploratory data mining and analysis. Leadership for the matrixed team could be placed either within a business department (e.g., finance or marketing) or within enterprise IT as an extension to an EDW or BI team, according to organizational needs and interests in big data analytics.

Manage the risk of initial big data BI projects by selecting projects that present only one or two of the extreme data challenges — volume, variety, velocity or complexity.

Although it would be ill-advised to ignore the increasing importance and future competitive advantage of big data, it is still a relatively new technology area, so start gradually and carefully. There is likely to be considerable strategic advantage by starting soon and building skills and infrastructure in planned phases. Mitigate risk in the early stages by tackling only one or possibly two of the extreme data challenges at a time (e.g., only high-volume data or high-complexity data but not both; see the Big Data as Synonym for Extreme Data (More Than Just Big) section for discussion of the four challenges). When selecting applications, consider that some analytic advantages of the Hadoop-MapReduce programming style can apply even on datasets that are not very "big." Consider leading-edge versus higher risk "bleeding edge" applications and use cases for the new technology, perhaps by following others who have achieved success with similar types of analytics or with similar data formats or data volumes. Avoid tackling problems so unique and ambitious that they might increase the chances of failure. Instead, reserve extra energy to invest in foundational activities, including basic architecture and infrastructure setup, team building, and skills development.

Develop the team's big data BI design and programming skills using a modest-sized data sample and working on a practical, promising business application. Hadoop-MapReduce applications can often be accomplished at relatively low cost due to the commodity hardware. For a very slow and controlled investment and careful ramp-up, consider a three-step approach: 1) start with a small-scale initial setup such as a small Hadoop cluster hosted externally by a cloud provider, or a single server cluster and a local installation, using data sampling if necessary to develop and test the program logic; 2) deploy the tested proof-of-concept application logic to a larger externally hosted Hadoop cluster at a service provider site to test scale-out to a larger cluster, which may be fairly easy and automatic due to the Hadoop auto-parallelism of the programs and data, and 3) bring the cluster technology in-house to host the production application. Step 3 applies only if IT resources exist to support the computing environment and the intent is to self-manage the open source technology internally within the enterprise.