University of West Bohemia

Faculty of Applied Sciences

Department of Computer Science and Engineering

Master Thesis

Tools and Methods for Big Data Analysis

Pilsen, 2016 Miroslav Vozábal

Declaration

I hereby declare that this master thesis is completely my own work and that I used only the cited sources.

Pilsen, June 20th, 2016 Miroslav Vozábal

....

Acknowledgements

There are a number of people without whom this dissertation might not have been written, and to whom I am greatly indebted.

To begin with, acknowledgments are reserved to my tutor Roman Mouček, Ph.D. for his valuable guidance, sustained interest, patience and supervision.

My thanks also go to my friends Jamie and Julie who reviewed my work more than once, and with dedication corrected grammatical and other mistakes, making it possible for the completion of this study.

Finally, I would like to thank all those people who supported and encouraged me during my studies especially my friends and all the lecturers.

Abstract

Tools and Methods for Big Data Analysis

Nowadays the volume of data generated by machines and human interactions is rapidly increasing along with the development of technologies that try to address this problem. Although Big Data is widely discussed in theoretical manners, there is a deficiency in publications and sources dedicated to its practical usage. The aim of this study is to explore the Big Data domain, provide overviews of free available biomedical databases and Big Data technologies, choose the most appropriate database and technology, and to apply the technology over the chosen data. Finally, the solution is presented and tested over two use cases.

Keywords: Big Data, Hadoop, Matlab, cluster, distributed computing, EEG/ERP

Table of Contents

1 Introduction

Every second sees huge amounts of exponentially growing data, being generated, acquired, stored, and analysed. The revolution in the generation of massive data amounts comes along with the Internet usage which allows data exchange between various electronic devices and humans. In this regard, the following fields are mentioned: mobile phones, social media, imaging technologies to determine a medical diagnosis, etc. The volume of available data continues to grow and it grows in different formats. Conversely, the price of data storage continues to fall which results in data storing being more reachable. Although creating data storage is getting cheaper and more available, the increasing volume of data in different formats and from diverse sources creates new problems with regards to data processing, including in its analysis and in integrating Big Data into business decision making processes.

In order to store and process Big Data, new technologies are evolving to address these problems. To deal with these challenges, there is a need for a new approach such as building a scalable and elastic architecture.

The purpose of this study is to explore the domain of the Big Data problem; particularly, to create an overview among free available repositories of biomedical Big Data and to discover appropriate technologies and methods along with their limitations and use cases to be applied over the chosen data.

Since the data, technologies and methods are chosen, a testing scenario is created and deployed over this data.

Finally, the achieved results are discussed and some recommendations are provided.

2 Big Data Overview

2.1 Data Evolution

To better understand what Big Data is and where it comes from, it is crucial to first understand some past history of data storage, repositories and tools to manage them. As shown in Figure 1, there has been a huge increase of data volume during the last three decades.

As we can see in the decade of 1990s the data volume was measured in terabytes. Relation databases and data warehouses representing structured data in rows and columns were the typical technologies to store and manage enterprise information.

Subsequent decade data started dealing with different kinds of data sources driven by productivity and publishing tools such as content managed repositories and networked attached storage systems. Consequently, the data volume was started being measured in petabytes.

As it is shown, the decade of 2010s has to deal with the exponential data volume driven by variety and many sources of digitised data. Almost everybody and everything is leaving a digital footprint. Some applications that generate a lot of data are shown in Figure 1. Due to much data, it has been considered to start measuring data in exabytes.

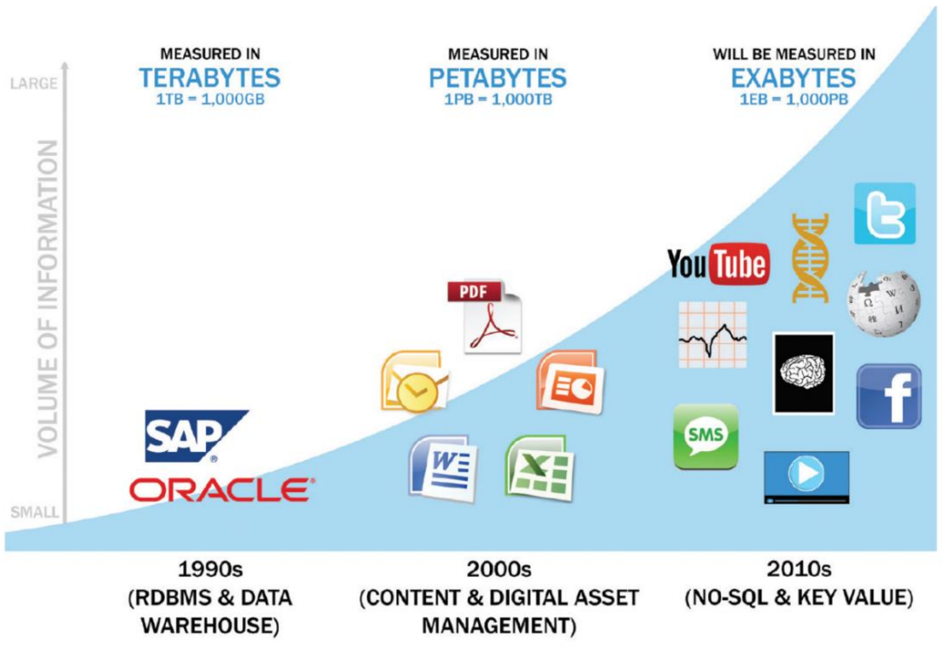


Figure 1: Data evolution and the rise of Big Data sources [1]

2.2 Sources of Data

The sources of increasing the amount of data can be divided into a few categories of data generation:

- Machine,
- Human Interaction, and
- Data Processing.

Typically, the first type is bound up with the spread of machines digitalisation which is related to sensors integration, the connectivity increase, and devices recording sounds, images or videos and to machines communication between each other. Particularly, devices such as cameras recording videos, cell phones collecting geospatial data, machines in production lines of industrial systems, are exchanging important information while processing their activities. More examples are mentioned below:

- Medical information – machines recording EEG (Electroencephalography), heartbeats, genomic sequencing,
- Multimedia – photos and videos uploaded on the Internet,
- Mobile devices – provide geospatial data (location, gyroscope), as well as metadata about phone calls, messages, internet usage and data gathered by mobile applications,
- Other devices – Warehouse Management Systems (WMS) providing inside location realised by Wi-Fi, identification of stored products or material by BAR, QR (Quick Response) codes or RFID (Radio-Frequency Identification) chips. There is a presence of many other technologies such as navigation systems, seismic processing, etc.

It is appropriate to consider other sources generated by activities on the Internet in general. Let us imagine having a set of servers which run web pages that can be determined for retail business. The servers can collect records of all activities of the websites' customers, users, transactions, applications and servers own activity and behaviour. For instance, there are logs which can be collected of [2]:

- Applications – users' activities and applications performance,
- Business processes – account changes, purchases and trouble reports,

- Clickstream data – to store customers’ interests,
- Configuration files - stored configuration of servers and applications,
- Database logs – to show who made database changes.

In order to become more conscious of the data production, it is convenient to mention another example that is related to a collection of astronomic data. In the year 2000, a project concerning sky mapping, by the name of Sloan Digital Sky Survey (SDSS) has been launched. During the first few weeks of the project, an extensive amount of data has been collected by a telescope; more data than there has ever been collected throughout astronomy history. The dataset consisted of approximately 140 terabytes of data in the year 2010. Large Synoptic Survey Telescope (LSST), a project, which is planned to be launched in Chile during this year, is supposed to collect the same data amount every five days. [3]

The next category concerns information exchange among people. For instance, some examples of social networks can be Facebook, Twitter, LinkedIn, etc. Every second these systems generate a huge amount of data shared by millions of people.” *For example, in 2012 Facebook users posted 700 status updates per second worldwide.”* [1] The number of messages generated by Twitter is increasing at a rate of 200% annually. In 2012 it has exceeded 400 million tweets per day. [3]

Data processing is concerned to deal with raw or already handled data to get an output or to get to another process phase that is involved in a particular project. To give an illustration of which fields are considered, let us look at the case of the health care industry. For instance, analysis of data generated by an EEG recorder can produce new data. Particularly, EEG data is considered as the input which is processed by an algorithm generating new data volume stored for further analysis.

The production, availability and the presence of a Big Data amount around us is described by [1] as Data Deluge. Figure 2 highlights several sources of the Big Data Deluge.

Figure 2: What is driving the data deluge [1]

In summary, there are many sources generating a lot of data that can be stored for further analyses and explorations which as we will see, could head toward desirable knowledge.

2.3 How Big Data can be applied

There are many fields where Big Data can be applied in. Largely it is seen as a source of information for making business decisions, improving competitiveness, reducing expenses and many others. Although the following example is not certainly a typical one for the enterprise, it shows us a way of a possible application in this area. It is software used to find the best time to book a flight which was created by an American professor Oren Etzioni. Due to not having access to the attributes of the decision process which determines prices of flying tickets, Etzioni decided to utilise Big Data to predict the best moment for buying a ticket to save people money. His application called Farecast, used a big volume of data processed by statistical methods to get the prediction. The model probability precision was 75%. It saved almost 50 USD of a customer per flight in the year 2012. Farecast used approximately 200 billion records for the prediction. Finally, the product was sold to Microsoft. Nowadays, it is not available anymore. Microsoft stopped running Farecast.

A wide application can be seen in the health-care industry. One of these applications can be the case of a doctor called Carolyn McGregor. Her scientist team is working on software which helps to the decision making of working procedures related to premature new-borns. This software records and processes child patient data in real-time. It tracks sixteen different data quantities such as the heartbeat, frequency of breathing, temperature measurement, blood pressure, etc. The system can detect changes of the state of a child’s health, which can signalise coming of an infection 24 hours before the first symptoms of the disease appearance. It is a prediction and it does not imply the cause of the change, but it can warn doctors that something in the background is happening. Due to it, they can start treatment in advance.

2.4 Definitions of Big Data

There are many definitions for Big Data. Somebody defines Big Data as data that is quite complicated rather than “easy going” and it is hard to acquire, store, manipulate and process it, due to the fact it is “big”. Another one, which is frequently mentioned is called “3V” described by three words (Volume, Variety and Velocity). These two definitions above and others are cited and mentioned below:

> *“Big Data can be defined as volumes of data available in varying degrees of complexity, generated at different velocities and varying degrees of ambiguity, that cannot be processed using traditional technologies, processing methods, algorithms, or any commercial off-the-shelf solutions.” [4]*

[1] Argues that there are three attributes standing out as defining Big Data characteristics:

> *“Huge Volume of data: Rather than thousands or millions of rows, Big Data can be billions of rows and millions of columns.”*
>
> *“Complexity of data types and structures**:** Big Data reflects the variety of new data sources, formats and structures, including digital traces being left on the web and other digital repositories for subsequent analysis.”*
>
> *“Speed of new data creation and growth**:** Big Data can describe high velocity data, with rapid data ingestion and near real time analysis.”*

Another definition of Big Data comes from McKinsey [5]:

> *“Big Data is data whose scale, distribution, diversity, and/or timeliness require the use of new technical architectures and analytics to enable insights that unlock new sources of business value.”*

The most known and used definition is originally described as “3V” (see Figure 3). Since then various authors have come up with other definitions and descriptions. The combination of these thoughts is described below:

- Volume – the volume is increasing exponentially. There are many sources generating a huge amount of data which takes a long time to process.
- Variety – Big Data is not always structured data. Actually, most parts of it are unstructured. In addition, it comes in different formats due to the variety of data sources; dealing with the data formats variety, increases the complexity of storing and analysing.
- Velocity – describes the rate of data change. Data volume changes dynamically. There is a need to manipulate with data in real-time. However, for some tasks in fields such as business and health-care, it can be somehow demanding.
- Value – it is often argued that the value is the most critical part of Big Data. Most of projects have to produce appropriate results, unless the company/institution can waste financial sources. It costs a lot of money to implement IT infrastructure systems to store data. If the subject which stores data is not able to extract data value, it is desirable to consider if the intention of the investment to an implementation of IT infrastructure systems was suitable.
- Veracity – not all data has to be perfectly good, but it does need to be almost good to give relevant sight. Upon considering the errors rate and the data incompleteness, ambiguity in the dataset is desirable and even necessary for further data analysis. Credibility is another aspect of data veracity which is worth mentioning.

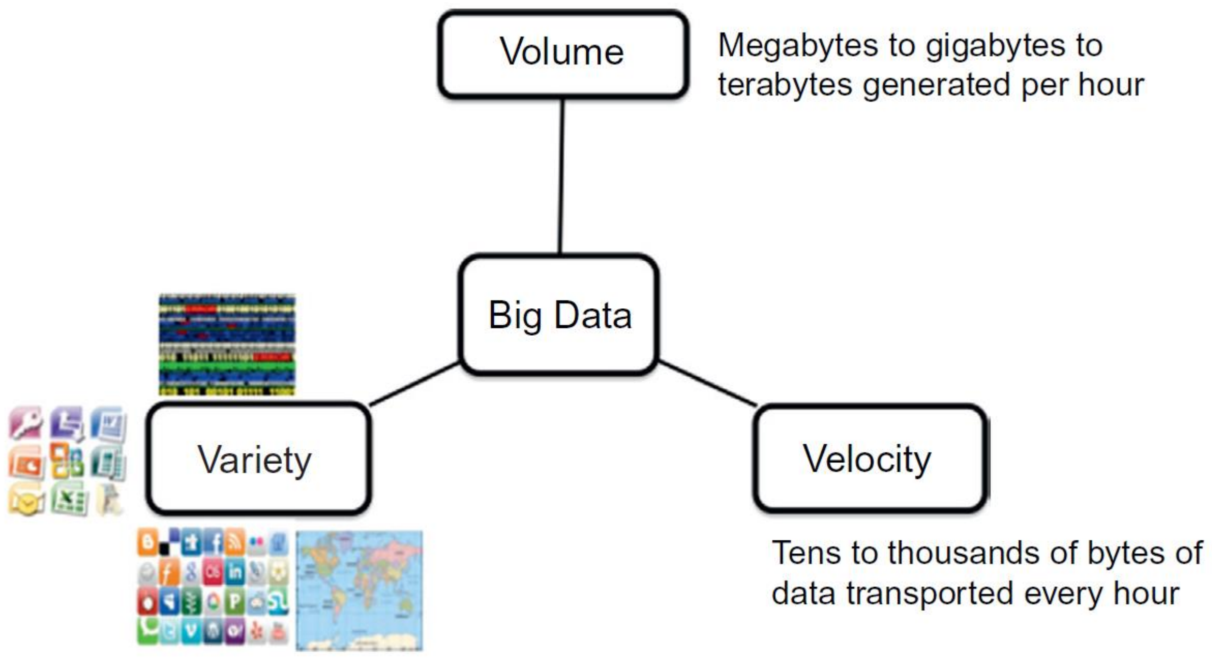

Volume
Megabytes to gigabytes to terabytes generated per hour
Big Data
Variety
Velocity
Tens to thousands of bytes of data transported every hour

Figure 3: Big Data characteristic [4]

2.5 Characteristics of Big Data

In relation to the definitions, the reason why there is intense complexity in processing Big Data is shown. Along with the Big Data there also exists ambiguity, viscosity and virality (see Figure 4).

- Ambiguity – emerges when there is less or no metadata in Big Data. An example can be a graph or something that usually needs a description. Letters M and F in a graph can depict genders or they can even represent Monday and Friday.
- Viscosity – this term is often used to describe the latency time in the data relative to the event of being described.
- Virality – describes how quickly data is shared throughout a network among people who are connected. The measurement result is the rate of spread of data in time. For instance, Twitter can be a relevant example when the tweets are spreading from the first (root one) original tweet among people throughout the network.

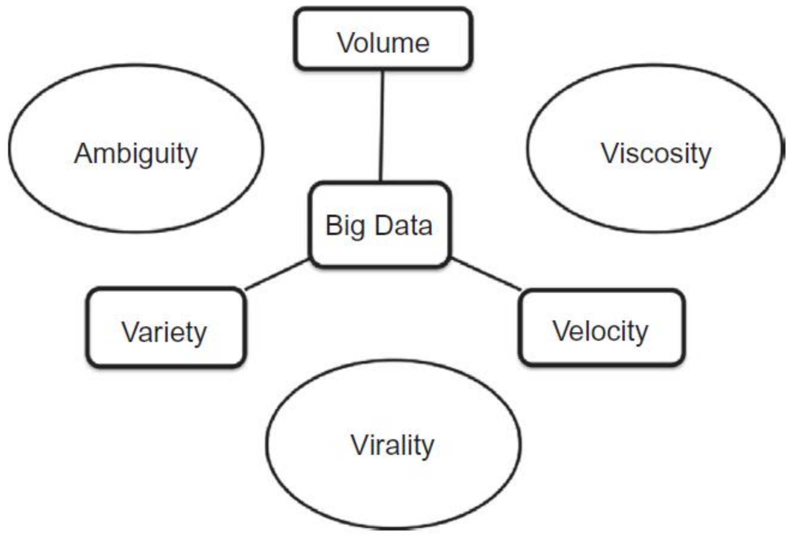


Figure 4: Big Data characteristics derived from '3 V' definition [4]

2.6 Big Data versus Small Data

Big Data is not simply small data that has grown. There are more aspects that define differences between these two categories. A subset of the aspects can be derived from the "3V" definition described above and the others are argued by [6] (see Table 1).

SMALL DATA	BIG DATA
GOALS	
Usually designed to answer a specific question or to achieve a particular goal	Nobody knows what the exact output of the project is. Usually it is designed with a goal in mind, but the goal is flexible
LOCATION	
Typically, small data is contained within one institution, often on one computer or even in one file	Big data are located throughout the company network or throughout the Internet. Typically, it is kept onto multiple servers, which can be everywhere
DATA STRUCTURE	
Ordinarily contains highly structured data. Commonly, the data domain is restricted to a single discipline or its sub-sequence. The typical forms of its storage are uniform records or spreadsheets	Has to be capable of absorbing unstructured data such as text documents, images, sounds and physical objects. The subject of disciplines can vary throughout the data
DATA PREPARATION	
In many cases, the data is prepared by its own user for his own purposes	The data is collected from many different sources. People who the data comes from are rarely the same people who use the data
LONGEVITY	

The data is kept for a limited time (academic life). After few years when the data project is finished, the data is usually discarded	A Big Data project typically contains data which has to be stored in perpetuity
MEASUREMENTS	
Typically, the data is measured using one experimental protocol	Many various types of data are measured in many different electronic formats
REPRODUCTIBILITY	
The projects are typically repeatable. If there is a problem with the data quality, the entire project can be repeated	Replication of the project is seldom feasible. There is nothing more than optimism that the bad quality data is found and flagged, rather than be replaced by a better one
STAKES	
Project costs are limited. The institution can usually recover from small data failure	Big data projects can be really expensive. A failed Big Data project can lead to bankruptcy
INTROSPECTION	
Data points are identified by rows and columns within a spreadsheet or a database. It enables to address a particular data point unambiguously	It is more difficult to access the data. The organisation and the context can be inscrutable. Access to the data is achieved by a special technique referred to as introspection
ANALYSIS	
In most cases, all the data in the project can be analysed together	Big data is typically analysed in incremental steps. It is extracted, reviewed, reduced, normalised, transformed, visualised, interpreted and reanalysed with different methods

Table 1: Big Data Versus Small Data

Additionally, in relation to the volume of Big Data and its incompleteness, corruption and ambiguity that are caused by its large volume, the Small Data is generally more organised, structured and it is also possible and desirable to avoid the incompleteness together with the other attributes of Big Data (see Figure 5).

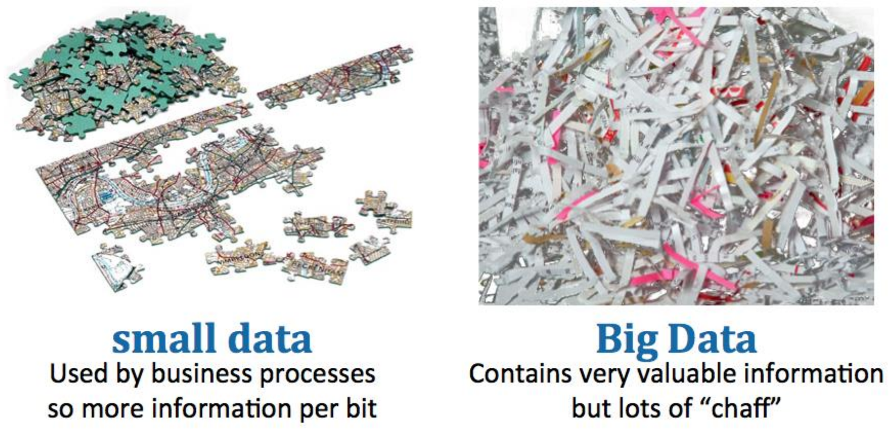

small data
Used by business processes so more information per bit
Big Data
Contains very valuable information but lots of "chaff"

Figure 5: Small and Big Data illustration[1]

2.7 Data Structure

Big Data usually comes in a structured or non-structured form such as multimedia files (videos, images and sounds), text files, geospatial and financial data, etc. Regardless of effort to store and analyse traditional data, most of Big Data is structured or semi-structured in nature, which requires different techniques and tools to store, process and analyse. There are four basic categories which are shown in Figure 6.

It is argued by [7] that approximately 80-90% of future data growth is coming from non-structured data types.

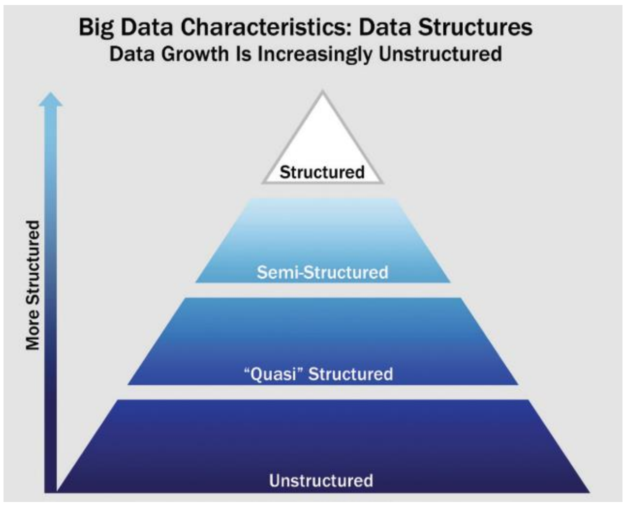

Big Data Characteristics: Data Structures
Data Growth Is Increasingly Unstructured
More Structured
Structured
Semi-Structured
"Quasi" Structured
Unstructured

Figure 6: Big Data Formats [1]

[1]Networkworld.com: http://governyourdata.com/

Big Data can be assigned into four groups:

2.7.1 Structured

It is data that is stored in a structure that defines its format. As a typical example can be considered a traditional RDBMS (Relation Database Management System), transaction data, data files such as spreadsheets in CSV (Column Separated Values). Typically, structured data is categorised into groups which define how they will be stored (data types: number, text, etc.), processed, accessed and restricted (a set of possible values: male, female). Due to its structure, this type of data can be easily stored, processed and queried.

2.7.2 Semi-structured

Although it is a type of structured data, it has no strict model structure. The best example is textual data files that enable being parsed such as XML (Extensible Markup Language). The data files describe themselves by an XML schema.

2.7.3 Quasi-structured

In spite of not being commonly mentioned, this group can be added to these three categories as the one in between the semi-structured and the unstructured. An example of this data set represents clickstream data. Let us consider visiting a few websites, the URLs (Uniform Resource Locators) are tracked into log files of the webpages' servers, the user's computer activity is being monitored. For example, there are three links below:

https://www.google.com/#q=EMC+data+science

https://education.emc.com/guest/campaign/data_science.aspx

https://education.emc.com/guest/certification/framework/stf/data_science.aspx

As [1] argues, the set of these URLs defines a clickstream that can be parsed and mined by data scientists to discover usage patterns and uncover relationships between clicks and areas of interest on websites. Then quasi-structured data can be defined as data which is rather unstructured and in an erratic format which can be handled with special tools. The clickstream data may contain inconsistencies in data values and formats). [1]

2.7.4 Unstructured

It is data that cannot be easily classified which makes it the opposite of structured data. They do not have any inherent structure. Typical examples which can be assigned to this group are images, videos and text documents.

2.7.5 Combination of the Four Groups

Although there are four different groups describing data structure, separate types of data are usually mixed together. For instance, there can be a classic RDBMS storing call logs for a software support call centre. In this case, all data such as date/time stamps, machine type, type of the problem and operation system, can represent the structured type. Moreover, the system might store quasi- or semi-structured data such as free-form call log information taken from an e-mail ticket of the problem, customer chat history or an actual phone description of the technical problem and its solution. There is much information that can be extracted from the available set of the data of different structures.

2.8 Data Analysts and Data Storage

In order for an analyst to process data in general, there has to be a repository where the data has to be stored. A few examples of repositories and aspects which are associated with it will be mentioned below. Due to its wide spread, it is appropriate to mention spreadsheets. They can be seen as the easiest way of data storage allowing business analytics to create simple logic on data structured rows and columns to create their own problems analyses. Additionally, the spreadsheets are easy to share and end users have control over their logic. Nevertheless, some disadvantages are emerging too. In spite of their controllable logic and simple sharing, one can deal with the problem of their versioning. Consequently, in relation to the sharing, finding the right (the most actual) version can sometimes be a hassle. Moreover, there is no implicit backup when a spreadsheet is lost; it is lost with the data and the logic.

A more scalable solution of storing data is data warehouses. They manage the data centrally, making the repository a unique source of data which users can rely on. In other words, the central way of dealing with data brings other benefits such as security and a possibility of backing up to avoid failure. On the other hand, these advantages come with some restrictions. The rules of keeping data organised in the meaning of versioning, central storing and being available only to users with appropriate rights to their access, come with their contrary capabilities. Data warehouses are usually handled and controlled by IT groups. This implies longer time for analysts to get data for creating datasets or making any changes, due to waiting for confirmations. Data warehouses solve problems related to data accuracy and availability. However, other problems emerge such as a lack of agility and flexibility, which is not desirable in the context of Big Data. In other words, data warehouses restrict analytics' work.

A solution comes with technology called analytical sandboxes. In comparison with traditional data warehouses, analytical sandboxes attempt to provide a less restricted environment to enable robust analytics being centrally managed and secured.

2.9 Emerging Big Data Ecosystem

A new ecosystem comes with the data deluge. Organisations and data collectors that can gather data from individuals start realising that a new economy is emerging as data business. Depending on the evolution of this economy, a new ecosystem is rising. As it is rising, the market sees the introduction of data vendors and cleaners that use crowdsourcing such as GalaxyZoo and Mechanical Turk. There are other vendors which offer added values by providing tools for data storage, processing and analysing. Most of them repack open source tools to make them easier for their customers. Typically, they add another functionality which makes their software commercial solution. We can discuss frameworks such as Hortonworks, Cloudera and Pivotal whose solutions are based on an open source framework called Hadoop.

As the ecosystem has been growing, groups of interest have been formed. Currently there are four main group which are associated with each other (see Figure 7):

- Data Devices,
- Data Collectors,
- Data Aggregators, and
- Data Users and Buyers.

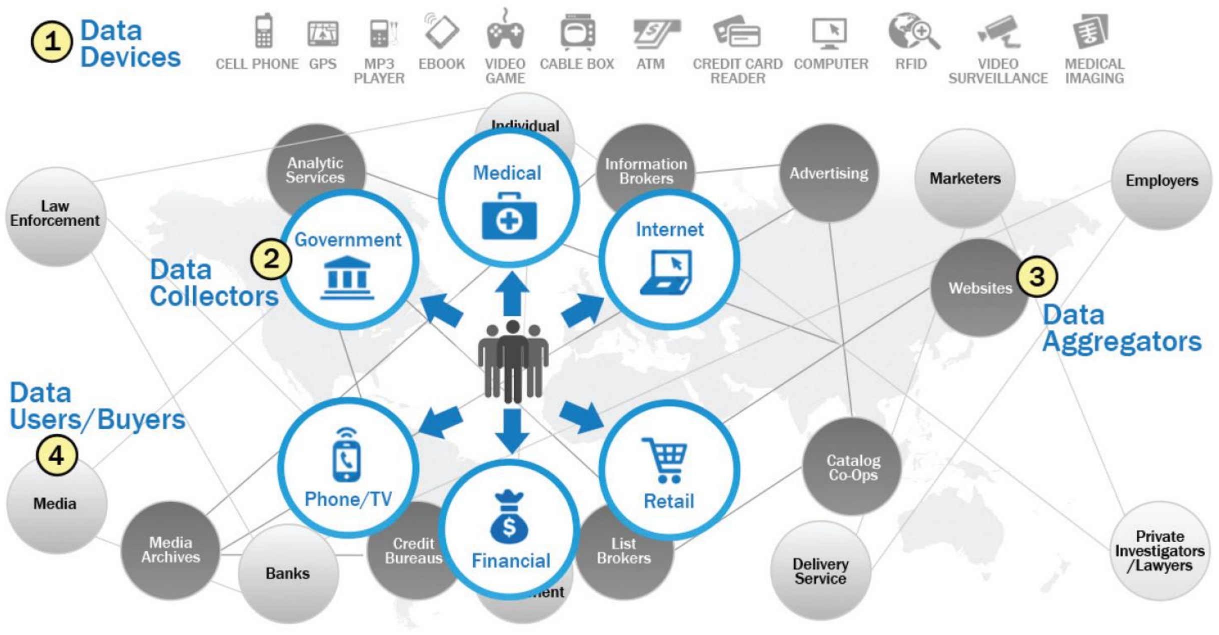


Figure 7: Emerging Big Data ecosystem [1]

2.9.1 Data Devices

As shown above, this is the layer which represents the sensor devices that are continuously generating and producing new data. The new data is associated with metadata which describes it. [1] Argues that approximately each gigabyte of new raw data comes with an additional petabyte of data created to describe them. Although these generators of data have been already described in detail in Chapter 2.2, there will be a few examples mentioned too. [1]

- Playing games – while playing an online video game on a PC, game console or smart phone, the video game provider can store data about the skills and levels reached by the player. If the provider has got an intelligent system that can monitor how the user plays the game and its duration, he can suggest equipment or other game artefacts for the character based on the user’s age, gender and interest which can be desired by the user. Additionally, the system can propose other games that the user can be interested in and consequently start playing them.
- Smart phones – another rich source of data which has been already mentioned before. The subject of interest is data about the Internet usage, SMS usage and real-time location. This data can be used for analysis of

smart phones in locations to track the speed of cars or to warn about busy roads nearby. Another application can be realised by GPS devices in cars which can provide real-time information about traffic delays or routes which the driver should take to avoid such delays.

- Retail shopping – it is commonly known that retailers' loyalty cards record purchased products, their amounts, money spent, the location of stores visited and also the kinds of products purchased within one transaction. Collecting this data provides desirable information which the retailer can base the advertisement on.

2.9.2 Data Collectors

The blue circles identified as (2) within Figure 7 describe sample entities which gather data from users and devices. [1]

- Government – collects personal data of its citizens, tax information, police records, car registrations, etc. If the data is well integrated, it is possible to prevent citizens from threats of terrorist attacks, crime commitments and frauds by finding abnormal patterns in behaviour of suspected individuals and making other predictions based on the analysis of the collected data.
- Retail Stores – tracking the path a customer takes through their store during pushing shopping basket with an RFID chip to be able to find out which products get more attention. Consequently, it helps to make a better strategy for the location of products and develop better publicity.
- Cable TV provider – which tracks the shows that a person watches. Through tracking of the channels, the provider can find out whether people would pay for watching particular channels and the amount of money that would be spent.

2.9.3 Data Aggregators

The dark grey circles in Figure 7 characterise entities which process collected data from the first layer or from the collectors described above and make them understandable. Particularly, they give them additional value to prepare them for the handing over process. Now the data is ready to be offered on the market. Typically, one of these data aggregators can transform and package the data as products to sell to list brokers that might want to generate marketing lists of people who may be good targets for specific ad campaigns. [1]

2.9.4 Data Users and Buyers

These entities represent a group of the final layer from the Big Data ecosystem. This group has the final benefits from the collected and aggregated data offered by the data aggregators. [1]

- Retail Banks – considered as data buyers might want to use aggregated data to determine which of their customers would like to apply for a second mortgage with the highest probability. The data can include demographic information about people living in specific conditions, people who appear to have a specific level of debt or if they pay bills on time and have a savings account.
- Data users may want to track or prepare for natural disasters by identifying which areas a hurricane will affect first hand. It can be observed by tracking tweets about it or discussing it in social media.

2.10 Roles for the New Big Data Ecosystem

The emergence of the new big data ecosystem comes with the need for an occurrence of new specialists who will take care of the new challenges of acquisition, cleaning, preparation, storing and analysis of Big Data. Additionally, the need for applying more advanced analytical techniques drives the emergence of new roles, analytical methods, and technology platforms.

There are three new roles which have emerged in the new Big Data ecosystem argued by [1] and depicted in Figure 8:

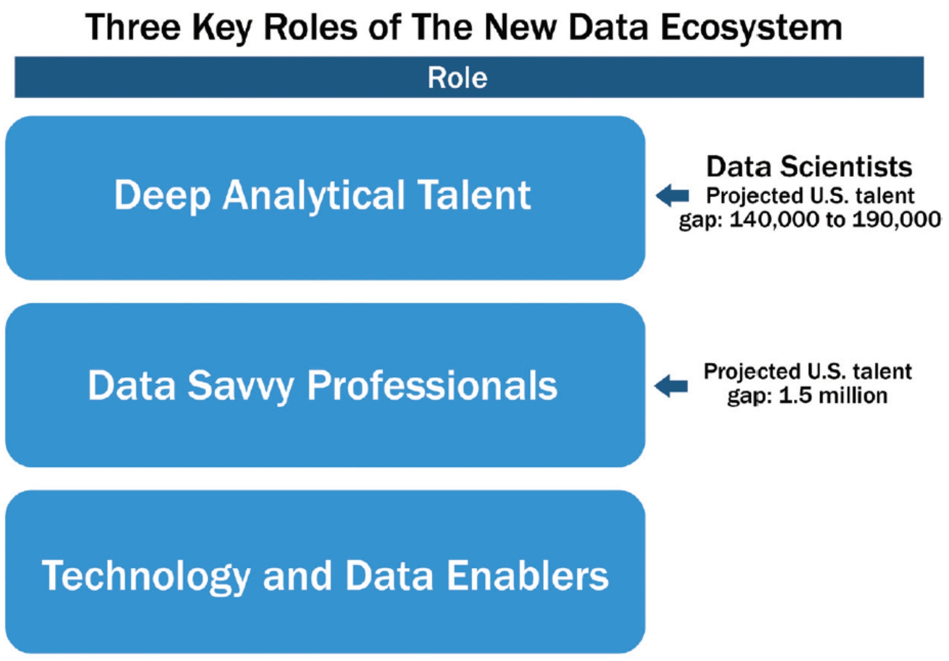

Three Key Roles of The New Data Ecosystem
Role
Deep Analytical Talent
Data Scientists
Projected U.S. talent gap: 140,000 to 190,000
Data Savvy Professionals
Projected U.S. talent gap: 1.5 million
Technology and Data Enablers

Figure 8: Roles of the new Big Data ecosystem [1]

- Deep Analytical Talent – it is described as a group with strong analytical skills. Members possess a craft to handle unstructured, raw data and to apply complex analytical techniques in a scalable environment such as clouds and clusters. Adequate knowledge in mathematics, statistics and machine learning is necessary. Additionally, the members of this group need to have access to an analytic sandbox or a workspace where they can apply their knowledge to explore Big Data.
- Data Savvy Professionals – this category represents people who have less technical knowledge but still possess the basics of statistics and machine learning. They specialise in business, finance, markets and management rather than deep technical knowledge. In contrary, people of this group usually tend to have basic knowledge of working with data to be able to cooperate with the other groups. Consequently, they should be able to define key questions that can be answered by using advanced analytics.
- Technology and Data Enablers – has people providing technical skills to support analytical projects such as deployment, administration of analytical sandboxes, workspaces and technologies which are related to this domain including large-scale data architectures deployed on clusters and clouds, enabling the definition of the company/organisation environment and determining data handling processes.

Cooperation among these three categories is necessary to solve sufficiently Big Data challenges. Although companies are commonly familiar with the last two mentioned groups, the first group Deep Analytical Talent is usually the newest role for them. Most people know this role as Data Scientist.

2.11 Data Scientist

A data scientist should be able to:

- Consider business problems in the context of analysis – it is a skill to diagnose a business problem, finding the core of the problem and determine which kinds of available analytical methods can be applied.
- Handle statistical models and data mining techniques of Big Data – this part consists of three phases:
    - Design,
    - Implementation, and
    - Deployment.

  Dealing with this mainly involves applying complex analytical methods to a variety of business problems which use data.
- Possess critical thinking with domain insights – it is important to think in the whole context of the company/institution interests. Consider having a company which processes data to improve its market position among rival companies. It is critical to use proper analytical methods which have values for the company and lead to processes improvements.

There are depicted desirable skills and behavioural characteristics in Figure 9 mentioned by [1]:

- Technical skills: software engineering, programming skills, machine learning,
- Quantitative suitcase: statistics and mathematics,
- Curious and creative: good attitudes in creative problem solving,
- Communicative and collaborative: abilities to articulate the business value, to be clear and collaboratively work with others who participate within the

department involving other parties such as project sponsors and stakeholders,

- Being sceptical – is bound up with critical thinking. One should be able to evaluate things from more than just one perspective.

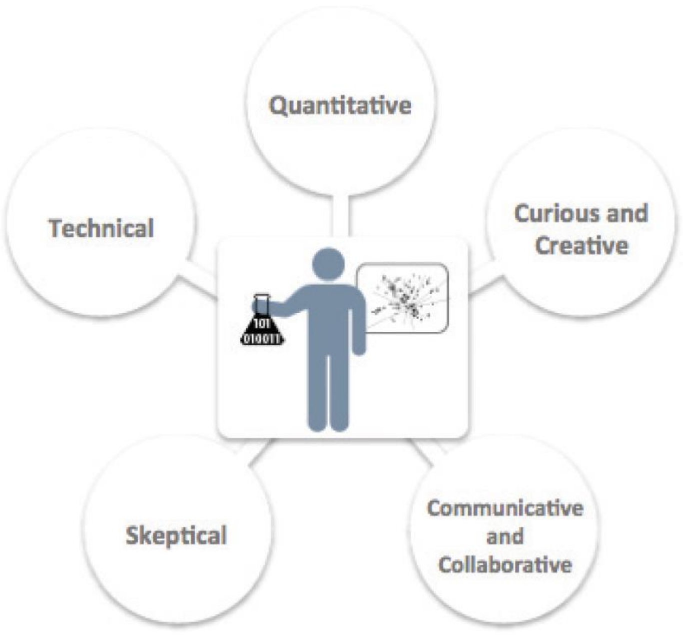


Figure 9: Big Data scientist desirable abilities [1]

2.12 Big Data Life Cycle

To get values by processing Big Data, it is necessary to apply an engineering approach that will assure appropriate incremental steps to increase the probability of the project success. Particularly, dealing with challenges of Big Data problems such as data heterogeneity, complexity, etc., there are a few typical phases which are addressed to the decomposition of the Big Data problem. It has been published by many authors that a typical cause of a Big Data problem projects' failures is excessive focusing on the analysis phase. Due to this fact, it is important to not underestimate other phases, especially the preparation phase, which is crucial for the next following phases where the output of the preparation phase is their input. For example, one cannot assume to derive particular value from data that is imperfectly, incompletely and with a little ambiguity captured. To deal with it there is a need to handle uncertainty and errors.

There are some tips and trends which are responsible for the evolution of the following phases. Newly occurring approach of processing, storing and analysing data in one complex place is mentioned by [8]. The same data are supposed to pass through a few phases. To provide an integrated data management and analytical capabilities for all the stages is to create an analytical sandbox. The concept of the analytical sandbox will be discussed further in this chapter. Another good practice mentioned is to consider a functionality, which can influence operations of cleaning and filtering data during process of extracting, transforming and loading data into an analytical sandbox. This approach seems to be an asset which comes with unprocessed, unfiltered and not transformed raw data that is useful for some types of analysis.

There are six main phases which capture and handle best practises, approaches and techniques which should be taken into account (see Figure 10). These best practises are mentioned by [8] and [1]:

1. Discovery – the team members learn the business domain. Available sources such as people, technology, time, and data are discussed. It especially involves finding data sources, data acquisition and the first assessment of its quality and suitability. A business problem is defined and its aims and hypotheses are formulated.
2. Data Preparation – this phase requires one central place called an analytical sandbox that the team can work with integrated data and perform analysis during the project time. Once the sandbox is ready, data is extracted, transformed, cleaned and loaded according to the project needs. Additionally, the team is becoming fully familiar with the data to be able to design analytical models.
3. Model Planning – there are techniques, methods and workflows determined which models will consist of. The choosing of the techniques and methods has to be related to the data characteristic and defined goals of the project.
4. Model Building – the models based on the previous phase are developed with suitable datasets for the execution. The team also considers if there is a need to use more powerful tools for run-time such as distributed environments or parallel computing that are necessary for dealing with Big Data.

5. Communicate Results – there is evaluated whether the project succeeded or failed. The evaluation is made in collaboration with project stakeholders. The decision process is based on the hypotheses and the project goals defined in the first phase of the discovery.
6. Operationalise – the team delivers final reports, code and technical documents. In addition, the team can develop a pilot project to deploy it in a production environment.

The process of moving from the actual stage to the next one is slightly different from standard software engineering approaches. Particular stages of solving the Big Data problem are bound up with some uncertainty and ambiguity as mentioned by [1]: *“Unlike many traditional stage-gate processes, in which the team can advance only when specific criteria are met, the Data Analytics Lifecycle is intended to accommodate more ambiguity.”*

It means that there has to be some checkpoints defined which confirm whether the team is ready to move to the subsequent stage.

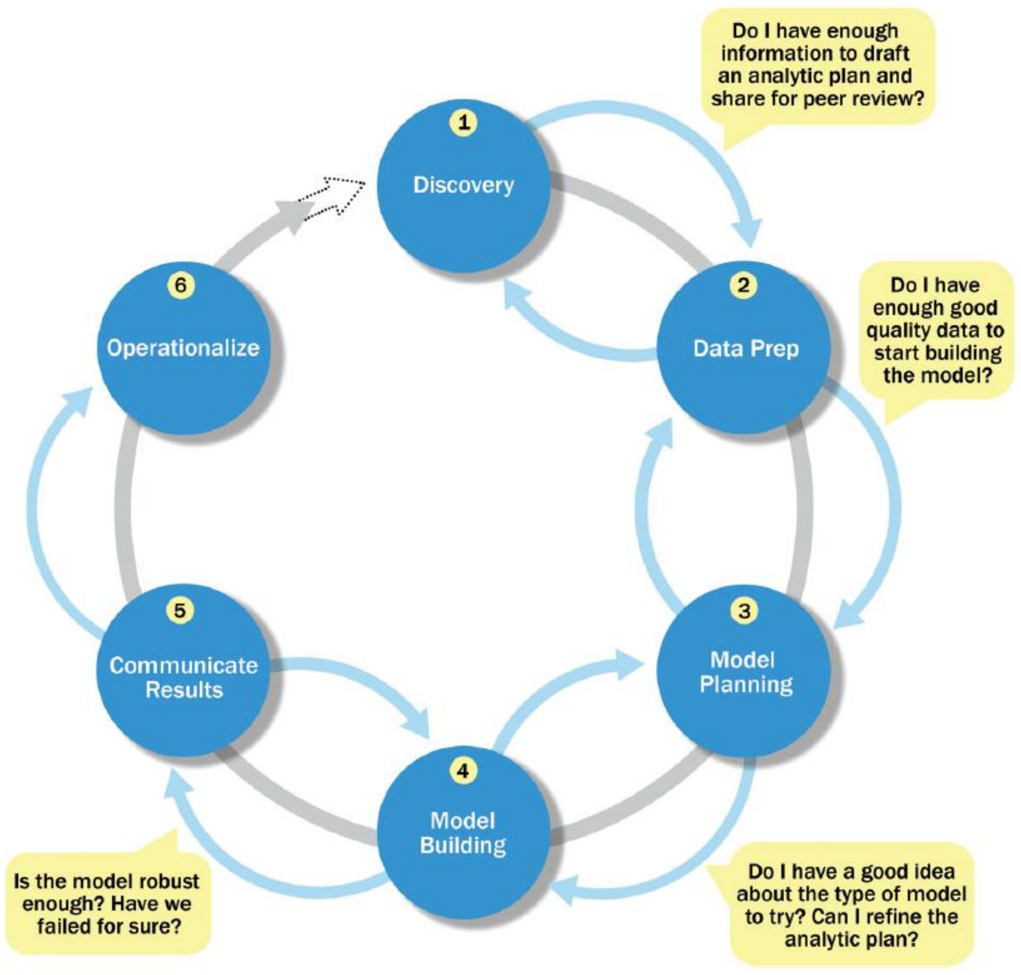

Do I have enough information to draft an analytic plan and share for peer review?
1
Discovery
2
Data Prep
Do I have enough good quality data to start building the model?
3
Model Planning
Do I have a good idea about the type of model to try? Can I refine the analytic plan?
4
Model Building
Is the model robust enough? Have we failed for sure?
5
Communicate Results
6
Operationalize

Figure 10: Big Data Analytics Lifecycle [1]

2.12.1 Discovery

2.12.1.1 Business Domain

In this part, the business domain context is identified. Although software engineers and data scientists have deep knowledge of the fields where they are specialised in, it is also important to understand the context of the area where the project will be performed.

2.12.1.2 Resources

Certain sources are needed which are evaluated to be available to support the project considering technology, tools systems, data and people. Particularly, in finding available technologies suitable for the needs of the following phases, related technologies for data acquisition, storing, processing, making models, tools for analysis, etc. Additionally, project teams are defined and various specialists are needed such as data scientists, marketing professionals, and technical supporters. The result should be a balanced team of various domain experts.

- Problem Framing – it is another important part of the discovery phase. Framing is considered to be the process of identifying the analytical problem to be solved. The analytical problem has to correspond to the business aim to meet the stakeholders' interests.
- Initial Hypotheses – this subphase forms ideas of basic analytical tests that the team will use in later phases as foundations for the analytical findings. Moreover, these hypotheses also influence characteristic data which acquisition will be based on. These hypotheses are crucial in terms of determining the analysis scope.
- Identifying Potential Data Sources – identifying the key data to solve the problem. Consideration of volume, types, sources and quality is involved in this part. This consideration should be related to the main characteristic of Big Data such as volume, variety and velocity mentioned in Chapter 2.5. It is crucial to choose the right data, which is the most suitable for the project goal and defined hypotheses. This decision will influence significantly the further phases.
    - Identification of Data Sources – making a list of candidate data sources which can meet the scope of initial hypothesis, freely available or commercial data sources has to be categorised.
    - Capture of Aggregated Data Sources – to get a better understanding it is desirable to see a data preview.
    - Raw Data Review – this is for finding relationships, dependencies among raw data for deep understanding and validation of its limitations and quality.
    - Evaluation of Tools and Data Tools that are needed – data structures and formats reveal that tools are suitable for the data analysis. Due to this, the team can start collecting appropriate tools.
    - Infrastructure – it is related to the chosen tools, data characteristics and the defined problem. Appropriate infrastructure should be taken into account involving storage, network capacity, and hardware performance.

2.12.2 Data Preparation

Data preparation is discussed to be the most intensive and iterative phase in a Big Data project lifecycle consuming the biggest piece of the project time. According to publications, it is common to spend at least 50% of the project by performing this part. The importance comes with the data quality. If the project team does not provide sufficient data quality, there might be a problem to perform subsequent phases. Sometimes teams tend to skip this phase, but most of the time they have to return back to this phase. All in all, data preparation is essential for the project success.

Data preparation consists of a few steps such as exploration, pre-processing due to modelling and analysis in further phases. An environment that is used for data manipulation has to be created and separated from a production environment. Typically, it is realised by creating an analytical sandbox. In addition, there is an occurring need to move data into the sandbox. Usually it is realised by performing a process called ETL (Extract Transform and Load). This process is mostly mentioned and related to Data Warehouses. It extracts, transforms, and loads data into the sandbox. When the data is in the sandbox, firstly the team needs to get familiar with it. To be able to do this, there are some techniques and tools which are usually helpful such as performing data visualisation which provides an overview over the data to uncover relationships among data variables and trends. The next following activity is to make a decision how to transform the data into a suitable format for the further data analysis.

2.12.2.1Analytical Sandbox

An analytical sandbox (sometimes mentioned as a workspace) is used as a tool for data storage, manipulation and exploration separated from a production environment due to getting flexibility and freelance that is caused by this separation. The policy of accessing a production environment is tightly controlled due to its importance and need for financial reporting.

The best practise is to collect all data within this tool, regardless of its format, structure and volume. It is developed to store aggregated, structured and unstructured data from various sources mentioned in Chapter 2.2. It should allow a virtualisation of data sources. An analytical sandbox is primarily assigned for data analysts who require access to all data throughout the organisation. Due to this, a tight cooperation with the company IT is of crucial importance since there is a need to negotiate about data availability of the analytical sandbox purposes. This technology allows us to move beyond traditional restricted Data Warehouses (DW) and Business Intelligence (BI) and contains raw and more ambitious data science projects.

The concept of an analytical sandbox is relatively new. Companies are working on the creation of a universal analytical sandbox where users can store, access data and manipulate with it. However, this solution is not available as an enterprise solution yet.

2.12.2.2 Performing ETL

ETL consists of three operations: extract from a datastore, transform data and load it into a datastore. To avoid the possibility of losing raw data, due to its significant value, the process of ETL is slightly different in the case of an analytical sandbox. Primarily, the data is extracted in its raw form and loaded into the datastore, where a data analyst can influence the transformation by taking a decision on whether to keep data unchanged or transform it into a new form. It allows the performance of raw data analysis. For example, if the team wants to perform analysis of fraud detection with the usage of ETL to get data into an analytical sandbox, the data would be inadvertently cleansed by the transfer operation. Additionally, this process might corrupt the analysis results because the traces of the fraud activity would disappear in the transformation process.

2.12.2.3 Data Pre-processing

It is a critical step within the lifecycle due to the processes of cleaning data, normalising datasets and performing transformation on the data. For instance, this step can involve joining and merging datasets to get the data into a form that enables further analysis. Actually, it is a subphase that can be named as the pre-processing stage of the data analysis. Additionally, the project teams are deciding which data is essential to keep or discard.

2.12.2.4 Data Overview

The next step after data pre-processing is a data overview to see and evaluate data patterns that enable deeper understanding in quick and efficient ways. An example can be data visualisation in order to examine data quality such as whether the data contains unexpected values or to monitor its consistency. In addition, it helps to deeply explore data by watching trends and suspicious activities or even interesting accumulations in the data.

2.12.3 Model Planning

There are candidate models identified in this phase such as clustering, classifying or finding relationships in the data depending on the aim of the project. A decision of choosing proper methods and framing the analysis concept for the further execution to achieve formulated targets should be related to the definition of the initial hypothesis defined in the first phase of the Big Data Lifecycle. The following points should be considered:

- Dataset Structure Evaluation – the structure of the datasets is in a tight relation with analytical methods and techniques. The analytical methods are dependent on these dataset structures. Moreover, the available set of these methods is restricted by them.
- Meeting Hypothesis and Business Context – to assess if the techniques can provide results for early defined targets such as meeting formulated hypothesis and business goals of the project.
- Single Technique or an Analytic Workflow – the team should consider if the model must consist of one method or a set of methods that will be a part of an analytical workflow.

2.12.3.1 Model Selection

An appropriate set of candidate analytical techniques should be discussed. A model is only an abstraction of reality due to a need of expressing a single technique or a complex set of techniques linked together. In addition, there are sets of machine learning and data mining techniques. These sets provide numerous methods that are related to dealing with different problems and approaches. Methods and techniques overview is mentioned in Chapter 2.13. Their suitability is also dependent on the type of data that the methods will be used on. Different approaches are involved when we are supposed to deal with structured or unstructured data.

Typically, teams design models by using mathematical and statistical packages such as Matlab, R and SAS (Statistical Analysis System). These tools provide computational environments with many additional libraries including machine learning and statistic libraries that enable to design complex models. Additionally, when the team deals with Big Data, it is important to ensure a suitable approach is chosen. Basic environments of these tools are usually suitable for not very large data sets due to their limitations. When we are dealing with Big Data, it is necessary to have a platform where created models can be handled in distributed environments with suitable and supportable technologies to deal with large volume or a need of real-time processing.

When the team is ready and has a plan on how to build a model, then it can move to the part of model building.

2.12.4 Model Building

In relation to data mining and machine learning, there is a need to adapt the environment for the models execution. It includes a development of datasets for training, testing and production purposes.

Analytical models are developed to be suitable for the training data and evaluated against the test data.

In this phase the data models defined in the previous phase are executed. To assess the model validity, the model is run on a small predefined testing set of data. During these activities, various configurations can be set up to improve or change the model behaviour by modifying variables or inputs.

The team has to be completely sure that the model being developed meets the business requirements defined in the first phase. The following questions should be asked:

- Does the model appear valid and accurate related to the launch on the dataset?
- Do the configuration parameters make sense in the domain context?
- Are there any tolerable or intolerable errors occurring?
- Can the model be deployed on the run time environment? Is there sufficient support to do it?

- Do we have enough data or we need more inputs? Are there any transformations or adjustments of the data to be performed?

When the model is evaluated, whether it is sufficient or not, the team can move to the following phase.

2.12.5 Communicate Results

After the executions, the team needs to discuss the results whether it succeeded or failed by proving or disproving the hypotheses. Additionally, the most successful models are chosen. The most important findings are delivered and shared with the stakeholders of the project. The stakeholders should assess how the project affects their processes. For example, the marketing department has to know how the project results should be used to reach and handle the project assets in a complex way. Best practices and experience should be recorded as recommendations for future projects.

2.12.6 Operationalise

In the last phase, the team sets up a pilot project to deploy the work in a controlled way before running it in a production environment of the company enterprise ecosystem. Due to the functionally of the analytical sandbox, the team is able to develop other models whose pilots will be realised and afterwards fully deployed. This approach enables the team to learn about the performance and limitations of the production environment. It is recommended to not deploy the solution throughout all the company departments at once. A better approach is to deploy it in incremental phases to avoid any complications.

2.13 Data Mining

Data mining is a process of analysing data from different perspectives. Particularly, it is a process of discovering patterns, trends and relationships from data using machine learning and processes. The patterns have to be meaningful to lead to some advantages. It is usually associated with business needs to identify trends and analyse the business environment to acquire knowledge as a business asset [9]. There are two classical Data Mining tasks:

- Predictive Tasks – the object of these tasks is to predict the future of data values by using historical data trends.

- Descriptive Tasks – their aim is to derive patterns such as correlations, trends, clusters and anomalies. Typically, they are used for assigning values into predefined groups defined by particular attributes.

Although Data Mining is in a tight relation with Big Data processing, most of the traditional Data Mining tools and libraries cannot deal with Big Data due to their limitations. Nowadays, some libraries and techniques are available which were developed for the purpose of dealing with Big Data. In comparison with the classical Data Mining tools, these are usually paralysed and deployed in distributed frameworks with high scalability. It is safe to note that the traditional Data Mining techniques were used as a pattern for the development of more robust versions handling Big Data problems. As an example there can be Hadoop's ecosystem which contains distributed machine learning tools.

2.13.1.1Machine Learning

Machine learning is an algorithmic technique which enables to learn from empirical data by using iterative algorithms. Particularly, it is used to forecast future behaviours, outcomes, trends and to find hidden insights in data. A list of widely used and the most famous techniques with a short overview are mentioned below:

Classification – it is a method of assigning objects to one of the predefined categories. It is usually used for detecting spam email messages based on the message header and content. It is widely used for prediction purposes. For instance, classification can help medical staff to diagnose heart disease patients.

Association Rule Mining – It is a descriptive method which is often used to discover interesting relationships hidden in a large set. It is usually used in marketing to identify customers who have similar interests to target their publicity more effectively. Another example can be a physical or logical placement of a product within related products' categories.

Clustering – it is an exploratory technique to discover hidden structures of the data and a task of processing a large data set of data objects which seem to have no relationship. Clustering methods find the similarities between objects according to the object attributes and assign the similar objects into clusters. For example, the clustering technique can be used in marketing and sales to better identify customers who have similar behaviours and spending patterns.

Regression – it is an explanatory task that can identify the input variables having majority statistical influence on the outcome. For example, a restaurant can predict the type and quantity of food which customers will consume based on the day of week, the weather, etc.

2.14 Big Data Privacy

With Big Data assets comes also the privacy concern. Some fields where data is collected in have stricter laws than others. For electronic health records, there are usually big restrictions when the data can be revealed. Other domains are not typically too restricted. Data restrictions are less regulated in the United States of America (USA) than they are in the European Union (EU). Apart from considering data privacy governance and policies of particular countries, there are usual principles which should be kept by organisations throughout the EU and the USA. [8]

Seven Global Privacy Principles [10]:

1. Notice (Transparency): *"Inform individuals about the purposes for which information is collected."*
2. Choice: *"Offer individuals the opportunity to choose (or opt out) whether and how personal information they provide is used or disclosed."*
3. Consent: *"Only disclose personal data information to third parties consistent with the principles of notice and choice."*
4. Security: *"Take responsible measures to protect personal information from loss, misuse, and unauthorized access, disclosure, alteration, and destruction."*
5. Data Integrity: *"Assure the reliability of personal information for its intended use and reasonable precautions and ensure information is accurate, complete, and current."*
6. Access: *"Provide individuals with access to personal information data about them."*
7. Accountability: *"A firm must be accountable for following the principles and must include mechanisms for assuring compliance."*

For better understanding, it is appropriate to mention data types which are usually under the control of privacy governance of particular countries and evaluated as the ones which could be abused and consequently violate the privacy.

These are types of usually protected information in the global context mentioned by [10]:

- Personally Identifiable Information (PII) – any information that directly or indirectly identifies a person.
- Sensitive Information – any information whose unauthorised disclosure could be embarrassing or detrimental to the individual.
- Other Information – any other non-identifiable information about an individual when combined with PII.

Examples of the particular categories are shown in Table 2.

Personally Identifiable Information	**Sensitive Information**	**Other Information**
Name	Race/ethnicity	Preferences
Postal address	Political opinions	Cookie ID
Email address	Religious/philosophical beliefs	Static IP address
Telephone/mobile number	Trade union membership	
Social Security Number	Health/medical information	
Driver’s license number	Marital status/sexual life	
Bank/financial account number	Age	
Credit or debit card number	Gender	
ZIP Code	Criminal record	

Table 2: Types of Protected Information [9]

Although there are these typical regulations applied, another concern rises. This concern is caused by a possibility of joining data together. For example, there is data collected from geospatial sensors generated by location-based services which require a user to share his location with the service provider. The service provider stores only some information about its user without recording his location or information about the place where he lives. Potential malicious software can infer the customer’s location from queries sent to the service provider server to determine the user’s identity.

Health issues can be another example. If there is a person with cancer, there are some patterns that can be tracked such as presence in a church, visiting a pharmacy, or presence in a cancer treatment centre. Consequently, it can reveal the user's private information.

Most of social media such as Facebook, Twitter and others often require us to share private information, but users do not know how the information is processed and how the data can be linked.

All in all, private information can be revealed by observing patterns of users' behaviour over time.

On the other hand, governance and data privacy policies are aspects that a data owner has to deal with. Organisations are concerned on how to process this data while retaining its asset. They have to also deal with questions such as how to share or sell the data without losing control.

If there are committed actions which are against the law and which violate restrictions, companies can be highly charged as it has happened to Google in 2012:

"Google Inc. has agreed to pay a record $22.5 million civil penalty to settle Federal Trade Commission (FTC) charges that it misrepresented to users of Apple Inc.'s Safari Internet browser that it would not place tracking "cookies" or serve targeted ads to those users, violating an earlier privacy settlement between the company and the FTC." [10]

One of the activities of the Federal Trade Commission (an authority in the USA) is to ensure if companies keep the privacy promises they make to customers. Google was charged with the largest penalty in the history of this agency. Google was also required to disable all the tracking cookies.

3 Big Data Tools

Although Big Data principles and approaches are frequently discussed, there are not many technologies which are convenient to deal with such data. Due to the definitions of the volume and the velocity, the tools which are supposed to deal with Big Data have to offer a distributed computing approach. There are the following approaches: multiple data and single program, and single data and multiple program.

In the first case, there is a single program, which is run on more nodes, where all nodes process different data. On the contrary, the second case is considered to have only one dataset, which is processed by a program divided on small tasks that are run on different nodes in parallel.

Due to it, there are tools that try to abstract from the physical distribution as much as possible. Since the Apache company released its new implementation of Map Reduce paradigm, a whole ecosystem called Hadoop has started evolving. The MapReduce paradigm offers the means to break a large task into smaller tasks, run in parallel, and consolidate the outputs of the individual tasks into the final output. The significant ecosystem expansion was caused by using simple programming models to process large datasets across clusters as well as was amplified by the fact that the whole solution has started as open source software.

Hadoop as the first publicly known and discussed technology of Big Data processing has been used as the base of open source and commercial extensions. In other words, most of the set of Big Data tools are based on the Hadoop solution.

These solutions offer methods and approaches to load, pre-process, store, query and analyse data.

In the following subchapters the Hadoop ecosystem will be described along with other technologies which have been evolved from it or others which are using its technologies.

3.1 Hadoop

3.1.1 MapReduce

As mentioned earlier, the MapReduce paradigm provides the means to break a large task into smaller tasks, run the tasks in parallel and consolidate the outputs of the individual tasks into the final output. MapReduce consists of two basic parts: a map step and a reduce step. [1]

- Map – performs an operation to a piece of data which generates some intermediate output.
- Reduce – gathers the intermediate outputs from the map steps, processes it and provides the collected final output.

The main advantage of MapReduce is the workload distribution over a cluster of computers (to run tasks in parallel). Particularly, MapReduce provides a technique, which allows the processing of one portion of the input which can be run independently of the other input parts. In other words, the workload can be easily distributed over the cluster.

3.1.2 Distributed File System – HDFS

The Hadoop Distributed File System (HDFS) is a file system which provides the capability to distribute data across a cluster to take advantage of the parallel processing of MapReduce.

HDFS is designed to run on common low-cost hardware. Consequently, it means there is no need to deploy it only on super computers. Although, it is implemented in Java, HDFS can be deployed on a wide range of machines apart from a node, which is dedicated to manage namespace services (see Architecture below).

3.1.2.1 Architecture

HDFS has a master/slave architecture as shown in Figure 11. It consists of a single master server which manages the filesystem namespace and manages access to files by clients and a single NameNode. In addition, there are DataNodes which are usually bound up with a node in the cluster. These DataNodes manage storage within their nodes that they run on. A file in HDFS is split into one or more blocks that are stored by a set of DataNodes. Moreover, they are responsible for serving read and write requests from the file system clients and performing blocks' creations, deletions and replications which are requested by NameNode. Whenever there is a request for an operation as opening, closing, renaming of a file or a folder, it is handled by the NameNode. Additionally, it is also in charge of the blocks mapping to DataNodes. [11]

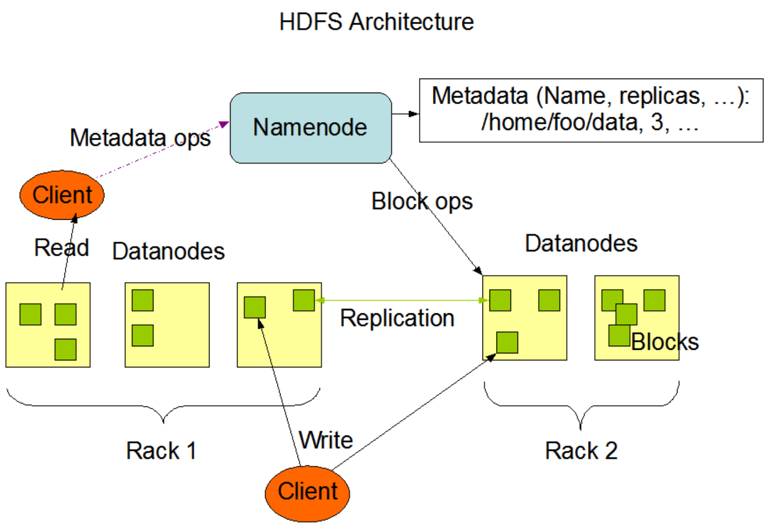


Figure 11: HDFS Architecture [11]

3.1.2.2 File Storage

HDFS breaks files typically into 64 MB blocks and stores the blocks throughout the cluster. Whenever possible, HDFS attempts to store file blocks on different machines to allow the map step to operate on each block of a file in parallel. If a file size is 200 MB, the file is stored in four blocks: three 64 MB blocks and one 8 MB block. If the file is smaller than 64 MB, the block represents the whole file. HDFS is determined to deal with large files such as files of up to 1 GB. Files with a size smaller than 64 MB are considered as small sizes which are not efficiently maintained by HDFS.

In order to enable high throughput data access, applications using HDFS have to keep write-once-read access model for files. It means, when a file is created, written and closed it should not be changed anymore.

Related to the file system namespace, HDFS still supports a traditional hierarchical file organisation where a user can create directories and also store files within them, and also performs other file system related operations.

3.1.2.3 Robustness

To avoid data failure, HDFS creates three copies of each block across the cluster to provide the necessary redundancy by default settings. If one of the machine fails, HDFS replicates an accessible copy of the lost data blocks to another available machine to get the system back to the state before the failure. To keep the data integrity, there are special files within the namespace, which contain check sums of all the file blocks. Whenever a block is fetched, moved or copied within HDFS, it is matched to the checksum to determine whether the block is corrupted or not. Additionally, HDFS is aware of the rack position, which means that the blocks are distributed across several racks (a hardware type of servers) to avoid a data failure which can be caused by stoppage of one rack. Moreover, the three replicas allow Hadoop to determine which machine to use for the map step on a particular block. Broadly, the data does not move to nodes, but the closest nodes related to the blocks location are chosen.

3.1.2.4 Accessibility

Hadoop provides applications' interfaces as well as user interfaces to be accessible from the outer environment.

3.1.2.5 Applications

There is more than one way how HDFS can be accessed. Native access for applications is through a Java API (Application Programming Interface). Additionally, there is a Java API wrapper for a programming language C and there is a possibility to access HDFS via HTTP (Hyper Text Transfer Protocol Secure) browser.

3.1.2.6 Users

- FS Shell – it is a command line interface which allows a user to keep data organised.
- DFSAdmin – it is a command set for administering an HDFS cluster.

Web interface – allows a user to view the contents of the HDFS and navigate through it via a web browser.

3.1.3 MapReduce Job in Hadoop

A typical MapReduce program in Java is composed of three classes: the driver, mapper and reducer.

- The driver – contains the job details and its configurations such as input file locations, details for assigning the input file to the map task, the names of the mapper and reducer Java classes and it also contains the location of the reducer task output.
- The mapper – represents the logic to be processed on each data block related to the defined input files in the driver code.
- The reducer – represents the logic of the gathering of intermediate results from the mappers.

To better understand the Hadoop MapReduce implementation we will demonstrate it via using an example of word counting mentioned by [1]:

Each map task is run on the node where a data block resides. All the map tasks process a fragment of the text, line by line, parses a line into words and provides a tag <word, 1> for each word according to their occurrence in the line. The key/value pairs are stored in the worker node's memory. When a worker finishes a line, the tags are passed to the reducer (the intermediate data). From the various map task outputs, for each unique key, lists or arrays are constructed. When the whole task is finished the output file is stored in HDFS.

3.1.4 The Hadoop Ecosystem

As mentioned above, Hadoop evolution comes all along with open source and commercial extensions to make Apache Hadoop easier to use and provide additional functionality and features. This subchapter examines the following Hadoop-related Apache projects which all together form the Hadoop Ecosystem.

- Pig – provides a high-level data-flow programming language,
- Hive – enables data querying similar to SQL (Structured Query Language),
- HBase – provides approaches for real-time operations of data reading and writing,
- Mahout – provides analytical tools.

Some of these technologies such as Pig and Hive abstract from MapReduce operations. It means, that a user does not have to know anything about MapReduce because Pig and Hive enable a developer to write high-level code that is later translated into one or more MapReduce programs. There is a need to mention that Pig and Hive are intended for batch processing as well as MapReduce programs.

Since the data is gathered, queried and pre-processed by Pig and Hive, it can be analysed by the tools provided by Mahout or Spark within a Hadoop environment.

If there is a problem which cannot be conducted by batch processing, HBase provides the ability to conduct real-time reads and writes of data stored in a Hadoop environment.

3.1.4.1 Pig

Pig provides high-level programming technique to conduct distributed operation over a Hadoop environment without having knowledge how MapReduce works. By using Pig, a user can save considerable time which he could have spent by writing MapReduce tasks. When a user runs a Pig script, Pig commands are automatically transformed on MapReduce operations which are running in the background.

Pig provides functions for several common data manipulations such as inner and outer joins between two or more files (tables), as would be expected in a typical relation database. Pig also provides a GROUP BY operation that is similar to the traditional functionality implemented in SQL which can be used together with other useful functions and operations related to aggregation, load/store, math, string and date-time categories.

3.1.4.2 Hive

Similar to Pig, Hive abstracts from MapReduce operations by providing high-level programming approach. The main difference between Pig and Hive is that the Hive language, HiveQL (Hive Query Language) is determined to query data (similar to SQL) rather than to be considered as a scripting language.

A hive table structure consists of rows and columns. The rows are typically related to a record, transaction or a particular entity. The corresponding columns represent the various attributes or characteristics for each row. Hive is typically used to apply some structure to unstructed data.

Although Hive's performance may be better than SQL database in certain applications, Hive is not intended for real-time querying. A hive query is first translated into a MapReduce job, that is afterwards run on the Hadoop cluster. For the real-time querying it is better to consider to use HBase as described below.

There are a few recommendations of when it is appropriate to consider using Hive mentioned by [1]:

- *"Data easily fits into a table structure."*
- *"Data is already in HDFS (Note: Non-HDFS files can be loaded into a Hive table)."*
- *"There is a desire to partition datasets based on time (For example, daily updates are added to the Hive table)."*
- *"Batch processing is acceptable."*

3.1.4.3 HBase

Unlike Pig and Hive, which are intended for batch applications, Apache HBase is capable of providing real-time read and write access to datasets with billions of rows and millions of columns. HBase is a technology which provides a distributed data store across a cluster. As the other Hadoop's projects, HBase is built upon HDFS to distribute the workload over nodes in the cluster. An HBase table is composed of rows, columns and the third dimension intended to maintain the different values of a row and column intersection over time. Hbase uses a key/value structure to store the contents of an HBase table. Each value is the data to be stored at the intersection of the row, column, and version.

As an example, HBase is used by Facebook as an approach to build a search indexes for user inboxes to ensure the order of the received messages.

3.1.4.4 Mahout

When a dataset is available in HDFS, the next step might be to conduct an analysis over the stored data. Tools such as R are useful for analysing relatively small databases due to the fact that they are not built upon a distributed environment. Mahout can be considered as a suitable option to apply analytical techniques within a Hadoop environment. This Hadoop ecosystem technology provides executable Java libraries which can be applied over a Hadoop cluster. The analysis methods are implemented by the usage of MapReduce operations. It means that this approach provides the needed scalability which is necessary in the manner of Big Data analysis.

Mahout provides Java code that implements the algorithms for several techniques in the categories such as classification, clustering and filtering.

3.1.5 Application

The following subchapters provide some examples of the real Hadoop ecosystem deployment within companies such as LinkedIn, Yahoo and IBM Watson.

3.1.5.1 IBM Watson

This is a computer system which was specifically developed to answer questions on the quiz show Jeopardy! in which it competed against two former winners. Contestants are provided with a few words which are associated with a particular object, situation, person, or a phrase. The aim of the game is to provide the sufficient answer. Finally, Watson defeated these two contestants. In this case, Hadoop was used to process various data sources such as news, dictionaries, electronic literature and the contents of Wikipedia. [1]

3.1.5.2 LinkedIn

LinkedIn is an online professional network gathering millions of people over many countries which provides services related to job posting, talent searches, participation in discussion, etc. LinkedIn uses Hadoop for: processing transaction logs, performing users' activities such as views and clicks, for developing and testing analytical models, etc. [1]

3.1.5.3 Yahoo

Yahoo as an internet browser is discussed as the largest publicly announced Hadoop deployments at 42,000 nodes over several clusters where is processed approximately 350 petabytes of raw data. In this case, Hadoop is associated with [1]:

- The maintenance and creation of the Yahoo's search index,
- Web content optimisation,
- Spam filters,
- Ad-hoc analysis and analytic model development.

As [1] argues: *"Prior to deploying Hadoop, it took 26 days to process three years' worth of log data. With Hadoop, the processing time was reduced to 20 minutes."*

3.1.6 A Big Amount of Small Files

As mentioned in HDFS description, files in Hadoop are stored in blocks whose default size is 64 MB. Due to the fact, that the HDFS architecture was designed to maintain large files within these blocks, it is not efficient in storing small files. If somebody wants to store a huge amount of small files, a various set of problems occur.

3.1.6.1 Problems with Small Files in HDFS

A small file is one which is significantly smaller than the HDFS block size. Each file, directory and a block is represented as an object in the namenode's memory where each of them occupies 150 bytes. It means that one million files would occupy approximately 3 GB. If rapidly more files are considered such a billion of them, this problem becomes crucial and feasible.

Due to the fact, that HDFS was not created to deal with small files, all operations associated with them are not efficient under the weight of their administration. In general, the storage of small files in Hadoop is inefficient.

3.1.6.2 Problems with Small Files and MapReduce

The first problem occurs when a lot of MapReduce processes are performing reading from discs. As it is generally known, access to a disc is one of the most expensive I/O (Input/Output) operation. When MapReduce operations read many small files, the reading itself does not consume a lot of time, but the effectivity which is bound up with the administration is considerably.

The second reason for performance degradation is associated with the MapReduce itself. When a MapReduce job launches, it schedules one map task per block of data being processed. If there are 10,000 files each containing 10 MB of data, a MapReduce job will schedule 10,000 map tasks. Typically, Hadoop is configured to run a map task within its own JVM (Java Virtual Machine).

3.1.6.3 Dealing with Small Files

Although there are a few techniques and recommendations on how to deal with small files, it does not make Hadoop a universal solution for all problems along with all possible data kinds in complex formats. Additionally, Hadoop is mainly determined for processing a huge data amount stored in a simple key/value structure. [12]

- Federated NameNodes – they allow to have multiple NameNodes to scale metadata. This technique isolates object metadata which in other words means that only one NameNode knows about any particular object. If data is shared among all cluster application, the solution is useless.
- Change the ingestion process – the easiest way how to get rid of small files is to change the source system to produce large files instead of a huge amount of small files. It is the best approach how to deal with small data but it cannot by applied every time due to the data or business restrictions.
- Batch file consolidation – propose to run a simple consolidating MapReduce job to read all of the small files on a folder and rewrite them into fewer larger files. If a file is important for processing, contains dependencies or it describes where its data is stored, this technique is not suitable. The process of consolidation is appropriate for files which contain raw data without any associations, dependencies such as csv files.
- Sequence files – in this case, the original filename is stored as the key in the sequence file and the file content is stored as values. For example, if 500 small files fit into one 64 MB block, it allows MapReduce jobs to launch

only one map task over 500 small files. This approach is convenient just in the case of having a huge amount of files.

3.1.7 Summary

The Hadoop ecosystem offers a scalable solution for loading, storing and pre-processing a large amount of data. All the tools and projects within the ecosystem are dependent on HDFS and MapReduce technologies. In other words, if we are considering to use Hadoop as a solution for data processing, it is necessary to take into account, that our data along with our intentions has to be compatible with the Hadoop architecture. As discussed in HDFS part, Hadoop is suitable for processing datasets which are composed of large files (gigabytes-exabytes). Hadoop is not suitable to process small files; the performance is considerably decreasing due to the administration related to dealing with them. Although there are some additional tools or Hadoop approaches which enable processing small files, they come along with many restrictions and they try to avoid the intention which Hadoop was made with.

3.2 Spark

It is an open source Big Data processing framework for performing data analytics on a distributed computing cluster such as Hadoop. Spark supports in-memory processing to increase speed and data process over MapReduce. It is discussed as a more powerful analysis replacement of Hadoop. Spark is deployed on the top of existing Hadoop cluster which enables Spark to access data via HDFS. Additionally, it can also process structured data via Hive and stream data from HDFS. [13]

As an alternative of the traditional bash MapReduce model, Spark comes with a new computation model which is supposed to be more efficient than Hadoop in areas such as real-time streams and iterative algorithms which are common in graph applications and machine learning. Additionally, it is also considered as an efficient environment for interactive data mining tools.

In comparison to Hadoop, Spark uses more RAM (Random Access Memory) instead of network and disk I/O. Due to the higher memory usage, Spark typically demands a high end physical machine for producing effective results. Hadoop uses replication to achieve fault tolerance whereas Spark uses a different data storage model called Resilient Distributed Datasets (RDD). To avoid failure, RDD saves additional data which is used for a lost block reconstruction. [14]

Since Hadoop 2.0 was released, Spark has become more popular because it can run on the top of HDFS along with other Hadoop components. Nowadays, it is considered as another part of the Hadoop ecosystem.

Spark provides four additional components which are separated from its core:

- Spark SQL – it provides access to the Spark datasets over JDBC (Java Database Connectivity) API and allows running SQL queries on Spark data. It also allows users to extract, transform and load (ETL) data from different formats.
- Spark Streaming – is a component for processing the real-time streaming data.
- MLib – is a distributed machine learning framework consisting of common learning algorithms and utilities, including classification, regression, clustering, etc.
- GraphX – is a distributed graph-processing framework which provides API for graphs and graph-parallel computation.

To be opened and accessible for application developers, Spark provides APIs for various programming languages such as: Scala, Java and Python.

3.3 Matlab

Matlab is a numerical computing environment and also a proprietary programming language developed by Mathworks. It is determined to leverage technical and scientific computations, analysis and to develop algorithms. Due to its popularity among many world scientific communities, there are many available sources such as e-books, forums, blogs, social media groups, etc. Additionally, Matlab as proprietary software also provides user support. Matlab involves the following five activities:

- matrix manipulations,
- plotting of functions and data,
- implementation of algorithms,

- creation of user interfaces,
- interfaces with programs written in other languages.

Although Matlab is mainly specialised in the domain of mathematical-scientific computing, it is also familiar with object-oriented programming techniques. Matlab framework is also considered as a tool for handling Big Data. [15]

3.3.1 Matlab Toolboxes

Matlab provides a toolbox mechanism, which is an efficient way on how to extend its own standard functionality. Particularly, a toolbox is just a collection of a source code stored in Matlab files. Additionally, this collection has to be defined in Matlab path.

Toolboxes can be distributed as open-source software or as proprietary modules. The proprietary toolboxes are also provided by Mathworks. There are many proprietary toolboxes determined for various areas such as:

- Data analysis and optimisation,
- Simulation tools,
- Automatisation of source code generation,
- Application and documentation development,
- Signal processing,
- Parallel computing,
- Finance and economy.

3.3.2 Big data

Tools and approaches considered with Big Data [15]:

- 64bit computing - due to large computations of Big Data, it is necessary to work on 64bit version of Matlab, deployed on the same bit architecture system to ensure not being dramatically limited by addressing just 2 GB in the case of using the 32bit version.
- Memory mapped variables - allow mapping of a file or its part to a Matlab variable in memory. They make it possible to access big data sets on discs which are big to hold them in memory or it takes them long to load.

- Disc Variables - enable access to Matlab variables directly from a special file stored on a disc without loading the full variables into memory. This is intended for avoiding memory overflow.
- Datastore - the datastore function can be used during accessing data which does not fit into memory. It allows an incremental import of data.
- Parallel Computing Toolbox – the possibility of adding plugins enables to connect toolboxes to handle these problems such as Parallel Computing Toolbox which enables parallel computing and even a connection to a Matlab Distributed Computing Server which can be additionally connected to Hadoop or to Amazon’s Elastic Computing Cloud (EC2) which lets you process big data without buying a cluster. The toolbox provides working with distributed arrays such as matrices and multi-dimensional arrays which are distributed across the memory of a cluster of computers. This approach is also useful when data does not fit into a single computer memory.
- MapReduce – a functionality used in Matlab to analyse data that does not fit into memory. A programming technique which can be used on a desktop, as well as on Hadoop. MapReduce in combination with the dataset function enables development on a local machine and executes the application on Hadoop without changes.
- Streaming algorithms – a possibility of stream processing on incoming data streams which cannot be fully held in memory.

3.3.3 Machine Learning

This technique includes a Statistics and Machine Learning Toolbox and also a Neural Network Toolbox. These toolboxes contain a set of variety machine learning algorithms that can extract insights from data and can make predictive models.

3.3.4 Interfaces

Due to available interfaces, it is possible to get, create, use and share add-ons to get more functionality integrated to Matlab. In the set of addable add-ons, applications, toolboxes, support packages and more are included. Matlab can call functions written in C or Fortran. Additionally, it can be connected to databases such as MySQL, Microsoft SQL Server and others via the Database Toolbox.

3.3.5 License

Matlab as proprietary software provides various types of licences. The licence type depends on the character of the end customer as well as on the options which can be particularly set up. There are three licence categories according to the groups where the licence is applied on:

- Individuals – it is determined for usage only by one person. The licence is bound up with the individual’s name.
- Academic – licences determined for academic purposes are considerably cheaper.
- Enterprise – there are licences determined for a commercial usage within a company.

There are two other categories considered with the count of individuals which the software will be available for:

- Individual – it is a licence for one person. The software can be installed and deployed on one or up to four computers.
- Network – this licence enables to launch Matlab software on a particular number of computers within a group at once regardless of the software installations deployed on the physical computers. Whenever the maximum number of the actual running software is reached, the other users which are trying to launch the software have to wait until one of the licences is released. Matworks provides an individual approach of the licence price determination. Depending on the licences count and their combinations, the company can offer some discounts.

3.4 NoSQL Databases

NoSQL (Not Only SQL) databases are databases which were designed to handle bigger data amounts more efficiently than traditional databases. They are more flexible with regards to the integration of new data, than data models, which are the basis of relational databases. NoSQL databases were also developed in order to make database queries more efficient and faster in the manner of getting their shorter responses. Additionally, these databases are specific by their scalability, which allows distributing the load which cannot be handled by a set of the current machines, easily to other new added machines. Due to the possibility of the data deployment over more machines, NoSQL databases allow to handle data such as leveraging queries in parallel over the participating machines. Although they are more flexible due to their structure, this capability causes a lack of some structural control mechanisms such as table integrity including and transactions reliability in comparison to relation databases. [16]

NoSQL databases are grouped into four main categories:

- Key/value – similar to a distribute hasmap where the data are represented by a key/value pair. The value can be a serialized object or a simple string. There is a limited set of the operations which can be run over these values: put, get, delete.
- Column-oriented – this type is similar to tables in relation databases, where the number of the columns is dynamic and it can be even different among all database records.
- Oriented document – the document oriented databases are also based on the key/value pairs. The values represent XML or JSON (JavaScript Object Notation) document type.
- Graph database – the model of the data representation is based on graph theory where a graph consists of nodes, relationship and attributes associated with them. Typically, these databases are used as tools for modelling the real world relationships in social networks.

The most famous and widely discussed databases are: Redis (key/value), HBase (column-oriented), MongoDB (oriented document), InfiniteGraph (graph database).

3.5 Evaluation

Although the Big Data theory is widely and frequently discussed, there are not many tools and practical examples on how to process Big Data. Almost all tools determined to deal with Big Data have been evolved from the free available Apache Hadoop Ecosystem which provides an effective solution on how to process a subset of Big Data defined by specific characteristics. Particularly, Hadoop is efficient to handle datasets which are composed of a small part of large files rather than countless small ones. In order to distribute data over a Hadoop cluster, HDFS divides data files on / in blocks. If the data characteristic does not allow it to be decomposed through the use of these blocks, it is inappropriate and inefficient to use Hadoop.

Matlab provides a different approach on how to handle Big data. There are numerous techniques which enable the prevention of problems of overflowed memory caused by loading a large amount of data at once. The most convenient Matlab Big Data technique is represented by Parallel Computing Toolbox along with Distributed Computing Server Toolbox. Particularly, these toolboxes allow Matlab scripts to run over a Matlab cluster. Data is stored in a shared filesystem instead of a distributed one. Due to this, the above mentioned problems caused by dividing files on blocks are eliminated. Consequently, the Matlab approach can be suitable to handle Big Data problems which can be hardly managed by Hadoop. Although Matlab is considered as a useful tool to deal with Big Data, it is necessary to keep in mind that Matlab is proprietary software requiring a valid licence.

To address Big Data problems which are associated with databases, a new technology called NoSQL has evolved. NoSQL databases enable a scalable and elastic solution on how to store and query a huge data amount faster and more efficiently than relation databases. On the other hand, NoSQL databases are not so strict about data consistency requirements.

All in all, it is necessary to consider the suitability of these technologies to deal with particular problems associated with specific datasets. Chapter 6 is dedicated to select the most suitable technology related to the dataset which is chosen as the most applicable one for the purposes of this study in further chapters.

4 Available Databases

The aim of this chapter is to explore and describe other available free databases of biomedical data with characteristics corresponding to Big Data and create an overview, which will enable us to make the final decision as to which database is the most suitable for further experiments. Moreover, there is a database of the EEG (Electroencephalography) / ERP (Event Related Potentials) project[1], which is deployed and used at the University of West Bohemia in Czech Republic. This database seems to have the potential of being a Big Data database. Finally, one of these databases will be chosen for further experiments.

The following suitable criteria aspects are considered:

- the rigorous data description (a lot of publications, documentations, and metadata) and its context,
- potential usefulness related to further experiments which should have a value and a positive impact on the EEG/ERP project,
- suitability of the data domain which requires biological data, and
- the potential meeting of at least one of the Big Data characteristics.

Although there are many databases accessible via the Internet which can be considered as being the Big Data, most of them are not associated to the biomedical area. Typically, there are government databases containing a huge amount of data, some data is available via Facebook, Twitter and LinkedIn. One of the Big Data resources are accessible via Amazon Web Services which provide public datasets. Servers containing digitalised documents such as books, journals, songs and music (all digital media) can be considered as another source.

Although there is a lack of potential Big Data databases of biomedical data, there are still some available datasets considered as significant sources of biomedical data, even though some of them meet the characteristics of Big Data. These databases, repositories or datasets are mentioned below.

[1] ERP/EEG project: https://eegdatabase.kiv.zcu.cz/

4.1 Genome Databases

A database project called Nucleic Acids Research (NAR) starts providing data collected by three major bioinformatics centres, the U.S.A National Cancer Centre for Biotechnology Information (NCBI), the European Bioinformatics Institute (EMBL-EBI) and Swiss Institute for Bioinformatics (SIB). Typically, this data is dedicated to research related to the genetic basis of disease (mainly cancer), drugs and their side effects.

Although the data is typically available along with metadata and publications, there is some heterogeneity between metadata and publications quality of particular projects that the whole database is comprised of. Nowadays the whole NAR project consists of 1685 databases such as Nucleotide Sequence Databases, RNA Sequence Database, Protein Sequence Databases, Structure Databases, Genomic Databases, Metabolic and Signalling Pathways Databases, Cell Biology Databases, Immunological Databases, etc. [17]

Data which is stored and available by NCBI via ftp[2] protocol has 614 TB in total including approximately 480,000 files within 42,060 folders.

4.1.1 Genome 1000 project

Human DNA is comprised of approximately 3 billion base pairs with a personal genome and around 100 GB is needed to store it.

Since the first human genome has been sequenced, and the prices of technologies have been reduced, some projects associated with genome sequencing have been launched. Although all humans share 99% of their DNA (Deoxyribonucleic Acid), the relatively few differences among us include diseases, personality, and other traits. The primary project purpose has been to identify the largest number of rare genetic variants throughout all ethnicities. Another goal of this project was to sequence at least 2000 human genomes. Nowadays, there are approximately 2500 genome sequences which are persisted and open to public without any charges to support their research. [18]

[2] Genome databases: ftp://ftp.ncbi.nlm.nih.gov/

The total size of this project folder which is available via ftp protocol is 559 TB.

4.1.2 GenBank

It is a public database of all known nucleotide and protein sequences, which is being built and distributed by NCBI. The primary sources of data are collected by authors and a few organisations (laboratories). NCBI makes the data available at no cost throughout the Internet with of ftp usage and web service access.

The data of GenBank is stored in the filesystem of NCBI in a directory called genomes or directories: genbank, refseq and all. This is data submitted directly to GenBank or INSDC (International Nucleotide Sequence Database Collaboration) database. The total size of GenBank storage publicly available is approximately 1.5 TB.

4.2 Machine Learning Repositories

Although machine learning repositories are not considered as containing Big Data, they are still useful data repositories which can be used for various research activities. These repositories are specific by rigorous descriptions of their datasets typically involving their characteristics such as:

- the dataset overview, its context and history including the purpose and the project aims,
- the data types and formats,
- the tasks which are usually associated with them,
- the number of instances and attributes,
- the quality status (if there are any values missing), and
- the list of projects where the dataset was used.

Although these repositories are generally considered to provide datasets through various interest areas, they still contain a significant set of biomedical datasets.

4.2.1 UC Irvine Machine Learning Repository

This machine learning repository[3] currently contains 350 datasets which are primarily dedicated to the machine learning community. It was created as an ftp archive in 1987 and nowadays it is maintained by the University of Massachusetts. Since the project foundation, the repository's datasets have been cited over 1000 times and they have got significant popularity among scientists' communities, institutions and universities.

There is a user friendly searchable interface which shows available datasets, their timelines, popularity and other characteristics. For example, there are datasets such as Skin Segmentation Dataset, Parkinson Dataset (voice measurements), E. Coli Genes Dataset, EEG Database Dataset, SPECT (Single Photon Emission Computed Tomography) Heart Data Set, Breast Cancer Dataset, etc. Three of these datasets are described in more detailed below:

- Diabetes 130-US hospitals for years 1999-2008 – a dataset recommended for the tasks of classification and clustering submitted on behalf of the Centre for Clinical and Translation Research in Virginia. The dataset represents 10 years of clinical care at 130 US hospitals and including over 50 patients and their hospital outcomes. The dataset includes 100,000 instances.
- EEG Database Dataset – this dataset is associated with a study of EEG correlations of genetic predisposition to alcoholism which provides 122 instances.

4.2.2 Challenges in Machine Learning

As discussed on the repository's websites[4], the project's aim is to provide support for machine learning projects and scientific research. It has started by funding of the EU Pascal network to establish a group which will provide an environment to gather research and students together throughout Europe to achieve better collaboration among specialist in Machine Learning, Statistics and Optimisation. The repository is actively updated and maintained. Additionally, it provides a huge

[3]UC Irvine Machine Learning Repository: http://archive.ics.uci.edu/ml/index.html
[4] Challenges in Machine Learning: http://www.chalearn.org/

amount of pre-programmed models which are either free or against a fee. As in the previous machine learning repository, most datasets come along with their descriptions, publications and other metadata containing information about the projects where the dataset were cited.

The repository provides dozens of datasets throughout various application domains, available in different formats and determined to be processed by different methods. The project emphasises to use real data, it means that the initiative of the project is to enable users to take advantages of the data which can establish a suitable and robust foundations to achieve valuable results. As an example of biomedical data, there are a few datasets such as Genomics Dataset, Abscising Acid Signalling Network Dataset, E.coli Gene Expression Dataset, REGED Dataset, Embryology Dataset, etc. Two datasets are described below:

- Embryology Dataset – a dataset called Zebra provides 154 feature representation of cells of zebrafish to determine whether they are in division or not.
- Genomics Dataset – a dataset called REGED whose goal is to find genes which could be the cause of lung cancer. It consists of almost 1,000 features, 5,000 training examples and different test sets of 20,000 examples.

4.3 National Health Care Services

During this decade, governments are urged to provide data maintained by national facilities to both their inhabitants and to the private sector. In order to satisfy the society requirements, governments are launching projects which can make the collected data open to public. Additionally, the goal of making data available comes along with the effort of improving the whole data network, exchange, sharing, storage and the communication among the facilities.

The opened data of these projects is widely discussed as Big Data sources characterised by its big volume and variety. However, these databases are not proper databases of biomedical data, they rather gather data related to daily activities of the health care services to provide customers information about their quality, availability and their costs to enable better transparency and control among these facilities.

4.3.1 HealthData.gov

It is a web portal[5] dedicated to publish health data and make it more accessible among groups of medical staff to deliver better care, researchers, companies to develop better products and services and even for making data accessible to USA inhabitants which provides them with a better overview over the quality and availability of the healthcare services and all this is free of charge. Additionally, in order to access this data, it is not required to be a member of these groups; everybody from all around the world can access this data. To get this data, the HealthData.gov provides web APIs which enables data to be machine accessible, searchable, readable and downloadable. The whole project is a result of the new health care reform law of the USA signed by President Obama in March 2010. Over 1000 datasets have been published since the project was launched in 2011.

People can access a number of separate federal agency databases through the portal. CDC Wonder and CMS database and Chronic Condition Data Warehouse are discussed below [19]:

- CDC Wonder – this database enables users to access a wide variety of public health information including data about deaths, births, cancer, HIV and AIDS, tuberculosis, etc. [19]
- Chronic Condition Data Warehouse – the data warehouse established by CMS to allow users to purchase some sets of data. [19]

Typically, these datasets contain data related to aggregated records of medical activities in hospitals and healthcare facilities such as given prescriptions, cost of particular healthcare procedures, statistics, overviews of variations in health service performance throughout the healthcare institutions, etc. Due to its size and various data, the datasets meet the volume and the variety of Big Data characteristics. In comparison to the genome databases, the datasets provided by this institution are not typically raw biomedical data determined for further

[5] HealthData.gov: http://www.healthdata.gov/

research. On the other hand, the data in different formats is easily accessible via web APIs by computers. In conclusion, various datasets associated with the public healthcare throughout the USA are available.

4.3.2 National Health Service

Similarly, to the USA case where the government affords to provide electronic information about healthcare on a national level, during April 2013, the English National Health Care[6] has launched a series of initiatives to begin opening its healthcare data intending to positively impact outcomes (primarily the healthcare quality, availability and transparency). The primary objectives of the project are to address information gaps held by the NHS, improve data flow throughout the healthcare facilities, provide publications of comparative provider outcomes and launch a service called Friends and Family Test to measure patients' satisfactions based on the number of patients who would recommend hospital services to friends and family. [20]

The data which is stored by this institution is divided into two categories [20]:

- Restricted Health Data - where the restricted data potentially carries persons or proprietary information which is anonymised through the removal of personally identifying information. To access this kind of data, it is necessary to submit a request which is later evaluated by the institution. Typically, this data is determined to research groups and charities.
- Open Health Data – data which is currently published under the Open Government Program and consists of aggregated data and healthcare quality data along with the availability of information and the general overviews of such healthcare.

In conclusion, the open data project run by NHS is considerably similar to the HealthData.gov.org project. It provides the amount of data which can be considered as Big Data due its volume and the variety but the data is typically

[6] NHS Health and Social Care Information Centre: https://www.england.nhs.uk/

consisting of small files containing aggregated, informational data of descriptions of quality and healthcare availability throughout the UK.

4.4 Local EEG/ERP Database at the University of West Bohemia

There is a project related to EEG research at the Department of Computer Science and Engineering within the University of West Bohemia which has been running for almost six years to leverage experiments within the neuro-informatics domain. The project research group possesses specialised measurement devices to record EEG signals among various experiments.

When an experiment is completed, its metadata such as the experiment description, the information about the person whose reactions were measured and recorded, the specification of the recording devices and the experiment scenario and others is stored along with the recorded raw binary data via EEG/ERP portal[7] into its databases.

The portal ensures a rigorous approach of uploading the experiments into the project databases insisting on a base set of required metadata of experiments to provide a sufficient level of its quality. Typically, the raw data in its binary form is uploaded into the project's relation database and the most of the metadata is uploaded to a NoSQL database. In general, the relation database is considered as the storage of all files which are not in the textual format, the NoSQL database is used for the rest of it. Since the experiment is uploaded, the project group can start making various analysis in order to carry out new research. Thus, the project is leveraged in the academic domain, there are plenty of people who participate on this project such as undergraduates, postgraduates, PhD students and scientists whose achieved results are well documented and some of them are published among scientific communities and even transformed into journals.

Due to the participation of various researchers and students who are contributing to the project expansion, the velocity of the production of new experiments has been increasing during the last years and the group of researchers want to adapt

[7] EEG/ERP portal: https://eegdatabase.kiv.zcu.cz/

to this trend by exploring the technologies which can prepare the project for the high expected data growth that is among the research discussed as a potential Big Data problem. The data volume which has been gathered since its start is supposed to have dozens of gigabytes. Additionally, there are new opportunities which are associated with a few subprojects which should influent and emphasise the growing trend as well. Particularly, the subprojects are considered to open the portal to public groups (acquisitioning, sharing, providing data) and to buy additional devices such as a portable-wireless device for recording EEG signals, the heartbeat, a temperature measurement and blood pressure.

4.5 Finding a Suitable Database

In conclusion, there are three types of databases that are considered, evaluated via defined criteria and depicted in Table 3.

Criteria	Genome Database	Machine Learning Repositories	National Healthcare Services	EEG/ERP database
Potential of meeting Big Data characteristics	YES	NO	YES	YES
Rigorous data description	NO	YES	NO	YES
Usefulness for the EEG/ERP project	NO	NO	NO	YES
Biomedical data	YES	YES	NO	YES

Table 3: The Comparison of the Data Sources Depending on The Defined Criteria

The genome database is truly a Big Data one which consists of hundreds of terabytes. Additionally, it is easily accessible via the ftp (File Transfer Protocol) and the whole project has a huge financial support. Although it has the considerable volume, the weakness of the database is the various quality of its projects/databases, their documentations and the data description along with the complexity of the genome research. The high number of databases and the complex structure where the databases are stored, make the movement through the storage unclear. Moreover, some of the databases are overlapping, which means that a set of them consists of other databases which makes all the database storage confusing.

The machine learning repositories are specific by the rigorous descriptions of their datasets with specification of what their potential application can be considered. Another quality aspect of these datasets is associated with the high number of their usage involving their citations and related publications which forms relations among similar projects/research. Contrary, the datasets which the repositories are made of, cannot be considered as Big Data due to their small volume.

Although the open healthcare databases seem to be really interesting for further processing, they do not usually contain suitable biomedical data due to the fact that the main aim of their foundation is to inform about the state, availability and reporting quality issues of the healthcare facilities throughout particular countries. On the other hand, the amount of collected data can be considered as the Big Data.

The most suitable database which was evaluated for this study is the EEG/ERP project. Although, this database might not have the volume and the aspects of Big Data these days, there are projects which can considerably support the hypothesis that it can grow to the volume of Big Data. Depending on further analysis, the Big Data approaches might be applied in this context. As a desirable fact of why this database should be considered as the most suitable one, is the rigorous description of the data and many available publications among the research group. Additionally, the application of a Big Data approach can lead to an improvement of the current state of the project and its success in the future.

5 EEG/ERP domain

5.1 Data Description

The data which the EEG/ERP portal deals with and stores it into the databases is divided into two groups:

Binary data – the binary data consists of the data generated by the EEG signal recorder and the data associated with the experiments such as videos, pictures, scenario files, archives, graphs, processed binary data and almost everything else that needs to be stored.

Textual data – textual data as metadata associated to their experiments. It consists of the experiment description, the related scenario description, the information about the environment where the experiment was leveraged within, information about participants, characteristics of the used hardware, etc.

5.1.1 Brain Vision Format

Experiment data is generated by a recording device of a company called Brain Products that provides solutions for neurophysiological research. The device itself is called BrainVision Recorder used for controlling EEG amplifiers and for recording EEG signals and sensor data with the aid of electrodes, sensors and a personal computer.

The derived data is stored on the PC as digitised raw data. This data can be further analysed at a subsequent time and by different software products.

The recorded data and information about the measurement generated by the recorder is stored within three binary files which the BrainVision Data Exchange Format is composed of: the header file, the marker file and the actual data.

- The header file – it describes the EEG. This file is an ASCII (American Standard Code for Information Interchange Computing) file with the extension “vhdr”. It consists of sections with different names containing keyword/value pairs which describe the measurement. These values typically contain the names of the marker and the raw data file, the data format and its particular alignment, number of participated electrodes and particular settings.

- The marker file – it is a file which contains markers defining particular points in the raw data file which the raw files consist of. The name is usually the same one as the one of the raw data file but with the extension “vmrk”.
- The raw data file – a file composed of the recorded binary values of the EEG signal with the extension “eeg”.

5.1.2 Databases

All data that are supposed to be persisted is divided into two databases. The decision where the data will be stored depends on the data characteristics related to the beginning of this chapter. The data is composed of two types: binary and textual.

The data that belongs into the binary group is uploaded into a relation database PostgreSQL. Contrary, most of the textual data is stored within a NoSQL database called Elasticsearch.

The main decision why the data is divided into two groups has two reasons:

- The data needs to be stored in a database which has appropriate technology of binary data storage,
- The variety of experiments needs a flexible structure for storing metadata.

Both databases are deployed in the production environment as the data layer of the EEG/ERP portal. The current occupied size is 43 GB in the case of the relation database and approximately 8 MB is kept by the NoSQL database.

5.1.2.1.1 PostgreSQL

PostgreSQL[8] is open source object-relational database system traditionally used on GNU/Linux system but it also supports a lot of other platforms such as Windows, Mac OS, Solaris, etc.

Related to the EEG/ERP project, the database schema of the production database consists of entities where the most of them are used as the portal data layer to persist data associated with experiments, users, used devices, etc. These

[8] PostgreSQL: https://www.postgresql.org/

entities do not occupy as much database space as the binary files which are stored within the database too.

PostgreSQL provides three ways of the binary data storage: BLOB (Binary Large Object), bytea and text. For the purposes of the binary data storage of the EEG/ERP experiments is used the first type BLOB. The maximum size of one file which can be stored in this RDBMS is 4 TB.

When a data file is supposed to be stored as the BLOB type, the file is divided into small files which indexes are stored in a special PostgreSQL entity called pg_large_object. For the purpose of handling BLOB files, PostgreSQL provides the API to deal with them. Due to it, there is some administration data which are additionally stored in the database.

5.1.2.1.2 Elasticsearch

Elasticsearch[9] is a NoSQL database which has been developing in JAVA programing language since the 2010 year. It is a document oriented database which is accessible via its web services. Additionally, this project is considerate to perform text analysis and full text searching too. [21]

Related to the EEG/ERP project, the main advantage of this database for the project demands is the database schema flexibility which can be easily adjusted during all the project lifecycle. Additionally, it is considered as a scalable solution due to the fact, that Elasticsearch is one of the group of NoSQL databases which possess this specific characteristic.

5.1.3 Data Analysis

In order to define whether the EEG database has got potential to become Big Data, there is a need to make analysis in a rigorous way to determine the growth trends and to predict some forecasts which can help to determine if the project has some characteristics of Big Data or eventually that there is high possibility to reach them in a short time.

[9] Elasticsearch: https://www.elastic.co/

Due to the really small size of the Elastic database, there is no need to analyse it because the data that is important for the prediction and the overview is stored within the relation database.

5.1.3.1 Information

The database was created and deployed at the department as a part of the EEG/ERP project in 2010. Since that time, new experiments have been uploaded there in random intervals which probably should not be longer than six months.

As mentioned before, the database contains production data and the logic of the portal's data layer stored in entities where some of them are in relations with each other. The whole database schema does not require a lot of free space as it will be mentioned in the following subchapters.

To be able to get the information about the data growth, it is necessary to have the trace of the particular date in which the experiments were leveraged or uploaded into the database. This segment of information is recorded in the entity called experiment and it represents the time when an experiment was performed. In combination with other entities the database was queried to obtain the defined results. These results are represented by overviews, statistics, trends and forecasts in the following subchapters.

To get more objective results, the interval of the data values aggregation was set up in the duration of six months.

Note: apart from the overall size of the database, the results are based on the database operations performed over the BLOB files.

5.1.3.2 General Overview

The rate between the size of the administration and the real stored data is shown in Figure 12.

Figure 13 depicts the rate between the amount of the experiments' data size (the BrainVision data) and the size of the data representing the other group such as videos, processed raw data, graphs, documents, etc.

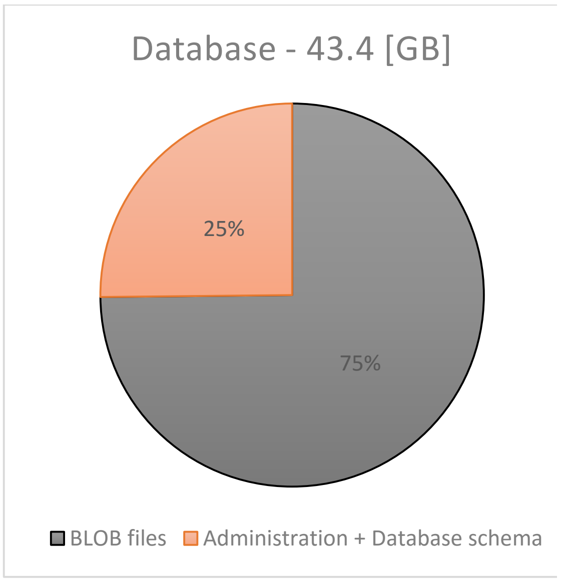


Figure 12: Database Overview

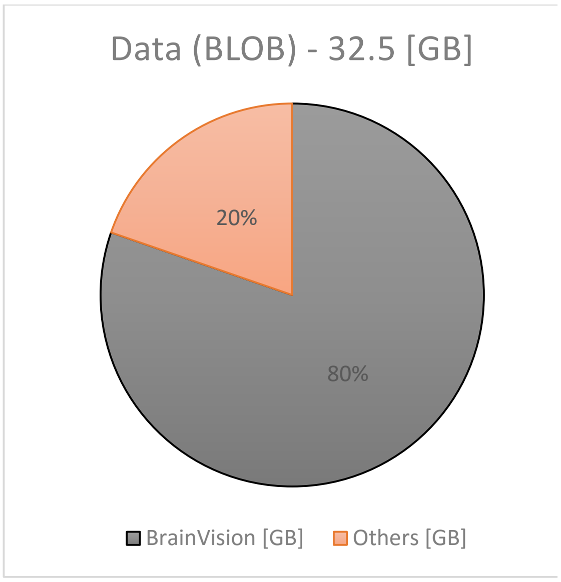


Figure 13: Data Overview

The particular values are depicted in Table 4 and Table 5 below.

Items	SIZE [GB]
BLOB files	32,5
Administration + database schema	10,9
All database	43,4

Table 4: Database Overview

Items	SIZE [GB]
BrainVision	26,1
Others	6,4
All data	32,5

Table 5: Data Overview

5.1.3.3 Statistics

To better understand the data it is convenient to mention some statistics which were created by data aggregations over the database.

The following values depicted in Table 6 are related to the size of the raw data recorded by the Brain Vision device. In other words, the table shows size statistics throughout experiments in the database. Due to their small size, the header and the marker files are not involved in the statistics.

Statistic	Value [MB]
Average	38,50
Minimum Value	0,08
Maximum Value	229,17

Table 6: Experiment Raw Data Statistics

To became more familiar with the data size distribution, we can see a histogram showing the most frequent experiments size as well as the less frequent ones. The histogram is depicted in Figure 14.

Note: Other charts are provided in the Apendix.

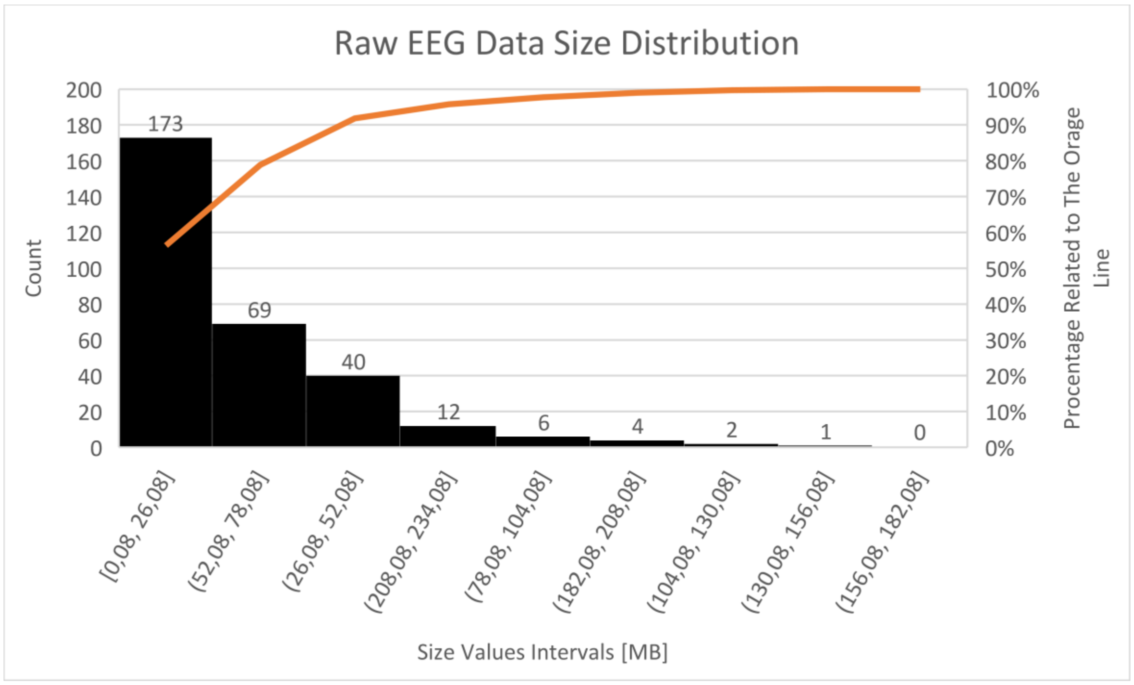


Figure 14: Histogram of the raw EEG data files sizes

As we can see, the most frequent experiment data values are usually between 0,08 – 26,8 MB in almost 63 % of cases, the second one is between 52,08 – 78,8 MB with approximately 20 % and the rest consists of the other sizes.

5.1.3.4 Graph Depiction

To depict trends and details about the data growth and characteristics there are three types of graphs described in the further subchapters. The graphs data sources are based on performed queries over the database, where the time of the leveraged experiments was restricted by the interval 1.2.2010 – 1.2.2016. There is a forecast of the trend lines for the next three years included in all the graphs. These forecasts are predicted by the algorithm called Exponential Smoothing which enables to assign different weights throughout the time periods. In other words, it means that the algorithm can take the most recent time period into account to which it assigns sufficient weight.

As a part of the algorithm, the confidence value of 95% is set for all graphs. This value represents the probability of the future data values that will be in the area restricted by the two orange curves within.

The X axis of all the graphs represent the timeline in years and the Y axis represent the size or the speed of the data growth in gigabytes.

5.1.3.5 Growth Velocity

The growth velocity represents the data speed throughout the timeline. As depicted in the graph below, (see Figure 15) the speed is linearly increasing until year 2015 where the top is reached and it starts decreasing to the bottom. This is probably caused by the unsystematic approach of uploading data into the portal. The data is typically uploaded in batches accumulated on the computers of the research members. Nevertheless, with this in our mind, we can state that the data growth velocity has got the linear characteristic. The forecast is copying the historical data in the linear data size growth.

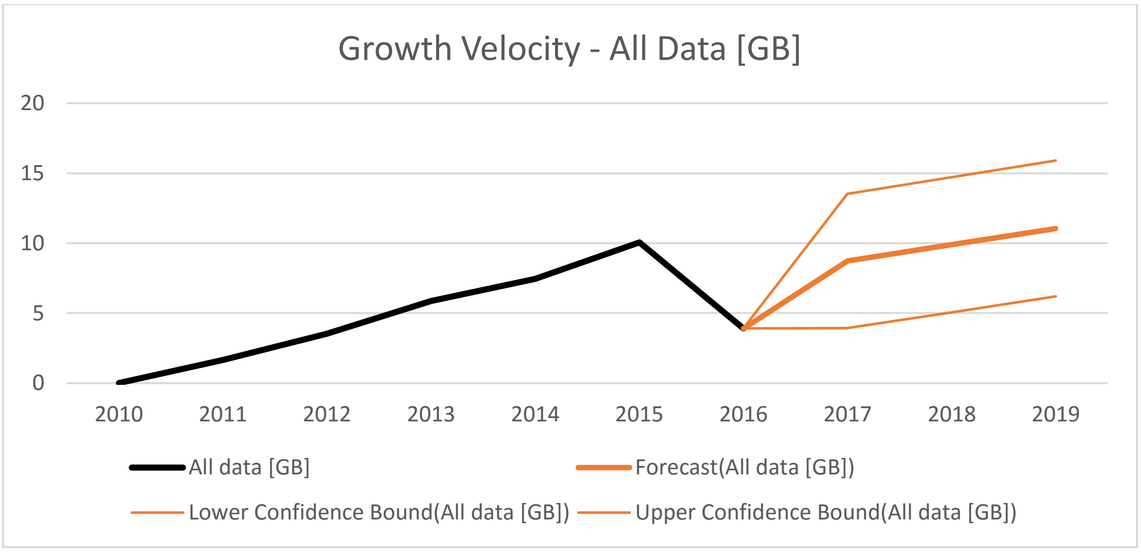


Figure 15: Growth Velocity - All Data

5.1.3.6 Incremental Data Growth

The graph shown in Figure 16 depicts how the velocity of the data growth affects the overall data increase. As we can see, the data growth is directly dependent on the growth velocity which made the overall data growth slower since the year 2015.

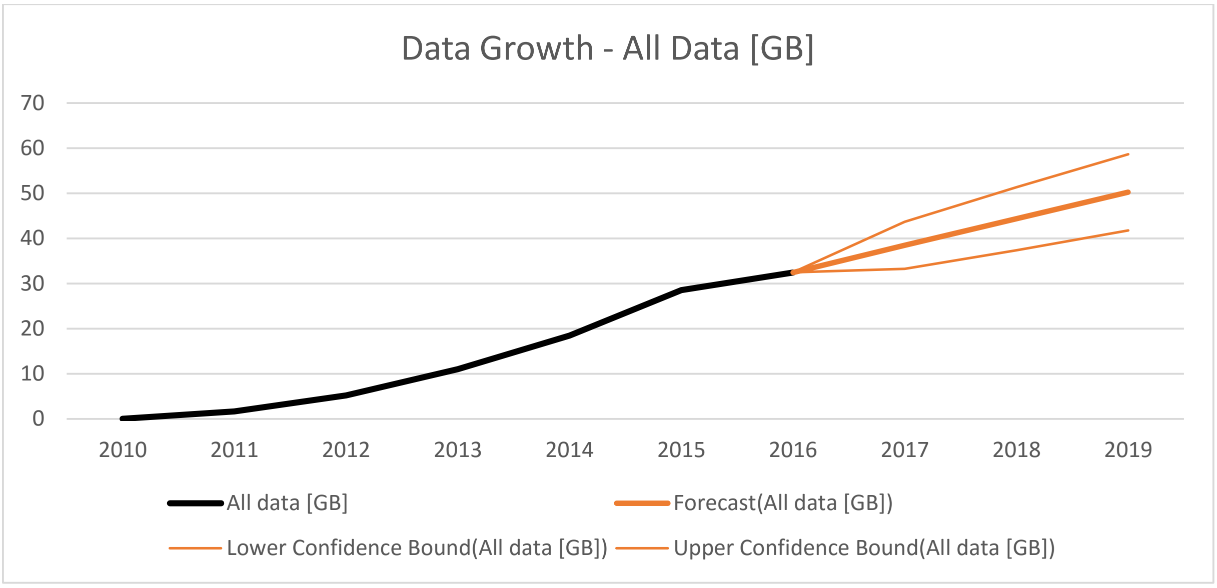


Figure 16: Data Growth - All Data

5.1.3.7 Long-time Forecast

The forecasts of the previous graphs were restricted to the three following years. The next graph (see Figure 17) is convenient to become familiar with the forecast which is related to long distance future. As seen below, the data does not seem to be growing as fast and the data growth seems to be linear. Without any external influences, according to the forecast, the data is supposed to reach the level of one terabyte in approximately the year 2032. It seems that the data cannot be considered as Big Data related to the volume characteristic without the external influence.

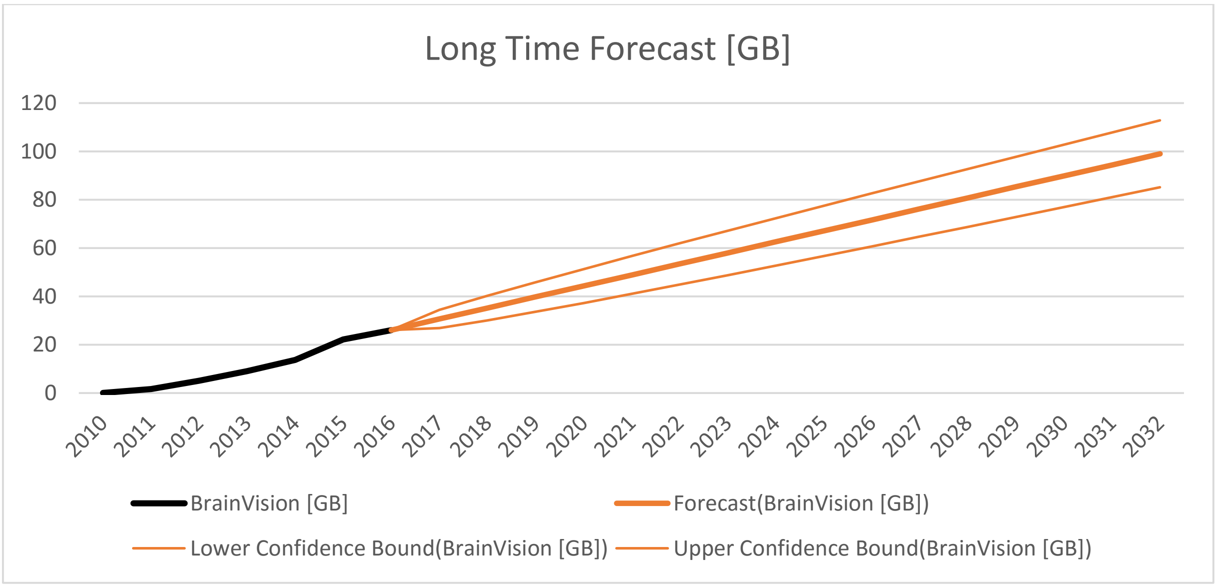


Figure 17: Long Time Forecast

5.1.4 Evaluation

In conclusion, as the result of the analysis, the growth rate is not the one which was expected. The size of the database does not meet the volume characteristic of Big Data. Moreover, the data growth, and the growth velocity values are linearly increasing.

5.2 EEG/ERP Project as Big Data or Small Data

- Volume – as the data analysis shown, the data volume and its growth velocity is not big/fast enough to meet these characteristic criteria.
- Variety – this characteristic includes some level of ambiguity. Data is stored in a structured form in the relation database but the binary data itself is considered as unstructured. Apart from the main part of the data in the BrainVision Format, other kinds of data such as graphs, videos and averages are stored.
- Velocity – The data is not changing dynamically and the velocity of the uploading new data into the database is rather slow. Although, nowadays there is not a need to process data in real-time, the raw data processing and analysis time starts consuming more time which can be a potential Big Data problem.

To better evaluate if the data has the potential to become Big Data, the comparison of the differences can be seen between Big Data and Small Data in Table 7 related to Table 1 from Chapter 2.6.

Criteria	Meeting the criteria	Description
GOALS	NO	Nobody knows what is the exact output of the analysis
LOCATION	?	The possibility of the expansion by the new projects
DATA STRUCTURE	YES	Although most of the data is stored in the relation database, there are many various types of it
DATA PREPARATION	?	Most experiments are repeatable
LONGEVITY	YES	The data is not kept for a limited time. The project has been running for six years
MEASUREMENTS	?	Although, there is only one device which provides the measurement, there is the possibility of the expansion by new subprojects

REPRODUCTIBILITY	NO	The experiments are typically repeatable
STAKES	NO	Due to its academic character, the project cannot consume much sources.
INTROSPECTION	NO	It is not difficult to access particular data
ANALYSIS	YES	The data is typically analysed in incremental steps

Table 7: Big Data vs Small Data Overview

In summary, there are some Big Data characteristics which the EEG/ERP project potentially meets but it is necessary to realise the proposed subprojects to declare the project as a Big Data one.

With this in mind, there are still some Big Data techniques which are applicable whose realization and implementation can become an asset for the project's future.

5.3 Project Workflow

To find an appropriate technique to apply, it is of crucial importance to primarily become familiar with the whole project workflow.

The project workflow is divided onto a few phases which are described in the following subchapters.

5.3.1 Preparation

Firstly, the experiment has to be designed along with the environment where it will be performed has to be suitable prepared including the room, stimuli generating devices (a data projector, speakers, etc.), staff who will perform the experiment, etc. Meanwhile, according to the experiment needs, appropriate individuals for the experiment are being gathered. [22]

For example, there is an experiment which is considered to investigate the driver's attention during a monotonous drive. As it was mentioned above, the project members have to gather appropriate hardware to generate stimuli such as a screen/projector, speakers and pedals in order to properly arrange the environment to be as close similar to the real environment as its possible. Particularly, there is a special room in the department, which tries to properly simulate the car environment (see Figure 18). [22]

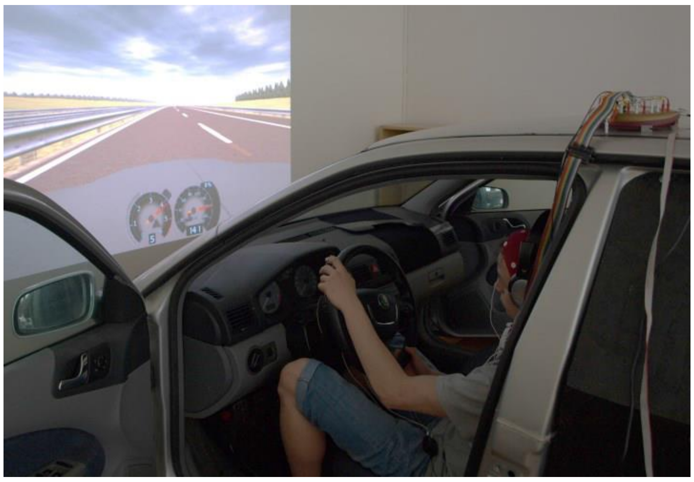


Figure 18: The Driver's Attention Experiment Environment [22]

5.3.2 Acquisition

The raw data is obtained by the BrainVision EEG recorder which is connected to a cap containing electrodes which are recording the signal from the participant’s scalp. The amount of the gathered data depends on the frequency of the measurement sampling (usually 1000Hz rate) as well as the number of active electrodes.

When the experiment scenario is ended, the metadata of the experiment are collected by participant questioning, the description of the environment, recording the day of the measurement, used devices, other participations, etc.

5.3.3 Pre-processing

The raw data is imported to software called EEGLAB which is an open source toolbox of Matlab. EEGLAB provides methods for the signal pre-processing such as an adjustment, the filtering or finding artefacts which are undesirable signal components usually created by unintended stimuli such as eye blinking.

5.3.4 Storage

Since the raw data is pre-processed, it is uploaded into the database via the project portal along with the metadata, where the part of them is stored in the Elasticsearch database.

5.3.5 Analysis

The experiment which needs to be analysed is downloaded from the portal and imported into Matlab via the EEGLAB plugin, or via projects programed in JAVA programming language, where convenient methods related to the character of the experiment are performed on the researchers' computers. Particularly, methods such as Matching Pursuit Algorithm, Hilbert Transformation, methods of deep learning, etc. are consider for the analysis.

Occasionally, some of the algorithms for the analysis phase are implemented in JAVA programming language. Depending on the algorithm complexity, the computation of some algorithms takes more time especially for large datasets. Due to it, most of the experiments are performed only over a subset of the experiment related data.

5.3.5.1 Matlab

Typically, the analysis of data is performed in Matlab, due to its wide functionality supported by additional toolboxes such as the EEGLAB and many others.

5.3.5.2 Java Modules

The department has implemented modules in JAVA language to load, pre-process and to analyse the raw data. As a part of some subprojects, the set of modules is being updated and expanded to provide new techniques and methods to deal with the data. Additionally, some of this functionality is integrated within the portal.

5.3.6 Results Discussion

When the analysis is finished, the results are evaluated and discussed among the members of the research group.

5.3.7 Results Publication

If the experiment results are significant or considered as an asset, they are published to become available among other neuroinformatic groups.

5.4 EEG/ERP Projects Phases in the Context of Big Data

Although the EEG/ERP project is not a Big Data one yet, most of the project phases are similar and overlap the Big Data Lifecycle. There is an overview of the EEG/ERP and a Big Data Lifecycle related to the Chapter 2.12.

Phase name	
EEG/ERP	Big Data
Data Preparation	Discovery
Acquisition/DataPre-processing/Storage	Data Preparation
Analysis	Model planning/building
Results Discussion	Communicate results
Result Publication	Operationalise

Table 8: Comparison of the EEG/ERP and a Big Data Project Lifecycle

The Data Preparation and the Acquisition phase is a little overlapped and similar to the Discovery phase of the Big Data lifecycle where the sources, technology and time have to be allocated, Similarly, the experiment has to be design and its environment has to be prepared (sources such as the hardware, people and environment have to be allocated).

Data Preparation/Acquisition can be compared with the Big Data Discovery phase where various activities such as data acquisition, cleaning which can be named as the filtering and the cutting of the signal on epochs as the transformation which is finally stored into the portal as loaded into an analytical sandbox.

To achieve any results, the research group finds appropriate subsets of data and algorithms suitable for experiments. These activities do not seem to be far similar to choosing techniques, determining workflows, building models and creating convenient environment for the models execution.

These phases are almost the same ones just with the slight difference, that the EEG project's experiment is one of the many performed. In the case of Big Data, the whole Big Data project is discussed and its results are critical for the whole organization. However, the EEG experiment is only considered if it is an asset which achieved expected results which can then be published.

Operationalise of the Big Data Lifecycle is considered to deliver final reports, code and technical document, in other words it is the last phase which summarizes and gathers the artefacts made during the whole lifecycle. The EEG/ERP project is just considered to gather achieved results into a presentable overview and publish it among public groups of neuroinformatic domain.

In conclusion, the phases are considerably similar to each other but the main difference is between the projects' scopes. As it was mentioned above, the phases of the EEG/ERP projects are considered to only one experiment at a time. When the last phase is finished, the whole cycle is repeated but usually in the context of another experiment, while the Big Data Project Lifecycle determines how to build up a new environment with sufficient models for processing Big Data, where the project success is critical for the company, because it is bound up with many sources. Moreover, the Big Data Lifecycle phases are often repeated in the case of failing to meet specified checkpoints which were declared at the start of the project.

5.5 Available Resources

Nowadays, the EEG/ERP project is deployed on hardware which is maintained by the Department of Computer Science and Engineering within the university. Due to its academic character, the department has restricted set of sources. On the other hand, there is an environment providing a huge hardware resources determined for academic purposes which can be considered as an alternative to the restricted resources of the department.

5.5.1 Local Resources

The department resources are centrally maintained by one of its employee. There are three supercomputers where each of them has 32 CPUs and 256 GB of RAM. The storage is provided by a disc array with a capacity of 100 TB. The hardware resources are distributed as virtual machines over projects within the department. If there is a need, the current hardware resources of a project can be additionally extended.

5.5.1.1 Hardware Resources assigned to EEG/ERP

The actual hardware resources assigned to the EEG/ERP project are composed of three virtual machines which are shown in Table 9 below:

Virtual machine	CPUs	RAM [GB]	Disc Capacity [GB]
1	1	4	100
2	1	4	100
3	1	4	60

Table 9: Actual Hardware sources assigned to the project

The first virtual machine is the production environment where the EEG/ERP portal along with its all parts is deployed. The second machine is a testing environment for developments The third machine is used as an experimental one.

In order to extend the data storage capacity, there is the possibility to obtain a few terabytes of additional disc space where the potential maximum limitation can be considered 10 TB.

Pre-processing and analysis phases are carried out on machines of the researchers where Matlab is installed and used via an academic network licence provided by the University of West Bohemia.

5.5.2 Metacentrum

Metacentrum which is coordinated by CESNET department is a project of operation and coordination of distributed computing and data storage infrastructure in the Czech Republic. It provides a dynamic network of resources throughout different locations determined to perform tasks whose memory and CPU requirements exceeds possibility of individual single computing centres. The highest priority of Metacentrum project is to support academic research efforts. Consequently, all academic subjects in the Czech Republic are able to use the services provided by Metacentrum without charging any fees. As part of the services there is a set of available software which has free usage as well such as Matlab along with a various set of toolboxes, Maple, Gaussian, etc.

6 Big Data Tool Selection

6.1 Storage

Nowadays, there are two databases where the data is being stored within the virtual machine which has 100 GB allocated. As result of the data analysis, without external stimuli the data volume will probably reach the level of 1 TB in approximately the year 2032. The current data storage system is sufficient for next following years. Most of PostgreSQL database size is occupied by binary files, which cannot be queries. With this in mind, there is no need to exchange this RDBMS for other more scalable technology such as a distributed file system. Moreover, the metadata of experiments are currently being stored in a NoSQL database where it can be queried in a more efficient way.

6.2 Pre-processing and Analysis

As said before, the whole process of analysis is performed either within Matlab along with EEGLAB or by applications written in Java.

6.2.1 Hadoop Based Solution

Most of technologies providing a scalable solution for dealing with Big Data are based on Hadoop. Due to the fact that Hadoop is implemented in Java, it provides suitable API which enables writing distributed Java programs based on MapReduce paradigm.

Hadoop is built on its distributed file system HDFS. HDFS solution is not efficient when the file system is supposed to deal with a huge amount of small files where a small file is the file which is smaller or almost equal to the size of block (default 64 MB). With this in mind, we can state that this solution could not be efficient in the domain of the EEG/ERP because the size of the most of the data files is less than 27 MB in 63 % of cases. Moreover, the remaining rest 37% of other files have different sizes, it means, which means, that if it they were stored in HDFS, most of the files would not fill HDFS blocks appropriately due to its inner fragmentation.

Although Hadoop provides some approaches how to deal with small files, neither of them is convenient to be applied in this context. The process of the data generation cannot be changed because the generated files are associated with particular experiments. Similarly, the same problem would occur if we used the method of batch file consolidation. Sequenced files technique, seems to be better, but if we consider that we would like to run a method over the dataset of one experiment, it would not be efficient either because one map operation is run over one data block, where the files of one dataset could be stored anywhere. In general, the solutions based on Hadoop are not appropriate Big Data technology for the domain of the department's EEG/ERP project. Moreover, there is no need to store data in a distributed file system.

6.2.2 MATLAB

Matlab provides a number of techniques and approaches to handle Big Data. Although there are tools such as memory mapped variables, disc variables and datastore which enable to deal with problems that occur when a large data has to be loaded into a computer's memory at once, they are considered as a method which provides a scalable solution. This functionality is enabled by Parallel Computing Toolbox along with Distributed Computing Server that provide a scalable distributed solution. This solution enables to run methods for data pre-processing and analysing over a scalable Matlab cluster.

6.3 Evaluation

A Matlab cluster seems to be the best solution for the application of a Big Data approach within the EEG/ERP project. Moreover, the researchers are familiar with this environment, which contains many available methods for data pre-processing and analysis either programed as Matlab scripts by the researchers or as functions provided by EEGLAB plugin which is widely used among this community.

7 Realization

7.1 Architecture of Sources for Processing EEG/ERP Experiments

The architecture depicted in Figure 19 represents software and hardware that is needed for EEG signal pre-pre-processing and analysis deployed over a Matlab cluster.

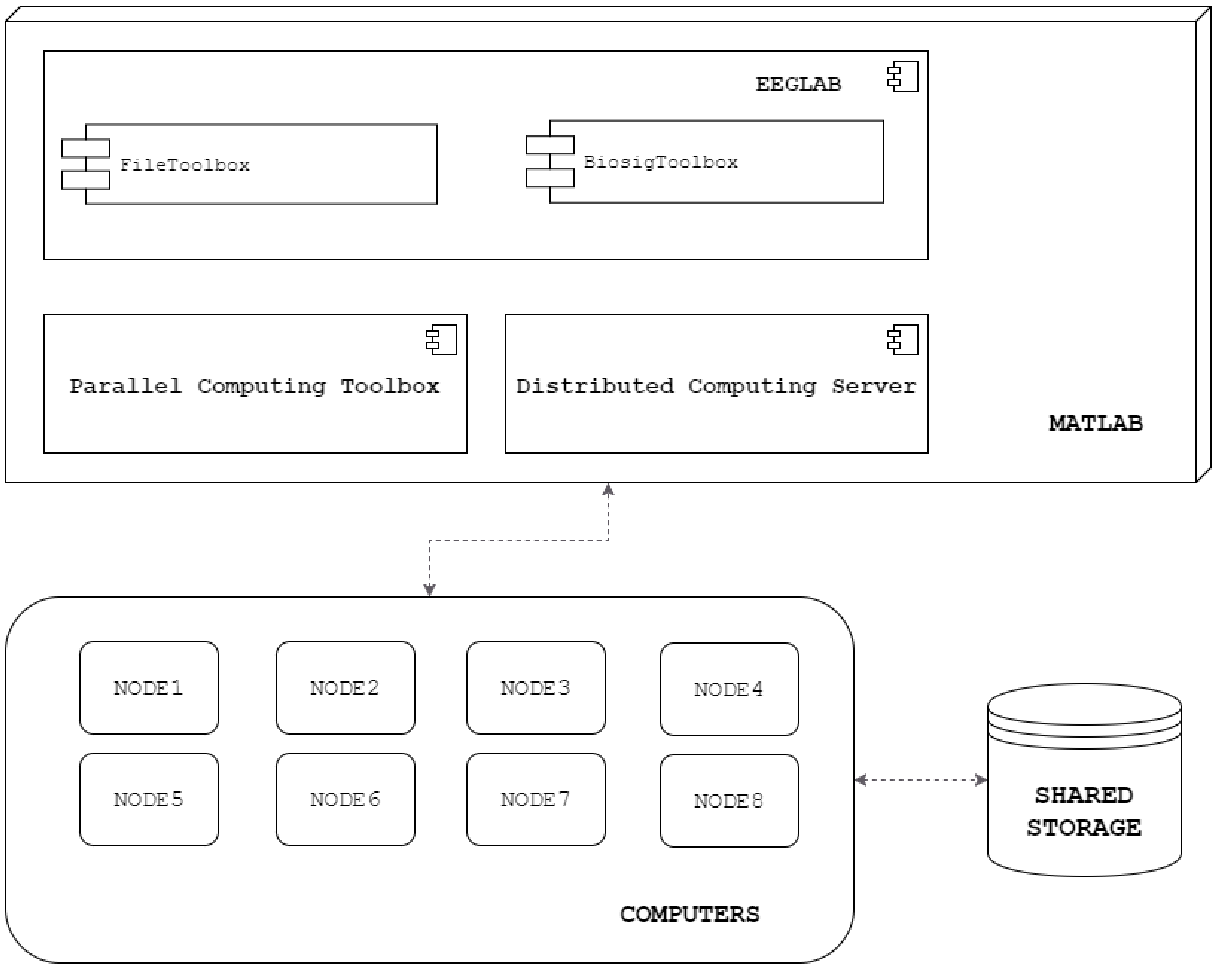


Figure 19: Architecture of Sources for Processing EEG/ERP Experiments

7.1.1 Software Requirements

7.1.1.1 Matlab

Due to the fact, that Matlab is proprietary software, to launch Matlab requires an appropriate licence. Moreover, to run a Matlab application over a cluster, other licences for the additional toolboxes are also required, particularly, licences for

Parallel Computing Toolbox and Distribute Computing Server. The following list describes the needed licences:

- Matlab – the licence required to run and use Matlab.
- Distrib_Computing_Toolbox – the licence is required to a run the master's node Matlab instance which manages a job and its tasks to be run on a Matlab Distributed Computed Server. Additionally, the licence can be used for parallel computations on a multi-core machine without using a cluster, the maximum count of available CPUs where an application can use with this toolbox is up to twelve.
- MATLAB_Distrib_Comp_Engine – the licence is required for tasks to be run on Matlab Distributed Computer Server's nodes.

7.1.1.1.1 Licences

To avoid buying the additional licenses, there is a possibility to use Matlab licences which are available within an academic project called Metacentrum.

7.1.1.1.1.1 The University of West Bohemia

There is a list of Matlab software licences which are available for academic purposes within the University of West Bohemia in Table 10.

TOOLBOX NAME	LICENCES' COUNT	TOOLBOX NAME	LICENCES' COUNT
MATLAB	450	MATLAB Compiler SDK	7
Simulink	150	Neural Network Toolbox	25
Bioinformatics Toolbox	15	Optimization Toolbox	50
Communications System Toolbox	10	**Distributed Computer Toolbox**	35
Computer Vision System Toolbox	2	**Distributed Computer Toolbox**	15
Control System Toolbox	50	Partial Differential Equation Toolbox	25
Curve Fitting Toolbox	27	Signal Processing Toolbox	50
DSP System Toolbox	25	SimBiology	5
Data Acquisition Toolbox	2	SimHydraulics	1
Database Toolbox	9	SimMechanics	4
Datafeed Toolbox	1	SimPowerSystems	1
Econometrics Toolbox	1	Simscape	5
Embedded Coder	1	Simulink 3D Animation	3
Financial Instruments Toolbox	2	Simulink Coder	2
Financial Toolbox	2	Simulink Control Design	10
Fixed-Point Designer	1	Simulink Desktop Real-Time	1
Fuzzy Logic Toolbox	10	Spreadsheet Link EX	1

Global Optimization Toolbox	1	Statistics and Machine Learning Toolbox	50
Image Acquisition Toolbox	2	Symbolic Math Toolbox	50
Image Processing Toolbox	31	System Identification Toolbox	25
Instrument Control Toolbox	1	Vehicle Network Toolbox	1
MATLAB Coder	7	Wavelet Toolbox	8
MATLAB Compiler	7		

Table 10: Matlab licences of the West Bohemia University

Although the licence of Distributed Computer Toolbox is available, licence of Distributed Computing Server is not included in the university package. Additionally, it can be bought in addition. Table 11 depicts the prices in the currency of Czech Koruna published by a company Humosoft which is an official distributor of Matlab licences in Czech Republic.

Number of CPU	Price [KC]
16	59 980
32	102 980
64	181 980
96	251 980

Table 11: Overview of Prices of Matlab Distributed Computing Server Licences

7.1.1.1.1.2 Metacentrum

There are licences which are required for running applications. Particularly, the elementary one is determined for opening and using Matlab as software. Additionally, there are other licences available which can be used too. Particularly, there is a set of available Matlab toolboxes providing additional functionalities related to the basic version. Some toolboxes are free of charge as open-source software. In contrary the others are commercial thus requiring a license for their usage.

There is a set of commercial toolboxes and their counts which are available for an academic usage on Metacentrum (see Table 12):

TOOLBOX NAME	LICENCES' COUNT	TOOLBOX NAME	LICENCES' COUNT
MATLAB	450	Target_Support_Package	1
SIMULINK	150	Embedded_IDE_Link	1
Bioinformatics_Toolbox	15	Simulink_HDL_Coder	1
Communication_Toolbox	25	Wavelet_Toolbox	8
Video_and_Image_Blockset	1	Neural_Network_Toolbox	150

Control_Toolbox	1	Compiler	7
Curve_Fitting_Toolbox	52	**MATLAB_Distrib_Comp_Engine**	320
Signal_Blocks	50	Symbolic_Toolbox	150
Data_Acq_Toolbox	2	Vehicle_Network_Toolbox	1
Database_Toolbox	9	Optimization_Toolbox	150
Datafeed_Toolbox	1	**Distrib_Computing_Toolbox**	15
Econometrics_Toolbox	1	PDE_Toolbox	50
RTW_Embedded_Coder	1	Real-Time_Win_Target	51
Fin_Instruments_Toolbox	2	Signal_Toolbox	50
Financial_Toolbox	2	SimBiology	5
Fixed_Point_Toolbox	1	SimHydraulics	1
Fuzzy_Toolbox	51	SimMechanics	4
GADS_Toolbox	1	Power_System_Blocks	1
Image_Acquisition_Toolbox	2	Simscape	5
Image_Toolbox	31	Virtual_Reality_Toolbox	3
Instr_Control_Toolbox	1	Real-Time_Workshop	2
MATLAB_Builder_for_Java	5	Simulink_Control_Design	50
MATLAB_Coder	7	Excel_Link	1
Identification_Toolbox	51	Statistics_Toolbox	50

Table 12: Matlab licences for academicals usage provided by Metacentrum

There are three marked licences in Table 12, which are required for our purposes. The others can be additionally used if there is a need.

7.1.1.2 EEGLAB

EEGLAB is an interactive opened source Matlab toolbox for processing EEG signals. It provides a lot of methods for signal pre-processing, filtering, averaging and analysis. User can use EEGLAB in two ways: as an additional library of Matlab, or as an interactive GUI (Graphical User Interface). EEGLAB also incorporates built-in tutorials as well as a help window. If the user uses EEGLAB as the GUI, there is an option to use a command history function that is considered as aid and a sample for further scripting. EEGLAB can be extended by users' programmed plugins as extension of current methods, compatibility with new input data formats, etc. Additionally, it can be deployed in a distributed environment.

7.1.1.3 Biosig plugin

EEGLAB Biosig plugin is an extension that provides compatibility with other input and export formats which are not included in the standard version of EEGLAB. Particularly, this plugin allows to import data from BrainVision format into EEGLAB.

7.1.1.4 FileIO plugin

It is a module which is used for source localization and importing some additional file formats.

7.1.2 Hardware Requirements

For the deployment of Matlab cluster, hardware sources need to be available, such as a set of nodes and storage which has to be available via file shared file system. There are two following options where a Matlab cluster can be deployed: on virtual machines of the Computer Science and Engineering Department or on available clusters of Metacentrum.

- The department resources – cluster can be administered and managed within the department which provides a certain level of elasticity that cannot be ensured by Metacentrum. The computation performance and other hardware sources do not have to be shared with other entities. As a big disadvantage is restricted scalability caused by restricted hardware resources of the department.
- Metacentrum resources – in comparison with the department resources, Metacentrum allows high hardware scalability, which includes hundreds of available CPUs and gigabytes of RAM along with terabytes of storage. The main disadvantage is that the sources are shared within the whole community, which can cause waiting for allocation of sources.

7.2 Environment Selection

As seen above, there are two options of environments where a Matlab cluster can be deployed (see Table 13).

Environment	Hardware Scalability	Flexibility	Licences	Maintenance	Ease	Deployment
Department	-	X	-	-	X	-
Metacentrum	X	-	X	X	-	X

Table 13: Environment Evaluation

Metacentrum is considered as a more suitable environment for running computations over a Matlab cluster. It has better hardware scalability, the maintenance is provided by Metacentrum staff and there are all needed licences. In other words, the department will not have to invest any money for required hardware, software licences, deployment and maintenance. Additionally, a Matlab cluster is already deployed within Metacentrum.

7.3 Implementation

In order to integrate these five technologies (Matlab, EEGLAB, BiosigToolbox and FileToolbox among each other in the environment of Metacentrum, it is necessary to become familiar with its architecture.

7.3.1 Metacentrum Architecture

The architecture of Metacentrum consists of three main parts: frontends, PBS (Portable Batch System) /Torque servers, computing nodes and disk storages. Some of the nodes can be virtual ones, in other words that there is one physical machine where are some virtual machines run on.

The frontends are machines used to users to log into the system directly without a reservation. Particularly, the frontends nodes provide the entry point of the Metacentrum system for users to prepare their tasks (jobs) to run, check the job actual state and to get generated results.

PBS/Torque servers look after jobs scheduling such as a job priority computation, scheduling jobs (putting them into queues), assigning resources and providing information about actual states of jobs.

Computing nodes are the end devices where the jobs are finally run. They are determined to leverage non-interactive tasks which are delivered to them by a scheduler system. They are placed together with disc storages in some cities throughout Czech Republic such as (Pilsen, Prague, Liberec, České Budějovice, Jihlava, Brno, Olomouc and Ostrava).

Disc storages are stored within particular clusters in the cities mentioned above to provide high performance without having undesirable delays. Additionally, all storages are available on all frontend nodes.

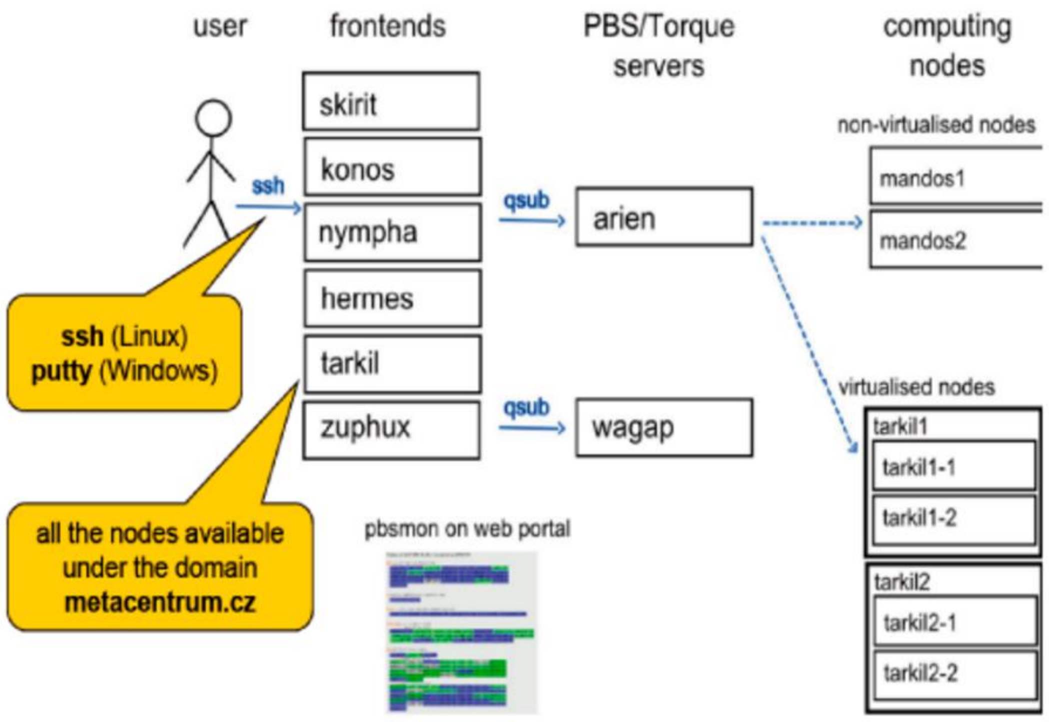


Figure 20: Metacentrum Architecture[10]

7.3.1.1 Scenario

As it is depicted in Figure 20, the typical activity workflow is composed of a few activities:

1. Log in – users logs into the system via an *ssh* (secure shell) client to one of available frontend nodes.
2. A job preparation – in this phase, the user prepares his job to run, he assigns various parameters to the job and submits it to the scheduler via a command called *qsub*.

[10] Metacentrum: https://wiki.metacentrum.cz/

3. Planning – the scheduler plans the job according to its priority and resource requirements.
4. Running – after job's waiting cycle, it is run on particular nodes chosen by the scheduler according to the user's requirements.
5. Finished job – when the job is finished the user can be informed via an email or he can check its actual state via *qstat* command from his frontend node where he is logged in or from Metacentrum websites.

7.3.1.2 Connection and Access to Sources of Metacentrum

Clusters, grids, servers and nodes are connected by a network which maximum transfer speed is up to 10 GB/s.

To access machines of Metacentrum *ssh* protocol is used to connect to a frontend node terminal which runs on GNU/Linux. Typically, it is recommended to connect via an application called PuTTy if one is using MS Windows platform or via *ssh* command on GNU/Linux platform. Authentication and security is realised by an authentication system called Kerberos.

To transfer files between a user and Metacentrum's storage a protocol called scp is typically used. For MS Windows platform, software called Winscp is recommended while for GNU/Linux is FilleZilla.

7.3.1.3 Torque Scheduler

Due to the fact, that Metacentrum is used by many users, jobs are run via a scheduler system called Torque which ensures a fair approach to the sources assignments to Metacentrum's users.

It is a PBS scheduler which enables you to run tasks in two different modes and with different tasks requirements. These modes are defined by the user's perspective where the first one allows an interaction (interactive mode) while the later one is run on the background (non-interactive mode). Additionally, the scheduler provides a definition of tasks' requirements such as performance, location, estimated time to process the task, etc. In order to serve many users, Torque has to deal with the management of sources allocation to provide a fair sources assignment among the cluster users.

Torque is the scheduler that is used throughout all clusters of Metacentrum. A basic term related to Torque is called a job. Due to the fact, that there are two modes of the interaction, when a user wants to create job, he has to determine if he wants to create an interactive or non-interactive (bash) job. If he chooses a bash job, after he submits the job by a particular command, the bash job is assigned to one of user's available queues, which was specified by the user before. The job is waiting in the server queue on a suitable time to be run. Meanwhile, PBS is making decisions depending on the actual state of available resources and users' priorities. Finally, the job is run as soon as there are necessary job resources available and also when the job has reached the relevant priority.

As it was mentioned above, jobs are waiting in queues. The queues are categorised by the maximum time duration of jobs to be finished. Particularly, there are eight groups: 2h, 4h, 1 day, 2 days, 1 week, 2 weeks and more than 2 weeks.

To enable user to handle a job, PBS provides a set of commands where the basic ones are involved in:

- submitting job to the queue,
- cancellation of a waiting or a running job,
- receiving information about the current node state and its properties,
- appearing graphical overview of queues and jobs, etc.

Another set of command is considered to the requirements specification for computational resources. Requirements for the properties and resources are specified by special marks which can be a part of the command of a job submit. These settings allow the user to set up the following tasks:

- the time queue where the job will be run in,
- how many nodes the job needs and their type,
- how many CPUs have to be allocated,
- which particular nodes or their physical location,
- required amount of the needed physical memory,
- temporary storage with short access duration for computational purposes,

- required software licences,
- other options such as the operation system, location in the cluster, networking cards and the file system.

There is a workflow of running a job:

- A user logs in the system via *ssh* protocol to one of frontend nodes,
- He runs a job by submitting a command called *qsub* which enables to set up sources for a particular time and passes it to a scheduler,
- After some time of waiting for the sources, the scheduler assigns the required sources (machines, processors, memory, licences) to the job.

There are four main types of job states:

- Q – The job is queued,
- R – The job is running,
- E – The job is exiting after running,
- C – The job is completed after running.

The priority of a job is associated with some rules and there is a procedure of the priority establishment. Firstly, jobs are sorted in by the following criteria:

- The priority of the queue – jobs in queues with the higher priority have an advantage,
- Fair share – jobs from the user who has spent less time by processing jobs has an advantage,
- Job time – the job which has the least time requirement has got an advantage.

Since the jobs are sorted by the previous steps, the scheduler goes through the sorted jobs and decides if the job is permitted to run regarding the sources demanded and which resources are available with consideration of the preferable location.

7.3.1.4 Data

Metacentrum uses a shared file system to guarantee a cohesive policy to handle data throughout the cluster which ensures that a user can access his data regardless to the location where it is stored and also provides a faster storage considered to meet data demands of being stored on a device that quarantines minimal delays while jobs are being computed.

There are to two main types of the storage:

- Storage – it consists of shared and backed-up NFSv4 volumes available from all the frontend as well as all the computing nodes. Disc arrays are related to the location of the particular clusters where they are physically placed nearby to ensure the lowest latency for the best possible performance.
- Home – it is a shared filesystem dedicated for users' home folders.
- Scratch – the scratch is a kind of storage providing the fastest devices where data can be stored. Mainly, it is dedicated for applications' temporal/working data. This local volume is available on all computing nodes throughout the whole cluster. Although this storage does not provide back-ups, it possesses basic protection of RAID 10 to avoid hardware failures.

7.3.2 Matlab with Distributed Computing Server

Matlab allows run programs on a cluster. It is based on the master-slave architecture where one node/instance is the master and other nodes are slaves.

A program determined to run on a Matlab Distributed Computing Server is decomposed on tasks which are parts of a job. The job's role is to represent a set of tasks as one object to be handled by a scheduler.

Tasks are distributed to available nodes of the cluster. The master runs only one instance of Matlab to manage the communication among slaves' nodes.

Tasks are distributed to available nodes of the cluster.

There are two job categories:

- Distributed job – a job consisted of tasks which do not communicate with themselves. There is no need for the communication and synchronization. The tasks are independent on each other.

- Parallel job – it is the opposite of the distributed job. In this case, tasks need to be synchronized and communicate among the job context to get a result with is reached by their cooperation.

7.3.2.1 Architecture

The Matlab session in which the job and its tasks are defined is called the client session. All clients accessing a Matlab Cluster have their own session/context in which their programs run. The session contains definition of paths, jobs and task. The internal Matlab Job Scheduler (MJS) distributes the tasks for evaluation to the server's individual sessions called workers. The architecture is depicted in Figure 22.

To be able run the client session, the user needs to have the license called Parallel Computing Toolbox. Similarly, the user has to possess other licences for running tasks on works which is managed by Distributed Computing Server. The licences' schema is show in Figure 21.

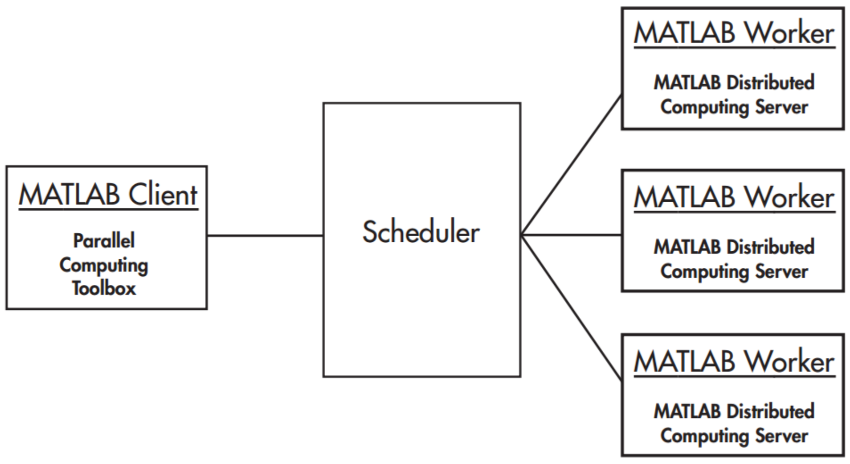

MATLAB Client
Parallel Computing Toolbox
Scheduler
MATLAB Worker
MATLAB Distributed Computing Server
MATLAB Worker
MATLAB Distributed Computing Server
MATLAB Worker
MATLAB Distributed Computing Server

Figure 21: Licences' Schema [15]

7.3.2.2 Matlab Job Scheduler

MJS is the part of the Matlab Distributed Computing Server that coordinates the execution of jobs and the evaluation of their tasks. MJS scheduler runs jobs in the order in which they are submitted, unless any jobs in its queue are promoted, demoted, cancelled, or destroyed. Additionally, if there is a need, there is a possibility to use MJS in a cooperation with another scheduler of a third party. The role of the scheduler and a job workflow is depicted in Figure 22.

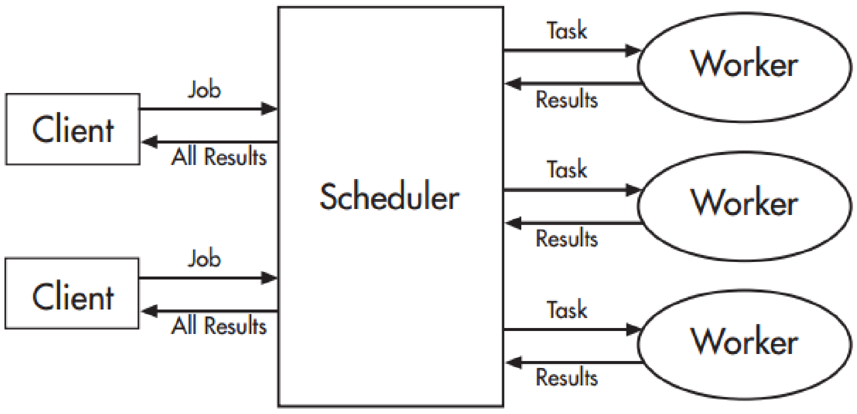


Figure 22: The Role of The Scheduler and The Job Workflow [15]

7.3.3 Overall Architecture

It represents the composition of the particular architectures' components such as: the Metacentrum access via *ssh/scp*, the shared file system, Torque scheduler, the Matlab session, MJS scheduler, Matlab, EEGLAB, Biosig toolbox, FileIO toolbox, Parallel Computing Toolbox and Distributed Computer Server. The overall architecture will be described in the distributed job type context.

7.3.3.1 Preparation

Firstly, a user has to create a Matlab function which he would like to run over multiple data (UserFunction.m).

To run it over a Matlab cluster, it is necessary to create another Matlab file which represents the client session (ClientSession.m). This file contains settings of the Matlab cluster and the decomposition of the application onto Matlab jobs and tasks (see 7.3.2 Matlab with Distributed Computing Server). In this case, we consider to have the application decomposed of just one job which consists of a number of tasks.

To run an application over a Metacentrum cluster, there is a need to create a job where the Matlab job and task will run. The job is created by submitting a bash script (Script.sh) to the Metacentrum's task scheduler Torque. The bash script contains Torque directives to define the job's parameters, settings, and the software which is supposed to be run on the cluster (Matlab), and to allocate hardware resources which are required for the running of the program such as the memory amount, number of processors and nodes. Moreover, the script has to also define required software licences such as one licence of Parallel Computing Toolbox for the Matlab session and Distributed Computing Server licences for Matlab workers, whose count depends on the number of the CPUs where the distributed application is supposed to be run on.

7.3.3.2 Software Dependencies

As mentioned above, there are six software components which are: Matlab, Parallel Computing Toolbox, Distributed Computer Server, EEGLAB Toolbox, Biosig Toolbox and FileIO Toolbox. There are two entities which need to have software sources assigned before they are run:

- A Matlab client session – one licence of Matlab and Parallel Computing Toolbox are required to run the client session,
- Matlab workers – for each Matlab worker one licence of Distributed Computer Server is required. Additionally, the paths to Matlab and to EEGLAB Toolbox (contains FileIO and Biosig Toolboxes) are also needed.

7.3.3.3 Activity Workflow

There are two protocols for the interaction with Metacentrum environment. The *ssh* protocol allows accessing a GNU/LINUX terminal to submit commands, while scp protocol enables the copying of files to the shared file system of Metacentrum.

When the script and the Matlab files are uploaded and everything is ready for the run (the user is connected to the terminal), the job is submitted into the queue of jobs by a command qsub Script.sh. The queue is managed by the Metacentrum's task scheduler Torque which assigns priorities to jobs depending on the allocated hardware sources, the user's group, etc. (see 7.3.1.3 Torque Scheduler).

When the required sources are assigned (hardware, Matlab licences such as Parallel Computing Toolbox and Distributed Computing Servers), the job is run by Torque. In order to run the Matlab script over the cluster, a Matlab client session is created as the master node in which Matlab jobs and tasks are defined. The whole job and its tasks (defined by UserFunction) are passed to MJS scheduler, which schedules and distributes the unfinished tasks from the task queue over the set of the Matlab workers that possess the Distributed Computing Server licences. While the tasks are being performed, the client session can gather their results or the tasks can store them via output files in the shared filesystem.

When the tasks are finished, the client session finishes the code and once done, the job is also ended. Job warnings and failures are stored in a file generated by Torque within the same shared filesystem folder as to where the job was submitted from.

Activity overview:

1. Creation of a bash script, a client session Matlab file, a user function.
2. Uploading the files to the shared file system on Metacentrum (via *scp*).
3. Accessing the GNU/Linux terminal on a frontend node (via *ssh*).
4. Submitting the job (qsub command passes the job into the queue of Torque scheduler).
5. Torque evaluates priorities and gathers resources for the job.
6. When the job is run, the Matlab client session commences.
7. A Matlab job is initiated (it is decomposed onto tasks).
8. The tasks are scheduled by MJS (assigned to free Matlab workers to be processed).
9. Results are generated (gathered by the Matlab client session or stored by tasks into output files).
10. The Matlab session is closed.
11. The job is finished.
12. Reports are stored into the folder where the job was started.
13. The user checks the results (evaluates and downloads them onto his local machine via *ssh* and scp).

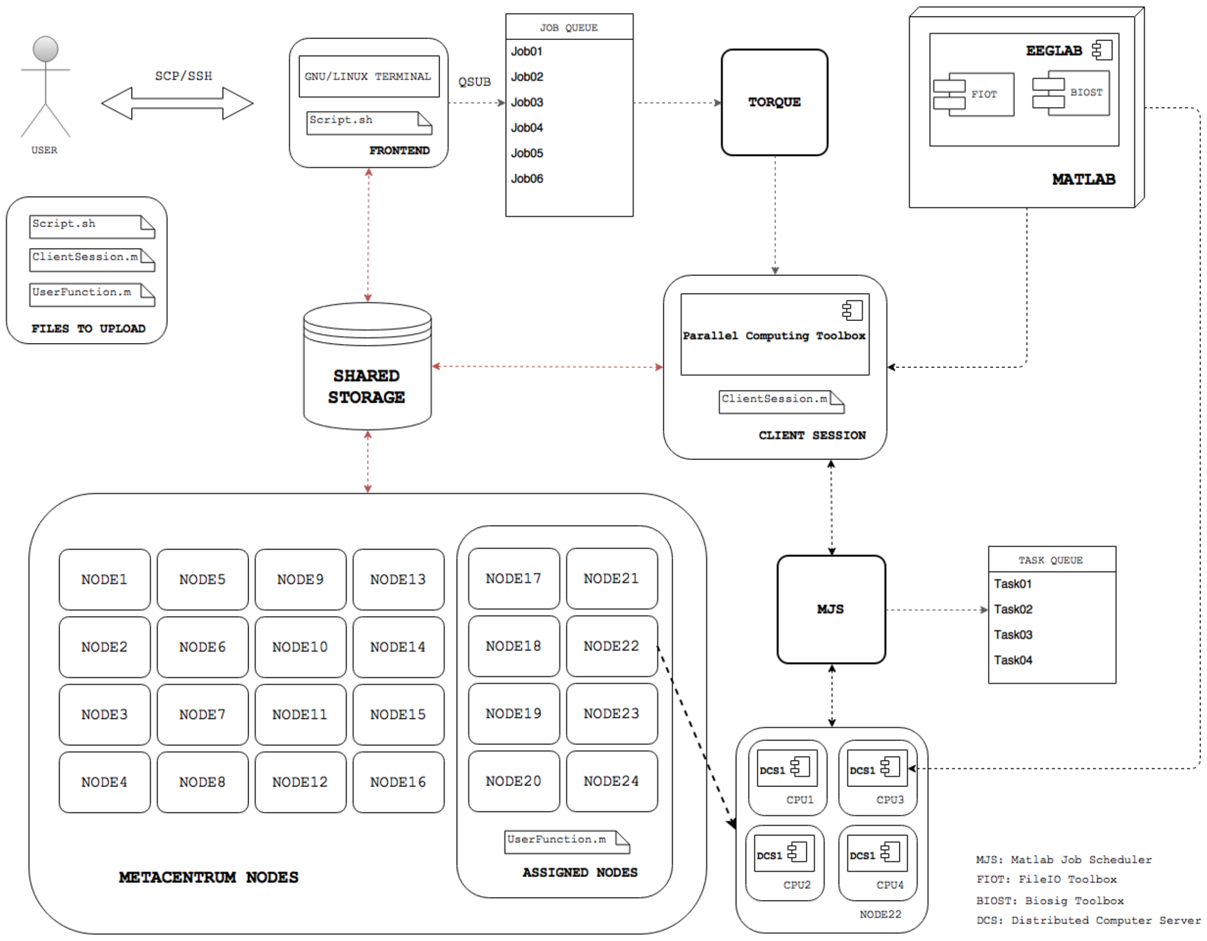


Figure 23: Overall Architecture

7.3.4 Task Selection

As mentioned above, there are two main options of Matlab Distributed Server application which are related to the job characteristics. The first option allows the user to run one task over multiple data (Simple Program Multiple Data) and the second one permits the running of a program in parallel.

Both of these use cases are considered as useable for high scalable processing and analysing EEG signals over a cluster.

Furthermore, the first option can be widely used, due to the generality of its deployment. Particularly, if we consider that an experiment has a dataset which consists of 100 or more measured signals, all these signals can be easily processed over a cluster, where a method which is determined for one signal processing can be easily run over the whole dataset.

This option will be demonstrated as a use case.

7.3.4.1 Model of SPMD (Simple Program Multiple Data)

To achieve running one program over multiple datasets it is necessary to create data and activity workflow. Particularly, this use case requires gathering input data and passing their names and their location into the main Matlab script which represents the client session. The data and activity workflow is depicted in Figure 24.

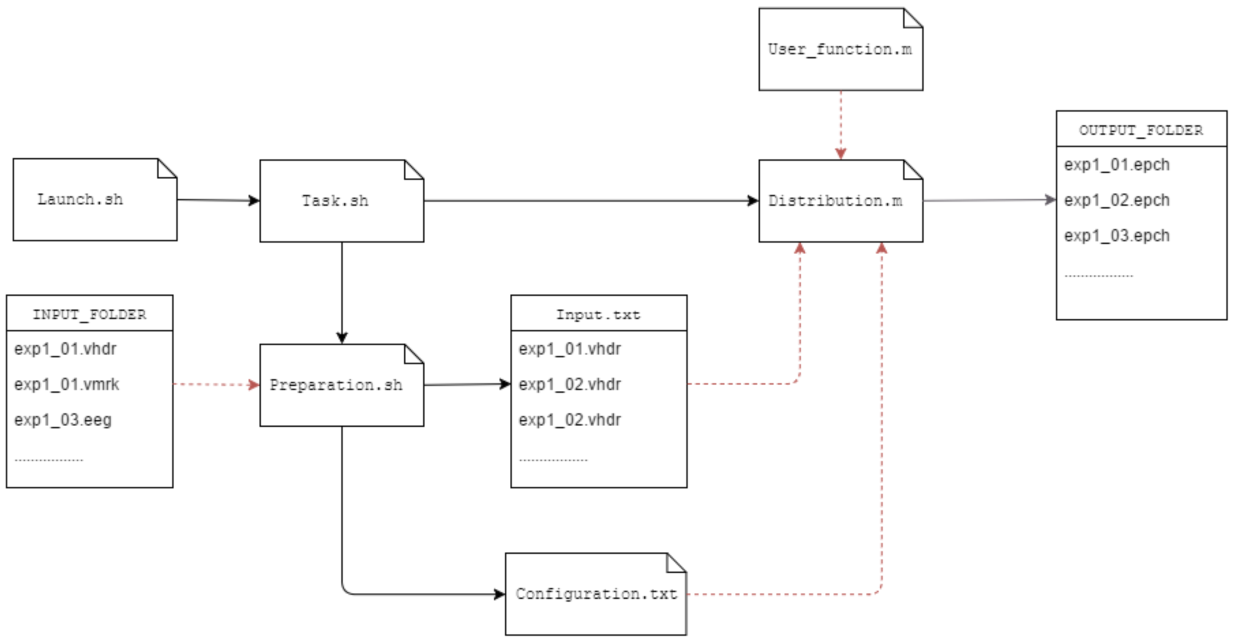


Figure 24: Use Case's Data and Activity Workflow

As is depicted in Figure 24, there are two types of entities: activities and data. The activities can be dependent on required input data. Data requirements of an activity are depicted as red dashed arrows. The activities represent bash GNU/Linux and Matlab scripts, where the bash scripts are used for the structure and data preparation and the Matlab scripts for performing the computation. The details are described below:

- Activities
  - Launch.sh:
    - clears the actual folder,
    - runs Preparation.sh,
    - submits the Task.sh to Torque.
  - Preparation.sh:

        - creates a list of header (.vhdr) files from the INPUT_FOLDER and saves it to Input.txt,
        - creates Configuration.txt, where additional data required for the computation can be found.
    - Task.sh – contains settings of Torque scheduler such as software requirements, licence requirements, performance requirements, the required location, etc.
    - User_function.m – a Matlab script which will be run over the Matlab cluster.
    - Distribution.m – represents the Matlab client session which defines the settings of the distributed environment and the definition of the decomposition on jobs and tasks. It involves:
        - the setting of a parallel pool,
        - checking licences' availability,
        - linking required libraries,
        - definition of input files,
        - creating of jobs/tasks which are passed to MJS scheduler,
        - reporting occurred failures,
- Data
    - INPUT_FOLDER – it contains all input data files (a dataset) in BrainVision Format which are supposed to be processed.
    - OUTPUT_FOLDER – it is the output folder where the results are gathered.
    - Input.txt – it represents a list of header (.vhdr) files which will be imported to Matlab via EEGLAB.
    - Configuration.txt – an input file which contains additional settings such as the path of INPUT_FOLDER, OUTPUT_FOLDER and the suffix of the output files.

Note: The task outputs do not have to be stored in files; they can be gathered and subsequently processed by the main Matlab program (the client session).

7.3.4.2 Use Cases

To demonstrate a practical usage of the model functionality, two use cases were implemented. These two are described in following subchapters.

Note: when an EEGLAB script is created and examined over one data file from a dataset, it can be run over the whole dataset within our SPMD model without any changes.

7.3.4.2.1 Dividing on Epochs

When an experiment is performed, the measurement device is recording EEG signals which are generated by the human brain. The parts of a signal represent a brain reaction on various stimuli. During the measurement process, devices such as a projector or a speaker generate outer stimuli that are perceived by the participant who is being measured.

The time when the stimuli were presented are recorded in the marker file (.vmr). The purpose of these records is to be able to associate a stimulus to the recorded brain activity. Typically, the brain activity is slightly delayed.

To analyse the activity inducted by a stimulus, the originally continuous signal is divided onto epochs which represent intervals that could have been influenced by particular stimuli.

For dividing a continuous signal to epochs, EEGLAB functions within Matlab can be applied. The script consists of the following three phases:

1. The header file is imported to EEGLAB,
2. An appropriate method is examined,
3. Epochs are saved in an output file.

7.3.4.2.2 Signal Filtering

Once a signal is recorded, it is necessary to remove frequencies which are associated with undesirable artefacts such as sweating artefacts, artefacts generated by electronic devices, etc. Frequencies lower than 0.1 Hz as well as higher than 30 Hz are typical for the occurrence of these artefacts. Mostly, the filtering takes a longer time to process than the deviation on epochs. Consequently, it makes this method suitable for the demonstration of the model usefulness. This use case consists of three steps:

1. The header file is imported to EEGLAB,
2. A method for the frequency filtering is examined,

3. The filtered signal is stored in an output EEGLAB file (.fdt) along with its configuration file (.set).

7.3.4.3 Testing

In order to test the model's appropriate functionality, two different datasets were chosen. The first dataset is determined to test the model functionality over a dataset which is characterised by a huge amount of small files. The second dataset is used to test the appropriate functionality over the few largest files of the EEG/ERP database.

- P300 Components Dataset – a dataset determined for finding P300 components. This dataset consists of 247 measurements recorded in BrainVision Format with an overall size of 1 GB.
- Driver's Attention Dataset – a subset of measurement which is characterised by the large size of the files. There are 14 EEG records with a total size of 3 GB.

The following Table 14 and Table 15 represent the hardware configurations and the elapsed times of the performed use cases which were run over the datasets.

Attribute	Setting 1	Setting 2	Setting 3	Setting 4	Setting 5	Setting 6	Setting 7
Number of nodes	1	1	1	2	3	4	5
Number of CPUs	4	8	16	16	16	16	16
Total available memory [GB]	10	10	20	20	20	20	20
Location	X	X	X	Brno	Brno	Brno	Brno
Number of workers	3	7	15	31	47	63	79
Elapsed time – P300 Dataset [Min]	34	17	14	9	9	8	8
Elapsed time – Driver's attention Dataset [Min]	9	6	6	7	6	8	8

Table 14: Dividing on Epochs - Hardware Configuration

Attribute	Setting 1	Setting 2	Setting 3	Setting 4	Setting 5	Setting 6	Setting 7
Number of nodes	1	1	1	2	3	4	5
Number of CPUs	4	8	16	16	16	16	16
Total available memory [GB]	10	10	20	20	20	20	20
Location	X	X	X	Brno	Brno	Brno	Brno
Number of workers	3	7	15	31	47	63	79
Elapsed time – P300 Dataset [Min]	84	31	18	12	10	10	10
Elapsed time – Driver's attention Dataset [Min]	187	46	31	46	42	35	41

Table 15: Signal Filtering - Hardware Configuration

As depicted in Figure 25 and Figure 26, the distribute computation of SPMD seems to be more efficient in the case of the signal filtering than of the cutting on the epochs, due to the higher complexity of the signal filtering method which can be seen in the considerable differences between curves of these two methods in Figure 28 .

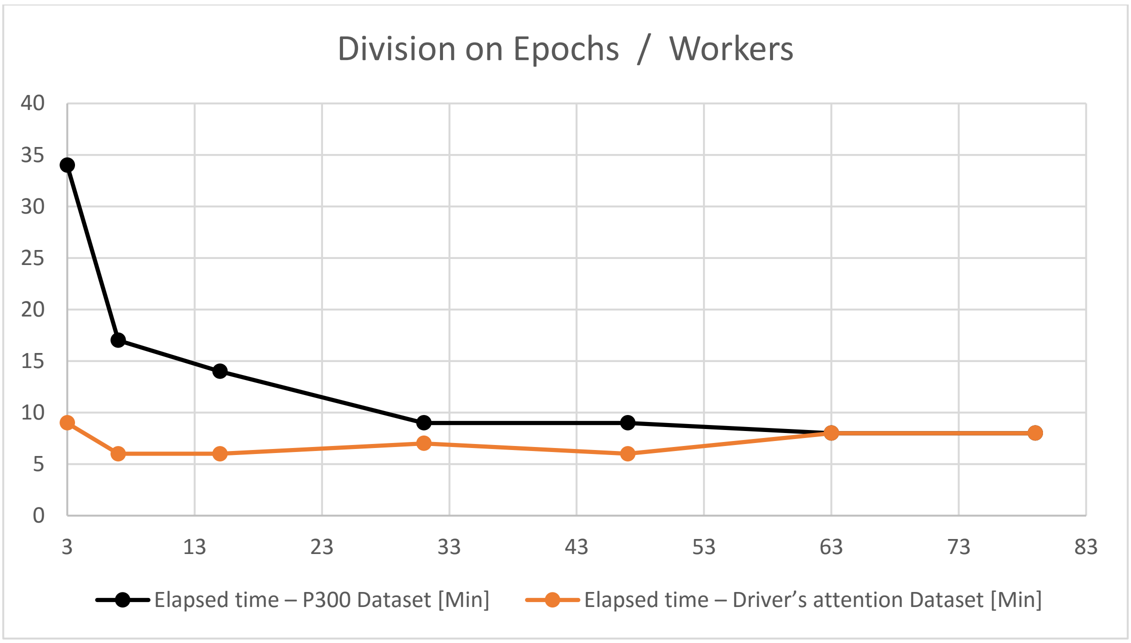


Figure 25: Division on Epochs / Workers

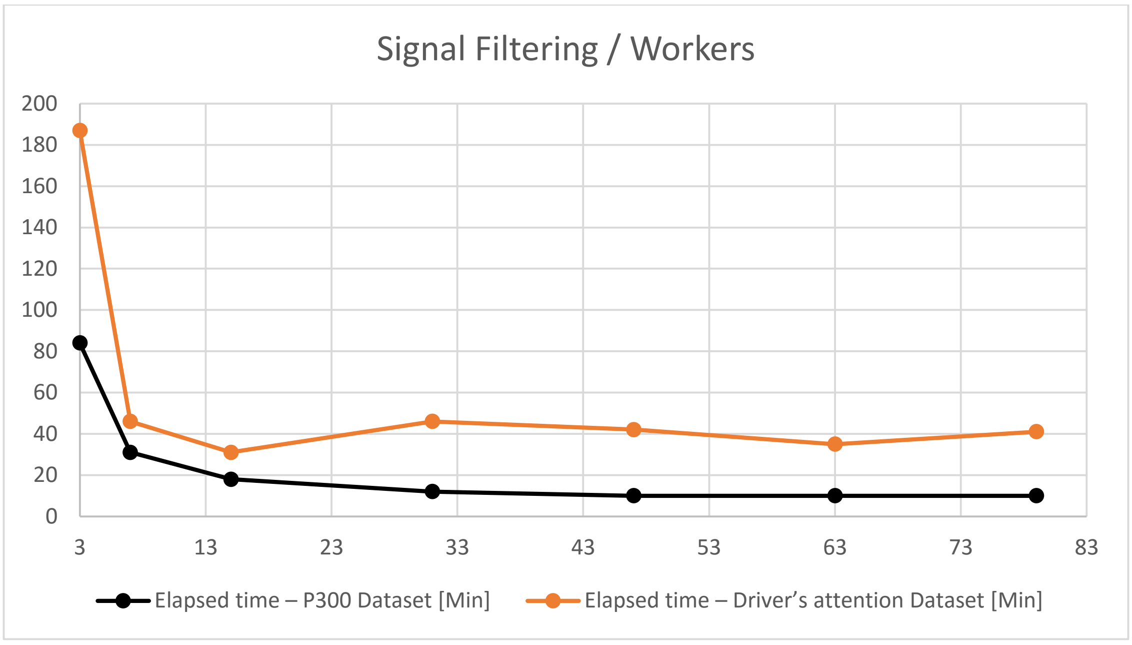


Figure 26: Signal Filtering / Workers

Additionally, Figure 27 and Figure 28 depict the processing dependency on the particular datasets. A huge amount of small files tends to have more deterministic characteristic than the small amount of large files of Driver's Attention Dataset. As seen in Figure 28, the ideal setting for the best efficiency of processing of the Driver's Attention dataset is to use the same amount of workers as the count of the dataset's files which is 13.

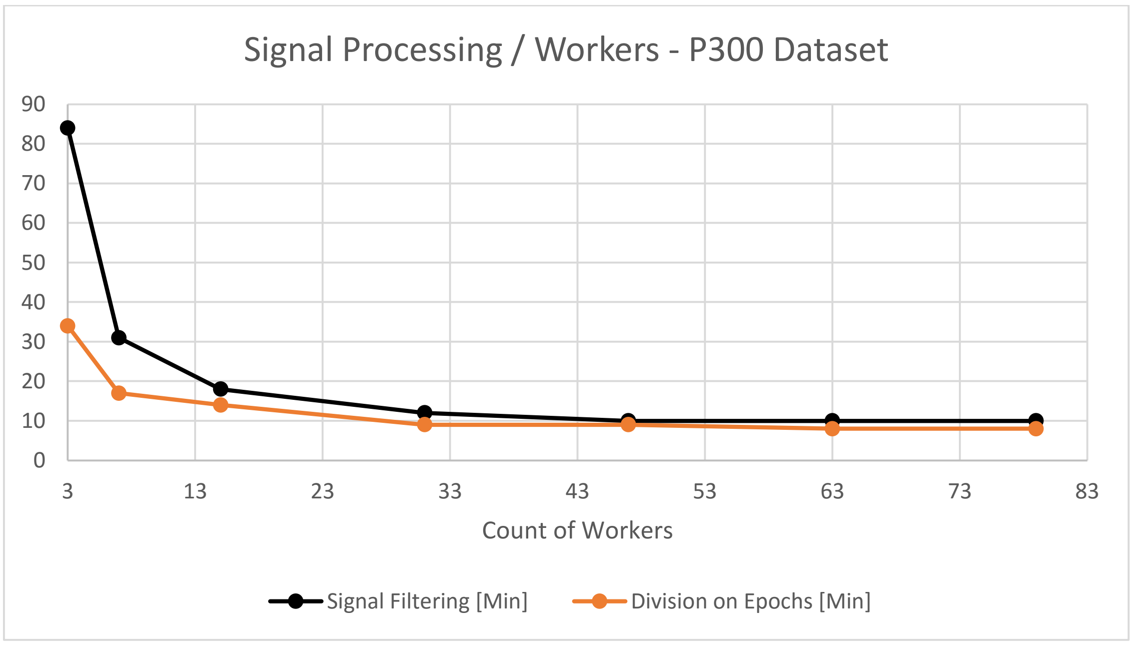


Figure 27: Signal Processing / Workers - P300 Dataset

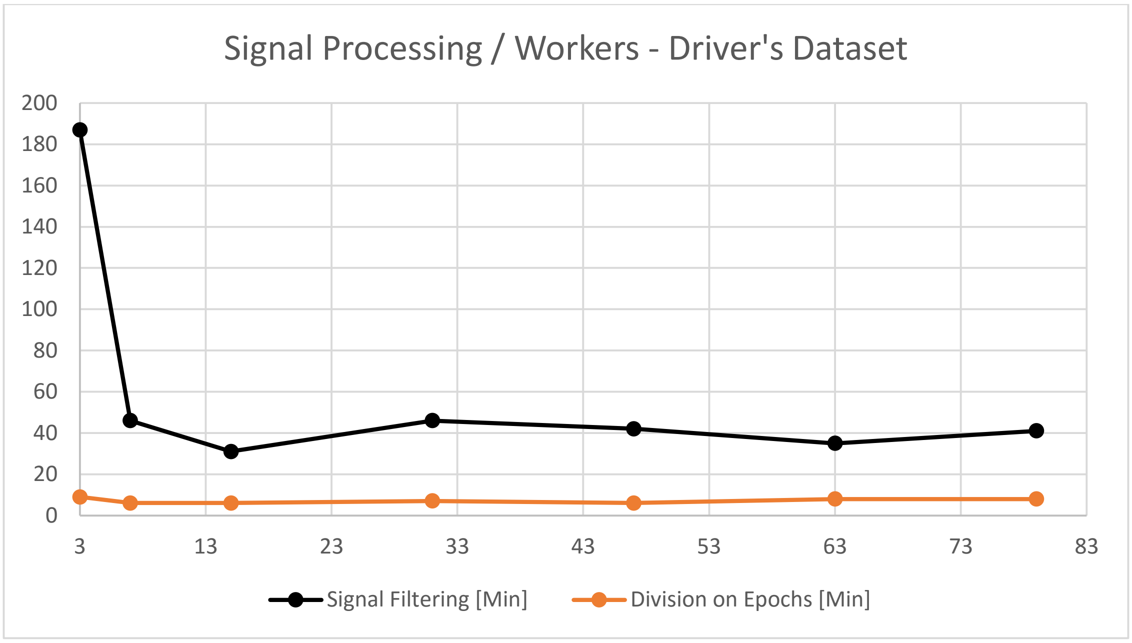


Figure 28: Signal Processing / Workers - Driver's Dataset

Although the files of epochs, filtered signals and their configurations were created from all measurements without any occurred errors, when the computation is distributed over more nodes, the location of each node has to be equal, otherwise the computation ends up with an error. Similarly, the same problem occurs when non-existent nodes are required. These problems have been discussed with the support of Metacentrum, likely the problems are caused by machine virtualisations along with the access to the file shared system where many technologies have to be integrated into each other.

7.4 Results

As the result of the testing, it can be conducted that the developed model is fully functional. Consequently, the integration of these four technologies Matlab, EEGLAB, Biosig toolbox and FileIO toolbox can be deployed together over a Matlab cluster.

This model can be a big asset for signal pre-processing and analysis within the domain of the EEG/ERP project. The main asset is considered to having a scalable solution for performing various methods over big datasets of data. All in all, a method for analysis of Big Data has been applied in the EEG/ERP domain.

7.5 Future Recommendations

In order to help to improve the future projects, the following three recommendations are considered in association to this study.

7.5.1 Parallel Computing

Matlab enables the user to make a Matlab program parallel and run it over the cluster. This functionality can be used farther to run algorithms which can allow them to decrease their running duration.

7.5.2 Integration of the model into the EEG/ERP Portal

The main advantage of the model is that the User_function representing the code which will be run over the cluster can be replaced. There is simply a need to retain its two input parameters such as the path along with the filename of a header (.vhdr) file and the path and name for the output file.

Furthermore, the whole architecture can be used all along with the EEG/ERP portal where the Matlab cluster can provide sufficient performance for processing methods with higher complexity (see Figure 29).

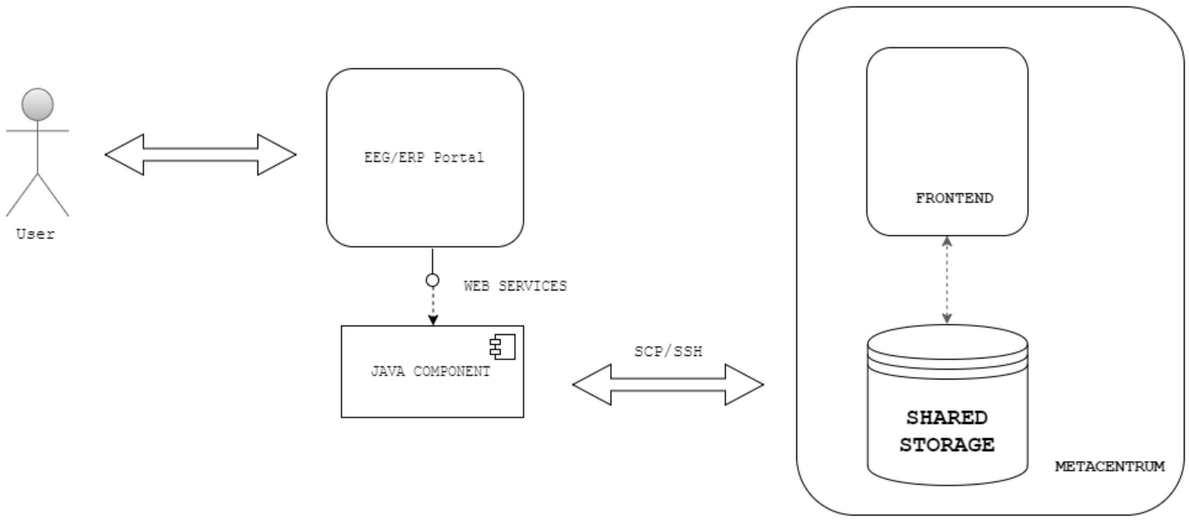


Figure 29: Portal and Metacentrum Integration

The portal can be integrated with Metacentrum via *ssh* and *scp* protocol and portal's web services. A Java module which implements the communication can be as a mediator between these two entities. Additionally, there are many java open source libraries that can be used for the implementation of the communication via scp and *ssh*.

There can be a few implemented methods for users provided by the portal. When a method would be supposed to be run over a big dataset, it can be passed to Metacentrum via the module which would manage the transfer of the dataset to Metacentrum, the method running, and the transfer of the generated results into the portal back to the user.

7.5.3 ITI Metacentrum Group of Cybernetic Department

There is a group within the Cybernetic Department of the University of West Bohemia which shares a computational cluster along with some licences of Matlab Distributed Server throughout Metacentrum. Metacentrum assigns a privilege of higher priority over the hardware which is provided by particular groups and entities. In other words, the group of the Cybernetic Department represented by a Metacentrum group called ITI has prioritised access to a cluster called alfrid. There is a possibility to be assigned to this group by its administrator and to use the priority advantage for sources allocation.

8 Conclusion

Each second sees a huge amount of data being generated either by human collaboration or by machines which are all around us. The Age of Big Data has come and there is a need to address the challenges which come along with it.

Consequently, the problem of Big Data is widely discussed; many books and journals have been published to address its challenges, definitions and recommendations on how to deal with it. Moreover, the terms such as privacy, security and ethical problems are also considered.

Although Big Data is a frequently discussed topic in theoretical manners, there is deficiency in publications and sources dedicated to its practical usage. Nevertheless, there are some evolving technologies such as Apache Hadoop along with its ecosystem. This technology is considered as the first open-source and widely used Big Data technology, building upon a distributed filesystem and an implementation of MapReduce paradigm. Most of the other technologies dedicated to deal with Big Data are based on the Hadoop solution. Although Hadoop is often discussed as a universal Big Data platform, it cannot address all Big Data problems. There are still some available solutions which are not based on Hadoop such as Matlab, a system which provides a different approach of a cluster computation run over a shared filesystem.

Furthermore, it is convenient to mention where Big Data is stored and how much it is available. Although there are many possibilities on how to access Big Data of various types, biomedical data is mostly not much available. There is only one publicly opened biomedical Big Data database which is intended to preserve genomes. On the other hand, due to its big volume, there is considerable quality variety among publications where the database is described. Other available databases cannot compete with the volume of the genome database. However, they are still characterised by some qualities which should be considered. (consider revising sentence to make it clearer) Nevertheless, due to its domain, availability, many publications and information within the project, the database of the EEG/ERP project was evaluated as the most suitable one.

Depending on the EEG data characteristics, which were obtained by the deep database analysis, Matlab solution was evaluated as suitable technology for application on EEG data. Additionally, Matlab is widely used within the EEG project for the data processing and its programs can be deployed over a Matlab cluster. Although the Computer Science and Engineering Department does not possess all required licences, there is the possibility to use the project Metacentrum where all licences are available along with countless hardware resources.

To confirm that a Matlab cluster is a solution which can be an asset for the EEG/ERP project, a model which allows the running of a Matlab program over multiple data on a Matlab cluster was created. The model was tested by performing two use case, where EEG signals were either divided on epochs or filtered over two experiment datasets. This testing has shown that the model is functional and can be considered as an asset for the EEG/ERP project. Additionally, a few recommendations are proposed on how the project can be improved further.

List of Abbreviations

Abbreviation	Meaning
API	Application Programming Interface
ASCII	American Standard Code For Information Interchange Computing
BI	Business Intelligence
BLOB	Binary Large Object
CSV	Column Separated Values
DNA	Deoxyribonucleic Acid
DW	Data Warehouses
EC2	Elastic Computing Cloud
EEG	Electroencephalography
EMBL-EBI	European Bioinformatics Institute
ERP	Event Related Potentials
ETL	Extract Transform and Load
FTC	Federal Trade Commission
FTP	File Transfer Protocol
GUI	Graphical User Interface
HDFS	Hadoop Distribution File System
HiveQL	Hive Query Language
HTTP	Hyper Text Transfer Protocol Secure
I/O	Input/Output
INSDC	International Nucleotide Sequence Database Collaboration
JDBC	Java Database Connectivity
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
LSST	Large Synoptic Survey Telescope
MJS	Matlab Job Scheduler
NAR	Nucleic Acids Research
NCBI	National Cancer Centre for Biotechnology Information
NHS	National Health Service
NoSQL	Not only SQL
PBS	Portable Batch System
PII	Personally Identifiable Information
QR	Quick Response Codes
RAM	Random Access Memory
RDBMS	Relation Database Management System
RDD	Resilient Distributed Datasets
RFID	Radio-Frequency Identification
SAS	Statistical Analysis System
SDSS	Sloan Digital Sky Survey
SIB	Swiss Institute for Bioinformatics
SPECT	Single Photon Emission Computed Tomography
SPMD	Simple Program Multiple Data

SQL	Structured Query Language
SSH	Secure Shell
URL	Uniform Resource Locator
WMS	Warehouse Management Systems
XML	Extensible Markup Language

References

[1] **EMC Education Services.** *Data Science & Big Data Analytics.* Indianapolis : John Wiley & Sons, 2015. 978-1-118-87613-8.

[2] **Splunk Inc.** Machine Data. *Splunk.* [Online] Splunk Inc., 2005-2016. [Cited: March 8, 2016.] http://www.splunk.com/en_us/resources/machine-data.html.

[3] **Mayer-Schönberger, Viktor and Cukier, Kenneth.** *A Revolution That Will Transform How We Live, Work and Think.* New York : Houghton Mifflin Harcourt Publishing Company, 2013. 978-0-544-00269-2.

[4] **Krishnan, Krish.** *Data Warehousing in the Age of Big Data.* San Francisco : Morgan Kaufmann Publishers, 2013. 9780124059207.

[5] **Manyika, James and Chui, Michael.** *Big data: The next frontier for innovation, competition, and productivity.* s.l. : McKinsey Global Institute, 2011. 978-0983179696.

[6] **Berman, Jules J.** *Principles of Big Data.* Boston : Elsevier, 2013. 9780124045767.

[7] **Iafrate, Fernando and Front, Matter.** *From Big Data to Smart Data.* Chap : John Wiley & Sons, 2015.

[8] **Jagadish, H. V., Gehrke, Johannes and Labrinidis, Alexandros.** Big Data and Its Technical Challenges. *Communications of the ACM.* Month, 2014, Vol. 57, 7.

[9] **Hurvitz, Judith, Kaufman, Marcia and Bowles, Adrian.** *Cognitive Computing and Big Data Analytics.* Hoboken : John Wiley & Sons,, 2012. 978-1-118-89662-4.

[10] **Minelli, Michael, Chambers, Michele and Dhiraj, Ambiga.** *Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses.* s.l. : John Wiley & Sons, 2013. 9781118562260.

[11] **Borthakur, Dhruba.** HDFS Architecture Guide. *Hadoop.apache.org.* [Online] The Apache Software Foundation, 8 4 2013. [Cited: 19 April 2016.] https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html#Data+Organization.

[12] **The Apache Software Foundation.** Hadoop. *Hadoop.* [Online] The Apache Software Foundation. [Cited: 15 May 2016.] http://hadoop.apache.org/.

[13] —. Spark. *Spark.* [Online] The Apache Software Foundation. [Cited: 22 May 2016.] http://spark.apache.org/.

[14] **Zaharia, Matei; Chowdhury, Mosharaf; Das, Tahagata; Dave, Ankur; Ma, Justin; McCauley, Murphy; Franklin, Michael; Shenker, Scott; Stoica, Ion;.** *Resilient Distributed Datasets: A Fault-Tolerant.* Berkeley : University of California at Berkeley, Electrical Engineering and Computer Sciences, 2011.

[15] **MathWorks.** Makers of MATLAB and Simulink. *Mathworks.com.* [Online] [Cited: 24 April 2016.] http://www.mathworks.com/.

[16] **Holubová, Irena, a další.** *Big Data a NoSQL databaze.* Praha : Grada, 2015. 978-80-247-5466-6.

[17] **Rigden, Daniel J. , et al.** The 2016 database issue of Nucleic Acids Research. *Nucleic Acids Research.* 2016,, Vol. 44, D1–D6.

[18] **Pennisi, Elizabeth.** 1000 Genomes Project Gives New Map of Genetic Diversity. *Science.* 2010, Vol. 330, 6004.

[19] *Citizen Science: The Law and Ethics of Public Access to Medical Big Data.* **Hoffman, Sharona.** 3, Cleveland : Case Western Reserve University School of Law, 2014, Vol. 30.

[20] **The GovLab.** The Open Data Era in Health and Social Care. *GOVLAB.* [Online] May 2014. [Cited: 5 June 2016.] http://images.thegovlab.org/wordpress/wp-content/uploads/2014/10/nhs-full-report-21.pdf.

[21] **Bydžovský, Martin.** *Relational and non-relational modeling for portal of electrophysiological experiments.* Pilsen : University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering, 2014.

[22] **Řeřicha, Jan.** *Software tool for management of neuroinformatics data.* Pilsen : University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering, 2013.

List of Tables

List of Figures

University of West Bohemia

Faculty of Applied Sciences

Department of Computer Science and Engineering

User Guide of The Matlab Cluster Model Within Metacentrum

Table of Contents

1 Introduction

This User Guide is established in order to explain precise steps on how to deploy the created model over the Metacentrum environment and run the two prepared use cases over an example dataset.

2 Metacentrum

Metacentrum[1] is accessible via *ssh* and *scp* protocols; the former allows users to be connected to a GNU/Linux terminal which is run on a frontend computer while the latter is used for file transfer from the user's machine onto the shared filesystem of Metacentrum.

2.1 Metacentrum Access

Metacentrum account can be possessed by whoever is a member of either an academic or a research institution in the Czech Republic (with research objectives). Additionally, people who are members of the eduID.cz federation can even create an account via the Internet[2]. Diversely, they would have to visit one of the organisations personally which provides identity checking. The purpose of the federation is to provide means for inter-organisational identity management and access control to network services, while respecting the privacy of the users. The eduID.cz federation is operated by CESNET.

2.2 Activity Workflow

There are numerous steps which need to be followed for the process of logging into getting the final results:

- Log in – the user logs into the system and connects to one of the available GNU/Linux frontend nodes via an *ssh* client.

1. Data Preparation – the user copies data and scripts to be run over the cluster via *scp* protocol.
2. Job Preparation – in this phase, the user prepares his job to run, he assigns various parameters to the job and submits it to the scheduler via a command called *qsub*.
3. Job Running – while the job is running the user can check its state with *qstat* command.
4. Checking Results – when the job is finished, the user checks the generated outputs and job reports to determine whether the job was finished successfully.

[1] Metacentrum: https://wiki.metacentrum.cz/wiki/
[2] Online applications for the membership: http://metavo.metacentrum.cz/cs/application/

5. Downloading Results – if the job was accomplished successfully and no errors occurred, the user can download the results back into the machine from the Metacentrum file system.

2.2.1 Log in

To access machines of Metacentrum *ssh* protocol is used to connect to a frontend node terminal which runs on GNU/Linux. Typically, if one is using a MS Windows platform, it is recommended to connect via an application called PuTTy or via *ssh* command on GNU/Linux platform. Authentication and security are realised by an authentication system called Kerberos.

Since the user is logged in, the system is ready to receive user's commands.

2.2.2 Data Preparation

To transfer files between the user's machine and Metacentrum's storage protocol, *scp* protocol is used. For MS Windows platform, software called Winscp is recommended while for GNU/Linux is FilleZilla.

This phase is usually used for transferring data along with scripts to prepare the environment for the further run.

2.2.3 Job Preparation

Since all resources such as scripts, data and required libraries are prepared, estimated hardware resources for the application running are required. The resources can be required via PBS (Portable Batch System) which assigns resources and also plans, runs and manages jobs. PBS currently manages all clusters and other machines over Metacentrum.

2.2.3.1 Job Resources

Requirements for the properties and resources are formulated as strings separated by commas and are usually entered as an argument of *-l option* of *qsub* command. Typically, the user creates a script where the requirements are stored for the repetitive run. The most common used options which can be defined are depicted below in Table 1:

Command	Meaning
nodes=1:ppn=2:brno	How many nodes the job needs and their type.
	How many CPUs have to be allocated.
	Which particular nodes or their physical location.
matlab=1	Required software licences,
walltime=240m	The time queue where the job will be run in
mem=2g	Required amount of physical memory per one node

Table 1: Resources' Requirements - Common Options

There are many other options which can be defined such as the operation system, temporary storage with short access duration, location in the cluster, networking cards and the file system. Table 2 shows an example of the requirements.

- The first row requires one node with two CPUs which is located in Brno.
- The second row requires one licence of Matlab, Distributed Computing Toolbox and Distribute Computation Engine.
- The third row requires the maximum time of the job's run which is 240 minutes.
- The fourth row requires 2 GB of RAM per one node.

```
#PBS -l mem=2g
#PBS -l nodes=1:ppn=2:brno
#PBS -l matlab=1,matlab_Distrib_Computing_Toolbox=1,matlab_MATLAB_Distrib_Comp_Engine=1
#PBS -l walltime=240m
```

Table 2: Resource's Requirements - Example

2.2.4 Job Management

To enable the user to handle a job, PBS provides a set of commands where the basic ones are involved in activities listed in Table 3:

Command	Meaning
qsub	Submitting job to the queue.
qdel	Cancelation of a waiting or a running job.
qstat	Receiving information about the current node state and its properties.

Table 3: Job Management - Common Options

Additionally, there is a possibility to show a graphical overview of jobs and queues.

The commands' examples are depicted in Table 4:

- Job called Task.sh is submitted to the system.
- Jobs submitted by the user are shown in the terminal.
- The job 1300388.arien.ics.muni.cz which was earlier submitted is cancelled

```
qsub Task.sh
qstat -s | grep 'user'
qdel 11300388.arien.ics.muni.cz
```

Table 4: Commands Example

2.2.5 Checking Results

Since the job is finished, the user checks the job reports which are generated by PBS to the folder from which the job was submitted. These reports contain the job outputs and information about error occurrences.

2.2.6 Downloading Results

After checking the reports, the user typically downloads the output files generated by the job to the machine via *scp*.

3 The Model

3.1 Description

As seen in Figure 1, the model consists of a three GNU/LINUX bash scripts and two Matlab files. The bash scripts are used for the structure and data preparation and the Matlab scripts for setting up the Matlab cluster and running the user function, which represents the method that should be run over multiple data.

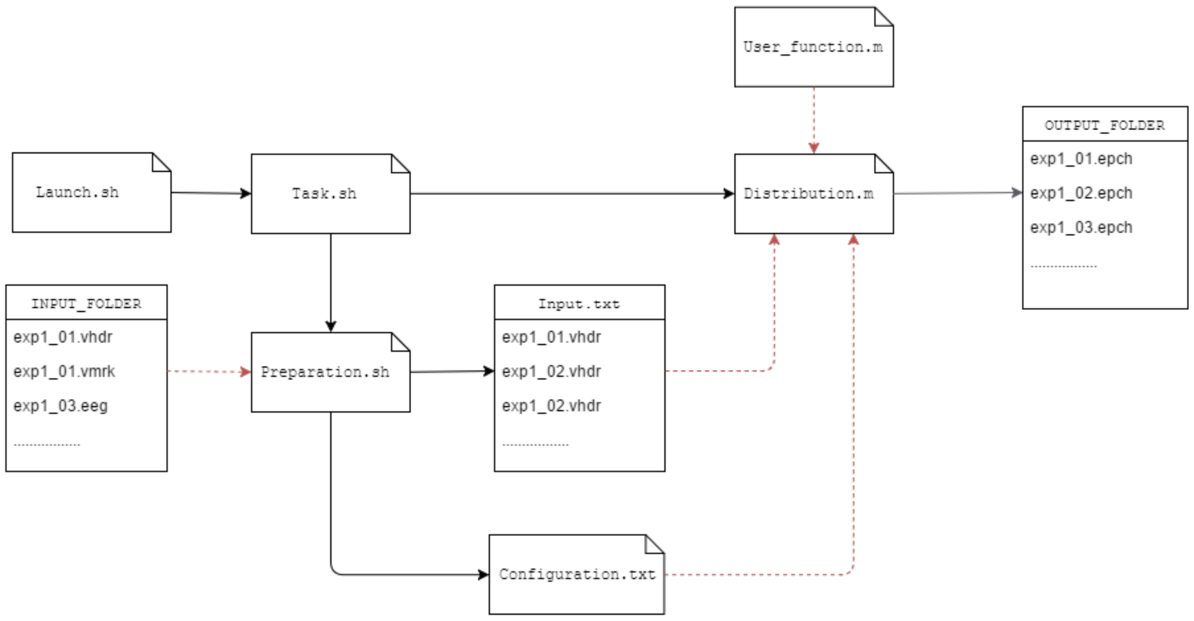


Figure 1: The Matlab Model

- Program and scripts:
    - Launch.sh:
        - clears the actual folder,
        - runs Preparation.sh,
        - submits the Task.sh to Torque.
    - Preparation.sh:
        - creates a list of header (.vhdr) files from the INPUT_FOLDER and saves it to Input.txt,
        - creates Configuration.txt, where additional data required for the computation can be found.
    - Task.sh – contains settings of Torque scheduler such as software requirements, licence requirements, performance requirements, the required location, etc.

  - User_function.m – a Matlab script which will be run over the Matlab cluster.
  - Distribution.m – represents the Matlab client session which defines the settings of the distributed environment and the definition of the decomposition on jobs and tasks. It involves:
    - the setting of a parallel pool,
    - checking licences' availability,
    - linking required libraries,
    - defining input files,
    - creating jobs/tasks which are passed to MJS scheduler,
    - reporting occurred failures.
- Data
  - INPUT_FOLDER – it contains all input data files (a dataset) in BrainVision Format which are supposed to be processed.
  - OUTPUT_FOLDER – it is the output folder where the results are gathered.
  - Input.txt – it represents a list of header (.vhdr) files which will be imported to Matlab via EEGLAB.
  - Configuration.txt – an input file which contains additional settings such as the path of INPUT_FOLDER, OUTPUT_FOLDER and the suffix of the output files.

Note: The task outputs do not have to be stored in files; they can be gathered and subsequently processed by the main Matlab program (the client session).

3.2 DVD Content

The DVD disc contains data and scripts which are vital for the program running.

- Realisation – contains four subfolders which contain source code, data, scripts and required libraries.
  - Data – contains subfolder P300 which represents a pre-prepared input set of data.
  - Model – this folder represents the whole model of distributed computation over multiple data.

    - Toolboxes – it contains the libraries such as EEGLAB along with Biosig and FileIO toolbox.
- Document – contains the folder which contains the document of this study.

3.3 Recommended Software

To avoid any problems during the setting up of the environment for running the model, there is a list of recommended software for performing further actions:

- PUTTY – an *ssh* client which has been already described.
- WinSCP – an *scp* client which has been also already described.
- Notepad++ - a simple text editor which can highlight the syntax of a source file.

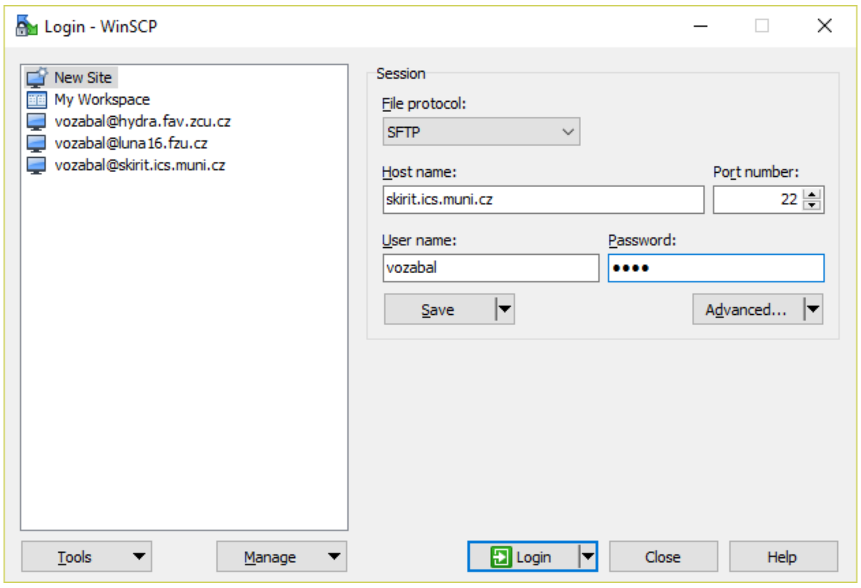
3.4 Activity Workflow

To run the model, it is necessary to:

- Copy Resources to Metacentrum and set up the script paths.
- Sign up onto a frontend terminal and run the model.

3.4.1.1 Copy Resources to Metacentrum and Set up The Script Paths

- Run WinSCP,
- Fill the following fields:
    - Host name – address of a frontend server: “*skirit.ics.muni.cz”*,
    - User-name and password – the credentials of the user’s Metacentrum, account (see Figure 2),
- Click on the Login button,
- Find the DVD root folder,
- Open Realisation folder,
- Select and copy all folders to the Metacentrum’s default folder which is currently opened (Home folder),



File protocol: SFTP
Host name: skirit.ics.muni.cz
Port number: 22
User name: vozabal
Password: ••••

Figure 2: WinSCP - Log in

- Open Model folder on the Metacentrum's panel
- Open *User_function_epochs.m* and *User_function_filter.m* in Notepad++,
  - Change the user name represented by (*REPLACE*) to yours in the *cd* command and the *addpath* command as it is depicted in Table 5 below:

Version	Text

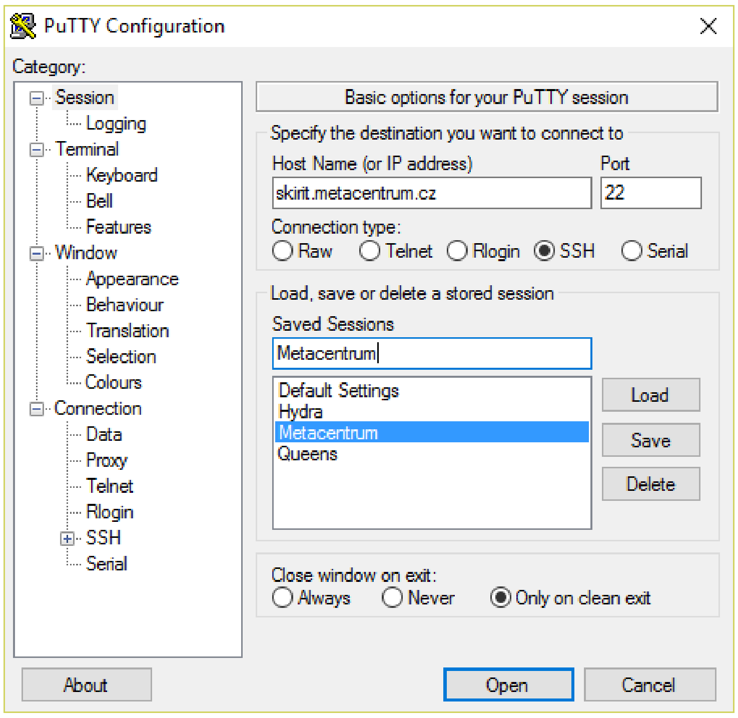
Original	cd('/storage/brno2/home/REPLACE/Model/');
	addpath('/storage/brno2/home/REPLACE/Toolboxes/eeglab/');

Edited	cd('/storage/brno2/home/vozabal/Model/');
	addpath('/storage/brno2/home/vozabal/Toolboxes/eeglab/');

Table 5: Path Settings

3.4.1.2 Sign up onto a frontend terminal

- Open PuTTY (see Figure 3),
- Fill the following fields:
  - Host name – address of a frontend server: "*skirit.ics.muni.cz*",
  - Port – fill it with value: "*22*",
- Select connection type: *SSH*,
- Click on Open,


PuTTY Configuration

Category: Session, Logging, Terminal, Keyboard, Bell, Features, Window, Appearance, Behaviour, Translation, Selection, Colours, Connection, Data, Proxy, Telnet, Rlogin, SSH, Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): skirit.metacentrum.cz

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions: Metacentrum

Default Settings
Hydra
Metacentrum
Queens

Load Save Delete

Close window on exit: Always Never [✓ Only on clean exit]

About Open Cancel

Figure 3: Putty - Log in

- Fill the user-name and the password fields,
- Enter command: `cd Model`
- Assign rights to the following scripts by command: `chmod 750 Launch.sh Preparation.sh Task.sh`
- Enter command: `./Launch` to run the model.

3.4.1.3 *Review Results*

There are some additional configurations that can be set up. Most of them are related to the activity workflow mentioned in chapter 3.4.

3.4.1.4 *Additional Settings*

There are some additional configurations that can be set up. Most of them are related to the activity workflow mentioned in chapter 3.4.

3.4.1.4.1 Changing the User Function

The previous steps are set up to run the use case within *User_function_filter.m* file. The Following steps describe how to run the other use case:

- Go to Model folder,
- Open *Distribution.m* file,

- Change the function's name which represents the second argument of createTask method depicted in Table 6,

Version	Text
Original	createTask(job, @`User_function_filter`, 0, {input{i}, output{i}});
Edited	createTask(job, @`User_function_epochs`, 0, {input{i}, output{i}});

Table 6: User function Channing

- Save the settings.

3.4.1.4.2 Checking the Job Status

If there is a need to check the job status, check the commands described in chapter 0 Job Management. The job progress can be checked by printing a file which describes the running job such as *"number.arien.ics.muni.cz.out"* where the number represents the number of the running job.

3.4.1.4.3 Changing Job Requirements

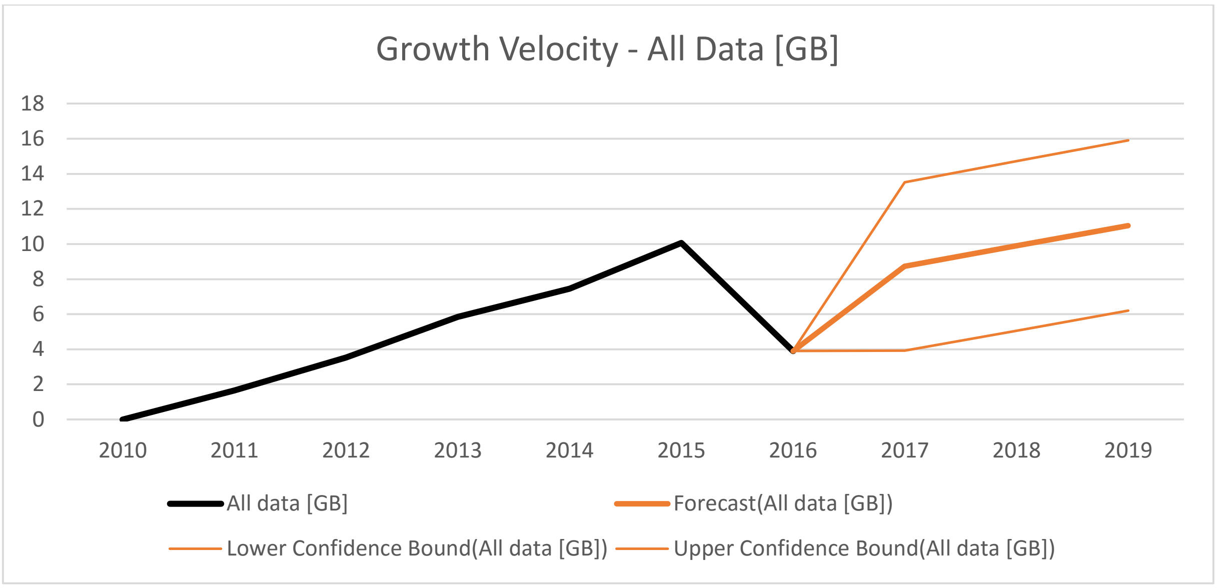
The job requirements are defined in Task file which can be modified along with the computation needs (see chapter 2.2.3 Job Preparation).

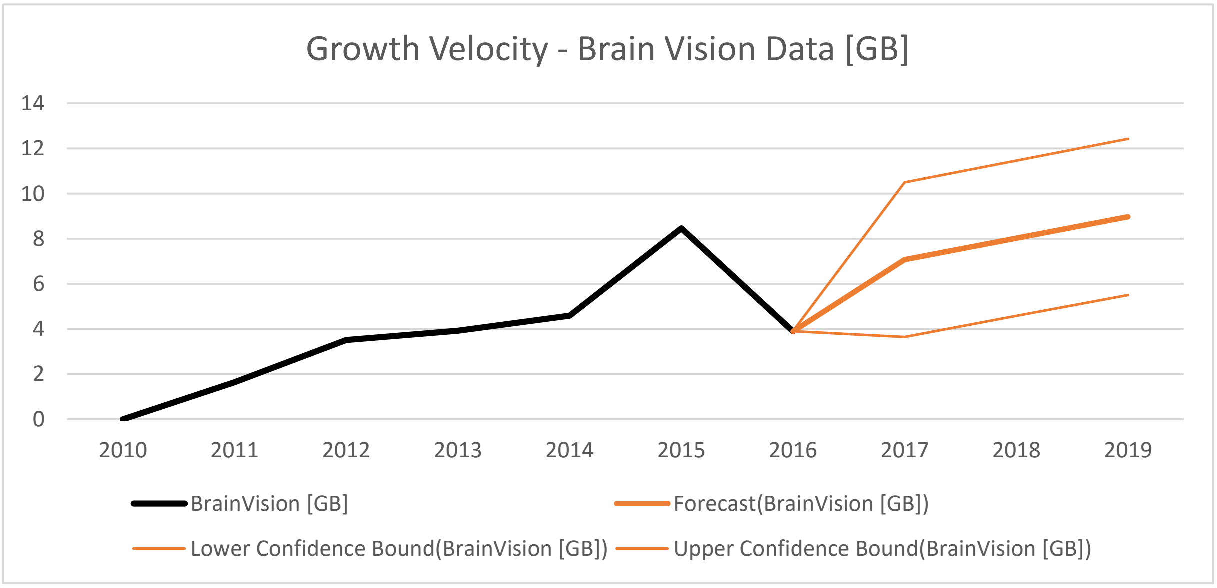
3.4.1.4.4 Error Occurrences

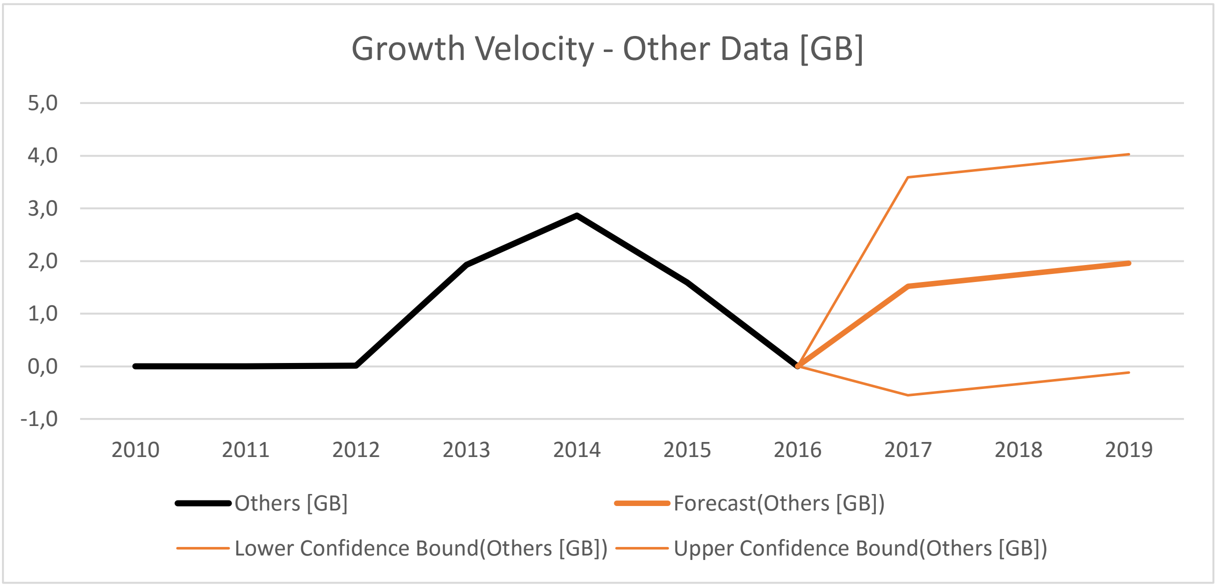
Due to Metacentrum complexity, some errors can occur. Typically, these errors are associated with the path definitions to link required libraries, licences, storage and the hardware configurations. If the job exceeds limits such as the wall time or the job, it is stopped by PBS. If it occurs, the PBS manager sends an email with the report to the user's email address.

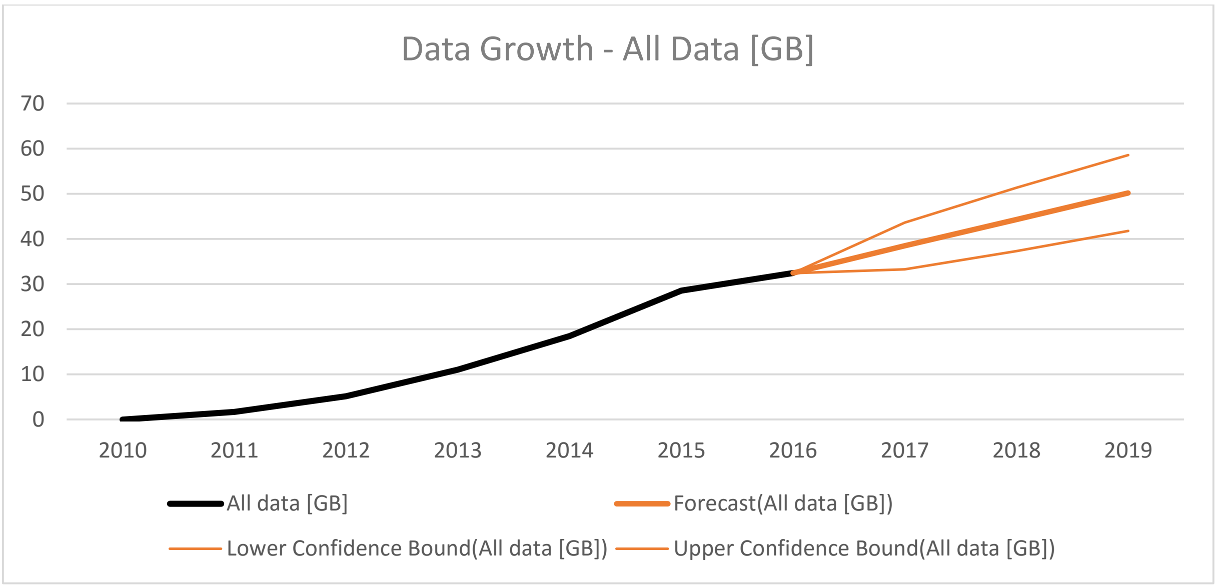
- Storage Paths – The model paths are set up on the storage of Brno: /storage/brno2/. If any errors occur, it is necessary to ensure that the path to the storage are valid ones. Check:
  - DATA_PATH in Preparation.sh
  - Path in the cd command of *User_function_filter.m* and *User_function_epochs.m*

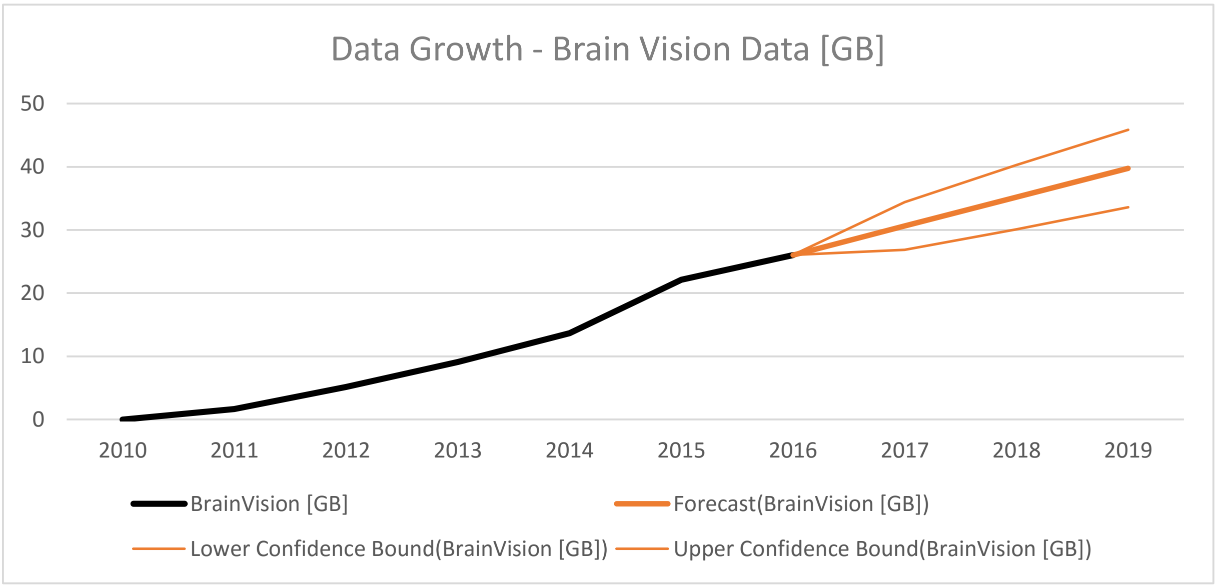
Configuration – some configuration of required resources can produce an error. Typically, the error occurs when nodes are not allocated in the same area, or when there are allocated nodes in which the configuration does not exist.

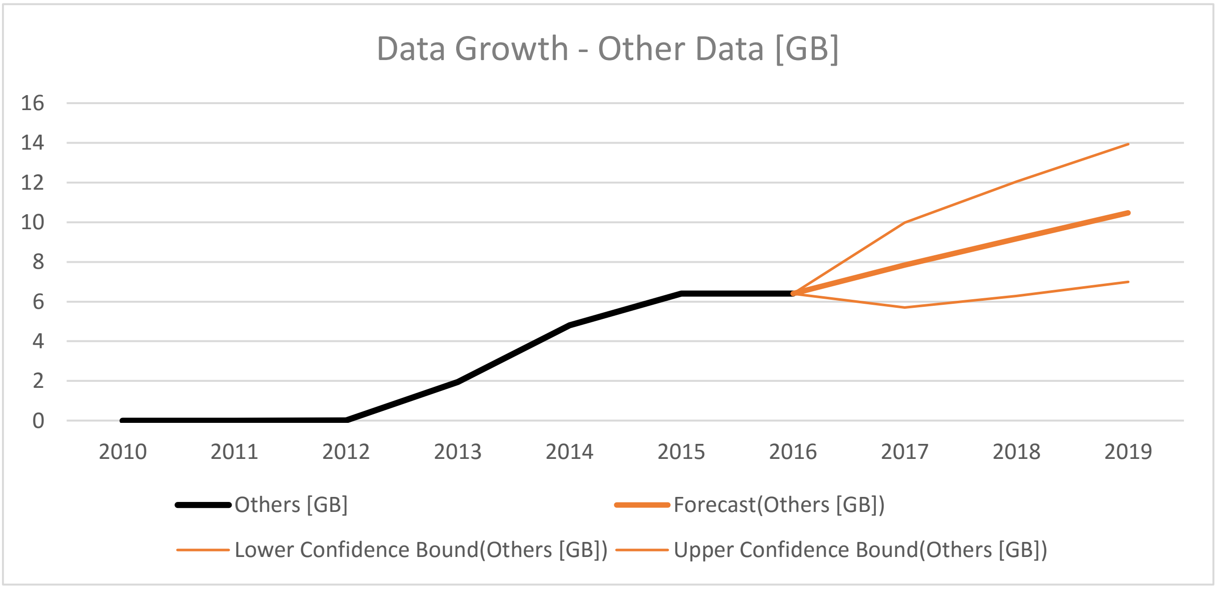

Growth Velocity - All Data [GB]
18
16
14
12
10
8
6
4
2
0
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
All data [GB]
Forecast(All data [GB])
Lower Confidence Bound(All data [GB])
Upper Confidence Bound(All data [GB])


Growth Velocity - Brain Vision Data [GB]
14
12
10
8
6
4
2
0
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
BrainVision [GB]
Forecast(BrainVision [GB])
Lower Confidence Bound(BrainVision [GB])
Upper Confidence Bound(BrainVision [GB])


Growth Velocity - Other Data [GB]
5,0
4,0
3,0
2,0
1,0
0,0
-1,0
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
Others [GB]
Forecast(Others [GB])
Lower Confidence Bound(Others [GB])
Upper Confidence Bound(Others [GB])


Data Growth - All Data [GB]
70
60
50
40
30
20
10
0
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
All data [GB]
Forecast(All data [GB])
Lower Confidence Bound(All data [GB])
Upper Confidence Bound(All data [GB])


Data Growth - Brain Vision Data [GB]
50
40
30
20
10
0
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
BrainVision [GB]
Forecast(BrainVision [GB])
Lower Confidence Bound(BrainVision [GB])
Upper Confidence Bound(BrainVision [GB])


Data Growth - Other Data [GB]
16
14
12
10
8
6
4
2
0
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
Others [GB]
Forecast(Others [GB])
Lower Confidence Bound(Others [GB])
Upper Confidence Bound(Others [GB])