FUJITSU

White paper
Solution Approaches for Big Data

Big Data becomes relevant for more and more organizations. They move to new fields of applications where large volumes of data are automatically and continuously generated at various data sources. Traditional IT meets its limitations when analyzing this data. How can you overcome the high level of complexity and the speed limitations? Various solution approaches have been successfully explored and are already productively used. Fujitsu will share its view on what to do in which situation.

Content

Some dreams of today's organizations

Improving profitability and revenue is usually the top priority of an organization. This requires ever improving performance and productivity of their employees, efficiency, effectiveness and competitiveness of the overall business, while minimizing potential risks. The exciting question is how to achieve this faster, better and to a greater extent than your competitors.

- What if you could better predict what will happen in terms of trends, customer behavior, or business opportunities?
- What if you could always take the best decisions?
- What if you could accelerate your decisions making?
- What if you could have critical actions initiated automatically?
- What if you could fully understand the root cause of issues or costs?
- What if you could skip useless activities?
- What if you could quantify and minimize risks?

Contemplating such questions, many managers immediately imagine the opportunities for their business. However, are these only nice dreams, or is there a chance that these dreams can come true?

Data – The most valuable asset of every organization

Apart from human resources, data is the most valuable asset of every organization. Already decades ago, people were aware of this fact and tried to turn their data into value. It was obvious that utilizing data in an intelligent way for their business could support decisions based on real facts rather than intuition, thus helping improve business processes, minimize risk, reduce costs and increase business in general.

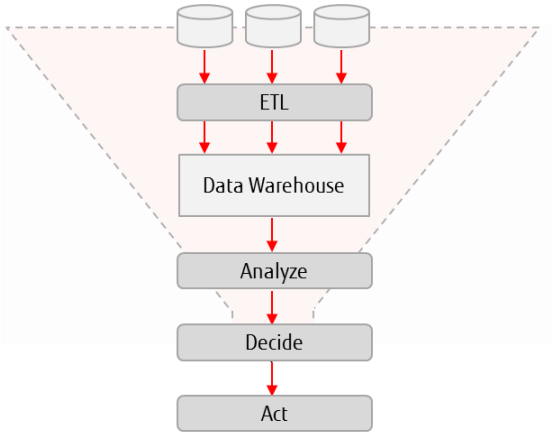
However, they also realized that data in its original form was usually of low value. Therefore data was collected from readily available data sources – mainly transactional databases – and then consolidated and transformed into a form convenient for analysis in order to discover relations, patterns and principles, to finally find the real value. Precisely this exactly was the idea of Business Intelligence (BI) in the early days.

Business Intelligence as it used to be

Typically in the context of Business Intelligence, the transformed data is loaded and stored in a dedicated database, the so-called Data Warehouse, which is separated from transactional systems in order to unload them from business analytics tasks, reports or visualization of query results in general. Data Warehouses are optimized for reporting.

For reasons of performance or authorization, multi-dimensional intervals or other special views are copied from the data warehouse; these so-called cubes or data marts can be used for in-depth analysis or role-aware reporting.

Traditional Business Intelligence considers mainly internal and historical views collected from a few data sources. Data is structured and typically stored in a relational database management system. Business analytics tasks are designed against a static data model, and happen periodically - every day, week or month in a batch process. As the average end user isn't trained to do his own sophisticated analysis, the number of direct users initiating queries or dealing with business analytics is strongly limited to a few specialists.



The situation has changed

Since the beginning of the Business Intelligence era, things have changed tremendously. There are versatile data sources which deserve to be taken into consideration. In addition to transactional databases, it is data from the web, be it blog contents or click streams which can help unveil valuable information, not to forget the content from social media which have evolved to the most commonly used communication platforms. There are multi-media files like video, photo and audio, from which important conclusions for the business can be drawn. There are huge text files including endless logs from IT systems, notes and e-mails which contain indicators that businesses are keen on. And not to forget the vast number of sensors built into smartphones, vehicles, buildings, robot systems, appliances, smart grids and whatever devices collecting data in a diversity which was unimaginable in the past. These sensors represent the basis for the ever evolving and frequently quoted Internet of things.

Having a closer look at specific industries, we should also mention medical scans in healthcare, RFID tags that track goods in motion, as well as geo-physical or spatial data (e.g. GPS-enabled location data) or data coming from observation satellites. This enumeration is far from complete.

It is a given, that every sort of data is constantly increasing in terms of volume, but especially sensors which generate event streams automatically and continuously, will have enormous influence. Therefore it is no surprise that we are all faced with an exponential data explosion.

Let us become more precise, what this exponential data explosion is all about. Analysts speak of 2.5 times 10^{18} bytes which are newly generated every day. 90% of all data originates from the last 2 years. There is a 65% growth of data per year, which equals a 100% growth every 18 months, or a 12-fold amount of data in 5 years compared to today. Consequently we are not just talking about Terabytes; we are talking about Petabytes, Exabytes, Zettabytes and even Yottabytes, and we do not recognize any real boundaries. As a result, many IT managers feel they are drowning in the vast floods of data.

It is not just the number of data sources and the amount of data that is increasing; it is also the types of data that proliferate. In the traditional Business Intelligence era, only structured data that reside in fixed fields of tables in relational database management systems was considered. Today, the majority of data – analysts speak of more than 80% - is unstructured. Unstructured data includes textual data, e.g. articles, e-mail and other documents, and non-textual data, e.g. audio, video and photos. In addition to structured and unstructured data, we should not ignore semi-structured data, which is not organized by fixed data fields, but has tags which separate consecutive data elements. Examples of semi-structured data are XML, HTML and PDF/A data, as well as RSS feeds.

Additionally to this, there is poly-structured data that includes a multitude of structures, which are even subject to change. Examples of poly-structured data are electronic records which represent XML documents with PDF/A elements, or different versions of the same document, which differ in the number of elements or even in the version of the underlying XML schema.

Today's demands are different

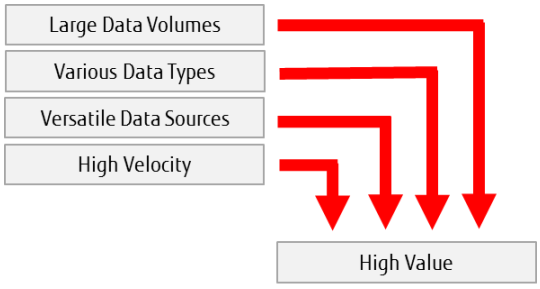
The exciting question is about the impact of all these considerations on Business Intelligence. People have recognized that within this vast amount of data from all the versatile data sources, no matter if structured, unstructured, semi-structured or poly-structured, which remained untapped in the past, much more value can be derived for organizations. But in contrast to traditional Business Intelligence, where reports were generated by batch processing within hours or days, ad-hoc queries with analytics results in real-time are now demanded in order to take immediate decisions proactively, or even initiate actions automatically. Moreover, the focus of analysis is on prediction about what will happen in the future rather than describing things which happened in the past. Due to the broad variety of opportunities all the data offers today, there are many more users who need direct access to business analytics, and this not only from the office but anywhere, from any device, be it a notebook, a smartphone or any other form factor.

Of course, a major requirement is that the solution to meet all these requirements has to be efficient and affordable.

With all this, we have set the stage for a new buzzword and one of the hottest topics and megatrends in today's IT market: Big Data.

Big Data – What it is all about

Big Data combines all the characteristics of data that we have just discussed. Big Data is defined by large data volumes in the range of many Terabytes and more – multiple petabytes is absolutely realistic, various data types (structured, unstructured, semi-structured and poly-structured data) from versatile data sources which are often physically distributed. Quite often, data is generated at high velocity and needs to be processed and analyzed in real-time. Sometimes data expires at the same high velocity as it is generated. From a content perspective, data can even be ambiguous, which makes its interpretation quite challenging.



However, Big Data is not just about the data itself; it is about affordable technologies to quickly store, discover and analyze massive data sets, sometimes even in real-time. High speed processing will enable you to submit queries repeatedly while gradually improving the queries and incrementally improving results. This enables many people, even those with little training, to be productive with analytics – something which would have been also absolutely unimaginable in the past.

In other words: Big Data stands for putting analytics and therefore knowledge at the fingertips of people that need it.

If you are wondering whether Big Data is a topic you should be interested in, there is a fairly simple answer. Just imagine that you, at the time being, utilize 5% of your available data for business analytics, which means in turn that 95% of your data is untapped. If you ignore Big Data and thus you are stuck at 5%, but your competitors from whom we assume that they use data with the same effectiveness are able to handle 15% of their data by using Big Data technologies, it will be pretty clear, which enterprise is more likely to win.

The value of Big Data

The value of Big Data for organizations is multifold. You may discover facts and insights about customers, suppliers and other business partners, about markets and operations, about the root causes of issues and costs, and the potential risks your organization might be faced with.

All these facts and insights would otherwise remain hidden. From newly discovered patterns and behaviors you will derive predictions about future trends and business opportunities which will definitely improve operational, tactical and strategic decision making in terms of speed, quality and relevance. Just avoiding some useless activities bears an enormous potential of cost savings. Big Data enables you to effectively use your data for business advantage and value creation. The option of initiating automatic actions will help accelerate this process.

Let us underline the value of Big Data using some examples. The impact of new insights can be an innovation of your business, your products and / or services. Customers who are likely to churn can be retained, and churned customers will be won back, by reliably evaluating customer sentiments, e.g. from the state of delivery processes and their calls to the helpdesk. New customers will be attracted by identifying current demand, e.g. through social media analysis. At the same time, a better-targeted marketing campaign will yield a higher ROMI (Return of Marketing Invest). Other examples are closely related to improved business processes. Examples are reducing search and processing time, achieving better planning and forecasting by a detailed analysis of historical data, improving the allocation of physical and human resources or increased performance and productivity by automated decision making in real-time. Finally, improvement of efficiency and effectiveness will increase profitability and drive growth. Business will benefit tremendously by being able to quantify and minimize risk. Capitalizing on information will help improve your competitiveness.

Why traditional solutions are not a good fit

As mentioned before, Data Warehouses are the data stores of traditional Business Intelligence solutions. Typically, they are based on relational databases. Relational databases require some sort of structure, which means that unstructured and semi-structured data would need some pre-processing. The tables resulting from pre-processing are often huge but sparse. This in turn means a high amount of metadata, high storage capacities and slow access to data. A row-wise data organization is well-suited for online transaction processing (OLTP), but when it comes to analytical tasks, much irrelevant data will have to be read from the large number of rows, because only information in certain columns is relevant.

Can server scale up improve the situation? No matter how powerful your server might be, there will always be a hard limit for each resource type. With increasing data size, these limits will become a challenge, sooner or later. For sure, in the future the limits will move upwards, but the totality of all data involved in your analysis might grow much faster. Moreover, with high-end servers and scale-up, cost for CPU, main memory and networking will always be relatively high.

If scale up is not recommended, the question remains about relational databases and server scale out. As the database is shared among several servers and accessed by them, the storage connections can prove as a bottleneck. Likewise, the effort to coordinate the access to shared data will increase with the number of database servers. This will – according to Amdahl's law - lead to decreasing efficiency of the individual server, and will strongly limit parallelization. Amdahl's law says that with every parallelization approach the inverse of the sequential portion which cannot be executed in parallel will limit the theoretically achievable performance increase. Hence, in order to achieve a 1000-fold performance increase by using 1000 server nodes, a serial portion of less than 0.1% has to be striven for.

Consequently, everything you try to improve the situation, be it scale up or scale out in conjunction with your relational database, would be time-consuming and expensive and far away from fulfilling real-time demands. Results would be returned too late and insights could become obsolete when being presented to the user. All told, due to the high volume of data, relational databases will outgrow economic feasibility and the required performance is unachievable.

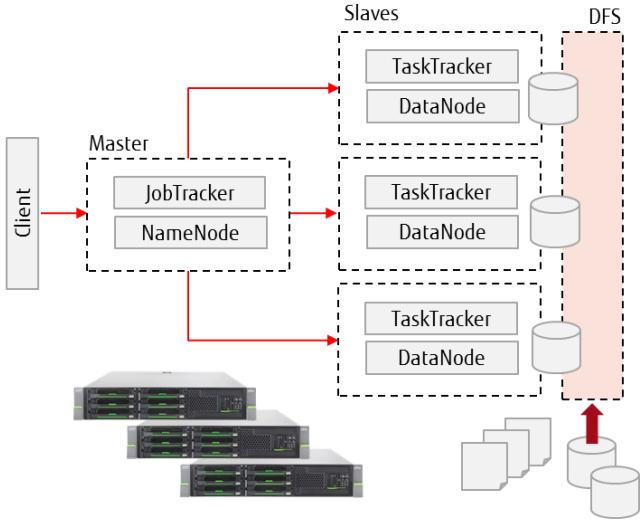
For sure, you could think about building distinct databases and Data Warehouses and split up your analytics tasks. However doing so, you would create disconnected data silos and conduct distinct disconnected analytics which would not give you the comprehensive insight that you expect.

As you would experience limitations everywhere when using traditional solutions, new approaches are required, which keep processing time constant, while data volumes increase.

Distributed parallel processing

The key to success is parallelization based on a "shared nothing" architecture and non-blocking networks ensuring a smooth communication among servers. You distribute I/O to many server nodes by moving subsets of data to the local storage of the servers.

Likewise, data processing is distributed by moving computing tasks to the server nodes where the data resides.



Distributed parallel processing provides several advantages. Executing a query or any other data operation by many nodes at the same time increases performance and delivers fast results. You may start small, with just a few servers, and you add more servers, as they are needed. Basically, your infrastructure will linearly scale out without any limits. Even if 1000s of servers are not enough, the game of adding new ones remains the same.

In order to make the overall configuration fault-tolerant, data is automatically replicated to several nodes. If a server fails, the respective task can be continued on a server where a data replica resides, fully transparently for software development and operation. Data updates need to be considered for all data copies, otherwise the system would be inconsistent.

As such a server farm can be built from industry standard servers, typically with dual-socket and some terabytes of local storage, but without any special additional hardware requirements, a distributed parallel processing solution will prove to be extremely cost-effective, which was not possible a couple of years ago. On the software layer the server farm is coordinated as a distributed storage system which coincidentally runs as a parallel processing cluster. This is seen as one of the biggest enablers for Big Data analytics.

Apache Hadoop

The de-facto standard for Big Data and distributed parallel processing is Hadoop, an open source framework primarily for batch operation, written in Java. Hadoop is a project and registered trademark of the Apache Software Foundation. It is designed to scale up to 1000s of nodes, to accept server crashes as "normal" in large farms, and to make data storage and analytics robust against failures.

The core components of Hadoop are the Hadoop Distributed File System (HDFS) and the Hadoop MapReduce framework.

Hadoop Distributed File System (HDFS)

HDFS is a distributed file system which is in particular designed and optimized for processing large data volumes and for highest availability. It spreads across the local storage of a cluster consisting of many server nodes.

HDFS includes a master server (NameNode) that partitions the original data and assign the data to the server nodes, based on defined rules. Each slave node of the cluster stores just a small fragment of the complete data set. The DataNode which is installed on each slave node is in charge of the local data management.

Each data block is replicated on more than one server for the purpose of high availability. By default, every data block exists three times. Besides the primary data block, one copy typically exists on a server in the same rack, while an additional copy will be on a server in another rack. To increase availability even more, data can be distributed to different locations. Of course, this will not protect against human errors when copying or deleting data; for this purpose, additional backup processes need to be applied.

The HDFS NameNode manages the metadata, thus being always aware of which data blocks belong to which files, where the data blocks are located, and where which storage capacities are occupied. By means of periodically transmitted signals, the NameNode will always know which DataNodes are still working. If the signal is missing, the NameNode will recognize the failure of a DataNode, will remove the failed DataNode from the Hadoop cluster, and will always try to distribute data load evenly across the available DataNodes. Furthermore, the NameNode ensures that the defined number of data copies is always available.

Hadoop MapReduce

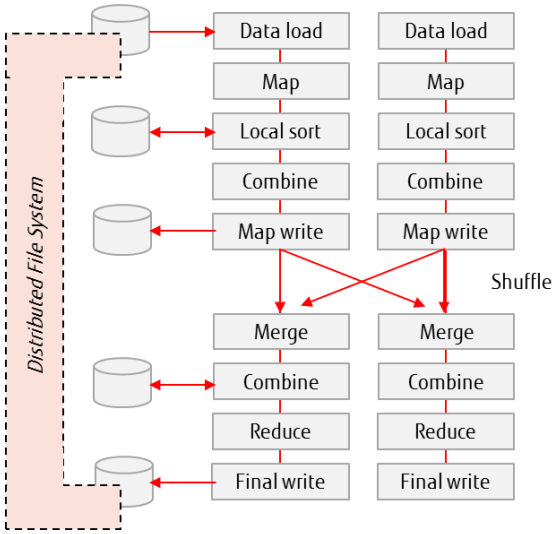
Similar to HDFS, the MapReduce framework works according to the master-slave principle. The master (JobTracker)divides a given problem (job) into multiple tasks (map tasks), and distributes these tasks across the network to a number of slave nodes (TaskTracker) for parallel processing.

Typically, the map tasks run on the same cluster nodes, where the processed data resides. If that server node is already heavily loaded, another node will be selected which is close to the data, i.e. preferably a node in the same rack.

Intermediate results will be exchanged among the nodes (shuffling), and thereafter merged by the reduce tasks to a final result. As the processing tasks are moved to the data and not vice versa, in the map phase basically all I/O activities can be parallelized while network load is almost completely avoided. To avoid any bottleneck regarding scalability in the shuffle phase, a non-blocking switched network without or with only low overprovisioning between the server nodes is required.

Optionally, intermediate results of the map phase and the shuffle phase may be aggregated (combine), in other to keep data volumes transferred from map tasks to reduced tasks as low as possible.

While the input data for MapReduce as well as final results reside in HDFS, the intermediate results are deposited in the local file systems of the DataNodes.



The JobTracker is steadily in touch with the TaskTrackers, monitors the slave nodes and takes care that disrupted or aborted tasks will be executed anew.

If a task does not notify any progress for a longer period, or if a slave node fails completely, all tasks not terminated yet will be restarted on another server, usually on a server with a respective copy of the data. If a task runs extremely slowly, the JobTracker will also restart the task on another server in order to execute the overall job in good time (speculative execution).

The only weak spot is the JobTracker itself, because it represents a Single Point of Failure. Hence, this server should include as many redundant components as possible, in order to keep the probability of failure on a low level.

It is true that MapReduce can be directly applied for executing business analytics. But a frequent use case is to transform data into an optimized shape for analytics.

YARN (Yet Another Resource Negotiator)

YARN is a further development of the MapReduce framework. Cluster resource management and application control which in MapReduce v1 are both covered by the JobTracker, are split up into separated instances. It is just the lean ResourceManager that still exists as a central instance, while the entire job control is delegated to the ApplicationMaster which can run on any slave node. For every job an ApplicationMaster is dynamically created; it is exclusively available for this job. MapReduce v2 will then only look after the parallel data processing.

The distributed application control increases the level of parallelization, removes bottlenecks and enables clusters with 10,000s of nodes.

Apache Hadoop subprojects

Beside the core components HDFS, MapReduce and YARN, there are various open source subprojects and supplements, which make Hadoop a comprehensive platform for analytic applications.

A selection of the most prominent function blocks and their dependencies is shown in the adjacent picture.

Pig with the script language „Pig Latin" enables the creation of scripts for queries and reports, which are compiled into MapReduce jobs.

Hive and the declarative query language HiveQL form a data warehouse used for ad-hoc queries and simplified report generation. The compilation into MapReduce jobs happens automatically. Beside others, Hive can process data from plain text files or HBase.

Sqoop is used for loading large data volumes from relational databases into HDFS and vice versa.

Flume is suited in particular for importing data streams, such as web logs or other log data into HDFS.

Avro serves for serializing structured data. Structured data is converted into bit strings and efficiently deposited in HDFS in a compact format. The serialized data contains information of the original data schema.

By means of the NoSQL databases **HBase, Cassandra** and **Accumulo** large tables can be stored and accessed efficiently.

Mahout is a library for machine learning and data mining

ZooKeeper is a library of modules for implementing coordination and synchronization services in a Hadoop cluster.

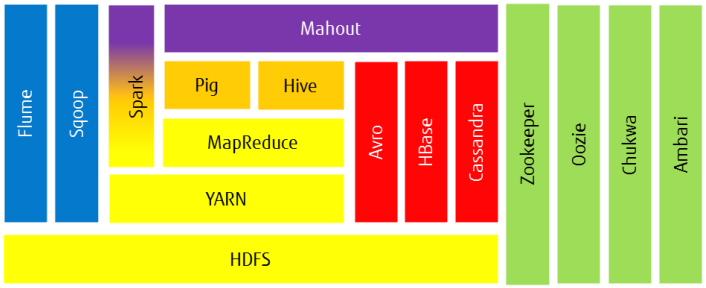
Workflows can be described and automated using **Oozie** by considering the dependencies between individual jobs.

Chukwa monitors large Hadoop environments. Logging data is collected, processed and visualized.

Ambari enables a simplified management of Hadoop environments. DataNodes are monitored with regard to their resource consumption and health status. Cluster installations and upgrades can be conducted fully automated. This increases availability and accelerates recovery.

Using **Spark**, you have the option to load data from HDFS or HBase into the memory of the cluster nodes for faster processing. A core feature is the effective management of in-memory data containers. This is of particular advantage, if multiple queries are applied to the same data sets or computation results serve as input for a next step, as for instance with machine learning algorithms.

Currently there are four APIs leveraging Spark technology: Spark SQL, Spark Streaming, Spark MLib and Spark GraphX. They deliver a datawarehouse functionality like Hive, an event processing service, a machine learning library like Mahout and a graph computaion library.



Distilled essence of Big Data

As mentioned before, distributed parallel processing based on distributed file systems can be used for comprehensive analytics tasks. However, frequently the pre-processing of large data volumes is in the foreground. Data is cleansed, redundancies and contradictions are eliminated, and finally data is transformed into a shape which is more appropriate for ad-hoc queries. The result of this transformation is usually much smaller than the initial data volume, but it contains all essential information needed for the respective use case. Thus, it may be seen as the distilled essence of the overall data.

It is beyond all questions that the access to data on disk storage systems or even AFA (All-Flash-Arrays) providing highest I/O performance and low latency can never be as fast as if data are resident in main memory, and hence closer to the applications. That's why for real-time demands the distilled essence of the transformed data is consolidated into a fast responding in-memory platform.

In-memory platform

An in-memory platform is a single server or cluster of servers whose main memory is used for fast data access. In practice, data accesses are accelerated by a factor of 1,000 to 10,000. Thus the analysis of business data can happen in real-time instead of taking hours, days or even weeks. Important decisions can be taken much faster.

To increase data availability, main memory contents can be mirrored between server nodes and therefore kept synchronously. It goes without saying that mirroring reduces the totally available net capacity of the main memory.

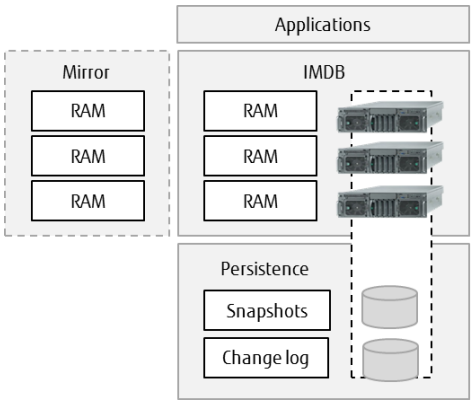
For analytics purposes, no disk storage is needed. However, it must not be ignored that data which is only available in the volatile main memory will be lost after server failure, e.g. caused by a power cut. Using a battery backup for the main memory would help only temporarily.

To prevent data losses, the data contents of main memory must be replicated to a persistent storage. Doing so, several solution approaches are imaginable.

A popular option is the continuous replication of memory data to disk. This guarantees an identical status of main memory and disk at any time.

Alternatively, in order to reduce I/O load, snapshots of the main memory contents may be generated in regular time intervals or when turning the total system off in a controlled manner. However, a system failure could cause the loss of all updates since the last snapshot. By logging the updates, this gap can be closed, too. The system will be able to recover automatically with the latest valid main memory contents from the last snapshot and the change log.

As storage for the data copies, snapshots and the change log, local disks of the server nodes involved in the in-memory platform as well as network storage systems are possible. Using Solid State Disks (SSD) will of course accelerate all synchronization activities and above all the recovery after system failure.



Due to steadily decreasing prices for main memory and the increasing performance of network components which contribute to forming memory contents of several servers to a logical unit, in-memory platforms with ever increasing data capacities become an important building block of infrastructures for Big Data.

Subsequently, various implementation options of in-memory platforms will be checked out.

In-Memory Databases (IMDB)

An In-Memory Database (IMDB) is characterized by loading its data entirely along with the database applications into the main memory of a server or server cluster to conduct rapid analysis. For analytics tasks, I/O is totally eliminated, because no disks are involved.

A relational IMDB is the solution of choice, if several independent applications access the data from different viewpoints and dynamically submit ad-hoc queries unknown beforehand, the results of which are expected in real-time.

While in the starting phase of IMDB primarily scale-up was supported, meanwhile the trend is to support both, scale-up and scale-out. However, it is worth mentioning that scale-out is not in the same high gear as with Hadoop configurations.

If an IMDB is the right choice for the distilled essence depends on various parameters. There is first of all the data size which is limited by the overall available main memory capacity of the server cluster. With column-oriented architectures, the compression factor can also be crucial. Depending on the IMDB implementation and the compression method used it will be between 10 and 50, thus adjusting the capacity limit upwards by the 10- to 50-fold. How many server nodes may be used in total may also vary from IMDB to IMDB. Moreover, the available budget might be relevant; although main memory is steadily getting cheaper, its cost compared to other storage media is still comparatively high.

Some IMDB products embrace OLTP production data and interference-free OLAP queries in the same database instance, thus enabling an analysis of production data in real-time. In such implementations, people frequently use hybrid tables consisting of an optimized column store and a row store. Doing so, updates during transaction processing are deposited in the row store, in order to avoid a permanent sort of the column store. Based on certain criteria, such as system load, size of the row store or after a given time interval, the contents of the row store will be sorted into the column store. Of course, queries are always executed using both parts of the table, as long as the row store is not empty.

Small tables are usually not converted into columns, just as system tables which are also small.

In-Memory Data Grid (IMDG)

If an IMDB is not in the run, an In-Memory Data Grid (IMDG) established between disk storage and the applications deserves attention.

An IMDG is a resident main memory area in which large data sets of external storage systems can be placed. The main memory area may spread across several servers. An IMDG is able to manage any data object which might be needed by the applications. This includes of course non-structured data as it occurs often with Big Data. In contrast to an IMDB, the entire data set need not imperatively fit into the available main memory.

Though, the I/O load is tremendously reduced. Applications are accelerated and analytics tasks can be conducted in real-time. The fact that scale-up and scale-out is supported enables growth with increasing data volumes.

While an IMDB is optimally suited for ad-hoc queries often still unknown upfront, submitted by various applications, the main use case of IMDG is rather for exclusive users with pre-defined queries, which are previously known. The application is responsible for processing the data objects. The IMDG looks after the access to the data, while the applications need not know where the data resides. Additional functions for searching and indexing the data stored in an IMDG make the borders between an IMDG solution and a NoSQL database blur.

An IMDG can be a cost-effective alternative to an IMDB, in particular, if its data is not subject to a high update frequency. The applications accessing the data can normally be used as they are. With the right skills in place to adapt the applications, the potential benefits of an IMDG can be fully exploited.

Where finally the data in an IMDG comes from, does not matter to the IMDG. Data can come from network storage systems or from HDFS. Less imaginable are NoSQL databases whose integrated caching function already provides some of the advantages of an IMDG.

Optimized relational database infrastructures

The question of how to integrate existing relational database infrastructures in Big Data analysis is frequently raised.

A relational database has the limitations discussed before. Therefore, it will definitely not be a good option for Big Data in general. But a relational database can be the data source for Hadoop, and it can be the harbor for the distilled essence, too.

In both cases, speed is important, especially when considering increasing database sizes.

So, what can you do in order to accelerate database accesses? How to reduce I/O, how to get more IOPS out of the storage infrastructure, and how to shorten latency? What are the options to optimize your database infrastructure?

As discussed, the in-memory database concept represents the fastest solution, and an in-memory data grid helps reduce I/O. In this section we will cover further alternatives.

One of the options is **caching**, in which frequently requested records are held in the main memory of the database server. This speeds up database reads, while every write is a write-through to the disk. The larger the cache size is, the more cache hits can be expected, and the less I/O load is generated. On the other hand, extra memory is needed for the cache and some memory and also CPU cycles for the cache algorithms are needed.

If certain data sets are chosen to be held in memory, because of their particular importance, then **memory tables** should be used. As before, extra memory is needed, as well as CPU cycles to maintain the memory tables.

A variation is **RAM disks**, a software solution which enables transparent use of the main memory, as if it were a disk.

While all these flavors take advantage of the fact that data processing in memory reduces I/O, using faster disk technologies in your storage systems is another option, for instance **SSD (NAND Flash based Solid State Disks)**. SSDs offer higher performance than hard disks, and the additional advantage of larger capacities compared to memory.

In order to shorten the route to your data, another option is to equip your database servers with **local flash memory** (e.g. PCIe SSD) that can hold a large portion of the data needed for processing with extremely reduced access time.

What could also be useful is an **All-Flash-Array (AFA)** in front of your storage array, in which you can ideally keep the entire database. However, such architectures will always end up in a tradeoff between size, required performance and cost.

No matter which of the options are chosen, whether access of the storage array is direct or whether a flash array is in between servers and the storage array, a **high-speed interconnect** between the servers and the storage array is strongly recommended. Infiniband with low latency and high bandwidth is a prominent candidate for this purpose. In order to integrate Infiniband with the various storage topologies and protocols in place, hardware or software gateways may be necessary.

Databases for Big Data

Distributed file systems, such as HDFS can cope with large data volumes of various types. However, compared with a file system, a database allows a far better and more efficient access to the data, primarily due to the query language included.

Currently the most widespread database is doubtlessly the relational database. Relational databases are a perfectly suited to transaction processing of structured data of a limited size. They are optimized for a record-wise parallel access by many users, as well as for inserting, updating and deleting records, while queries cause a higher timely effort.

Big Data exceeds the size limits, includes a lot of data which is not structured, and it is about analyzing data which includes frequent queries. All these are aspects that show that relational databases are not the best choice for Big Data.

NoSQL (Not only SQL) databases are especially designed for Big Data and help overcome the limitations of the relational database paradigm. They are not based on a rigid schema, can easily cope with new data types; they allow format changes without disrupting applications. In contrast to the general purpose relational databases, NoSQL databases address just special use cases, and achieve due to their simplicity a higher throughput for the immense data volumes which are typical for Big Data. The queries are similar to SQL.

NoSQL databases are designed to be distributed across the nodes of a server cluster and for scale-out, allowing basically an almost linear and unlimited scalability. Replication of data to several server nodes enables fault-tolerance and an automatic recovery after failure.

As there is a high speed demand regarding data access and data processing, in many NoSQL implementations a caching function iis integrated, keeping frequently used data resident in main memory, thus reducing I/O.

In the field of NoSQL databases there are various data models optimized for different problems.

Key-value stores

The first variant we are going to look at is the key-value store, which consists of a large number of key / value pairs. The value is accessed using the key which uniquely references the value. The structure of the values is not interpreted by the database; this is solely up to the application.

Key-value stores are suitable in particular if Internet data, e.g. click streams in online shopping applications, have to be processed at high speed. Search functions in e-mail systems are another frequent use case.

Document stores

In document stores, data is stored as documents. Similar to key-value stores, the documents (values) are referenced by unique names (keys). Every document is absolutely free with regard to its schema, i.e. you may use the schema which is needed by the application. If the demands change, adaptations will be rather simple. New fields may be added or already used fields may be removed.

Document stores do not include any functions to process data; it is again the application which has to look after this, which increases the development efforts of the applications.

Document stores are used for storing contiguous data in a document (e.g. HTML pages) or serialized object structures (e.g. in JSON format). Twitter for example uses document stores for managing their user profiles while considering followers and tweets.

Columnar stores

The most common and probably most frequently used variant of NoSQL databases is the columnar store or column-oriented database, whose main use case is large amounts of structured data which cannot be reasonably accessed and processed in a relational database.

Imagine large tables with billions of rows representing the data records. The number of columns per record is relatively small, especially in comparison with the large number of rows. Many queries might relate to an even smaller number of columns. In a row-oriented data store, all rows have to be read for every query, what is extremely inefficient.

Storing data in columns helps increase efficiency. Access can be restricted to those columns which are of interest for the query; there is no need to read unnecessary data. Due to the limited number of columns per record, usually an entire column can be accessed in one read. This minimizes the amount of data to be read dramatically. If necessary, a column can be split into multiple sections for simple parallelized processing, in order to accelerate analysis even more.

Row-oriented store

Key	Region	Product	Sales	Vendor
0	Europe	Mice	750	Tigerfoot
1	America	Mice	1100	Lionhead
2	America	Rabbits	250	Tigerfoot
3	Asia	Rats	2000	Lionhead
4	Europe	Rats	2000	Tigerfoot

Column-oriented store

Key	Region	Product	Sales	Vendor
0	Europe	Mice	750	Tigerfoot
1	America	Mice	1100	Lionhead
2	America	Rabbits	250	Tigerfoot
3	Asia	Rats	2000	Lionhead
4	Europe	Rats	2000	Tigerfoot

SELECT SUM (Sales) GROUP BY Region

Data required

Data read, but not required

Column-oriented databases are self-indexing. While providing the same performance benefit as indices, no extra space is required for indexing, and no index area needs to be maintained.

As a certain column contains only one type of data, and quite often there are only few distinct values per column, it is possible to store only these few distinct values and references to these values. Doing so, you achieve an extremely high compression. Typical compression factors are in the range from 10-50. This helps reduce storage capacities and consequently storage costs, too.

The only drawback is the fact that more CPU cycles are needed when inserting or updating data records.

Dictionary compression

Region	Product	Sales	Vendor
0 Europe	0 Mice	0 750	0 Tigerfoot
1 America	1 Rabbits	1 1100	1 Lionhead
2 Asia	2 Rats	2 250	
		3 2000	

0	0	0	0
1	0	1	1
1	1	2	0
2	2	3	1
0	2	3	0

Run-length compression

Region	Product	Sales	Vendor
0 Europe	0 Mice	0 750	0 Tigerfoot
1 America	1 Rabbits	1 1100	1 Lionhead
2 Asia	2 Rats	2 250	
		3 2000	

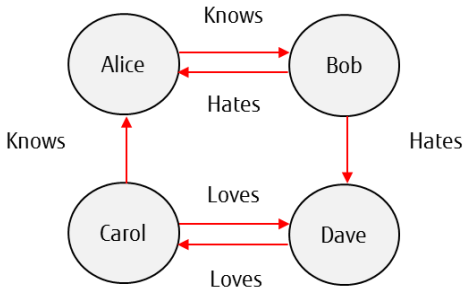
0 x1	0 x2	0 x1	0 x1
1 x2	1 x1	1 x1	1 x1
2 x1	2 x2	2 x1	0 x1
0 x1		3 x2	1 x1
			0 x1

Columnar Stores are optimally suited if data in large sparse tables is needed for ad-hoc queries.

Some columnar stores support **column families**. The database schema describes just the rough structure of these families. Every column family typically represents, from a content perspective, a contiguous area of data with similar access patterns. A subdivision of the family into several columns can happen dynamically without the need to modify the schema or reorganize the database. This enables a very flexible use of the database. New or updated entries are supplied with a time stamp, thus storing in addition to the current situation even a certain history. This makes the dynamic behavior traceable and gives you the option to correct inconsistencies afterwards.

Graph databases

In graph databases, information is represented by vertices and edges including their characteristics. A typical operation applied to a graph database is looking for simple and efficient traversals through the entire graph. Use cases are schedule optimization, geographical information systems (GIS), web hyperlink structures, and determining relationships between people in social networks.



If the number of vertices and edges becomes too large for a single server, the graph has to be partitioned what basically corresponds to scale-out. Doing so, the total graph is divided into sub-graphs what may prove as a difficult or sometimes even insolvable task. In the latter case, some vertices must be allocated to several sub-graphs; this is denoted as overlapping partitioning.

For large databases with a high read intensity and low write load, replicating data to several servers will accelerate processing.

Complex Event Processing

The focus of the previous sections was on data at rest, i.e. data on disk or in memory is waiting for being processed. Now we are going to have a look at data in motion,

Event streams are generated continuously at a very high velocity, for instance by sensors. These event streams have to be collected and analyzed on the fly in real-time. Depending on their relevance, events are filtered and then correlated.

The analysis is based on a set of pre-defined rules that include a condition and an action. If the condition (which may be a correlation of logical and timely aspects) is fulfilled, the action will be triggered in real-time. This is how a CEP (Complex Event Processing) engine works.

Depending on the use case, you have to cope with 100,000s to millions of events per second. Another critical parameter is latency, e.g. the time elapsed between the event input and the event output. Typical latency figures are in the range of microseconds to milliseconds.

For these reasons, a CEP solution has to be fast and scalable. In order to avoid time-consuming I/O, the collecting tank for the event streams is organized in-memory. Just the operational logs are written to disk, if you make use of this option. In such logs, amongst others, bottlenecks with regard to resources are noted, or anomalies, the cause of which is intended to be determined afterwards.

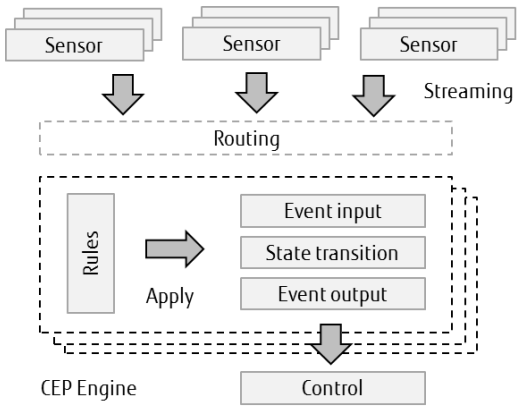
If the load caused by the event streams cannot be managed by a single server, the load will be distributed.

Large event streams are distributed to several servers each of them being equipped with a CEP engine. Incoming events processed or forwarded to other servers.

If many rules and queries have to be processed, a distribution to several servers with CEP engines independent from each other is recommended. In this case, it is attempted to forward different queries related to each other to the same server, if possible.

If certain rules require high main memory capacity, e.g. due to long time windows to be considered, an IMDG will help, which spreads across several servers. Data loses can be avoided due to the high availability features implemented in the IMDG solution (e.g. automatic data replication to another server, or to a persistent storage.

Complex queries, which cannot be executed by a single server, may be split into sub-tasks and distributed across several servers.



Depending on the use case, results of a CEP engine are forwarded to HDFS (e.g. log data), NoSQL databases, In-Memory Databases or In-Memory Data Grids to be used for further purposes, such as analytics, visualization or reporting.

In theory, it is even imaginable to preserve data generated by sensors in other storage systems, e.g. in use cases where everything has to be traceable.

Big Data reference architecture

After having discussed the essential technologies required for Big Data, we are now in a position to present the overall Big Data reference architecture in which these technologies are taken into account. Depending on the use case, some of these technologies are only needed alternatively, or they are not even needed at all. The reference architecture represents the building blocks from which for every customer-specific situation an optimum solution can be designed by combining various technologies.

Let us now give a brief summary of what is shown in the figure below.

Data is extracted from versatile data sources. These data sources can be transactional systems and data warehouses, IT logs, click streams, Internet pages, which can even be linked, e-Mails, text files and multimedia files. If an IMDB is already in place, it may be used as a data source, too.

For accelerating the extraction of data on disk storage, an IMDG can optionally be established. This is not just an advantage for Big Data applications, but also for any other application accessing this data.

Data from all the data sources mentioned is imported into a distributed data management system, e.g HDFS, which spreads across the local disks of server nodes and serves as a storage basin for large data volumes. By means of distributed parallel processing and MapReduce the data is cleansed and pre-processed.

The analysis and visualization of the analytics results may be applied directly to the distributed data management system. Alternatively, data transformed by distributed parallel processing may be exported as distilled essence into another data management system. This will usually be a SQL or NoSQL database. Analytics tasks will then be applied to this database.

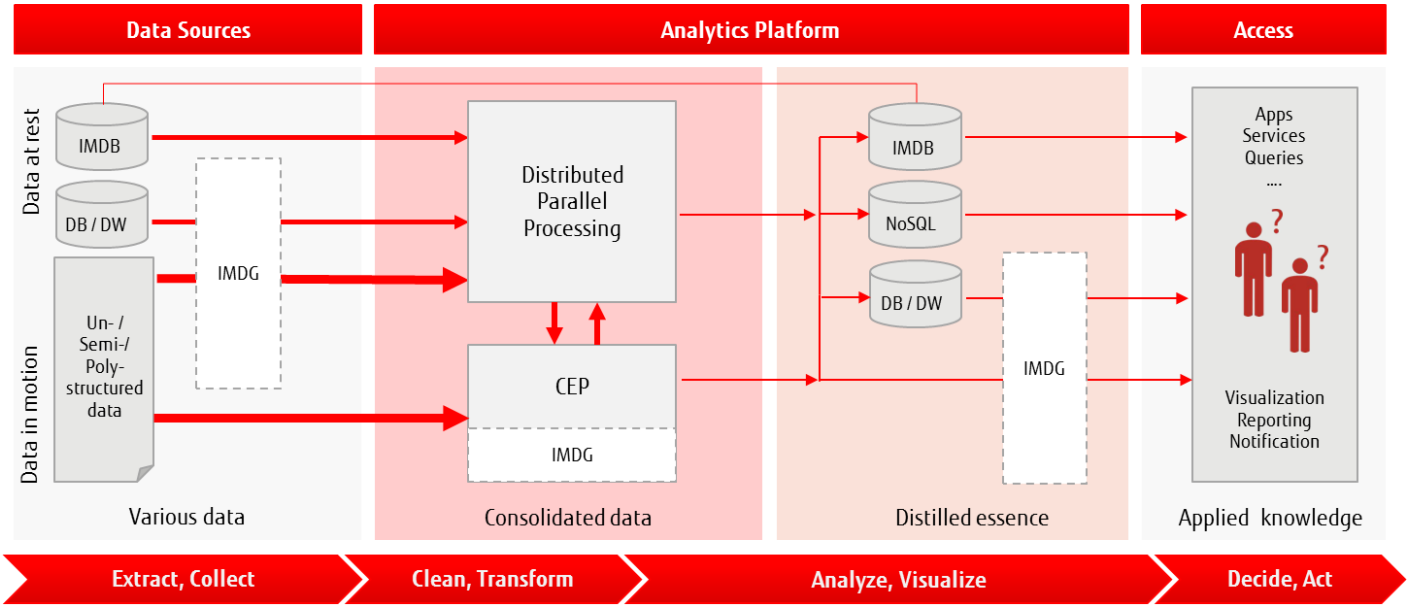
In case of less time-critical use cases, the SQL database may be located on a central disk storage system, which advantageously is equipped with a high-speed interconnect to the respective database server. Using fast storage systems, such as All-Flash-Arrays (AFA) can provide further advantages.

However, if analytics results are demanded in real-time, an IMDB is a good choice. Alternatively an IMDG may be established in order to keep the distilled essence either completely or at least major parts of it main memory resident and accelerate the analytics applications drastically.

While data from the data sources mentioned above are typically processed by Hadoop MapReduce in batch operation, sensor data, intrusion attempts, credit card transactions or other event streams generated at high frequency must be collected, and analyzed in real-time by CEP. Only then, the respective actions can be triggered in real-time, too. The data store for CEP is the main memory; depending on the use case an IMDG is necessary. Results of CEP may also be forwarded to Hadoop, or to the distilled essence outside Hadoop for further processing or visualization. Likewise Hadoop data are sometimes used by CEP.

The magnitude of the arrows symbolizes the data volumes transferred between the individual instances of the Big Data solution architecture.

As every hybrid solution approach, the reference architecture might look highly complex. The more important it is not to face the end user with this high complexity. This is achieved by corresponding connectors between the individual subsystems and by data adapters enabling a smooth transition. Most crucial for the end user is that the analytics functions needed will - fully transparently for the end user – help themselves with the right data from the right data stores. For the end user, not the technology is decisive, but the insight into the available data that will boost his business.



Big Data is not just about infrastructure

We have discussed what Big Data is, which benefits we can expect from it, which infrastructure solution approaches and even which combinations of these might make sense, and what the overall solution architecture looks like.

However, it would be unfair to finish here, because Big Data is not just about infrastructure; it is also about data and processes. Going for Big Data processing means having high expectations with regard to valuable results; for this purpose, transforming data into high quality information is a key requirement. Poor quality will result in poor output and poor user experience, and the Big Data undertaking will be a waste of time and money.

The most common obstacle we observe in organizations is that there is too much data and too few resources, as well as a lack of analytic and technical skills. This leads to a lot of questions, such as

- Which data from which data sources?
- What to look for?
- Which questions to ask?
- Which actions to trigger?
- Which analytic methods?
- How to visualize results?
- Which tools? How to use them?
- How to discover the meaning of data?
- How to do iterative and exploratory analysis?
- What about data retention and deletion?
- How will the first steps look like?

Especially the analytics area deserves special attention. The role profile more and more organizations are looking for is denoted as data scientist, a person with a background in analytics, mathematics and computer science, a profound industry knowledge, with a mandate to think deeply about the data.

Your roadmap to Big Data

Let us have a look at typical steps an organization should take in order to successfully introduce Big Data analytics.

In a first step it is important to align your big data strategy with the key business strategy. Therefore choose areas in which new insight offers biggest impact.

Then break down your big data strategy into manageable objectives, ask yourself what output you desire or which decisions you have to make, and what you need to know. This does not necessarily need to be the maximum possible outcome.

Focus on one objective at a time. This will help shorten project time and accelerate time-to-value.

Create a cross-functional team consisting of data owners, system and tool owners and end user representatives. Data owners know their data; they know which data is required and where it comes from. System and tool owners know system architectures and tools; they are well versed with the tools coming into consideration and know how to integrate the distributed data sources. The end user representatives should have a clear view on what output is required and what the output has to look like.

Having your cross-functional team on board, you can start building test cases. Prepare the right data for the analysis, get the right equipment and tools, and start with a small infrastructure.

Do not try to re-invent the wheel. Rather go the library approach. Identify useful algorithms which are already in place and customize them according to your needs. This will save time and money.

Then produce results and visualize them. Explore combinations of data to create new insight, while asking previously unexplored questions. And not to forget: Open up the results to everybody in the enterprise who can take advantage of them. Doing so, you will take advantage of any feedback you can get.

Beyond this, it is also necessary to address potential security, privacy, compliance, or liability issues.

After all of this has happened successfully, you should increase the scope and also your infrastructure.

Along with all this, we should not leave the required cultural change unmentioned. Especially the data owners and process owners have to give up control of things which were previously exclusively in their hands. Decision makers should learn to accept and respect analytics results. Pretty often this will only work with the management contribution and managerial encouragement.

Operation of Big Data infrastructures

After having extensively discussed the Big Data reference architecture and the various technologies, the question remains how to operate the IT infrastructure for Big Data.

One of the options is certainly self-management in your own data center. Doing so, pre-defined, pre-tested and pre-integrated configurations, appliances and optimally harmonized hardware and software components will contribute to simplify the introduction and the management.

However, the deployment, the integration and the operation of a Big Data infrastructure is complex, requires human resources and special skills. In particular, if there is a lack of resources or a lack of skills, it might be useful to focus on the core competencies and leave the operation of the infrastructure to a service provider, who can deliver this service faster, better and due the achievable scale effects even more cost-effectively.

More and more organizations are no longer interested in keeping complex infrastructures in their own data center. In this case, Big Data from the cloud could be an attractive alternative, relieving the organizations from pre-investments and all operational tasks. The efforts regarding installation, configuration and maintenance are completely omitted. Capacity planning is no longer necessary, too. The required capacities can be flexibly adjusted to changing demands, in particular during occasionally or periodically occurring peak loads.

Then the frequently discussed question if the large volumes of external data really need to be moved through the corporate firewall becomes superfluous. The billing model is "pay per use". Which parameters exactly influence the costs depends on the cloud service and the cloud provider.

Anyway, Big Data services from the cloud provide a huge potential for cost reduction, thus paving the way to Big Data benefits despite small budgets.

It is worth mentioning that Big Data from the cloud provides the greatest advantage, if the essential data sources are also hosted by the same cloud provider (Storage as a Service). Otherwise huge amounts of data would have to be transferred across networks, which would be contradictory to the high velocity demands.

But we should not ignore the fact that there can be good reasons to keep the data in your own datacenter. For instance, in the case of internal data, such as sensor data from your production plant, or video data from surveillance and security systems, which would cause an enormous network traffic; likewise, in the case of sensitive data whose loss would pose an enormous risk for the organization, or which even must not be moved into the cloud for compliance reasons; or huge volumes of volatile data accruing at such a high frequency that would prevent a fast and timely transfer into the cloud.

IaaS, PaaS, SaaS or even Data Science as a Service?

If you fancy with Big Data from the cloud, the question will come up, what exactly is to be delivered from the cloud. It might be most convenient, if you need neither care about the analytics applications and the middleware, nor the infrastructure including servers, storage systems and networks.

Organizations may use the infrastructure of a cloud provider while taking care of middleware and analytics tools themselves.. This variant is known as Infrastructure as a Service (IaaS). If in addition to the infrastructure, Big Data middleware, e.g. Hadoop, is delivered by the cloud provider, we speak of Platform as a Service (PaaS). In case of Software as a Service (SaaS), organizations even use the analytics tools delivered by the cloud provider.

Consequently, only one task is left for the organization: finding the frequently quoted needle in the haystack, i.e. clarifying all the data science questions for which the hardly available data scientists are looked for. As these data scientists are rare, some cloud providers go a step further and even offer Data Science as a Service, a service that quasi helps the customer look for the needle in the haystack and even find it.

How Fujitsu can help

Big Data does not just represent a big opportunity for organizations; it also poses challenges with regard to processes and infrastructure which should not be underestimated.

Fujitsu addresses all dimensions of Big Bata challenges, which means that we cover the aspects of data, processes and infrastructure. In an end-to-end service approach, we consult our customers in terms of what can be achieved by Big Data and jointly define the roadmap to Big Data. We schedule and prioritize the use cases, design and implement the solution, we integrate it into the existing IT landscape, and we maintain the overall solution, even consistently across country borders or globally, if required.

Our services are based on proven methodology and our project experience across all industries.

Fujitsu supports all relevant Big Data infrastructure concepts, and will thus always elaborate the optimum combination of technologies according to the customer-specific situation and requirements. No matter if you need a server cluster running Hadoop Open Source Software, no matter how large the cluster has to be, no matter whether in-memory technologies for real real-time analytics demands are needed, no matter if on-disk solutions are sufficient, or if even a combination of various approaches is the solution for you, Fujitsu will have the right answer to establish the appropriate solution for you.

The well-designed solution stack includes proven servers, such as our PRIMERGY industry standard servers and our PRIMEQUEST servers meeting mission-critical demands, and storage systems from Fujitsu, such as ETERNUS DX, respectively AFA from our partner Violin Memory. Software is either from Fujitsu itself, open source, or from leading independent software vendors with whom we have a close partnership. For backup or archiving, Fujitsu's data protection appliance ETERNUS CS is a great choice.

Fujitsu's products are based on standards, thus helping you avoid any vendor lock in.

Integrated Systems lower the entry barrier and enable a fast time-to-value. Especially worth mentioning are FUJITSU Integrated System PRIMEFLEX for Hadoop (a Hadoop cluster enabling even business users to conduct analytics in an easy manner), FUJITSU Integrated System PRIMEFLEX for SAP HANA (SAP's IMDB solution) and the IMDG solution FUJITSU Integrated System PRIMEFLEX for Terracotta BigMemory (which is supplemented by special tuning services for maximum performance).

Attractive financing options enable the introduction of Big Data solutions without any pre-investment.

If you want to go a step further, if you want to be relieved from the operation of your Big Data infrastructure, in order to be able to focus rather on your strategic projects, Fujitsu will provide the respective managed services, while charging you on a monthly basis, which turns your CAPEX into OPEX.

If you do not want to care about any infrastructure aspects at all, you can even get Big Data as a service delivered from the Fujitsu Cloud.

In other words: Fujitsu is a one-stop shop for Big Data where businesses can get complete solutions including all sourcing options from a single source. This will reduce complexity, time and risk.

Summary

Big Data is more than just high volumes. It is also characterized by a variety of types, versatile data sources and interpretation options, velocity in terms of continuous generation and the need to quickly get analytics results, and finally by technologies which enable all this in an affordable way.

Big Data offers an enormous potential for business value. Decisions based on real-time data rather than (frequently wrong) intuition will enable you to act smarter in the future. If you want to be competitive, there will be no way around Big Data. Be aware: If you ignore Big Data, your competitor will not! Therefore it is advisable to learn how to swim in the ocean of Big Data as early as possible.

Depending on the business requirements, the current infrastructure of the organization and further aspects, different solution approaches or even a combination thereof are imaginable.

Fujitsu is a one-stop shop for Big Data where organizations can get optimized infrastructure solutions based on a blend of various technologies, including hardware, software and services –all from a single source. This makes Fujitsu the partner of choice for Big Data solutions and big success.

Contact
FUJITSU Technology Solutions GmbH
Address: Mies-van-der-Rohe-Strasse 8,
80807 Munich, Germany
Phone: +49-7203-922078
Fax : +49-821-804-88429
E-mail: gernot.fels@ts.fujitsu.com
Website:www.fujitsu.com/fts