To Hadoop or not to Hadoop

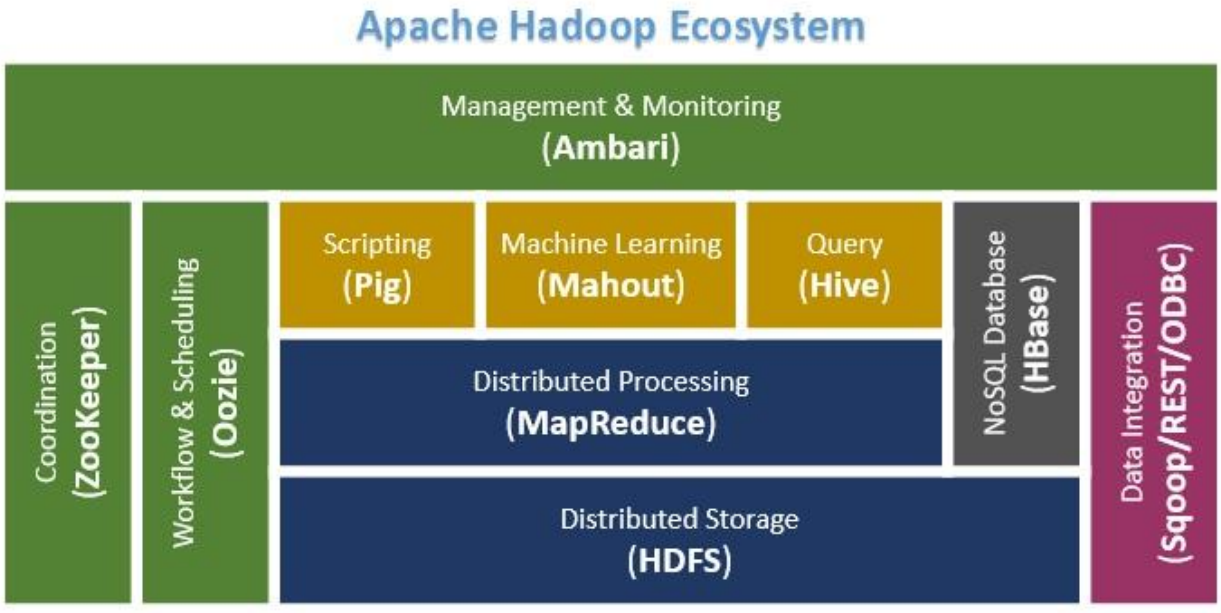
Tom Breur
19 December 2016

Introduction
In some circles, Big Data gets equated with Hadoop. This, I would argue, is utter nonsense. But an interesting question remains: how did Hadoop capture so much mind space? Big Data and data strategy have become topics in corporate boardrooms, nowadays. Gartner predicts that by 2020, 30% of companies will have a Chief Data Officer on their board. Hadoop surely *did* play a pivotal role through all of this, and is featured prominently in hype cycles that many thought leaders are talking about. Let's see if we can shed some light on the question what role Hadoop ***can*** play in your information ecosystem.

Personally, I think Big Data is a big deal. I may not think it is as radically *new* as some vendors would like us to believe, but I welcome the fact that senior management awareness is at an unprecedented level. Hype cycles have inherent risks, and of course those need to be managed. When leveraged intelligently, Big Data can be a game changer. Hadoop has been highly disruptive for the business intelligence (BI) space. However, due to the relative immaturity of some of this technology, and ongoing shortage of skilled talent, there are substantial risks involved. In particular when it comes to managing security, and ensuring adherence to privacy policies and compliance, you face considerable challenges that need to be addressed.

The rise of Hadoop
Hadoop broke the frontier of non-relational querying, and because it was Open Source, this enabled the growth of an elaborate eco-system. Hadoop has been around for more than 10 years, although from 2003-2008 it was only used by Google and Yahoo. Only for the past 7-8 years has it received wide attention, especially so in the last five years. Cloudera played an important role in raising awareness for the possibilities Hadoop has to offer. Its success triggered development of an array of solutions that got built on top of it, like Hive, Pig, Zookeeper, Impala, Mahout, and many, many others.

Summary overview of Hadoop ecosystem:



credit: Dattatrey Sindol

Hadoop provides linear scalability that the traditional (relational) vendors could not offer. The flexibility of the Hadoop Distributed File System (HDFS) allows you to store all kinds of data types and formats, either structured or unstructured. This dramatically lowers the threshold for capturing and storing of data, the fundamental premise of so-called data lakes. Now companies can afford to store data that previously may have been too expensive to keep. This has the potential of taking a holistic (360°) view of the customer to a whole new level. Connecting data from disparate sources, often with a wide variety of data formats and types, provides the most promising avenue for transformational insights and business innovation.

HDFS and MapReduce leverage MPP architecture, and will run on clusters of inexpensive commodity hardware. MapReduce in conjunction with Yarn enables parallelization of data processing tasks across an arbitrary number of nodes. It scales linearly, some large instances run thousands of clusters in parallel. Resilience to failing hardware is "built into the system" which makes these operations reliable and fault tolerant. The libraries are designed to detect and handle failures (like a disk crash) gracefully, in stark contrast with traditional BI platforms. With HDFS you can inexpensively load large Volumes of data, it elegantly handles a large Variety of data, and due to the linear scalability, you can quickly and gracefully deal with growing Velocity of data, or alternatively with more stringent requirements for latency.

Big Data and the three V's

At the beginning of the Big Data hype, there was frequent mention of these three V's. Then some people, like thought leader Mark Madsen, started pointing out that those same three V's

didn't explain much about Big Data. Moreover, they are not always present in Big Data applications, so they aren't really defining characteristics for it. Big Data applications don't always rely on Terabytes of data (may even have fairly low Volume). Velocity doesn't mean much when you still rely mostly on batch processing, as many organizations still do. And what if the Variety of data types is low? Would any of these disqualify an application as "Big Data"?!? Hardly. The three V's have certainly helped to give laypeople a notion of Big Data, but have done little or nothing to define it.

Traditional BI systems showed two major constraints that have made business leaders cringe when managing such projects. Poor scalability and prohibitive cost of relational systems meant that –in particular– semi-structured data like log files, XML, or JSON were cumbersome candidates to "load" into a traditional data warehouse. Defining ETL processes is a prerequisite for loading in traditional architectures. Determining 'correct' transformations for such sources is even harder when they are subject to frequent change, and their record layout is imperfectly specified. It will be no surprise that those two happen to go together quite often…

NoSQL solutions (of which Hadoop is but one special case) have circumvented this painful bottleneck that always existed to get the data "in" by allowing schema on read (defining 'the' correct data model), rather than requiring schema on write. The HDFS system has no trouble ingesting semi-structured files, deals elegantly with changes to their format (in stark contrast to traditional ETL), and source format variety across source systems is also largely a non-issue. All of this means that capturing data has become much easier and cheaper.

Even though storage has become so much less expensive, there is still no free lunch. Extracting information from Hadoop still (and will always) requires deep insight in the embedded structure of the data, and familiarity with the logic behind the underlying business processes. But at least uncovering and testing those structures –which likely requires access to scarce human expertise– can be *deferred* until priorities for information *extraction* have been determined. This aligns nicely with principles from Lean and Agile development methods, like delaying decisions until the last responsible moment.

The second part of this equation is that in traditional BI solutions, data were moved from a place that was optimized for transaction processing (like a normalized, relational database system) to a platform and database specifically designed for analyzing and retrieving data, commonly a dimensional data warehouse. As the volumes of data become excessive, this moving of data creates costly redundancy and eats up tremendous network bandwidth and I/O. Hadoop solutions, for the most part, enable both storage (HDFS) and analysis (MapReduce) in the same environment. As the volumes of data have been growing dramatically, no longer having to move data around has proven an extremely valuable feature that can help dramatically drive down cost as well as latency in data access.

Some tools like Apache Spark simulate the existence of temporary data structures in memory without actually landing the data (it stays in HDFS). That way you can build a very complex data pipeline without moving any data at all, at least not until you write it to a data mart specifically

designed for reporting purposes and ad hoc analytical queries (like Impala, for instance). Also, you can compute transformations once, cache the results in memory (without any intermediate write operations), and query these data sets at will – obviously with significant performance gains.

Traditionally, in BI we have been trying to use "all encompassing" RDBMS systems that handle collection, storage, computation, as well as presentation of data. Some argue that in an attempt to be a jack of all trades, you risk becoming a master of none. An RDBMS that is good at everything cannot be particularly good at anything. The new wave of data pipeline solutions *decouple* collection, storage, computation and presentation. That way you can use the most appropriate technology for each task like HDFS for storage and HBase for fast access. This approach also enables you to evolve some of the components, without disturbing any other parts of the pipeline. The more your data volumes and need for low latency push the envelope, the better these contemporary data architectures will enable you to keep pace with changing business requirement demands.

Hadoop and hype cycles

Sentiments around Hadoop have waxed and waned. Cloudera, and to a lesser extent MapR and Hortonworks, have been a driving force to promote Hadoop as an enterprise ready solution in support of BI, especially the last five years. Awareness at all levels of organizations has grown to the point where Big Data often gets equated with Hadoop. Clearly that demonstrates how effective the marketing has been. Along with awareness comes expectation, and as Hadoop and Big Data evolve, Gartner's familiar hype cycle has been evolving right in front of our eyes. As may be expected, in many cases Hadoop has not lived up to expectations.

The concept of a data lake proved an enticing marketing concept, despite the fact that it has no architectural founding. But since data lakes were so strongly associated with Hadoop, adoption of data lakes went hand in hand with adoption of Hadoop. The dramatic price drop for storage has fueled exponential growth in data storage. As data lakes get filled, or nearly flooded, a potential bottleneck accrues when teams make inroads to leverage all those data for competitive advantage. They may quickly have more data at their hands than they have time to process and explore.

As a symptom of hype cycle dynamics, it is illustrative to note that after Intel invested $760 million in Cloudera, their valuation was determined at $4.1 billion. Since then, due to arguably typical disillusionment at the tail end of Gartner's hype cycle, their market valuation has been faltering. This year, (WSJ, 30 March 2016) Fidelity marked down their holdings in technology startups like Cloudera by up to 38%. Startups in this space have come and gone, underscoring volatility and dramatically lowered expectations. Gartner survey results in the recent past have also pointed to a slowing down or holding off on Hadoop investments. Even "… the early adopters don't appear to be championing for substantial Hadoop adoption over the next 24 months" (Gartner, May 2015).

The most commonly voiced cause for "failed" Hadoop projects is lack of sound business application, and failure to demonstrate tangible commercial benefits. The second most common inhibitor for adoption continues to be the skills gap. Although the tool vendors claim their solutions will help close this gap, their solutions are –still– mostly targeted at highly skilled users. Rather than enabling a substantial broadening of the resource pool in support of enterprise wide applications, they seem to be mostly running and supporting niche Hadoop instances.

Of all the Hadoop implementations that were listed in the Gartner survey, 70% of them reported 20 users or less. These numbers suggest they can hardly be considered enterprise wide adoptions, but more likely high powered data scientists. As valuable as their work may be, it is of a different nature from the "bread and butter" analytics that most BI users perform. As Gartner notes: "… the low numbers of users relative to the cost of cluster hardware, as well as any software support costs, may mean Hadoop is failing to live up to this promise."

Hadoop implementation pains

Since Hadoop originated in the Open Source world, and was initially only implemented in few and very advanced organizations with high maturity of data management (like Google and Yahoo), the need to make launch and implementation easy and simple was not very high at the outset. Achieving extremely high performance and scalability, and being able to tweak and tune were much more important features. This history resulted in technology with notably more manual effort than most "mainstream" installations would experience for the rest of their enterprise IT stack.

Hadoop offers cost effective storage for a very wide variety of data types. This makes it such a compelling choice as the medium for comprehensive persistent storage, often labeled as a "data lake" by clever marketers. Bear in mind, though, that there exists *no* architectural concept like a "data lake", and data does not flow into it as effortlessly as rivers seem to do. Nor does it trickle down for free. But that being said, Hadoop *does* provide relatively inexpensive storage, and as such a concept from early data warehousing called "persistent staging" appears to have made a comeback through the backdoor.

Hadoop offers the option of expandable parallel processing (adding additional nodes later on) which future proofs processing capacity "as needed." Hadoop can elegantly handle relational, semi-structured and unstructured data alongside each other. As a result of this, you can now consider storing sensor data, smartphone data, RFID, machine generated data, and lots of formats that are likely to hold *some*, as of yet to be determined information promise. Because of the financial economies, roughly $1000/TeraByte, you do not need to be as concerned about hard and proven business cases before you consider storing these source data.

Adequate sourcing, finding skilled data engineers, has been a consistent bottleneck for most Hadoop projects. The ability in Hadoop to tweak and tune for optimal performance (i.e. by managing data distribution across nodes) has become somewhat of a liability rather than an

asset. Many organizations are desperately trying to reduce the amount of manual effort. Manual coding has the potential for error, tends to increase maintenance costs, and on top of that also taps into already scarce resources. Fortunately, some of the corporate service providers in the Big Data space have recognized this, and attempt to alleviate some of this pain in their offerings.

Although Hadoop has been around for more than 10 years as a data platform, it still does not offer much in the way of data management and data logistics functionality. Hopefully this will be provided in the years to come. As the number and diversity of data sources grow, managing all these data flows can become a source of "human" overhead. If you plan Hadoop to be your enterprise data hub, or persistent staging area, or data lake, you want to be able to ingest new sources quickly and inexpensively. That was and is the luring promise of the data lake.

Some sources, like a financial reporting system, may provide data at regular intervals, and almost entirely in a structured and consistent format. Some of these cycles may be slow (like monthly), others may come (much) faster. Some data source formats are documented and consistent, others are not (say, machine generated data, or log files). Yet you want to offer storage for all these sources at a low threshold, fast and inexpensive. Hadoop does not offer such capabilities natively, but within the eco-system there are several products you can leverage to provide what some have called "Data-as-a-Service."

One of the ways to make development more robust, and to expedite delivery, is through metadata driven development. Models or templates allow dynamic data ingestion, and can standardize the addition of new sources, thus dramatically driving down the time and cost to connect to ever more data sources. Some of these "code generation" tools allow you to create sets of jobs that can be scheduled as well as run at will. This is commonly considered a "best practice" in data warehouse and BI development. Note that any 'hacking' (tweaking, changes) that developers (need to) do in that code may reintroduce errors, and can make this architecture brittle.

Some developers like "drag and drop" interfaces, as they are accustomed to from many ETL tools. As intuitive as that may seem (which *does* in fact lower the barrier to entry), those interfaces usually do not scale very well, unless they also come equipped with some scripting functionality. To drive down cost for adding more data sources, automation is the name of the game. GUI based tools are useful, but not sufficient to deliver on the fundamental premise of data lakes, namely the ability to be able to add more sources quick and inexpensively. Human involvement never scales well.

For data source profiling, and several other use cases, Data Scientists and advanced analysts often want to query the data through means of a SQL-like layer sitting 'on top' of Hadoop (rather than manually write MapReduce jobs themselves). Solutions like Hive or Impala provide just that: an interface that professionals who are comfortable with writing their own SQL queries can quickly and easily become acquainted with. Almost every Hadoop implementation

that offers more functionality than merely a data scientists' sandbox, will provide such relational data access.

As the number of interfaces to source systems continues to grow, managing data logistics becomes an important job. Especially when there are (can be) dependencies in data sources, scheduling of loading jobs needs to be carefully designed. Oozie seems the most commonly used scheduling solution at the moment, where Sqoop often gets used for transferring data from a relational data base into Hadoop, typically landing the data in Hive or HBase. Pig, which provides scripting functionality and Zookeeper are often used for similar purposes as well. One of the legitimate complaints about the Hadoop ecosystem has been that data management functionality is not nearly at a par with traditional, "old school" BI vendors. This shortcoming, at the moment, is preventing it from taking over more relational data warehouse implementations.

Hadoop and Data Governance

Data Governance is not all of a sudden required *because of Hadoop*, this should already be in place. However, when you implement Hadoop, security and privacy considerations are exacerbated due to the nature of the technology, and the sheer volume and variety of data that are likely being stored. If your project was inspired because you decided to store a new and large data source, there are perforce more data to be concerned about. Many databases have some logging functionality which allow DBA's to monitor usage, access and changes; no such functionality is automatically built into Hadoop. Therefore, if monitoring data access is a (business) requirement, you'll need to cater to that "user behavior tracking" feature yourself. This is sometimes also referred to as operational metadata. As of late, Sentry appears one of the few solutions that can enable role based security, albeit exclusively at the SQL access level.

Beside *technical* measures like logging who does what in your Hadoop environment, you also need to have *procedural* guidelines in place about who is allowed to retrieve data or make changes, and when. People with access to such rich data sources need to be aware of the responsibilities that implies. Typically, this gets institutionalized by training, as part of the employee induction program. It may get formalized by behavior protocols that employees need to sign off on, in order to meet compliance rules. But even if compliance doesn't call for it, all these "human" (procedural) measures demonstrate awareness of data sensitivity to employees. They also reflect care for customers' privacy and a genuine appreciation for the responsibility that gathering, keeping, and consolidating all this information brings with it.

An example of a technical measure to safeguard privacy is building a so-called "Chinese Wall" to physically separate analytical information (like behavioral data) from personally identifiable information (PII). Statistical analysts and data scientists can do their job just fine when a customer is identified by a surrogate key (an artificially assigned unique random number). They do not need to know what his name or address is, unless they use the latter to match with ZIP code data. ZIP codes themselves can even be removed, once the relevant external data like a procured ZIP code databases has been joined to the analytical data set.

Companies that are even more concerned about protecting their customers' privacy, can encrypt and shield PII from analysts who have unfettered access to detailed transactional information, since that part of the data set *by itself* is still relatively sensitive. It can be quite remarkable and illuminating how much you can learn about an anonymous ID once you get access to a 360° view of the customer. There is no legitimate need to further jeopardize customers' privacy by giving analysts access to information like name and address that is needed almost exclusively to communicate with the customer.

In some industries there are well established rules and guidelines like HIPAA, SOX, Basel II, etc. In other industries, there may not exist a commonly accepted standard, yet, which means that the onus is on data managers to make sure they meet or exceed the standards that senior management expects. You need to strike a balance between leveraging all the available data, and at the same time doing so in a careful and controlled manner. Besides regulations and compliance, damage to the brand value is a genuine concern. When Target hit the news with their prediction of teenage pregnancy (supposedly), immediately the press was all over this case (even though it probably didn't happen at all). This just goes to show the very real publicity risks of careless analytics.

Conclusion

There are certainly valid use cases for the application of Hadoop. In fact, some of the internet giants like Ebay, Facebook, LinkedIn, Yahoo, just to name a few, have built their success around it. When data volumes become excessive, moving and copying data becomes too expensive, too slow, and a hindrance to real-time processing. Especially when there is a mix of source data *types* (XML, text, JSON, sensor data, geospatial information, rich media files, etc.), Hadoop becomes a much more attractive option. Many organizations consider Hadoop in an attempt to modernize their DWH because of the flexibility with regards to data types, and its scalability, when significant growth in data volumes and variety is imminent on the horizon.

Because of cost-efficient solutions like Hadoop, the threshold for *considering* whether to store or discard data has been lowered dramatically. I would argue that this technology evolution alone has been the single most important driver of Big Data innovations. I consider a close second behind it the steady stream of management books in the past 10 years, that pointed at innovative ways that data-driven organizations have disrupted their respective markets. The pursuit of competitive advantage by leveraging Big Data has triggered a sea change in storage and information management paradigms. As soon as wholesale text analytics becomes within range, I expect yet another big push forward.

Given some of these rather compelling benefits that Hadoop has to offer, the question may arise: "The why aren't *more* organizations benefitting from this?" An important reason is that setting up Hadoop is (still) rather cumbersome. No free lunch there, either. There has been a consistent shortage of talent, and Hadoop is anything but a turnkey solution. Talent in the BI space has usually relied on SQL; with native Hadoop you need Java and scripting skills. There is

a fair amount of manual coding and tuning involved, which brings extra debugging with it. Although there are ever more platforms that will connect with Hadoop, integrating it in your enterprise data architecture is likely an additional hurdle.

Hadoop does not replace the data warehouse as we have known it for a few decades. If you consider a tiered structure, it can offer a worthwhile *complement* to it. Enterprise wide implementations will leverage analytical relational databases that reside downstream from HDFS storage. For cost effective data ingestion, and liberal persistent storage of data sources that are deemed to hold promise for analytics, Hadoop has been a boon. Needless to say, storing data in and of itself doesn't help; only smart analytics can contribute to success in the marketplace.

The advantages of Hadoop usually don't amount to much until you have a need to persist about 1-10 TeraByte of data, or more. Above, say, 50 TeraByte, NoSQL solutions like Hadoop tend to be considerably more cost efficient than relational solutions at the current price points. The reality is that not many organizations can dream up use cases that would require storing so much data, which may well be a chicken-and-egg problem: what *if* you could have all of those source data at your fingertips – what commercial applications might you be able to dream up?

As mentioned before, security and monitoring of data access is not at all easy to guarantee with Hadoop. As a result, governing Big Data in Hadoop presents considerable challenges. The same qualities that computer scientists built into Hadoop to enable speed and scalability, are at odds with comprehensive logging of all user activity. The inherent distributed nature of MPP systems makes them notoriously cumbersome from the standpoint of auditing and control. At the SQL access level (typically Hive, or Impala) role based security is available that can suppress access to certain columns (variables a user may see) or even rows (which subsets of the data is available, like when regions only get to see their "own" detailed data).

Hadoop aggressively replicates data to enable parallel processing, as well as to ensure continuity in the face of failing hardware. Concern should not *just* be over legal repercussions in the event of a data breach. Equally important are problems that can negatively affect customer satisfaction and brand perception. What if a customer demands that his data be removed from all of your systems? How much effort will that cost? Can you provide guarantees that all of his data has been removed? These liabilities with regards to data management need to be acknowledged, understood by all stakeholders, and properly managed and mitigated. No small feat, especially when this platform purports to scale out to an enterprise data warehousing level.

Running SQL queries on top of Hadoop that get translated to the corresponding MapReduce "instructions" are known to be slow, at times. This may have to do with the way data have been distributed across clusters, or it may be that there are either memory or I/O constraints. For ad hoc analytical queries, that can trigger very heavy loads and perforce are unpredictable, there is never a universally best solution. The old adage of "horses for courses" still holds. You can only

optimize against a known objective, and different queries have different targets that would need to be optimized.

Kudu is a new Apache development effort that is meant to provide a middle ground between the HBase speed layer, and the HDFS storage layer. Not optimized for either, it provides a sensible "best-of-breed" that should give consistent adequate response times for all but the largest of organizations that need to push the boundaries of what currently can be provided by the best computer scientists. Possibly solutions like this will lead to technology that behaves less like a Ferrari with temperament: incredibly fast *when* it runs, but in frequent need of maintenance. Most organizations would be better served by a less exciting, but more versatile and reliable vehicle.

A Gartner survey in 2015 showed that 70% of Hadoop deployments will not meet cost savings and revenue generation objectives due to skills and integration challenges. Determining how to deliver business value from Hadoop still represents the second most frequent challenge for those considering this technology, a close second after the skills gap. When you have more data readily available, you may be able to do analyses across business silos –typically blending structured with unstructured data– that were previously out of reach. That is where competitive advantage can be built.

For competitors without these Big Data capabilities it would take a considerable runway to emulate advanced analytics applications. In today's markets where competitors seem to be leapfrogging each other all the time, senior leadership will aggressively leverage *any* source of sustainable competitive advantage. Because proprietary data will never become available to the competition, it is clear that superior decision-making on the basis of those data, gives a competitive edge that may last. Products are easily copied, talent can be lured away, but quicker and better decisions that utilize your own data, can never be copied at all.