

Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky



Komparace distribucí frameworku Apache Hadoop

DIPLOMOVÁ PRÁCE

Studijní program: Aplikovaná informatika

Studijní obor: Podniková informatika

Autor: Mgr. Petr Todorov

Vedoucí diplomové práce: doc. Ing. Ota Novotný, Ph.D.

Praha, duben 2020

Prohlášení

Prohlašuji, že jsem diplomovou práci Komparace distribucí frameworku Apache Hadoop vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 27. dubna 2020

.....

Petr Todorov

Poděkování

Na tomto místě bych rád poděkoval především doc. Ing. Otovi Novotnému, Ph.D., za cenné rady, konzultace a metodické vedení při tvorbě této diplomové práce. Současně bych chtěl poděkovat své rodině za podporu nejen při tvorbě této práce, ale po celou dobu studia na Vysoké škole ekonomické.

Abstrakt

Práce se zaměřuje na komparaci distribucí frameworku pro zpracování big data Apache Hadoop. Teoretická část přináší stručný vhled do oblasti big data, detailní popis frameworku a ekosystému Apache Hadoop. Práce rovněž poskytuje přehled o situaci na trhu distribucí frameworku pro zpracování big data Apache Hadoop. Praktická část práce představuje možnosti zpracování big data v reálném čase v rámci vybraných distribucí frameworku Apache Hadoop formou realizace typové úlohy příjmu a zpracování příspěvků ze sociální sítě Twitter. Na základě zjištěných informací a výsledků provedení příjmu a zpracování big data je následně provedena komparace vybraných distribucí frameworku Apache Hadoop. Informace, které práce přináší, lze využít pro rychlou orientaci na trhu distribucí frameworku Apache Hadoop a výběr distribuce frameworku Apache Hadoop vhodné pro zpracování big data v reálném čase.

Klíčová slova

Big data, Apache Hadoop, Cloudera, Hortonworks, MapR, zpracování big data v reálném čase.

JEL klasifikace

C88 Other Computer Software, L86 Information and Internet Services • Computer Software, M15 IT Management.

Abstract

This thesis focuses on comparison of the Apache Hadoop big data processing framework distributions. The theoretical part brings a brief insight into the big data area, a detailed description of the Apache Hadoop framework and ecosystem. The thesis also provides an overview of the Apache Hadoop big data processing framework distributions market situation. The practical part of the thesis introduces the possibilities of real-time big data processing within selected Apache Hadoop framework distributions in the form of performing a type Twitter social network data ingestion and processing task. Based on information found and data ingestion and processing results the comparison of selected Apache Hadoop framework distributions is accomplished. The information introduced in this thesis can be used in order to get oriented in the Apache Hadoop framework distributions market situation quickly and also for the selection of the distribution suitable for real-time big data processing.

Keywords

Big data, Apache Hadoop, Cloudera, Hortonworks, MapR, real-time big data processing.

JEL Classification

C88 Other Computer Software, L86 Information and Internet Services • Computer Software, M15 IT Management.

Obsah

Úvod	22
Motivace práce	22
Cíle práce	23
Výstupy a přínosy práce	23
Struktura práce	23
Teoretická část práce	24
Praktická část práce	24
Předpoklady a omezení práce	25
Metodologie práce	27
1 Rešerše literatury	28
1.1 Odborné publikace	28
1.2 Akademické práce	36
1.3 Analýzy trhu	37
1.4 Internetové zdroje	38
1.5 Shrnutí	39
2 Big data	40
2.1 Vymezení pojmu big data	40
2.1.1 Data, informace, znalosti	40
2.1.2 Metadata	40
2.1.3 Big data	41
2.2 Charakteristiky big data	42
2.2.1 První vymezení charakteristik big data – 3V	42
2.2.2 Standardní množina charakteristik big data – 4V	42
2.2.3 Další vývoj definice charakteristik big data – xV	43
2.3 Analýza big data	45
2.3.1 Vymezení pojmu analýza big data	45
2.3.2 Vztah analýzy big data k ostatním oblastem analýzy dat	46
2.3.3 Proces analýzy big data	49
2.4 Koncepty a technologie pro analýzu big data	51
2.4.1 Distribuované uložení dat	51
2.4.2 Distribuované zpracování dat	56
2.4.3 Machine learning	62

2.4.4	Cloud computing	63
2.5	Praktické aplikace analýzy big data	64
2.5.1	Zdroje big data.....	64
2.5.2	Praktické aplikace analýzy big data.....	65
2.6	Shrnutí	66
3	Framework Apache Hadoop	68
3.1	Vznik a řešení potřeb efektivního uložení a zpracování big data	68
3.1.1	Vznik a řešení potřeby efektivního uložení big data	68
3.1.2	Vznik a řešení potřeby efektivního zpracování big data.....	70
3.2	Vznik a rozvoj frameworku Apache Hadoop	71
3.2.1	Vznik projektu Hadoop	71
3.2.2	Rozvoj projektu Hadoop.....	74
3.2.3	Současný stav projektu Hadoop	76
3.3	Architektura a funkcionalita frameworku Hadoop	78
3.3.1	Projekt Hadoop	80
3.3.2	Ekosystém Hadoop.....	93
3.4	Praktické aplikace frameworku Apache Hadoop.....	111
3.4.1	Twitter	111
3.4.2	Spotify.....	113
3.4.3	LinkedIn	114
3.5	Shrnutí	116
4	Distribuce frameworku Apache Hadoop.....	120
4.1	Přehled dostupných distribucí frameworku Apache Hadoop	120
4.1.1	Možnosti nasazení a provozu frameworku Apache Hadoop	120
4.1.2	Trh poskytovatelů frameworku Apache Hadoop	123
4.1.3	Přehled dostupných distribucí frameworku Apache Hadoop	129
4.2	Cloudera´s Distribution Including Apache Hadoop	139
4.2.1	Vznik a rozvoj distribuce	139
4.2.2	Architektura a funkcionalita distribuce	144
4.2.3	Model poskytování distribuce	146
4.3	MapR Data Platform.....	147
4.3.1	Vznik a rozvoj distribuce	147
4.3.2	Architektura a funkcionalita distribuce	150
4.3.3	Model poskytování distribuce	155
4.4	Hortonworks Data Platform	156

4.4.1	Vznik a rozvoj distribuce	156
4.4.2	Architektura a funkcionalita distribuce	160
4.4.3	Model poskytování distribuce	162
4.5	Shrnutí	163
5	Zpracování big data v reálném čase	165
5.1	Nasazení distribuce frameworku Apache Hadoop	165
5.1.1	Proces nasazení distribuce frameworku Apache Hadoop	165
5.1.2	Nasazení distribuce Cloudera´s Distribution including Apache Hadoop ...	169
5.1.3	Nasazení distribuce MapR Data Platform	171
5.1.4	Nasazení distribuce Hortonworks Data Platform	173
5.2	Zajištění zdroje big data.....	174
5.2.1	Registrace vývojářského účtu na platformě Twitter Developer	174
5.2.2	Vytvoření Twitter aplikace	175
5.2.3	Výběr relevantního Twitter API	177
5.3	Příjem big data v reálném čase	179
5.3.1	Specifikace konkrétního procesu příjmu big data v reálném čase	179
5.3.2	Příjem big data a analýza sentimentu nad jednotnou množnou big data....	181
5.3.3	Příjem big data z jednotného zdroje do clusteru distribuce Hadoop	187
5.4	Zpracování a analýza získaných big data.....	190
5.4.1	Příprava a transformace získaných big data	190
5.4.2	Aplikace základního programovacího modelu	196
5.4.3	Aplikace window operací.....	200
5.5	Shrnutí	204
6	Komparace vybraných distribucí frameworku Apache Hadoop.....	206
6.1	Vymezení metody a kritérií komparace	206
6.2	Aplikace kritérií komparace.....	209
6.3	Vyhodnocení výsledků komparace	211
6.4	Shrnutí	212
	Závěr	214
	Použitá literatura	217
	Přílohy	I
	Příloha A: Milníky vývoje distribuce CDH.....	I
	Příloha B: Milníky vývoje distribuce MDP	III
	Příloha C: Milníky vývoje distribuce HDP	IV
	Příloha D: Verze a komponenty nasazených distribucí frameworku Apache Hadoop...VII	VII

Příloha E: Srovnání kroků průvodce instalací distribuce frameworku Apache Hadoop IX

Seznam obrázků

Obr. 2-1 Model charakteristik big data 9V – vlastnosti v doméně dat, statistiky a BI.....	44
Obr. 2-2 Životní cyklus analýzy big data	49
Obr. 3-1 Architektura MapReduce v rámci Hadoop 1	73
Obr. 3-2 Posun od architektury Hadoop 1 k Hadoop 2	75
Obr. 3-3 Architektura HDFS	82
Obr. 3-4 Grafické rozhraní HDFS – výpis obsahu adresáře HDFS.....	83
Obr. 3-5 Architektura YARN	86
Obr. 3-6 Grafické rozhraní YARN – stav využití defaultního oddílu fronty zdrojů	88
Obr. 3-7 Grafické rozhraní YARN – přehled stavu všech aplikací v clusteru Hadoop	89
Obr. 3-8 Schéma průběhu jednotlivých fází MapReduce aplikace (WordCount).....	90
Obr. 3-9 Ambari – průvodce nasazením clusteru distribuce Apache Hadoop	96
Obr. 3-10 Ambari – dashboard základních metrik monitoringu clusteru Hadoop	97
Obr. 3-11 Ambari – správa konfigurace služby se zajištěním automatického verzování	98
Obr. 3-12 Architektura distribuovaného big data NoSQL DBMS HBase.....	101
Obr. 3-13 Architektura big data datového skladu Hive.....	103
Obr. 3-14 Schéma průběhu realizace Spark aplikace	106
Obr. 3-15 Grafické rozhraní Spark History Server	107
Obr. 3-16 Grafické rozhraní frameworku Tez	110
Obr. 3-17 Vrstvy analytické platformy Twitter	112
Obr. 3-18 Využití GCP v rámci analytické platformy Spotify	114
Obr. 3-19 Nástroje využívané v rámci BDA společností LinkedIn	116
Obr. 4-1 Forrester Wave : Big Data Hadoop Solutions, Q1 2014	125
Obr. 4-2 Forrester Wave : Big Data Hadoop Distributions, Q1 2016.....	126
Obr. 4-3 Forrester Wave : Cloud Hadoop/Spark Platforms, Q1 2019.....	127
Obr. 4-4 Magic Quadrant for Data Management Solutions for Analytics – 2016, 2018... .	128
Obr. 4-5 Vysokoúrovňová architektura distribuce CDH.....	145
Obr. 4-6 Grafické rozhraní Cloudera Manager	145
Obr. 4-7 Vysokoúrovňová architektura distribuce MDP	151
Obr. 4-8 Grafické rozhraní MapR Control System.....	154
Obr. 4-9 Vysokoúrovňová architektura distribuce HDP	160
Obr. 4-10 Grafické rozhraní NiFi.....	161
Obr. 5-1 Detail Twitter Developer aplikace – přístupové klíče a tokeny.....	176
Obr. 5-2 Proces příjmu big data v reálném čase.....	180
Obr. 5-3 Údaje prvních získaných Tweetů	186
Obr. 5-4 Struktura získaných Tweetů.....	187
Obr. 5-5 Výkonnostní charakteristiky procesu příjmu dat v rámci distribucí Hadoop	189
Obr. 5-6 Výstup dotazu na metriky sentimentu u kandidátů v primárních volbách	194
Obr. 5-7 Výstup dotazu na metriky sentimentu v intervalu jedné hodiny	195
Obr. 5-8 Celkový sentiment a celkové počty Tweetů po jednotlivých kandidátech	197
Obr. 5-9 Počty Tweetů dle polarity sentimentu po jednotlivých kandidátech.....	197
Obr. 5-10 Výsledky jednotlivých kandidátů zveřejněné po tzv. volebním superúterý	198
Obr. 5-11 Výkonnostní metriky Spark úloh v rámci grafického rozhraní Spark Server UI	199
Obr. 5-12 Výkonnostní charakteristiky výpočtu základních metrik sentimentu	200

Obr. 5-13 Vývoj celkového sentimentu v intervalu jedné hodiny.....	200
Obr. 5-14 Vývoj celkového počtu Tweetů v intervalu jedné hodiny	201
Obr. 5-15 Vývoj počtu negativních Tweetů v intervalu jedné hodiny.....	201
Obr. 5-16 Vývoj počtu neutrálních Tweetů v intervalu jedné hodiny.....	202
Obr. 5-17 Vývoj počtu pozitivních Tweetů v intervalu jedné hodiny	202
Obr. 5-18 Výkonné charakteristiky výpočtu vývoje metrik sentimentu v čase	203
Obr. 6-1 Vyhodnocení výsledků komparace – přidělené body.....	212
Obr. 6-2 Vyhodnocení výsledků komparace – výsledné pořadí.....	212

Seznam tabulek

Tab. 2.1 Vývojové fáze na poli analýzy – od Analytics 1.0 k Analytics 4.0	48
Tab. 2.2 Srovnání charakteristik databázových systémů – RDBMS, NoSQL, NewSQL.....	54
Tab. 2.3 Srovnání kategorií databázových systémů z hlediska CAP teorému	55
Tab. 2.4 Srovnání charakteristik základních distribuovaných programovacích modelů....	59
Tab. 3.1 Srovnání vlastností architektury Hadoop 2 a Hadoop 3	78
Tab. 3.2 Kategorizace dat v rámci HDFS z hlediska erasure coding.....	85
Tab. 3.3 Kategorizace komponent projektu a ekosystému frameworku Apache Hadoop z hlediska fází procesu BDA	117
Tab. 4.1 Základní možnosti nasazení a provozu frameworku Apache Hadoop	121
Tab. 4.2 Kategorie poskytovatelů Hadoop řešení	124
Tab. 4.3 Přehled poskytovatelů Hadoop, kteří jsou v souladu se specifikacemi ODPi	129
Tab. 4.4 Přehled distribucí frameworku Apache Hadoop.....	131
Tab. 4.5 Srovnání komponent tvořících cloudové distribuce frameworku Hadoop	138
Tab. 5.1 Parametry clusteru pro nasazení distribuce frameworku Apache Hadoop.....	166
Tab. 5.2 Verze klíčových nasazených komponent clusteru distribuce Hadoop	168
Tab. 5.3 Přehled funkcionality Tweets API	178
Tab. 6.1 Zvolená kritéria komparace, stanovené váhy kritérií	207
Tab. 6.2 Výsledky aplikace zvolených kritérií komparace.....	210

Seznam výpisů programového kódu

Výpis 5-1 <code>kafka_producer.py</code> – kód pro příjem big data z Twitter Filtered Stream API (vlastní zpracování dle (Twitter, 2020b; Nielsen, 2011; Sarkar, 2018)).....	182
Výpis 5-2 <code>consumer.properties</code> , <code>producer.properties</code> – konfigurace Kafka Mirror Maker (vlastní zpracování)	189
Výpis 5-3 <code>spark_sentiment_static_quries.py</code> – příprava dat (vlastní zpracování dle (ASF, 2019w)).....	191
Výpis 5-4 <code>spark_sentiment_static_quries.py</code> – výpočet základních metrik sentimentu ze získaných Tweetů (vlastní zpracování dle (ASF, 2019w))	198
Výpis 5-5 <code>spark_sentiment_static_quries.py</code> – výpočet vývoje základních metrik sentimentu ze získaných Tweetů v čase (vlastní zpracování dle (ASF, 2019w))	203

Seznam zkratek

Zkratka	Anglický význam	Český ekvivalent
ACID	atomicity, consistency, isolation, durability	nedělitelnost, konzistence, izolovanost, trvalost
AI	artificial intelligence	umělá inteligence
API	application programming interface	rozhraní pro programování aplikací
ASF	Apache Software Foundation	-
AWS	Amazon Web Services	-
BASE	basically available, soft state, eventual consistence	základní dostupnost, měkký stav, konečná konzistence
BDA	big data analytics	analýza big data
BDR	backup and disaster recovery	zálohování a obnova z výpadku
BI	business intelligence	-
BSP	bulk synchronous parallel	masivně synchronní paralelizace
CAP	consistency, availability, partition tolerance	konzistence, dostupnost, tolerance výpadků sítě
CDF	Cloudera DataFlow	-
CDH	Cloudera's Distribution Including Apache Hadoop	-
CEP	complex event processing	komplexní zpracování událostí
CLDB	Container Location Database	-
CLI	command line interface	rozhraní příkazové řádky
CPU	central processing unit, processor	procesor
CRISP-DM	cross-industry standard process for data mining	standardní model procesu dobývání znalostí z databází
CSV	comma-separated values	hodnoty oddělené čárkami
DAG	directed acyclic graph	orientovaný acyklický graf
DB	database	databáze

DBMS	database management system	databázový systém
DFS	distributed file system	distribuovaný souborový systém
DL	deep learning	hluboké učení
DNS	domain name system	systém doménových jmen
DS	data science	datová věda
DSL	domain specific language	doménově specifický jazyk
DW	data warehouse	datový sklad
EBS	Elastic Block Store	-
EC	erasure coding	-
ELT	extract, load, transform	načtení, nahrání, transformace dat
EMR	Elastic MapReduce	-
EPS	event processing systems	systémy zajišťující zpracování událostí
ETL	extract, transform, load	načtení, transformace, nahrání dat
ETLT	ETL and/or ELT	ETL a/nebo ELT
EULA	end user license agreement	Licenční smlouva s koncovým uživatelem
FIFO	first in, first out	-
FQDN	fully qualified domain name	plně kvalifikované doménové jméno
FS	file system	souborový systém
FUSE	Filesystem in Userspace	-
GB	gigabyte	gigabajt
GCP	Google Cloud Platform	-
GFS	Google File System	-
GPDB	Greenplum Database	-
GPU	graphics processing unit	grafický procesor
HA	high availability	vysoká dostupnost

HaaS	Hadoop as a service	Hadoop jako služba
HDD	hard disk drive	pevný disk
HDDS	Hadoop Distributed Data Store	Distribuované datové úložiště Hadoop
HDF	Hortonworks DataFlow	-
HDFS	Hadoop distributed file system	distribuovaný souborový systém Hadoop
HDP	Hortonworks Data Platform	-
HPE	Hewlett Packard Enterprise	-
HTTP	hypertext transfer protocol	-
HW	hardware	fyzické prostředky
IaaS	infrastructure as a service	Infrastruktura jako služba
IBM	International Business Machines	-
IDH	Intel Distribution for Apache Hadoop	-
IoT	internet of things	internet věcí
IT	information technologies	informační technologie
JAR	Java archive	-
JDBC	Java Database Connectivity	standardizované API Java pro přístup k DBMS
JDK	Java Development Kit	-
JSON	JavaScript object notation	JavaScriptový objektový zápis
JVM	Java virtual machine	virtuální stroj Java
KDD	knowledge discovery in databases	dobývání znalostí z databází
LTS	long-term support	dlouhodobá podpora
MB	megabyte	megabajt
MCS	MapR Control System	grafické rozhraní pro správu distribuce MDP
MDP	MapR Data Platform	-

MEP	MapR Ecosystem Pack	-
ML	machine learning	strojové učení
MPP	massively parallel processing	masivně paralelní zpracování
MR	MapReduce	-
NDFS	Nutch distributed file system	distribuovaný souborový systém Nutch
NFS	network file system	sítový souborový systém
NIST	National Institute of Standards and Technology	Národní institut standardů a technologie
NLP	natural language processing	zpracování přirozeného jazyka
NoSQL	non SQL/not only SQL/non-relational database system	nerelační/nejen relační databázový systém
ODBC	Open Database Connectivity	standardizované API pro přístup k DBMS
ODPi	Open Data Platform initiative	-
OLAP	online analytical processing	analytické zpracování v reálném čase
OLTP	online transaction processing	zpracování transakcí v reálném čase
ORC	optimized row columnar	-
OS	operating system	operační systém
PaaS	platform as a service	platforma jako služba
QFS	Quantcast File System	-
QJM	Quorum Journal Manager	-
RAM	random-access-memory	operační paměť
RBAC	role-based access control	řízení přístupu na základě rolí
RCFile	record columnar file	-
RDBMS	relational database management system	relační databázový systém
RDD	resilient distributed dataset	-
REST	representational state transfer	-

RFID	radio frequency identification	identifikace na radiové frekvenci
RHEL	Red Hat Enterprise Linux	-
RPC	remote procedure call	vzdálené volání procedur
S3	Simple Storage Service	-
SaaS	software as a service	software jako služba
SCM	Service and Configuration Manager	-
SDK	software development kit	-
SOR	schema-on-read	DB schéma definované při čtení
SOW	schema-on-write	DB schéma definované při zápisu
SPOF	single point of failure	jediný bod selhání
SQL	structured query language	strukturovaný dotazovací jazyk
SSD	solid-state drive	-
SSH	secure shell	-
TCP	Transmission Control Protocol	-
TDH	Teradata Open Distribution for Hadoop	-
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution	-
UDP	User Datagram Protocol	-
URL	Uniform Resource Locator	jednotná adresa zdroje
vCPU	virtual CPU	virtuální CPU
WAL	write-ahead-log	-
XML	extensible markup language	-
YARN	yet another resource negotiator	-
YCSB	Yahoo! Cloud System Benchmark	-
ZKFC	ZooKeeper failover controller	-

Terminologický slovník

Termín	Význam/definice
analýza big data	„ ... proces zkoumání rozsáhlého množství dat rozličných typů (<i>big data</i>) za účelem odhalení skrytých vzorů, neznámých korelací a dalších užitečných informací.“ (Hwang, a další, 2017, str. 24)
analýza dat	„ ... proces odhalování smysluplných vzorů v rámci dat a jejich komunikace ...“ (Jain, 2017, str. 15)
big data	„ ... informační aktiva, která se vyznačují vysokým objemem, vysokou rychlostí vzniku a zpracování a/nebo vysokou různorodostí, a vyžadují nákladově efektivní, inovativní způsoby zpracování informací zajišťující rozsáhlejší vhled, podporu rozhodování a automatizaci procesů.“ (Gartner, 2019)
business intelligence	„ ... sada procesů, aplikací a technologií, jejichž cílem je účinně a účelně podporovat rozhodovací procesy ve firmě. Podporují analytické a plánovací činnosti podniků a organizací a jsou postaveny na principech multidimenzionálních pohledů na data.“ (Novotný, a další, 2005, str. 19)
cloud computing	„ ... model zajišťující na základě potřeb zákazníka prostřednictvím sítě všudypřítomný, pohodlný přístup ke sdílenému poolu konfigurovatelných výpočetních zdrojů (např. sítí, serverů, úložiště, aplikací a služeb), které lze rychle zřizovat a rušit pouze s minimálním úsilím pokud jde o správu těchto zdrojů nebo interakci s poskytovatelem služeb.“ (Mell, a další, 2011, str. 2)
cluster	„ ... množina počítačů neboli uzlů (<i>node</i>), které jsou vzájemně propojeny prostřednictvím síťové technologie ... představují integrovanou množinu zdrojů a mohou mít jediný systémový obraz zahrnující všechny uzly clusteru ...“ (Prahbu, 2011)
data	„ ... formalizovaný záznam lidského poznání pomocí symbolů (znaků), který je schopný přenosu, uchování, interpretace či zpracování.“ (Gála, a další, 2015)
distribuce Hadoop	„ ... množina komponent zahrnující projekt frameworku Apache Hadoop a další open source projekty ekosystému Hadoop, jež jsou společně zabaleny do jediného distribučního balíku, který lze snadno použít a spravovat.“ (Singh, a další, 2019, str. 18)

dobývání znalostí z databází	„ ... netriviální proces identifikace platných, nových, potenciálně užitečných a srozumitelných vzorů z dat.“ (Fayyad, a další, 1996, str. 40-41)
Hadoop	„ ... framework zajišťující distribuované zpracování rozsáhlých souborů dat napříč clustery počítačů s využitím jednoduchých programovacích modelů.“ (ASF, 2019b)
HDFS	„ ... distribuovaný souborový systém navržený pro provoz nad komoditním hardware ... je vysoce odolný proti chybám a je navržen pro nasazení na nízkonákladovém hardware ... poskytuje přístup k datům aplikací s vysokou propustností a je vhodný pro aplikace zpracovávající rozsáhlé soubory dat.“ (ASF, 2019o)
informace	„ ... pojmenování pro obsah toho, co se vymění s vnějším světem, když se mu přizpůsobujeme a působíme na něj svým přizpůsobováním.“ (Gála, a další, 2015, str. 13)
komplexní zpracování událostí	„ ... realizace komplexních operací nad kombinací různých událostí z jednoho či více prostředí a vazeb mezi těmito událostmi ...“ (Buyya, a další, 2016, str. 44)
komunikace	„Proces výměny, respektive přenosu informací“ (Gála, a další, 2015, str. 14)
MapReduce	„ ... programovací model a související implementace pro zpracování a generování rozsáhlých souborů dat.“ (Dean, a další, 2004, str. 137)
metadata	„ ... data popisující data jiná. Jejich prostřednictvím se lze na data dotázat, jsme schopni data doplňovat, konsolidovat je, vzájemně je synchronizovat a integrovat.“ (Gála, a další, 2015, str. 54)
pokročilá analýza	„ ... autonomní nebo semi-autonomní zkoumání dat nebo obsahu s použitím sofistikovaných technik a nástrojů, typicky za hranicí tradičního business intelligence.“ (Gartner, 2019)
ODPi	„ ... nezisková organizace zaměřená na zjednodušení a standardizaci ekosystému big data prostřednictvím obecných referenčních specifikací a testovacích sad.“ (ODPi, 2019a)
Ozone	„ ... škálovatelné, redundantní a distribuované objektové úložiště pro Hadoop.“ (ASF, 2019s)
proud	„ ... kontinuální tok informací a/nebo dat ...“ (Buyya, a další, 2016, str. 44)

strojové učení	„ ... Prostřednictvím výpočetních prostředků navrhuje systémy, které jsou schopny se na základě trénování učit z dat. V čase a s postupným získáváním dalších zkušeností se tyto systémy mohou učit a zlepšovat a zdokonalovat model, který lze, na základě předchozího učení, využít pro predikci odpovědí na otázky.“ (Bell, 2015, str. 2)
událost	„ ... výskyt určitého jevu v prostředí systému.“ (Buyya, a další, 2016, str. 43)
YARN	„ ... framework pro plánování úloh a správu zdrojů clusteru.“ (ASF, 2019b)
znalost	„Informace v souvislostech (kontextu) formuje znalost. Ta reprezentuje porozumění získané zkušeností nebo studiem, je srozumitelná a použitelná k řešení problému nebo k rozhodování.“ (Gála, a další, 2015, str. 14)
zpracování událostí	„ ... realizace operací nad událostmi, které jsou reportovány systémem zajišťujícím jejich příjem z určitého prostředí.“ (Buyya, a další, 2016, str. 44)

Accumulo, Ambari, Apache, Apache Software Foundation, Apex, Atlas, Avro, Beam, BigTop, Calcite, Cassandra, CouchDB, Crunch, DataFu, Drill, Druid, Falcon, Flink, Flume, Giraph, Hadoop, Hama, HAWQ, HBase, HDFS, Heron, Hive, Chukwa, Ignite, Impala, Kafka, Knox, Kudu, Kylin, Livy, Lucene, Mahout, Mesos, MiNiFi, Myriad, NiFi, Nutch, Oozie, Ozone, Parquet, Phoenix, Pig, Ranger, Samza, Sentry, Slider, Solr, Spark, Sqoop, Storm, Submarine, Superset, Tajo, Tez, Thrift, Trafodion, Whirr, Zeppelin a ZooKeeper jsou ochrannou známkou (™) nebo registrovanou obchodní známkou (®) společnosti Apache Software Foundation. Všechny ostatní názvy zmíněné v textu mohou být ochrannými známkami nebo registrovanými obchodními známkami příslušných subjektů, reprezentujících jejich vlastníky.

Úvod

Motivace práce

V průběhu posledních dekád a zejména let lze pozorovat rapidní rozvoj technologií (sítě, úložiště, výpočetní prostředky, atd.), které jsou častěji a rychleji adoptovány nejen širokou veřejností, ale ve snaze získat a/nebo udržet díky inovacím konkurenční výhodu a krok s vývojem trhu, také především ze strany firem všech velikostí napříč všemi obory. Technologie jsou v současné době chápány jako „motor“ inovací a jsou dnes dostupnější z hlediska znalostí potřebných pro jejich adopci i z finanční stránky věci a je možné je označit jako takřka všudypřítomné.

S rostoucím rozvojem technologií, jejich masivním zaváděním ve firmách a využitím ze strany široké veřejnosti dochází k masivnímu nárůstu nejen objemu dat, ale také frekvence, s níž jsou generována. Jestliže velikost fotografií prvních běžně používaných digitálních fotoaparátů dosahovala řádu kilabajtů, moderní fotoaparáty mobilních zařízení produkují fotografie o velikosti jednotek až desítek megabajtů. Objevují se rovněž nové typy zdrojů dat a výrazně se mění charakter produkovaných dat. Příkladem nového typu zdroje dat mohou být zařízení internetu věcí, která jsou schopna generovat toky dat v reálném čase prakticky neustále nebo data produkovaná uživateli v rámci sociálních sítí. Stále častěji se jedná nikoli o dříve majoritně zastoupená strukturovaná data (dokumenty, tabulky, aj.), ale především o data semi-strukturovaná nebo nestrukturovaná (například záznamy činnosti uživatelů na webu, logy, videozáznamy, apod.).

Data vznikají prakticky bez přestání a je nezbytné je nejen ukládat, evidovat a archivovat, ale především také zpracovávat a analyzovat, získávat z nich nové souvislosti, informace a znalosti. Tradiční nástroje pro příjem, zpracování, uložení a analýzu dat nicméně přestávají na efektivní zpracování dat splňujících kritéria big data vlivem daných okolností a dalších faktorů stačit, a proto je nezbytné využít pro tyto účely specializované nástroje pro příjem, zpracování a analýzu dat – frameworky pro zpracování big data.

V současné době lze jako de facto standard frameworku pro zpracování big data označit Apache Hadoop. Nasazení a provoz množiny projektů ekosystému Apache Hadoop mohou být časově náročné, spojeny s nemalými náklady a provázeny značným rizikem neúspěchu. Vznikají tudíž distribuce frameworku Apache Hadoop, které celý proces zavedení, provozu a využití Hadoop značným způsobem zefektivňují a snižují/eliminují riziko neúspěchu. V době vzniku byl Apache Hadoop zaměřen na distribuované dávkové zpracování big data. Je ale možné použít distribuce frameworku Apache Hadoop pro efektivní zpracování big data v reálném čase? Jakým způsobem bude zpracování big data s využitím frameworku Apache Hadoop v reálném čase probíhat? Kterou z dostupných distribucí frameworku Apache Hadoop pro tyto účely zvolit? Text této práce poskytuje odpovědi na všechny uvedené otázky.

Cíle práce

Práce je členěna na teoretickou a praktickou část, přičemž dané struktury odpovídá také rozložení dvou hlavních cílů, které byly pro tuto diplomovou práci stanoveny:

1. analyzovat dostupné distribuce frameworku Apache Hadoop,
2. provést komparaci vybraných distribucí frameworku Apache Hadoop z hlediska možností zpracování big data v reálném čase.

První hlavní cíl je realizován v rámci teoretické části práce, přičemž s ohledem na potřebu uvedení čtenáře do dané oblasti, je tento cíl realizován sekvenčně formou dílčích cílů, mezi které patří:

- 1.1. představit oblast big data,
- 1.2. detailně charakterizovat framework a ekosystém Apache Hadoop,
- 1.3. analyzovat dostupné distribuce frameworku Apache Hadoop.

Realizace druhého hlavního cíle je součástí praktické části práce, přičemž také tento cíl je dekomponován na dílčí cíle, mezi které patří:

- 2.1. zajistit zdroj produkující big data v reálném čase,
- 2.2. prezentovat použití vybraných distribucí frameworku Apache Hadoop pro příjem a zpracování big data z daného zdroje v reálném čase,
- 2.3. provést komparaci vybraných distribucí frameworku Apache Hadoop z hlediska možností zpracování big data v reálném čase.

Výstupy a přínosy práce

Tato diplomová práce poskytuje čtenářům základní vhled do oblasti big data, detailní informace o projektu a ekosystému frameworku pro zpracování big data Apache Hadoop a podrobný přehled o situaci na trhu distribucí frameworku Apache Hadoop. Současně práce prezentuje způsob použití hlavních distribucí frameworku Apache Hadoop pro zpracování big data v reálném čase. V neposlední řadě poskytuje informace, které lze využít pro kvalifikovaný výběr distribuce frameworku Apache Hadoop pro účely zpracování big data v reálném čase.

Struktura práce

Práce je členěna na teoretickou a praktickou část. Kromě úvodu a závěru je text tvořen šesti hlavními kapitolami, včetně nezbytné rešerše literatury, přičemž první čtyři kapitoly obsahují teoretickou část a praktická část je realizována v rámci páté a šesté kapitoly. Dále je text doplněn o seznam obrázků, seznam tabulek, seznam výpisů programového kódu, seznam zkratek a terminologický slovník, které předcházejí úvodu. Součástí je také seznam použité literatury a přílohy, přičemž tyto části následují po závěru práce.

Teoretická část práce

Teoretická část diplomové práce zahrnuje celkem pět kapitol, jejichž stručný popis následuje.

Kapitola 1 – Rešerše literatury

První kapitola zahrnuje rešerši literatury umožňující zmapovat současný stav v dané problémové oblasti, a to s primárním zaměřením na komparaci distribucí frameworku Apache Hadoop. Výsledky rešerše literatury dokumentují aktuální stav dané problémové oblasti a poskytují přehled nejvýznamnějších titulů, které se zabývaly problematikou komparace distribucí frameworku Apache Hadoop nebo s ní případně souvisí.

Kapitola 2 – Big data

V rámci této kapitoly je vymezen pojem big data a charakterizován postupný vývoj jeho chápání v čase. Kapitola dále prezentuje vztah oblasti big data k dalším oblastem analýzy, především business intelligence (BI) a dobývání znalostí z databází (KDD). Současně kapitola zahrnuje charakteristiku procesu analýzy big data a oblasti praktického využití big data a popis nejvýznamnějších technologií, které s oblastí big data úzce souvisejí.

Kapitola 3 – Framework Apache Hadoop

Tato kapitola obsahuje popis vzniku a řešení potřeb efektivního uložení a zpracování big data a s tím související vznik a rozvoj frameworku Apache Hadoop. Dále je v rámci kapitoly prezentován detailní popis architektury a funkcionality frameworku Apache Hadoop. Text je doplněn o popis možností nasazení a provozu Hadoop a přehled nejvýznamnějších poskytovatelů frameworku Apache Hadoop (na rozdíl od následující kapitoly, která poskytuje detailní analýzu trhu producentů distribucí frameworku Apache Hadoop, jež jsou využívány a poskytovány množstvím uvedených poskytovatelů Hadoop).

Kapitola 4 – Distribuce frameworku Apache Hadoop

Obsahem kapitoly je podrobná analýza trhu distribucí frameworku Apache Hadoop. V rámci dílčích podkapitol jsou podrobně prezentovány všechny distribuce Apache Hadoop, které jsou v současné době k dispozici. Součástí analýzy každého řešení je popis vzniku a rozvoje dané distribuce frameworku Apache Hadoop, charakteristika architektury daného řešení, popis funkcionality a přiblížení modelu poskytování distribuce, který se může napříč jednotlivými distribucemi značně lišit. V závěru této kapitoly je na základě získaných informací vymezena množina distribucí, jejichž nasazení a využití pro realizaci typové úlohy zpracování big data v reálném čase je předmětem praktické části práce.

Praktická část práce

Praktická část diplomové práce zahrnuje dvě kapitoly, jejichž stručný popis následuje.

Kapitola 5 – Zpracování big data v reálném čase

V rámci této kapitoly je charakterizována typová úloha zpracování big data v reálném čase reprezentovaná analýzou sentimentu pro vybrané pojmy z příspěvků sociální sítě Twitter v reálném čase, včetně postupu realizace procesu příjmu a zpracování big data ze sociální sítě Twitter v rámci vybraných distribucí frameworku Apache Hadoop.

Postupně tak kapitola dokumentuje provedené nasazení komparovaných distribucí frameworku Apache Hadoop od producentů Cloudera, MapR Technologies a Hortonworks, zajištění zdroje big data, realizaci příjmu big data v reálném čase a zpracování a analýzu získaných big data.

Kapitola 6 – Komparace vybraných distribucí frameworku Apache Hadoop

Závěrečná kapitola zahrnuje vlastní komparaci vybraných distribucí frameworku Apache Hadoop, která byla realizována na základě stanovených srovnávacích kritérií s využitím informací a zjištění získaných v rámci předchozí kapitoly. V závěru poslední kapitoly bylo provedeno souhrnné vyhodnocení komparace jednotlivých vybraných distribucí frameworku Apache Hadoop a shrnutí důležitých informací získaných v průběhu komparace jednotlivých distribucí frameworku Apache Hadoop.

Předpoklady a omezení práce

Hlavním předpokladem diplomové práce je úspěšné nasazení jednotlivých distribucí frameworku Apache Hadoop a jejich komponent, což determinuje možnost realizace praktické části práce. Klíčovým předpokladem je tudíž především dostupnost repositářů obsahujících artefakty a odpovídající dokumentaci, s jejichž využitím je možné jednotlivé distribuce nasadit a zprovoznit. Z tohoto důvodu byly pro vlastní realizaci praktické části práce zvoleny pouze distribuce společností Cloudera, MapR Technologies a Hortonworks, v jejichž případě je k dispozici open source a/nebo volně dostupná edice distribuce frameworku Apache Hadoop.

Významné omezení práce představuje způsob nasazení a provozu jednotlivých distribucí frameworku Apache Hadoop. Pro distribuce frameworku Apache Hadoop je symptomatické nasazení a provoz v distribuovaném prostředí, tedy formou clusteru vzájemně propojených řídících a pracovních strojů (uzlů). Existuje řada možností nasazení a provozu distribucí frameworku Apache Hadoop. Při plánování clusteru distribuce Apache Hadoop je nezbytné reflektovat řadu faktorů a specifikovat především následující parametry cílového prostředí¹:

¹⁾ Na tomto místě není uvedena citace, protože autor vychází z vlastních zkušeností z návrhu, nasazování a provozu distribuovaných systémů a/nebo prostředí. Nejedná se o úplný výčet všech parametrů, které je nezbytné řešit při plánování clusteru distribuce frameworku Apache Hadoop. Smyslem daného výčtu je pouze dílký ilustrace potenciální komplexnosti plánování, nasazení a provozu clusteru frameworku Apache Hadoop a odůvodnění uvedených omezení této práce.

- virtualizace zdrojů – nasazení nad:
 - fyzickými stroji,
 - virtuálními stroji,
 - kontejnery (Docker, Kubernetes, apod.),
 - případně kombinací uvedených možností,
- prostředí pro umístění a provoz clusteru:
 - on-premise,
 - v rámci veřejného cloutu (Amazon Web Services – AWS, Google Cloud Platform – GCP, Microsoft Azure, aj.),
 - v rámci privátního cloutu,
 - jiná varianta nebo kombinace,
- rozsah nasazení distribuce:
 - pseudo-distribuovaný cluster (1 uzel),
 - distribuované prostředí (2 a více uzlů, z toho minimálně 1 řídící a 1 pracovní) – v tomto případě je nutné určit počet řídících a pracovních uzlů,
- parametry jednotlivých uzlů clusteru, zejména:
 - operační paměť (RAM),
 - počet procesorů (v)CPU,
 - disková kapacita (HDD),
- množství dalších, především kvalitativních parametrů:
 - způsob řešení dostupnosti jednotlivých komponent a/nebo prostředí jako celku (vysoká dostupnost – HA, failover, replikace, zálohování, apod.),
 - úroveň zabezpečení komponent, prostředí a dat (nasazení komponent pro řízení a/nebo auditování přístupu a využití funkcí a/nebo dat, atd.),
 - cílové využití clusteru (big data lake, zpracování big data v reálném čase, kombinace, aj.),
 - další.

Současně s rostoucími požadavky na nasazení v rovině jednotlivých uvedených i dalších parametrů rostou náklady nejen na nasazení, ale především na provoz clusteru. S ohledem na možnosti autora práce bylo zvoleno nasazení nad virtuálními stroji v rámci veřejného cloutu, formou středně velkého clusteru (2 řídící uzly a 3 pracovní uzly), bez pokročilé konfigurace kvalitativních parametrů (dostupnost, zabezpečení a další parametry, které by byly v případě produkčního nasazení nezbytné), s následujícími parametry:

- řídící (master) uzel – 2x – 16 GB (gigabajt) RAM, 8 vCPU, 50 GB HDD,
- pracovní (worker) uzel – 3x – 4 GB RAM, 2 vCPU, 100 GB HDD,
- celkem – 44 GB RAM, 22 vCPU, 400 GB.

Tato kategorie clusteru umožňuje v základním rozsahu nasadit a provozovat cluster každé jednotlivé distribuce a využít jej pro realizaci úlohy zpracování big data v reálném čase. Hlavní omezení práce představuje dostupnost zdrojů, jež limituje prostředí, v jehož rámci je realizována praktická část, na stanovený rozsah zdrojů pro cluster každé jednotlivé distribuce, bez možnosti simultánního nasazení a provozu těchto distribucí.

Metodologie práce

Praktická část práce byla realizována s využitím (Švecová, a další, 2016, str. 153-216) metod vícekriteriálního hodnocení variant aplikovaných s cílem stanovení preferenčního uspořádání hodnocených variant, tj. identifikace optimální varianty. Nejprve byla zvolena kritéria hodnocení komparovaných distribucí frameworku Apache Hadoop a stanoveny váhy kritérií metodou postupného rozvrhu vah. Kritéria komparace pak byla aplikována na komparované distribuce frameworku Apache Hadoop. Použitím metody vícekriteriálního hodnocení variant stanovením hodnoty (užitku) variant s využitím přímého stanovení dílčích hodnocení jednotlivých kritérií bylo získáno preferované pořadí zkoumaných variant a následně provedeno závěrečné vyhodnocení komparovaných distribucí frameworku Apache Hadoop.

1 Rešerše literatury

Tato kapitola je zaměřena na provedení rešerše literatury umožňující do jisté míry zmapovat aktuální stav v rámci dané problémové oblasti. Rešerše literatury zahrnuje prameny použité v rámci realizace teoretické části práce, tedy publikace a prameny zabývající se oblastí big data a frameworku Apache Hadoop a dále zdroje, s jejichž využitím byla provedena praktická část práce. Zejména v kontextu praktické části práce jsou charakterizovány odborné a akademické práce s podobným zaměřením, které sice nebyly v rámci práce přímo citovány, nicméně reprezentovaly nezanedbatelné součásti aktuálního stavu poznání v dané problémové oblasti. Kapitola je členěna na dílčí podkapitoly věnované jednotlivým typům zdrojů, mezi které patří odborné publikace, akademické práce a internetové zdroje, přičemž samostatná podkapitola je věnována analýzám trhu společností Gartner a Forrester. Po přečtení kapitoly by měl čtenář získat dobrou představu o aktuálním stavu poznání v oblasti big data, frameworku a ekosystému Hadoop.

1.1 Odborné publikace

Pokud jde o odborné publikace, v rámci české literatury se tištěné publikace zaměřené na téma big data dosud vyskytovaly pouze vzácně. Výjimku představuje monografie *Big Data a NoSQL Databáze* (Holubová, a další, 2015), která vymezuje a představuje oblast big data, podrobně charakterizuje jednotlivé typy nerelačních/nejen relačních databázových systémů (NoSQL) a detailně popisuje pokročilé aspekty zpracování big data s využitím NoSQL dotazových systémů, představujících logický evoluční krok, který následoval po dlouholetém monopolu relačních databázových systémů. Daná publikace je orientována primárně na NoSQL databáze, poskytuje úvod do problematiky big data. Framework a distribuce Hadoop jsou zde zmíněny pouze okrajově.

V české literatuře existují také překlady některých zahraničních prací, nicméně autorská publikace věnovaná oblasti big data nebo distribucím Hadoop se prakticky neobjevuje. Naproti tomu v anglosaské literatuře lze nalézt řadu odborných publikací, které jsou orientovány na oblast big data, projekt a/nebo ekosystém Hadoop, distribuce Hadoop, případně kombinaci těchto témat. V rámci této práce byly pro zpracování teoretické části práce využity primárně zahraniční zdroje zaměřené na big data a projekt a/nebo ekosystém Hadoop.

Rozsáhlý vhled do oblasti big data přináší kniha *Big Data & Hadoop* (Jain, 2017), která podrobně vymezuje koncept big data, představuje okolnosti vzniku a rozvoje této oblasti, relevantní technologie, možnosti praktického využití. Dále publikace detailně charakterizuje framework Apache Hadoop a distribuci Hadoop společnosti IBM. Majoritní část knihy je věnována jednotlivým komponentám frameworku a ekosystému Apache Hadoop. V rámci každé kapitoly je daný systém představen, charakterizován jeho

možnosti, detailně popsán způsob použití a relevantní vazby na ostatní systémy ekosystému Apache Hadoop.

Tématicce big data a frameworku Apache Hadoop se věnuje publikace *Big Data : Principles and Paradigms* (Buyya, a další, 2016). Tato kniha sestává ze čtyř částí, které se podrobně věnují oblastem big data science, big data infrastrukturám a platformám, bezpečnosti a soukromí na poli big data a praktickým aplikacím big data. V rámci teoretické části práce byla využita především první část dané knihy, konkrétně kapitoly zaměřené na analýzu big data a zpracování big data v reálném čase. Většina kapitol druhé části je zaměřena na využití zdrojů a optimalizaci v rámci prostředí pro zpracování big data. Třetí část knihy se zabývá výzvami na poli soukromí v rámci sociálních sítí, zabezpečením a ochranou soukromí v kontextu big data a otázkami masivní lokalizace uživatelů a zařízení s využitím big data technologií spadajících do kategorie internetu věcí. V části praktických aplikací je obsažena detailní charakteristika a závěry z realizace frameworku pro dolování názorů veřejnosti, analýzy sentimentu nad příspěvky ze sociální sítě Twitter mapující vztah nálady a počasí, řešení pro optimalizaci využití pásma bezdrátových sítí za nejistoty a v neposlední řadě detekce událostí a anomalií.

Kniha *Hadoop Essentials* (Achari, 2015) poskytuje úvod do problematiky big data a Hadoop, přehled projektů a technologií tvořících ekosystém Hadoop, podrobný popis klíčových pilířů Hadoop v podobě komponent HDFS (distribuovaný souborový systém Hadoop), MapReduce a YARN, dále charakteristiku architektury a funkcionality komponent Hive, Pig, HBase, Sqoop, Flume, Storm, Spark. Tato publikace vymezuje roli projektu Hadoop v rámci procesu zpracování big data a podrobně charakterizuje architekturu, funkcionalitu a možnosti využití klíčových komponent projektu i ekosystému Hadoop.

Obsáhlá monografie *Expert Hadoop Administration* (Alapati, 2017) sestává z pěti oddílů, v jejichž rámci autor komplexně pojednává o projektu Hadoop a jednotlivých aspektech řízení životního cyklu clusteru Hadoop. Jako jeden z hlavních zdrojů byla tato publikace využita především v rámci čtvrté kapitoly této práce. Potenciální čtenář zde nalezne odpověď na prakticky jakoukoli otázku týkající se architektury projektu a clusterů Hadoop, nasazení testovacího a plně distribuovaného clusteru Hadoop, zpracování big data prostřednictvím jednotlivých aplikačních enginů/enginů pro zpracování big data, které jsou v rámci projektu a především ekosystému Hadoop k dispozici, dále řízení a ochrany dat a zajištění vysoké dostupnosti v rámci clusterů Hadoop, správy zdrojů, řízení výpočetních úloh a zabezpečení v distribuovaném prostředí Hadoop, monitoringu, optimalizace výkonu a řešení problémů. Vysoká přidaná hodnota publikace spočívá především v podílu informací, které je možné využít v průběhu řízení životního cyklu reálně nasazovaných a provozovaných clusterů Hadoop. Přestože se publikace může jevit jako příliš obsáhlá, na zlomku prostoru oproti oficiální dokumentaci poskytuje v nezbytném kontextu klíčové informace z oblasti praktického nasazení Hadoop a především zkušenosti a aspekty, reprezentující každodenní agendu správy clusteru Hadoop.

Kniha *Big Data Analytics* (Ankam, 2016) poskytuje naprosto vyčerpávající charakteristiku využití enginu pro zpracování big data Apache Spark v kombinaci s frameworkem Hadoop

pro efektivní realizaci procesu zpracování big data. V jednotlivých kapitolách autor podrobně vysvětuje a na praktických příkladech demonstruje v podstatě veškerou funkcionality Spark, od „základního“ rozhraní pro programování aplikací (API) a core abstrakcí, přes obě API pro zpracování proudů dat, funkcionality z oblasti strojového učení a grafových výpočetních úloh. V knize jsou rovněž popsány možnosti a způsoby integrace/využití Spark v kombinaci s data science notebooky Zeppelin, Jupyter, Hue a také data science notebookem H2O, který je zaměřen na oblast strojového učení a umělé inteligence. V knize jsou také popsány možnosti interaktivní analýzy s využitím SparkR. Daná publikace přináší rozsáhlý vhled především do možností praktického využití Apache Spark pro zpracování big data.

Základní vhled do projektu a ekosystému Hadoop poskytuje monografie *Professional Hadoop* (Antony, a další, 2016), v jejímž rámci jsou popsány jednotlivé komponenty projektu a ekosystému Hadoop v rámci kategorií úložiště, výpočetních úloh, uživatelské zkušenosti, integrace s externími systémy, zabezpečení a výpočetních úloh realizovaných v paměti (in-memory computing). Oproti jiným publikacím je v rámci této publikace věnována kapitola distribuci Apache BigTop. Daná kapitola popisuje základní koncepty dané distribuce, přičemž jako primární účel je stanoveno využití pro vývoj vlastních derivátů projektu Hadoop a dále je popsán způsob testování a nasazení. Podrobnější vysvětlení okolností vzniku a smyslu použití distribuce BigTop představuje jeden z hlavních aspektů, který knihu diferencuje od dalších podobně orientovaných publikací.

Monografie *Big Data & Hadoop* (Bhushan, 2018) je zaměřena na zpracování big data v rámci clusteru Hadoop a na rozdíl od ostatních publikací je koncipována spíše z pohledu vývojáře, který potřebuje Hadoop využít. V jednotlivých kapitolách jsou popsány podobné oblasti jako v jiných dříve uvedených publikacích, např. big data, NoSQL, Hadoop, instalace Hadoop, MapReduce, HBase, Cassandra, Pig, Hive, reálné případy zpracování big data s využitím Hadoop, administrace a správa clusteru Hadoop. V případě všech komponent je kláden důraz na vysvětlení technické podstaty (příkladem může být popis Java rozhraní HDFS vymezující možnosti interakce s HDFS prostřednictvím protokolu HTTP (hypertext transfer protocol), programovacího jazyka C, souborového systému v uživatelském prostoru či API souborového systému) a praktickou demonstraci na reálných příkladech. Pro netechnické uživatele může publikace působit poněkud „nízko úrovňově“ zaměřená nicméně technicky vybavení uživatelé praktický přístup ocení.

Pro zorientování v oblasti data science a analýzy big data lze využít knihu *Data Science & Big Data Analytics* (EMC, 2015). Kolektiv autorů představuje klíčové koncepty analýzy big data, proces a životní cyklus analýzy dat, objasňuje možnosti a roli programovacího jazyka R v rámci statistické analýzy big data, a dále charakterizuje vybrané oblasti, algoritmy a metody datové vědy a analýzy dat, mezi které patří shlukování (clustering), asociační pravidla, regrese, klasifikace, časové řady, analýza textů. V poslední části práce jsou popsány možnosti a způsoby využití Hadoop pro realizaci úloh datové vědy a analýzy dat nad nestrukturovanými big data.

Problematiku správy zdrojů a jejich využití pro distribuovanou realizaci výpočetních úloh zpracování big data v clusteru Hadoop s využitím komponenty YARN řeší publikace *YARN Essentials* (Fasale, a další, 2015). V první části jsou popsány okolnosti vzniku YARN a

úloha této komponenty v rámci clusteru Hadoop. Dále následuje popis architektury a funkcionality správce zdrojů YARN. V další kapitole autoři popisují způsob nasazení YARN v rámci clusteru Hadoop, rozdíly mezi koncepcí Hadoop 1 a Hadoop 2, v jehož rámci byla funkcionalita správy a řízení zdrojů vyčleněna z MapReduce. Část práce je věnována administraci YARN a popisu vývoje vlastní aplikace zpracování big data a jejího distribuovaného nasazení s využitím YARN. Podrobně jsou zde popsány enginy pro zpracování big data, které lze nad YARN nasadit (Samza, Storm, Spark, Tez, Giraph, HBase, Kafka). Závěrečná část knihy je věnována způsobu řešení problémů a chyb při nasazování aplikací běžících nad YARN. Kniha také uvádí alternativní řešení správy zdrojů, což ji poněkud odlišuje od ostatních publikací s analogickým zaměřením.

Podrobného průvodce architekturou, funkcionalitou a možnosti využití enginu pro zpracování big data Apache Spark představuje monografie *Mastering Apache Spark* (Frampton, 2015). V úvodní části publikace jsou představena jednotlivá API frameworku Spark, architektura, možnosti nasazení a výkonnostní aspekty provozu. V následujících kapitolách jsou postupně podrobně charakterizována jednotlivá API poskytující funkcionalitu z oblasti strojového učení, SQL (structured query language) dotazování dat, zpracování proudů dat, tvorby realizace výpočtů založených na grafech. Také v této knize je vcelku podrobně charakterizován data science notebook H2O zaměřený na podporu strojového učení a umělé inteligence. Na rozdíl od jiných publikací věnovaných enginu Spark je zde vcelku podrobně představena komerční platforma DataBricks umožňující využívat Spark formou služby.

Monografie *Signal Processing and Networking for Big Data Applications* (Han, a další, 2017) poskytuje podrobný přehled konceptů a algoritmů masivně paralelního distribuovaného zpracování dat v kontextu zpracování big data v prostředí clusteru Hadoop. První část dané práce obsahuje představení a popis role architektury clusteru Hadoop v procesu distribuovaného zpracování big data. Druhá část knihy je zaměřena na sjednocení metodologických a matematických východisek a pojednává o metodách prvního řádu, optimalizaci, sub lineárních algoritmech, algoritmu Tensor a hlubokém učení. Poslední část představuje jednotlivé metody a algoritmy v kontextu reálných big data aplikací, konkrétně analýzy big data (BDA) založené na vnímání (sensing), distribuovanou masivní optimalizaci, optimalizaci big data v komunikačních sítích a systémech chytrých sítí. Závěrečné kapitoly jsou věnovány zpracování rozsáhlých souborů dat s využitím MapReduce a možnostem masivního shromažďování dat prostřednictvím bezdrátových senzorových sítí.

Publikace *Stream Processing with Apache Flink* (Hueske, a další, 2019) se zabývá enginem pro zpracování proudů dat v reálném čase Apache Flink, představujícím alternativu k systémům zajišťujícím zpracování proudů dat formou tzv. mikro dávek (např. Spark) i k enginům pro plně real-time zpracování proudů dat (např. Storm). V úvodu je popsán a vysvětlen koncept stavového (stateful) zpracování proudů dat a základy zpracování proudů dat. Dále následuje popis architektury a funkcionality Apache Flink a návod k nasazení vývojového prostředí pro tuto komponentu. V dalších kapitolách je prezentována a demonstrována funkctionalita API Apache Flink, především základního DataStream API, operací nad časovými okny, stavové operátory a aplikace, integrace s externími systémy. Závěrečná část práce je věnována problematice distribuovaného

nasazení Flink (nad YARN nebo Kubernetes) a nasazování a provozu aplikací kontinuálního zpracování proudů dat.

Internet věcí jako klíčový zdroj big data ve formě nikdy nekončícího proudu dat v kontextu analýzy big data popisuje kniha *Big-Data Analytics for Cloud, IoT and Cognitive Computing* (Hwang, a další, 2017). V první kapitole autoři představují koncepty strojové inteligence a big data science. Další část je zaměřena technologicky a popisuje oblasti cloud computingu virtualizace a senzorových sítí a dále senzorické kompetence internetu věcí, mobilních zařízení a dalších kognitivních systémů. Následují kapitoly věnované algoritům strojového učení s učitelem a bez učitele a hlubokému učení. V závěrečné části je popsána řada praktických úloh analýzy big data využívajících dříve charakterizované algoritmy a technologie, a to z oblastí zdravotnictví a analýzy sociálních médií. Publikace je ve větší míře než dříve uvedené monografie zaměřena na využití strojového učení v rámci procesu analýzy big data.

Podobně je zaměřena také publikace *Big Data* (Chen, a další, 2014). Nejprve jsou popsány okolnosti vzniky a rozvoje éry big data a související technologie cloud computingu, internetu věcí, datových center a Hadoop. Následuje stručný popis jednotlivých etap procesu analýzy big data, od vzniku big data, přes přijímání, uložení, až po analýzu. V závěrečné části jsou stručně představeny jednotlivé oblasti využití a praktické aplikace analýzy big data. Na rozdíl od předchozí rozsáhlé monografie tato kniha poskytuje pouze základní vhled do oblasti big data a procesu analýzy big dat, bez zaměření na jednotlivé algoritmy a oblasti strojového učení.

Tématem knihy *Architecting Data-Intensive Applications* (Kumar, 2018) je rovněž oblast big data a systémů pro jejich příjem, zpracování a analýzu. Na rozdíl od jiných publikací je tato práce zaměřena více na možnosti správy a řízení big data (big data governance). Autor vymezuje oblast big data, popisuje referenční architekturu pro analýzu big data na příkladu řešení společnosti Oracle, vymezuje základní architektonické vzory řešení pro zpracování big data. Dále kniha obsahuje charakteristiku procesů shromažďování a normalizace rozsáhlých souborů dat, popis způsobů a postupů tvorby flow pro kontinuální zpracování proudů dat a vymezení specifického zpracování big data a zpracování proudů dat. Hadoop je v rámci této práce nahlížen pouze optikou možností uložení a základního dotazování big data. Enginy pro analýzu big data, které ekosystém Hadoop poskytuje, zde nejsou téměř vůbec zmíněny.

Podrobný vhled do projektu a ekosystému Hadoop poskytuje monografie *Professional Hadoop Solutions* (Lublinsky, a další, 2013), která charakterizuje projekt Hadoop, komponenty tvořící ekosystém Hadoop, vznik a rozvoj distribucí Hadoop a roli těchto kategorií v rámci procesu zpracování a analýzy big data. Následuje popis jednotlivých komponent zajišťujících/participujících na uložení dat v rámci Hadoop. Jelikož se jedná publikaci relativně staršího data (z hlediska dosavadního vývoje Hadoop), jsou celé tři kapitoly věnovány dávkovému zpracování big data prostřednictvím frameworku MapReduce a optimalizaci takových big data aplikací. Další část je věnována především zabezpečení a možnostem využití Hadoop v cloudu AWS.

Analytický engine Apache Spark je tématem knihy *Beginning Apache Spark 2* (Luu, 2018), která je zaměřena ryze technologicky a prakticky. V úvodu jsou představeny

základní koncepty a architektura Apache Spark, včetně jednotlivých knihoven, resp. API. Dále je probrána instalace a konfigurace Spark a také základní interakce s klienty Spark. Není bez zajímavosti, že v rámci této knihy je stručně představena možnost využití Spark formou služby v rámci řešení Databricks. Další kapitoly obsahují podrobný popis a prezentaci jednotlivých API, od základních abstrakcí, přes zpracování proudů dat formou mikro dávek, až po funkce knihoven Spark pro strojové učení. Tato prakticky zaměřená publikace umožní čtenáři rychle získat představu o rozsáhlých možnostech využití Apache Spark a v případě využití reálných příkladů a ukázek také nabýt zkušenosti s tvorbou Spark úloh pro zpracování big data.

Algoritmy analýzy dat, dolování dat a dobývání znalostí z databází využívané v rámci procesu analýzy big data představuje monografie *Data Analytics* (Runkler, 2016). Postupně jsou v jednotlivých částech charakterizovány postupy předzpracování a vizualizace dat a algoritmy z kategorií korelace, regrese, predikcí, klasifikace a shlukování. Publikace poskytuje základní přehled v oblasti metod dobývání znalostí z databází a charakterizuje možnosti jejich využití v procesu analýzy big data.

Praktické využití projektu a ekosystému Hadoop popisuje kniha *Learning Hadoop 2* (Turkington, a další, 2015). Nejprve je představen projekt Hadoop, včetně jednotlivých distribucí frameworku Apache Hadoop a možnosti využití Hadoop v rámci AWS. Následuje popis architektury a funkcionality HDFS s podporou ZooKeeper, YARN a MapReduce. V dalších kapitolách jsou popsány vybrané projekty ekosystému Hadoop, konkrétně Samza, Spark, Pig, Hive, Oozie, Kite, Crunch. Samostatná kapitola je věnována aspektům provozu clusteru Hadoop. V závěru jsou stručně charakterizovány jednotlivé distribuce Hadoop a také alternativy enginů pro distribuované zpracování big data. Předností daného zdroje je především řada praktických ukázek a příkladů.

Projektem Hadoop v zatím poslední majoritní verzi se zabývá monografie *Mastering Hadoop 3* (Singh, a další, 2019). V první části knihy jsou detailně popsány jednotlivé komponenty projektu Hadoop, tedy HDFS, YARN, MapReduce. Druhá sekce knihy je zaměřena na charakteristiku vybraných komponent tvořících ekosystém projektu Hadoop, především Presto, Hive, Impala, Spark, Flink, Storm/Heron, Pig, HBase, Kafka, Flume. Další část je věnována aspektům praktických úloh zpracování big data, a to včetně dávkového zpracování big data, zpracování proudů dat, strojového učení, využití Hadoop v rámci cloudu, ladění výkonu a benchmarkingu. Poslední část knihy je věnována otázkám zabezpečení clusteru Hadoop. Hlavní přínosem dané publikace je především detailní popis nové architektury, konceptů a funkcionality komponent projektu a ekosystému Hadoop.

Kromě samostatných publikací je v literatuře dostupná řada statí a příspěvků, které představují součást obsáhlejších a/nebo šířejí zaměřených publikací.

Ve sborníku ze sympózia týkajícího se návrhu a implementace operačních systémů byla poprvé zveřejněna (Dean, a další, 2004) studie distribuovaného výpočetního modelu MapReduce. V rámci daného textu je podrobně vysvětlen způsob jakým autoři řešili distribuované zpracování rozsáhlých souborů dat, zajištění odolnosti proti výpadkům v rámci daného procesu, otázky umístění (locality) dat během zpracování dat a další otázky výkonnosti a spolehlivosti daného výpočetního modelu.

Tématem příspěvku (Demchenko, a další, 2014) prezentovaného na mezinárodní konferenci o kolaborativních technologiích a systémech byly komponenty tvořících architekturu big data ekosystému. Autoři zde představili oblast a základní koncepty big data a dále definovali referenční architektonický framework ekosystému big data zahrnující komponenty a nástroje umožňující pokrýt komplexně celý proces analýzy big data. Dále byly jednotlivé komponenty tohoto frameworku blíže popsány a vysvětleny.

Koncept distribuovaného souborového systému pro uložení rozsáhlých souborů dat byl publikován v rámci (Ghemawat, a další, 2003) sborníku ze symposia týkajícího se principů operačních systémů. V textu jsou detailně popsány architektonické vlastnosti souborového systému pro distribuované uložení dat a způsoby řešení vlastního uložení dat, správy metadat napříč jmenným prostorem přesahujícím více fyzických strojů, otázky zajištění konzistence při operacích ukládání a čtení dat, ochrany proti výpadku řídících komponent a zajištění vysoké dostupnosti prostřednictvím replikace. V závěru textu jsou uvedeny výkonnostní charakteristiky této referenční implementace distribuovaného souborového systému pro uložení big data.

Otázkou zajištění konzistence v cloudových úložištích se zabývá práce (Chihoub, a další, 2014). Autoři nejprve představují tzv. CAP teorém definující, že v distribuovaných systémech lze dosáhnout pouze dvou ze tří záruk neboli garancí, mezi něž patří konzistence, dostupnost a odolnost proti výpadku. Na tomto základě jsou dále v rámci jednotlivých podkapitol vysvětleny různé modely zajištění konzistence v distribuovaných souborových systémech, které jsou orientovány na zajištění určité množiny uvedených záruk (silná, slabá, eventuální, kauzální, časová konzistence). Dále jsou představeny vybrané reálné systémy pro distribuované uložení dat, včetně předností a nedostatků. V závěru je popsán přístup k zajištění konzistence v distribuovaných souborových systémech, který je označován termínem automatizovaná auto-adaptivní konzistence.

Problematikou programovacích a výpočetních modelů pro zpracování big data (Wu, a další, 2017a) a konceptů big data úložišť (Wu, a další, 2017b) se zabývali také autoři Wu, Sakr a Zhu v příspěvcích, které jsou součástí příručky big data technologií. První zmíněný text obsahuje kategorizaci a popis klíčových modelů pro distribuované zpracování big data, tj. MapReduce (MapReduce), funkční programování (Spark, Flink), SQL dotazování (Hive, Cassandra, Spark SQL, Drill, Impala, Presto), actor model (Akka, Storm, S4), statistika a analytika (R, Mahout), datové flow (Oozie, Dryad), masivně synchronní paralelizace (Giraph, Pregel, Hama), vysokoúrovňový doménově specifický jazyk (Pig, Dryad, Trident, Green Marl, Jaql). V závěru příspěvku je uvedeno srovnání daných modelů včetně uvedení předností, nedostatků a relevance jejich použití pro typové praktické aplikace. Druhý z uvedených příspěvků je zaměřen na vymezení srovnání datových modelů a modelů úložišť pro distribuované uložení rozsáhlých souborů dat. Postupně jsou vysvětleny koncepty a reálné příklady blokových (Amazon EBS, OpenStack Cinder), souborových (sítový souborový systém (NFS), HDFS), objektových (Amazon S3 (Simple Storage Service), EMC Atoms, OpenStack Swift) úložišť a relačních i nerelačních databázových systémů. Oba zmíněné příspěvky poskytují základní vhled do modelů a možností distribuovaného uložení a zpracování big data.

Zejména v zahraniční literatuře lze nalézt množství článků publikovaných v rámci odborných časopisů, představující významnou kategorii odborných publikací.

Komparativní studii databázových systémů z kategorií SQL, NoSQL a NewSQL přinesl článek (Binani, a další, 2016). Autoři zde vymezili zmíněné kategorie databázových systémů, jejich architekturu, funkcionality a možnosti využití v oblasti big data. Daný text především v závěru představil databázové systémy NewSQL překonávající omezení tradičních relačních a nových nerelačních/nejen relačních databázových systémů a umožňující dosahovat optimální úrovně konzistence i v případě ukládání a zpracování big data, včetně zajištění funkcionality SQL dotazování a OLTP (zpracování transakcí v reálném čase) charakteristické pro relační databázové systémy.

Tématikou architektury řešení pro zpracování big data založených na frameworku Hadoop se zabýval článek (Erraissi, a další, 2017). V úvodu byly především z architektonického hlediska představeny jednotlivé distribuce frameworku Hadoop. U každé distribuce byl uveden seznam projektů frameworku Hadoop, komponent vlastních, z ekosystému Hadoop i dalších. Postupně byly popsány distribuce společností Hortonworks, Cloudera, MapR, Pivotal a IBM. Hlavním předmětem článku byla komparace uvedených distribucí frameworku Apache Hadoop z hlediska 34 převážně funkčních kritérií. V závěru autoři uvedli, že na trhu distribucí Hadoop neexistoval jednoznačný vítěz a pro zpracování big data bylo možné bez problémů využít prakticky kteroukoli ze zkoumaných distribucí.

Oblasti dobývání znalostí z databází byly věnován článek (Fayyad, a další, 1996). V úvodní části tohoto textu byla popsána role a význam procesu dobývání znalostí z databází v reálném světě a současně uvedeny některé příklady praktické aplikace metod z dané oblasti. Dále byl charakterizován vztah pojmu dolování dat a dobývání znalostí z databází a podrobně opsán proces dobývání znalostí z databází, v jehož rámci představuje dolování data jednu z realizovaných etap. Také zde byly popsány jednotlivé metody a algoritmy používané v dané oblasti, zejména klasifikace, regrese, shlukování, summarizace, modelování závislostí, detekce změn a odchylek, rozhodovací stromy, aj. V závěru článku pak byl popsán potenciál využití metod umělé inteligence v oboru dobývání znalostí z databází.

Oblast big data, jednotlivé technologie a oblasti využití analýzy big data byly popsány v článku (Sangeetha, a další, 2015). V textu byla vymezena oblast big data a charakteristiky big data, pospána architektura systémů pro uložení a zpracování big data. Autoři v článku také popsali souvislost etablování a rozvoj oblasti big data s širším technologickým rozvojem zahrnujícím například rozmach cloud computingu. V závěru byly popsány stručně popsány možnosti praktického využití analýzy big data.

Téma analýzy sentimentu nad příspěvky ze sociální sítě Twitter v reálném čase bylo věnován článek (Mane, a další, 2014). Autoři zde nejprve stručně představili sociální síť Twitter, oblast analýzy sentimentu a framework Hadoop. Následoval popis způsobu realizace analýzy sentimentu, který autoři použili, přičemž tato byla provedena formou MapReduce úlohy. V závěru byly diskutovány výsledky především z hlediska přesnosti a časové efektivity.

1.2 Akademické práce

V české literatuře lze v současné době nalézt množství diplomových prací, které se zabývaly tématikou frameworku Apache Hadoop a jeho využití při realizaci specifických úloh zpracování big data, integrace v rámci jiných systémů, atd. Většina těchto prací charakterizovala některé z distribucí frameworku Apache Hadoop. Pouze část z dané množiny byla zaměřena přímo na komparaci distribucí frameworku Apache Hadoop. Akademické práce s tímto zaměřením budou popsány v rámci následujícího textu.

Diplomová práce *Analýza nástrojů pro práci s Big Daty* (Ngo, 2018) představila v rámci teoretické části významné poznatky z oblastí big data, nerelačních databází, datových skladů, vztahu big data a business intelligence, přístupů k analýze a zpracování big data, cloud computingu, data miningu a základních úloh analýzy a zpracování big data. V textu byl charakterizován framework Apache Hadoop a další projekty z ekosystému Hadoop. Dále zde autor popsal distribuce společností Cloudera, Hortonworks, MapR. Následovala charakteristika veřejných cloudů společností Amazon, Microsoft a Google poskytujících služby umožňující automatizovaně nasadit a provozovat určitou distribuci frameworku Apache Hadoop. Stručně byla charakterizována pozice, kterou jednotlivé distribuce frameworku Apache Hadoop dosahovaly v rámci analýz trhu společnosti Gartner. Závěrečná část práce obsahovala porovnání distribucí frameworku Apache Hadoop společností Cloudera, Hortonworks a MapR, přičemž autor použil pro hodnocení celkem 46 kritérií rozdelených do 4 různých kategorií. Zvolená kritéria byla aplikována v rámci vícekriteriálního hodnocení jednotlivých distribucí metodou TOPSIS. Výslednému pořadí dominovala distribuce Hortonworks následována řešeními MapR a Cloudera.

Také diplomová práce *Nástroje pro Big Data Analytics* (Miloš, 2013) byla orientována na využití frameworku Hadoop pro zpracování big data. Teoretická část práce zahrnovala vymezení oblasti big data, vztahu big data a business intelligence. Autor dále popsal framework Hadoop a nad rámec tohoto projektu také nástroje HBase, Pig a Hive. Současně byla představena klíčová řešení na trhu, a to „tradiční“ distribuce i cloudová řešení. V rámci praktické části autor shromáždil data ze sociální sítě Twitter pomocí nástroje Flume a uložil je do distribuovaného souborového systému Hadoop. Následně provedl jejich deserializaci pomocí Hive. Pro analýzu a vizualizaci dat využil autor proprietární nástroj Tableau.

Vybrané projekty z ekosystému Hadoop popisovala také práce *Nová generace datového skladu reklamního systému Sklik.cz* (Kvasnica, 2017). Autor nejprve vymezil základní teoretické koncepty datových skladů. Následoval popis aktuálního stavu datového skladu používaného společnosti Seznam pro službu Sklik.cz. Jako alternativu ke stávajícímu řešení, které využívalo framework a určité projekty ekosystému Hadoop, představil autor podstatu, přednosti a nedostatky variant, mezi které patřily MySQL, Apache Impala, Hive, Druid, Infobright a Kylin. V rámci praktické části práce bylo provedeno nasazení varianty založené na Apache Impala a zajištěna integrace do stávajícího Hadoop prostředí a migrace na dané nové řešení datového skladu. Nové řešení bylo z hlediska výkonnostních charakteristik porovnáno s původní variantou.

Diplomová práce *Tools for big data analysis* (Macák, 2018) se rovněž zaměřila na charakteristiku projektů ekosystému big data. Byl zde uveden a popsán projekt Hadoop a ekosystém Hadoop. Autor postupoval tak, že nejprve specifikoval charakteristiky big data, požadavky na systémy zajišťující příjem, zpracování a analýzu big data a specifika architektury těchto systémů. V dalších podkapitolách byly popsány jednotlivé kategorie nástrojů pro zpracování big data, především databázové systémy všech kategorií, distribuované souborové systémy a systémy/enginy pro zpracování a analýzu big data. Hlavní výstup práce byl reprezentován diagramem pro výběr nástrojů relevantních pro konkrétní specifické případy užití. Pozitivním rysem dané práce byla skutečnost, že autor se neomezil pouze na vymezení projektů frameworku a ekosystému big data, ale v případě většiny kategorií nástrojů pro zpracování big data představil relevantní alternativy pokrývající danou oblast funkcionality.

Na podobnou problémovou oblast byla zaměřena také práce *Apache Hadoop jako analytická platforma* (Brotánek, 2017). V textu byly charakterizovány vlastnosti big data a výpočetní modely využívané pro zpracování big data. Dále byla podrobně popsána architektura a funkcionalita jednotlivých komponent frameworku Apache Hadoop. Následně autor představil vybrané projekty ekosystému Hadoop. V práci byly popsány distribuce Hadoop: Cloudera's Distribution Including Apache Hadoop (CDH), Hortonworks Data Platform (HDP) a MapR Data Platform (MDP). Poté následoval popis zásad práce s clustery frameworku Hadoop. V závěru praktické části byly podrobně vymezeny funkce vybraných komponent ekosystému Hadoop pro analýzu big data.

V neposlední řadě se projektem a ekosystémem Hadoop zabývala práce *Integrace Big Data a datového skladu* (Kiška, 2017). Autor nejprve podrobně popsal oblast big data, včetně možností praktické aplikace analýzy big data. Dále byly charakterizovány oblasti datových skladů a business intelligence. Následně byl detailně popsán framework a ekosystém Hadoop. V praktické části práce byla provedena integrace existujícího datového skladu s big data platformou Hadoop formou nasazení clusteru Hadoop a vytvoření workflow pro přenos dat z existujících datových skladů do prostředí Hadoop, včetně evidence metadat o realizovaných přenosech dat.

1.3 Analýzy trhu

Mezi literární prameny a zdroje spadající do oblasti big data a projektu a ekosystému Hadoop patří také analýzy trhu, které pravidelně publikují přední světové analytické a konzultační společnosti. V rámci této práce byly využity především analýzy trhu společností Gartner a Forrester. Společnost Gartner vydává tržní analýzy nesoucí označení Magic Quadrant, zatímco druhá jmenovaná společnost publikuje analýzy trhu pod názvem The Forrester Wave.

Přestože analýza trhu *Magic Quadrant for Data Warehouse and Database Management Solutions for Analytics* (Nemschoff, 2016) byla zaměřena na oblast řešení analytických datových skladů a databázových systémů, součástí hodnocení daného trhu byly také společnosti Cloudera, MapR Technologies a Hortonworks. Shodně byla řešení těchto společností, zaměřených primárně na produkci vlastní distribuce Hadoop, umístěna do

kvadrantu nesoucího označení „vizonáři“. Za lídry v dané oblasti byly označeny společnosti Oracle, Teradara, Microsoft, IBM a SAP.

Na řešení systémů správy dat podporujících oblast analýzy byla orientována analýza trhu nesoucí název *Magic Quadrant for Data Management Solutions for Analytics* (Ronthal, a další, 2018). Také v tomto případě byly součástí hodnocení řešení společností Cloudera, Hortonworks a MapR Technologies. Uvedení producenti distribucí frameworku Hadoop byly shodně hodnoceni jako „specializovaní hráči“ (niche player), přičemž nejlépe byla hodnocena společnost Cloudera. U všech uvedených společností přitom bylo autory zdůrazněno, že využití open source může determinovat i negativní aspekty, jako např. zaostávání za aktuálním vývojem z hlediska poskytovaných verzí jednotlivých komponent distribuce.

V roce 2014 byla publikována analýza trhu *Big Data Hadoop Solutions* (Gaultieri, a další, 2014). Tato studie identifikovala na trhu 6 typů distribucí Hadoop, od open source projektů a producentů zaměřených výhradně na distribuci Hadoop, až po poskytovatele big data řešení a tvůrce doplňkových projektů nad rámec open source ekosystému Hadoop. V rámci analýzy byla komparována řešení společností Amazon, Cloudera, Hortonworks, IBM, Intel, MapR Technologies, Microsoft, Pivotal software a Teradata. Majoritní část těchto řešení byla označena za lídry trhu Hadoop big data řešení vyznačující se silnou nabídkou a silnou strategií, a tedy silnou pozicí na daném trhu. Jako centrální prvek architektury Hadoop big data řešení bylo v rámci textu označeno data lake pro ukládání big data.

Tržní analýza *Big Data Hadoop Cloud Solutions* (Gaultieri, a další, 2016a) se zaměřila na poskytovatele cloudových řešení Hadoop big data. V rámci dané studie byla hodnocena řešení společností Altiscale, Amazon, Google, IBM, Microsoft, Oracle, Qubole a Rackspace. Z výčtu je patrné, že v daném okamžiku se producenti distribucí Hadoop zaměřovali spíše na on-premise nasazení, než na prostředí cloutu. Z analýzy rovněž vyplynulo, že ředitelé organizací využívající on-premise řešení Hadoop hledají alternativy především s cílem využít předností cloutu před on-premise nasazením, např. nižších pořizovacích nákladů, vyšší úrovně flexibility, atd.

V roce 2016 byla zveřejněna analýza trhu *Big Data Hadoop Distributions* (Gaultieri, a další, 2016b), jejímž předmětem byly výhradně distribuce Hadoop. V rámci dané analýzy byla hodnocena řešení společností Cloudera, Hortonworks, IBM, MapR Technologies a Pivotal Software. Za lídry trhu byly v tomto případě označeny čtyři z uvedených společností s výjimkou Pivotal Software. Jednoznačným vítězem hodnocení byla společnost Cloudera reprezentující průkopníka v oblasti distribucí Hadoop. Výsledky analýzy mimo jiné potvrdily kulminující konkurenci na trhu distribucí Hadoop.

1.4 Internetové zdroje

Nedílnou součást zdrojů z oblasti big data i projektu a ekosystému Hadoop představují zdroje dostupné na internetu. V první řadě se jedná zdroje oficiální dokumentace, která je zpravidla dostupná pouze online. Tento typ zdrojů byl využit především v rámci kapitol

věnovaných frameworku Hadoop a jednotlivým distribucím Hadoop a představoval významnou část množiny zdrojů využitých v rámci této práce.

Nadace Apache Software, která zastřešuje řadu open source projektů nejen z oblasti big data, poskytuje na stránkách jednotlivých projektů vždy podrobnou dokumentaci, v některých případech spíše technického rázu, v jiných i uživatelského charakteru.

Společnost Cloudera, producent distribuce CDH poskytuje (Cloudera, 2019b) obsáhlou dokumentaci online a rovněž řadu zajímavých článků dostupných na blogu společnosti a oficiálních tiskových prohlášení (Cloudera, 2018c). Také společnost Hortonworks zajišťovala pro uživatelskou komunitu obsáhlou online dokumentaci (Hortonworks, 2019e), včetně návodů a uživatelsko-technologického blogu. Po sloučení společností Cloudera a Hortonworks byla naprostá většina této dokumentace přesměrována pod doménu cloudera.com. V případě některých starších zdrojů bylo nutné využít funkcionality webu web.archive.org, který zajišťuje pravidelné vytváření a evidenci snímků vybraných internetových stránek. Díky dané službě bylo možné i zpětně dohledat některé stránky, které již nebyly na původních adresách k dispozici a dokonce i stránky, které jsou stále dostupné pod doménou Hortonworks.com.

Oficiální dokumentaci a kvalitní technologický blog poskytuje také společnost MapR Technologies (MapR, 2018c). Přestože byla tato společnost předmětem akvizice ze strany HPE, majoritní většina internetových stránek společnosti, včetně dokumentace a blogu je zatím stále k dispozici pod původní doménu mapr.com. Pouze stránky věnované podpoře komunity a jednotlivých produktů MapR byly přesměrovány pod doménu HPE.

Dále byly v rámci práce využity vybrané příspěvky z blogů a/nebo oficiální dokumentace poskytovatelů cloudových platform, např. Alibaba Cloud (Alibaba, 2019), AWS (Amazon, 2019a), GCP (Google, 2019b) nebo Azure (Microsoft, 2019b). Současně je k dispozici řada příspěvků z technologických blogů společností, které Hadoop v praxi využívají. V rámci kapitoly obsahující praktickou úlohu zpracování big data byla využita oficiální dokumentace API (Twitter, 2020b), jehož prostřednictvím bylo možné získat big data v reálném čase. V průběhu realizace jednotlivých kapitol bylo využito množství dalších internetových zdrojů.

1.5 Shrnutí

Oblast big data je z hlediska aktuálního stavu poznání podrobně zmapována především v rámci zahraniční odborné literatury, převážně anglosaské provenience. V české odborné literatuře dané téma není opomíjeno, nicméně obsáhlejší monografie zabývající se oblastí big data a/nebo frameworkem a ekosystémem Hadoop představují spíše výjimku. V kontrastu k tomuto stavu bylo nicméně v české literatuře identifikováno množství kvalitních diplomových prací, které se oblastí big data a frameworkem a ekosystémem Hadoop včetně možností praktického využití pro analýzu big data zabývaly.

2 Big data

Tato kapitola obsahuje charakteristiku základních teoretických poznatků z problémové oblasti big data, jejichž prezentace je nezbytná pro sjednocení terminologie a teoretických poznatků představujících východisko pro realizaci dalších částí této práce. V rámci kapitoly jsou nejprve vymezeny obecné pojmy, mezi které patří data, informace a znalosti, dále je podrobně vymezen termín big data a doplněn detailní popis množiny charakteristik big data tak, jak se postupně vyvíjela. Kapitola dále vymezuje vztah big data k dalším oblastem analýzy dat, popisuje proces analýzy big data, představuje praktické aplikace analýzy big data a stručně charakterizuje technologie a nástroje pro analýzu big data.

2.1 Vymezení pojmu big data

2.1.1 Data, informace, znalosti

Před vymezením pojmu big data je nejprve nezbytné uvést definici pojmu data a dalších souvisejících termínů, aby bylo následně možné na základě definice kategorie big data snadno pochopit rozdíl oproti „tradičním“ datům.

Data jsou vymezena jako (Gála, a další, 2015, str. 14) „... formalizovaný záznam lidského poznání pomocí symbolů (znaků), který je schopný přenosu, uchování, interpretace či zpracování.“

Jestliže data vstupují do procesu výměny, pak je možné prostřednictvím dat předávat a získávat určité informace, definované jako (Gála, a další, 2015, str. 13) „... pojmenování pro obsah toho, co se vymění s vnějším světem, když se mu přizpůsobujeme a působíme na něj svým přizpůsobováním.“ Proces výměny, respektive přenosu informací je označován (Gála, a další, 2015, str. 14) termínem komunikace.

Informace pak determinují možnost získávat znalosti, kdy jedna z definic znalostí uvádí, že (Gála, a další, 2015, str. 14) „Informace v souvislostech (kontextu) formuje znalost. Ta reprezentuje porozumění získané zkušeností nebo studiem, je srozumitelná a použitelná k řešení problému nebo k rozhodování.“

2.1.2 Metadata

S daty úzce souvisí pojem metadata. Jedná se o (Gála, a další, 2015, str. 54) „... data popisující data jiná. Jejich prostřednictvím se lze na data dotázat, jsme schopni data doplňovat, konsolidovat je, vzájemně je synchronizovat a integrovat.“ Příkladem metadat mohou být v případě souborů fotografií údaje o datu a čase pořízení fotografie, použitých hodnotách nastavení fotoaparátu, atd. V závislosti na metadatech neboli schopnosti popsat organizaci, respektive strukturu dat, je možné data kategorizovat na (Gála, a další, 2015, str. 53-54):

- strukturovaná data – pomocí metadat je možné exaktně vymezit strukturu dat a určit význam jednotlivých částí dat; příkladem mohou být záznamy složené z konkrétních datových položek, například objednávka, faktura, atd.,
- semi-strukturovaná data – s využitím metadat je možné strukturu dat charakterizovat a určit význam jednotlivých částí dat pouze částečně; příkladem může být email či jiné entity, v jejichž případě je část dat strukturovaná a část nikoli,
- nestrukturovaná data – prostřednictvím metadat není možné popsat strukturu dat ani určit význam jednotlivých částí dat; příkladem mohou být audio nebo video záznamy, grafika, aj.

Schopnost popsat organizaci, respektive strukturu dat prostřednictvím metadat představuje (Gála, a další, 2015, str. 53) jeden z faktorů možné míry automatizace zpracování dat prostřednictvím počítače. Vlastnosti dat týkající se metadat současně představují jednu z charakteristik big data.

2.1.3 Big data

Přestože data splňující některé z charakteristik big data byla dostupná i dříve a pojem big data² se v literatuře již objevil, je použití tohoto termínu v „moderním významu“ nad rámec pouhé velikosti/objemu dat, připisováno současnému řediteli marketingového výzkumu společnosti O'Reilly Media, Rogeru Mougalasovi. Tento v roce 2005 big data definoval jako (Sangeetha, a další, 2015, str. 3269) „... rozsáhlou množinu dat, které je téměř nemožné spravovat a zpracovávat prostřednictvím tradičních nástrojů business intelligence.“

V obecné rovině představují big data (Chen, a další, 2014, str. 2) „... soubory dat, jež není možné v rámci přijatelného času vnímat, přijímat, spravovat ani zpracovávat prostřednictvím tradičních IT a softwarových/hardwarových nástrojů.“

Uvedená definice kromě zobecnění technologií a/nebo nástrojů použitých pro příjem, zpracování, uložení a analýzu big data (nikoli databázové nebo business intelligence nástroje, ale pouze informační technologie – IT a softwarové/hardwarové nástroje) zdůrazňuje také faktor času. Definice implikuje, že určitá data z množiny big data je potenciálně možné prostřednictvím tradičních nástrojů zpracovat, ale nikoli v rámci přijatelného časového horizontu.

Společnost Gartner definuje big data jako (Gartner, 2019) „... informační aktiva, která se vyznačují vysokým objemem, vysokou rychlosťí vzniku a zpracování a/nebo vysokou různorodostí, a vyžadují nákladově efektivní, inovativní způsoby zpracování informací zajišťující rozsáhlejší vhled, podporu rozhodování a automatizaci procesů.“

²) V rámci české oborové literatury neexistuje shoda ohledně používání ani překladu termínu big data. Lze se tak setkat s přístupem, kdy je používán český překlad „velká data“ či „veledata“ a rovněž s praxí, kdy je používán původní nepřeložený tvar, a to „Big Data“, „Big data“ nebo „big data“. V rámci zahraničních zdrojů se nejčastěji setkáme s tvarem „Big Data“. V rámci této práce je v souladu s běžnou praxí v oboru informačních technologií, kdy množství zahraničních termínů není překládáno, nadále používán pouze původní nepřeložený tvar, a to ve formátu „big data“.

Definice pojmu big data se postupně vyvíjela a nadále vyvíjí, přičemž jednotná všeobecně akceptovaná definice tohoto termínu dosud neexistuje, což dokládá například studie amerického Národního institutu standardů a technologie (NIST), která (Grady, a další, 2015, str. 1-19) identifikuje více než dvacet různých definic big data.

Z vybraných uvedených definic big data je patrné, že velikost/objem dat není jediným ani hlavním kritériem zařazení dat do kategorie big data. Z tohoto důvodu je nezbytné uvedené definice doplnit o popis množiny vlastností charakteristických pro big data. Společně s vývojem definice pojmu big data přitom docházelo také k evoluci množiny charakteristik big data.

2.2 Charakteristiky big data

2.2.1 První vymezení charakteristik big data – 3V

Historicky první vymezení vlastností/charakteristik big data publikoval v roce 2001 analytik Doug Laney, který ve svém reportu (Laney, 2001, str. 1-3) na základě reflexe vývoje především v oblasti e-commerce predikoval budoucí rapidní rozvoj v produkci dat a potřebu řešení jejich efektivní správy a zpracování a upozornil na tři klíčové vlastnosti dat, které budou muset podniky v procesu zpracování a analýzy řešit a překonávat limity stávajících nástrojů pro zpracování a analýzu dat. Jedná se o následující charakteristiky (Laney, 2001, str. 1-3):

- objem dat (volume),
- rychlosť vzniku a zpracování dat (velocity),
- různorodost dat (variety).

Autor tak vymezil fundamentální množinu vlastností big data, která je dodnes všeobecně akceptována, což ilustruje například výše zmíněná definice pojmu big data společnosti Gartner, která akcentuje právě tyto charakteristiky big data. Pro danou množinu vlastností big data je v literatuře používáno (Jain, 2017, str. 3) označení model 3V charakteristik big data.

Pro big data je charakteristický (Laney, 2001, str. 1-3) kontinuálně rostoucí objem (volume) generovaných dat, která je nutné přijímat, zpracovávat, ukládat, analyzovat a dále spravovat, dále zvyšující se rychlosť (velocity) vzniku dat determinující potřebu zrychlení celého procesu zpracování dat. V neposlední řadě jsou big data symptomatická (Laney, 2001, str. 1-3) zvyšující se různorodost (variety) dat zahrnující růst počtu prvků tvořících množinu typů zpracovávaných dat (struktuovaná a častěji semi-struktuovaná a nestrukturovaná data) a také zdrojů generujících data, která stále častěji vykazují některou z charakteristik big data.

2.2.2 Standardní množina charakteristik big data – 4V

Společnost International Business Machines (IBM) později (Buyya, a další, 2016, str. 8) rozšířila základní množinu vlastností big data o další charakteristiku, a to věrohodnost (veracity), která implikuje, že big data mohou být charakteristická nejednoznačnou věrohodností, a to napříč celým procesem zpracování big data a současně z různých

hledisek, mezi která patří kvalita dat (úplnost, přesnost, duplicita/redundance, atd.), důvěryhodnost zdroje/původce dat, nejistoty spojené s přesností a/nebo správností dat, a další.

Společně jsou charakteristiky objemu (volume), rychlosti a zpracování (velocity), různorodosti (variety) a věrohodnosti (veracity) dat označovány jako (Goyal, a další, 2018, str. 245) model 4V charakteristik big data. Jestliže bylo u předchozí množiny vlastností big data 3V uvedeno, že se jedná o široce akceptovanou základní množinu charakteristik big data, pak model 4V charakteristik je v této práci označován za standardní za účelem odlišení od dalších modelů, jež z množiny 4V vlastností big data vycházejí.

2.2.3 Další vývoj definice charakteristik big data – xV

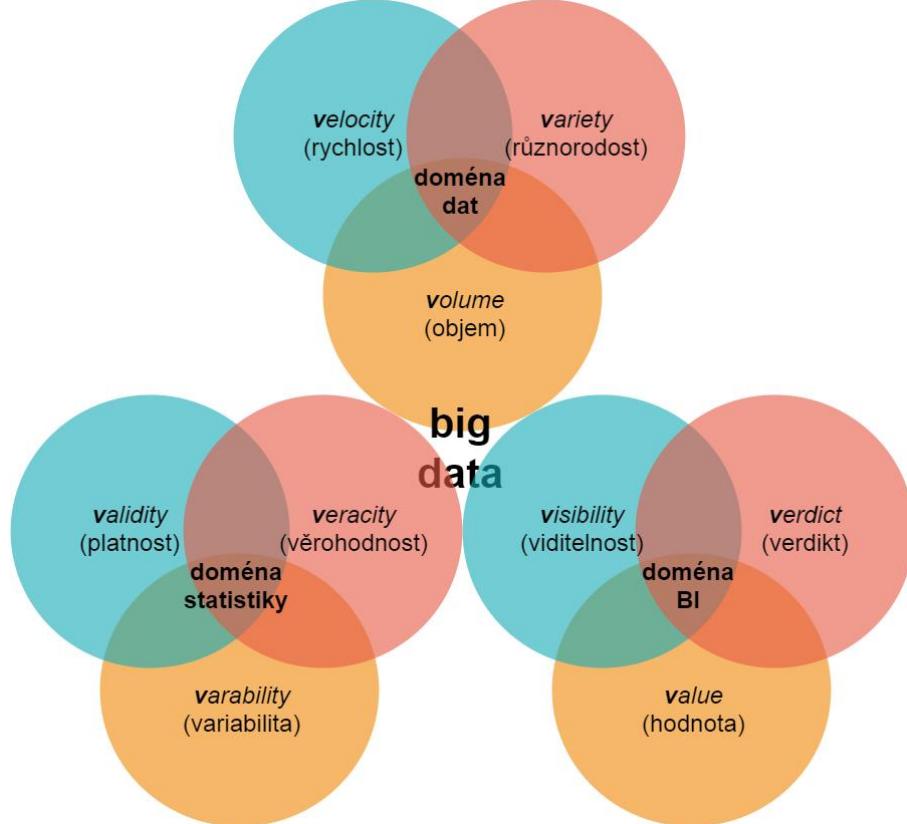
Postupně se objevily (Buyya, a další, 2016, str. 8-15) další modely charakteristik big data, pro které je charakteristické, že k množině vlastností 4V přidávají různý počet dalších charakteristik, a proto jsou v rámci této práce zařazeny do kategorie modelů charakteristik big dat xV.

Společnost Microsoft představila (Buyya, a další, 2016, str. 8-9) vlastní model charakteristik big data 6V, který množinu 4V doplňuje o hodnotu (value) a viditelnost (visibility). Dimenze hodnoty (value) přitom vyjadřuje myšlenku, že (Jain, 2017, str. 4) jakákoli organizace by měla zdroje na realizaci procesu zpracování big data alokovat pouze v případě, že big data a vhledy získané jejich prostřednictvím přinesou určitou měřitelnou hodnotu pro podnik (business value). Charakteristika viditelnosti (visibility) pak vyzdvihuje (Buyya, a další, 2016, str. 9) význam dostupnosti úplného obrazu dat pro přijetí informovaného rozhodnutí.

V roce 2014 byl publikován model 5V (Demchenko, a další, 2014, str. 106), zahrnující charakteristiky big data, mezi které patří objem (volume), různorodost (variety), rychlosť a zpracování (velocity), hodnota (value) a věrohodnost (veracity) dat. Oproti předchozímu modelu big data charakteristik pak autoři modelu 5V ještě navíc jako doplňkové vlastnosti uvádějí (Demchenko, a další, 2014, str. 106) proměnlivost (variability) a provázání (linkage) dat. Příčinu těchto charakteristik přitom autoři spatřují ve skutečnosti, že (Demchenko, a další, 2014, str. 106) v průběhu celého životního cyklu dat dochází v důsledku transformace a/nebo zpracování ke kontinuálním změnám obsahu a/nebo stavu dat, což determinuje jejich značnou variabilitu a potřebu provázání původních dat na nová transformovaná a/nebo zpracovaná data.

Dále byl představen model charakteristik big data 3V² neboli 9V, který na rozdíl od předchozích množin vlastností big data (Buyya, a další, 2016, str. 9) není omezen primárně na aspekt dat, ale naopak na základě reflexe různých typů definic big data vymezuje charakteristiky big data ve vztahu k celému procesu analýzy big data (BDA). Tento model charakteristik big data kategorizuje do tří domén, mezi které patří (Buyya, a další, 2016) doména dat (data domain), doména business intelligence (business intelligence domain) a doména statistiky (statistics domain), jež jsou většinou graficky znázorněny prostřednictvím tří Vennových diagramů, na jejichž pomyslném průniku se nachází big data.

Obr. 2-1 Model charakteristik big data 9V – vlastnosti v doméně dat, statistiky a BI



Zdroj: data (Buyya, a další, 2016, str. 14), vlastní zpracování

Jak je patrné z Obr. 2-1, doména dat zahrnuje tradiční kritéria modelu 3V, tedy objem (volume), rychlosť vzniku a zpracování (velocity) a různorodost (variety) dat, přičemž autoři modelu 9V v této rovině (Buyya, a další, 2016, str. 11) zdůrazňují především význam objemu dat, v jehož případě, na rozdíl od rychlosti a různorodosti, dochází v průběhu času k výrazným změnám (růstu).

Doména BI zahrnuje charakteristiky (Goyal, a další, 2018, str. 250) představující motivaci k realizaci procesu BDA realizovaného s cílem dosažení BI, konkrétně tedy viditelnost (visibility), hodnotu (value) a verdikt (verdict). Z perspektivy business intelligence tudíž (Goyal, a další, 2018, str. 250) viditelnost zajišťuje možnost získávat z big data vhledy, zpětné pohledy a/nebo předpovědi týkající se určitého problému a odpovídajícího řešení daného problému, hodnota vyjadřuje potenciál big data a z hlediska přínosu pro potřeby, respektive řešení určitého problému podniku, tedy pro BI, a verdikt reprezentuje rozhodnutí, které je nutné přijmout z hlediska rozsahu řešeného problému neboli specifikace a výběru hypotéz „co když“ (what-if), kvantifikace dostupných zdrojů a výpočetní kapacity. Za nejdůležitější charakteristiku je přitom v rámci domény BI označována (Goyal, a další, 2018, str. 250-251) viditelnost, jelikož determinuje možnost zajištění dalších dvou charakteristik big data v rovině BI.

Mezi kritéria z domény statistiky patří (Buyya, a další, 2016, str. 13) věrohodnost (veracity), platnost (validity) a variabilita (variability) determinující možnost aplikace statistických modelů založených na správných hypotézách na základě důvěryhodných a spolehlivých zdrojů dat. Tato doména je pro celý proces BDA klíčová zejména proto, že v případě (Buyya, a další, 2016, str. 13) špatně zvolené hypotézy, nedůvěryhodného zdroje

dat a/nebo nesprávného statistického modelu mohou být výstupem procesu analýzy big data nesprávné závěry. Zatímco (Buyya, a další, 2016, str. 13) vlastnost věrohodnosti je tedy zaměřena na zajištění důvěryhodnosti a spolehlivosti dat, platnost je orientována na validní použití dat s využitím relevantního statistického modelu a variabilita implikuje potřebu efektivního zvládnutí komplexity a variace dat. Za klíčovou vlastnost z domény statistiky je přitom označována (Buyya, a další, 2016, str. 13) spolehlivost, která zdůrazňuje relevantní způsob vytvoření statistického modelu blízkého realitě.

Uvedený výčet modelů ani samotných charakteristik big data není úplný, nicméně poskytuje dostatečný výhled do vlastností big data a parametrů, jež odlišují big data od tradičních dat. Pokud jde o otázku velikosti big data, bude jakákoli odpověď zřejmě validní pouze po určité omezenou dobou, protože tato hranice se neustále posouvá, od (Jain, 2017, str. 2) terabajtů v roce 2012 po petabajty v současné době, v budoucnu exabajty, atd.

Z uvedených definic big data je přitom patrné, že aby bylo možné určitý soubor dat označit za big data, nemusí být nutně splněny všechny charakteristiky big data současně, jelikož například (Jain, 2017, str. 2) data relativně malého objemu mohou být tak komplexní, že na jejich zpracování v rámci rozumného časového horizontu tradiční nástroje zpracování dat nebudou stačit. Zejména (Holubová, a další, 2015, str. 21) tzv. proudy dat (streams) kontinuálně bez přestání generují tok dat, která jsou z hlediska objemu relativně „malá“, nicméně z hlediska dalších charakteristik mohou odpovídat kategorii big data a jejich zpracování prostřednictvím tradičních nástrojů big data nemusí být v rámci relevantního časového horizontu možné.

Moderní definice big data se pak zaměřují především na vlastnosti big data ve vztahu k procesu analýzy big data a potenciálu business intelligence realizovaného nad big data pro dosahování potřeb a/nebo řešení problémů podniku.

2.3 Analýza big data

2.3.1 Vymezení pojmu analýza big data

Termín analýza big data (big data analytics) je v literatuře definován například jako (Hwang, a další, 2017, str. 24) „... proces zkoumání rozsáhlého množství dat rozličných typů (big data) za účelem odhalení skrytých vzorů, neznámých korelací a dalších užitečných informací.“

Analýza big data představuje (Pyne, a další, 2016, str. 4) „... odlišnou formu nebo rozšíření tradiční analýzy dat ...“, která musí v praxi řešit problémy spojené právě s charakteristikami big data.

Jiná definice pak vymezuje BDA jako (Ankam, 2016, str. 1) „... proces analýzy big data za účelem poskytování minulých, současných a budoucích statistik a užitečných výhledů, které lze využít pro zkvalitnění podnikových rozhodnutí.“

Primárním cílem BDA je (Hwang, a další, 2017, str. 24) podpořit podniky v přijímání lepších business rozhodnutí prostřednictvím analýzy rozsáhlých objemů dat, která by s využitím tradičních nástrojů BI zůstala nevyužita.

2.3.2 Vztah analýzy big data k ostatním oblastem analýzy dat

Za účelem sjednocení terminologie a pohledu na problémovou oblast bude v rámci této podkapitoly provedeno vymezení nejdůležitějších termínů a domén, které s BDA úzce souvisí.³

V obecné rovině představuje analýza big data (big data analytics) podoblast dobývání dat z databází⁴ (knowledge discovery in databases), jež je definováno jako (Fayyad, a další, 1996, str. 40-41) „... *netriviální proces identifikace platných, nových, potenciálně užitečných a srozumitelných vzorů z dat.*“, přičemž (Berka, 2003, str. 24) životní cyklus dobývání znalostí dle metodiky standardního modelu procesu dobývání znalostí z databází (CRISP-DM) zahrnuje fáze porozumění problematice, porozumění datům, příprava dat, modelování, vyhodnocení výsledků a využití výsledků.

Současně je BDA zařazována pod oblast analýzy dat (data analytics), která je definována jako (Jain, 2017, str. 15) „... *proces odhalování smysluplných vzorů v rámci dat a jejich komunikace ...*“ využívající poznatky, metody, postupy, nástroje a technologie především z oblasti statistiky, programování a operačního výzkumu. Jiná definice uvádí, že analýza dat spočívá v (Runkler, 2016, str. 2) „... *použití počítačových systémů pro analýzu rozsáhlých souborů dat s cílem podpory rozhodování.*“ Projekty analýzy dat přitom obecně zahrnují fáze, mezi které patří (Runkler, 2016, str. 3) příprava (plánování, sběr dat, generování vlastností, výběr dat), předzpracování (čištění, filtrování, doplňování, opravy, standardizace, transformace), analýza (vizualizace, korelace, regrese, předpovědi, klasifikace, shlukování) a následné zpracování (interpretace, dokumentace, vyhodnocení).

V souvislosti s analýzou dat je nutné zmínit další oblast, kterou je „datová věda“ (data science – DS), které se liší zejména zaměřením z hlediska času, kdy (Ankam, 2016, str. 2) zatímco analýza dat se primárně zaměřuje na analýzu minulých a současných jevů, datová věda využívá prostředky explorativní analýzy s cílem poskytování doporučení a předpovědí pro budoucnost. Analýza dat se tudíž zaměřuje (Ankam, 2016, str. 2) spíše na deskriptivní a diagnostickou analýzu (descriptive and diagnostic analytics), na rozdíl od datové vědy,

³) Cílem této podkapitoly není dokumentace všech oblastí analýzy dat, které s BDA souvisí, ale pouze zcela základní charakteristika vybraných oborů, a to především KDD a BI. Současně také není cílem daného textu zcela jednoznačně vymezit vztah uvedených oblastí ve smyslu podřazenosti/nadřazenosti, a to především s ohledem na skutečnost, že všechny uvedené obory se i nadále vyvíjejí, přičemž rozvoj je výrazně determinován mimo jiné rychlým pokrokem inovací a vývoje v oblasti softwarových i hardwarových technologií.

⁴) Dříve byl (Fayyad, a další, 1996, str. 39) pro činnosti související s KDD používán (kromě jiných) termín dolování dat (data mining), přičemž poprvé byl termín KDD použit v roce 1989 pro zdůraznění výstupu procesu dobývání, jímž by měly být právě znalosti. Termín KDD je v současnosti používán (Fayyad, a další, 1996, str. 39) pro označení procesu dobývání užitečných znalostí z dat, zatímco pojem dolování dat označuje jednu z fází tohoto procesu, tj. etapu dolování či modelování.

která je orientována na prediktivní a preskriptivní analýzu (predictive and prescriptive analytics). Pro BDA je pak charakteristické, že umožňuje realizovat všechny uvedené typy analýzy/analytických úloh, tedy (Pyne, a další, 2016, str. 15-17) deskriptivní BDA zaměřenou na odhalování vzorů z historických dat (co se stalo?) a využívající například metody regrese a vizualizace, prediktivní BDA orientovanou na generování předpovědí budoucího vývoje z existujících dat všech možných zdrojů (co se stane?) a využívající například techniky předpovídání, predikce či hodnocení a také preskriptivní BDA, jejímž cílem je pomáhat při hodnocení dopadu potenciálních rozhodnutí (jaký to bude mít dopad/jak toho docílit?), a jež pro tyto účely využívá například koncepty optimalizace, číselného modelování nebo simulace.

Další klíčovou oblast, s níž BDA úzce souvisí, představuje business intelligence, disciplína vymezená jako (Novotný, a další, 2005, str. 19) „... sada procesů, aplikací a technologií, jejichž cílem je účinně a účelně podporovat rozhodovací procesy ve firmě. Podporují analytické a plánovací činnosti podniků a organizací a jsou postaveny na principech multidimenzionálních pohledů na data.“ Architektura řešení BI přitom obecně zahrnuje (Novotný, a další, 2005) vrstvu pro extrakci, transformaci, čištění a nahrávání dat (ETL), pro ukládání dat, pro analýzy dat, pro prezentaci dat a vrstvu oborové znalosti. Jednu z komponent BI přitom představuje dolování dat, stejně jako v případě KDD. Podobně jako BDA, je také BI v některých zdrojích zařazována do kategorie analýzy dat.

Vztah BI a BDA je nahlížen různě, především v závislosti na tom, zda je hodnocen primárně z technického, respektive technologického či strategického hlediska. Z čistě technologického pohledu se lze nezřídka setkat s vymezením BDA vůči BI, kdy BI je zařazováno (Ankam, 2016, str. 3-4) do kategorie nástrojů zaměřujících se převážně na deskriptivní a prediktivní analýzu realizovanou především nad relačními databázovými systémy (RDBMS) a tedy převážně strukturovanými daty s pevně daným schématem definovaným při zápisu dat (schema-on-write – SOW). V protikladu k tomu je BDA zařazována do kategorie (Ankam, 2016, str. 3-4) nástrojů orientovaných spíše na preskriptivní analýzu realizovanou především nad nerelačními/nejen relačními⁵ databázovými systémy a platformami pro distribuované uložení dat, tzn. nad daty převážně semi-strukturovanými a/nebo nestrukturovanými, v jejichž případě není schéma pevně dáné při zápisu dat, ale je načteno/specifikováno/odvozeno/objeveno při každém dalším načtení dat (schema-on-read – SOR).

Přestože může být uvedená kategorizace BI a BDA v některých aspektech logická, považuje ji autor této práce za značně nepřesnou, především s ohledem na neustálý vývoj obou oblastí, kdy nástroje z kategorie BI již v současnosti mohou efektivně využívat data

⁵) Písmena „No“ ve zkratce NoSQL jsou v zahraniční literatuře používána ve významu „non“, „not only“ nebo také „non-relational“. Tato situace byla determinována mimo jiné historicky, kdy v době vzniku byly NoSQL systémy vymezovány v opozici k RDBMS („non“ SQL), později jako další vývojová etapa/stupeň („not only“ SQL) v oblasti databázových systémů (DBMS). Ani jeden z těchto výkladů zkratky nicméně (Holubová, a další, 2015, str. 25) není správný, především s ohledem na další vývoj v oblasti databázových systémů, kdy například některé RDBMS již v současné době poskytují určité vlastnosti/funkce dříve dostupné pouze v rámci NoSQL. V dalším textu je pro slovní označení NoSQL databázových systémů používáno označení „nerelační“.

uložená i v NoSQL databázích, skutečnost, že i v případě BI je možné využít semi-strukturovaná data, již zmíněný fakt, že BDA umožňuje realizovat analytické úlohy všech typů (deskriptivní, prediktivní i preskriptivní) a také skutečnost, že proces ETL (ač v odlišné formě s využitím jiných nástrojů a technologií) představuje komponentu BI i BDA, atd.

Z uvedených důvodů se autor této práce přiklání spíše ke strategickému pohledu na vztah BI a BDA, který respektuje rozdíly v procesu a technologiích využívaných v oblastech BI a BDA, nicméně nestaví tyto domény do protikladu, ale naopak (Buyya, a další, 2016, str. 11) zdůrazňuje úlohu BDA pro podporu rozhodování a upozorňuje, že význam BDA neumožňující dosahovat BI by nebyl relevantní. Kombinace využití tradičního BI a BDA představuje (Davenport, a další, 2017) jednu z vývojových fází na poli analýzy, tzv. „*Analytics 3.0*“, která pro zajištění operativní podpory rozhodování využívá všechna dostupná data a technologie, jež zajistí nejen zpracování dávkově, ale především v reálném čase. Jednotlivé vývojové etapy jsou blíže charakterizovány v Tab. 2.1, přičemž vývoj od jedné etapy k další nezahrnuje eliminaci stávajících oborů a technologií, ale spíše další rozšíření množiny využívaných disciplín, které je zpravidla determinováno technologickým pokrokem. Z tabulky je rovněž patrné, že v současné době se již nacházíme v další fázi charakteristické masivní adopcí a využitím disciplín a technologií umělé inteligence (AI).

Tab. 2.1 Vývojové fáze na poli analýzy – od Analytics 1.0 k Analytics 4.0

Vývojová fáze	Rok	Typy úloh	Disciplíny, technologie
Analytics 1.0	Do 2005	Deskriptivní, prediktivní	BI, ETL, DW
Analytics 2.0	Od 2006	Základní BDA	Big data, BDA, Hadoop
Analytics 3.0	Od 2010	+ úlohy v reálném čase	Analytics 1.0 + 2.0, IoT
Analytics 4.0	Od 2015	+ autonomous analytics	AI, DL, kognitivní technologie

Zdroj: data (Davenport, a další, 2017, str. 2-15), vlastní zpracování

Oblasti KDD, BI i BDA tedy představují samostatné etablované obory, pro které je charakteristické, že stále nejsou zcela jednoznačně vymezeny, nezřídka se v určitých aspektech více či méně překrývají, respektive hranice mezi těmito doménami není jednoznačně vymezena. V určitých případech také využívají některé stejné/podobné postupy, prostředky, algoritmy či metody, shodně jsou označovány za interdisciplinární. Pro všechny současně platí, že úspěch a přínos výsledků z realizace do značné míry závisí na kvalitě dat, která jsou zpracovávána, a v neposlední řadě, že v podnicích jsou zpravidla nasazovány formou projektu.

Na druhou stranu pokud jde o vývoj, pak lze konstatovat, že BDA představuje další vývojovou etapu následující po BI, na níž navázala etapa charakteristická využitím obou těchto oblastí, tedy jak BI a tradičních dat, tak BDA a big data. Uvedený vývoj je spojen s etablováním tzv. pokročilé analýzy (advanced analytics), která je definována jako (Gartner, 2019) „... autonomní nebo semi-autonomní zkoumání dat nebo obsahu s použitím sofistikovaných technik a nástrojů, typicky za hranicí tradičního business intelligence.“ Oblast pokročilé analýzy odpovídá výše zmíněné vývojové fázi Analytics 4.0 a zahrnuje technologie a/nebo techniky, mezi které kromě BDA dále patří (Gartner, 2019)

dolování dat/textu, strojové učení, rozpoznávání vzorů, tvorba předpovědí, vizualizace, sémantická analýza, analýza sentimentu, síťová a distribuovaná analýza, multivariantní statistika, analýza grafů, simulace, komplexní zpracování událostí či neurální sítě, atd.

Rozvoj této oblasti je spojen především s technologickým pokrokem umožňujícím postupovat na cestě od business intelligence k umělé inteligenci, jejíž význam stoupá, a jež se prosazuje takřka ve všech oborech.

2.3.3 Proces analýzy big data

V základní rovině tvoří životní cyklus projektu analýzy big data fáze (Ankam, 2016, str. 5) identifikace business problému a požadovaných výsledků projektu, identifikace potřebných dat, sběr dat, předzpracování a ETL, provedení analýz, vizualizace informací. V tomto a dalším odstavci následuje popis jednotlivých fází, který vychází ze zdroje (Ankam, 2016, str. 6-7). V první fázi životního cyklu BDA je nejprve vymezen business problém, který má projekt řešit a současně očekávaný výsledek realizace BDA projektu, což napomáhá jasně vymezit rozsah potřebných dat i analýz, jež bude nezbytné provést. Následuje identifikace stanovení požadované kvality, množství, formátu a interních i externích zdrojů dat, které bude nezbytné pro realizaci BDA zajistit. Do procesu přitom mohou vstupovat kromě big data také další zdroje a data nesplňující charakteristiky big data.

Obr. 2-2 Životní cyklus analýzy big data



Zdroj: data (EMC, 2015), vlastní zpracování

V další fázi je proveden návrh a realizace sběru dat ze stanovených zdrojů prostřednictvím relevantních technologií pro příjem a případně uložení dat. Jakmile jsou data přijata, dochází k jejich předzpracování s cílem převodu do požadovaného formátu a zajištění odpovídající kvality (čištění dat, doplňování dat, anonymizace, atd.). Následně jsou realizovány příslušné úlohy deskriptivní, diagnostické, prediktivní a/nebo preskriptivní analýzy, a to s využitím technologií pro zpracování big data dávkově a/nebo v reálném čase. Výsledky realizace analýz, jež mají poskytnout odpovědi na otázky související se

stanoveným business problémem, jsou pak v poslední fázi vhodným způsobem vizualizovány tak, aby co nejvíce přispěly k přijetí relevantního rozhodnutí řešícího daný business problém.

V uvedeném obecném pojetí se proces BDA, s výjimkou použitých metod a technologií, příliš neliší od tradičních projektů analýzy dat. Detailní specifikace životního cyklu projektu BDA zahrnuje (EMC, 2015) šest fází, mezi které, jak je patrné z Obr. 2-2, patří objevování (discovery), příprava dat (data preparation), plánování modelu (model planning), tvorba modelu (model building), komunikace výsledků (communicate results) a operacionalizace (operationalize) ve smyslu aplikace či uvedení do praxe. Následující tři odstavce obsahují bližší charakteristiku jednotlivých fází životního cyklu BDA vycházející z pramene (EMC, 2015).

Ve fázi objevování získává tým poznatky z dané doménové oblasti, včetně dříve realizovaných projektů podobného zaměření. Současně tým hodnotí dostupné zdroje relevantní pro realizaci projektu, mezi které patří především lidé, technologie, čas a data. Důležitou součást první fáze představuje vymezení řešeného business problému a formulace vstupních hypotéz. Pro realizaci dalších etap je klíčová dostupnost vhodné analytické platformy. Ve fázi přípravy dat je realizován proces extrakce dat ze zdrojových systémů, transformace a nahrání analytické platformy, který může mít podobu ETL a/nebo ELT, jež jsou společně označovány jako ETLT. Tým se tak podrobně seznamuje s daty a realizuje jejich čištění a transformace, které zajistí stav dat umožňující efektivní analýzu a využití v průběhu dalších fází.

Plánování modelu zahrnuje výběr metod, technik a workflow, jež budou aplikovány v rámci následující fáze tvorby modelu. Na základě průzkumu dat jsou odhalovány vztahy mezi proměnnými, probíhá selekce klíčových proměnných a nevhodnějších modelů. Do etapy tvorby modelu spadá vytvoření testovacích, tréninkových a produkčních kolekcí dat (data set) a následně vytvoření a aplikace modelů, které byly připraveny v rámci předchozí fáze. Přitom je nezbytné zhodnotit, zda budou stávající analytická platforma a prostředí (zdroje) pro aplikaci modelů a workflow dostačující.

Realizační tým následně v průběhu fáze komunikace výsledků ve spolupráci s hlavními zainteresovanými stranami rozhodne o tom, zda výsledky realizace představují z hlediska hodnotících kritérií stanovených v první fázi úspěch či neúspěch. Dále jsou identifikována klíčová zjištění, kvantifikována hodnota pro business a sestaven popis obsahující sumarizaci a sdělení získaných zjištění a poznatků pro zainteresované strany. V závěrečné fázi životního cyklu BDA jsou dodány finální reporty, informace, kód a technická dokumentace. Realizační tým může rovněž v této etapě provést pilotní projekt, jehož prostřednictvím dojde k implementaci modelů v produkčním prostředí.

Autoři uvedené specifikace životního cyklu BDA zdůrazňují, že (EMC, 2015) jednotlivé fáze životního cyklu nejsou realizovány ryze sekvenčně, ale naopak vykazují charakter procesu a měly by být prováděny iterativně až do okamžiku, kdy tým disponuje potřebnými zdroji pro přechod do další fáze. V případě, že určitá fáze nepřinese požadované výsledky nebo není úspěšná, je možné se kdykoli vrátit k předchozí fázi a realizovat další iteraci.

2.4 Koncepty a technologie pro analýzu big data

Jak již bylo zmíněno, rozvoj na poli big data (i dalších) je do značné míry determinován pokrokem a inovacemi v oblasti technologií. Objevují se nové koncepty a na trh jsou uváděny nové technologie zajišťující možnost zpracovávat big data novými způsoby, rychleji, výkonněji a s vyšší nákladovou efektivitou. V dalších následujících podkapitolách jsou proto klíčové koncepty a technologie související s big data stručně představeny.

Jak vyplývá z předchozích kapitol, big data a proces jejich zpracování jsou charakteristické především tím, že uložení a zpracování přerůstá dimenzi jednotlivých strojů a je tudíž nezbytné řešit jejich uložení a zpracování distribuovaně, tedy napříč více uzly tvořících společně tzv. cluster. Koncepty a technologie⁶ pro analýzu big data lze rozdělit do dvou základních kategorií, mezi které patří (Jain, 2017, str. 30) provozní (operational) big data zabývající se především distribuovaným příjmem a uložením big data a analytické (analytical) big data zaměřené na distribuované zpracování big data.

2.4.1 Distribuované uložení dat

MODELY DISTRIBUOVANÉHO ULOŽENÍ DAT

Model uložení představuje (Wu, a další, 2017b, str. 3) jádro všech systémů operujících s big data, přičemž existuje více modelů uložení a organizace dat, které lze na tomto poli aplikovat, a to nikoli nezbytně konkurenčně, ale naopak samostatně nebo častěji v určité kombinaci. Pro distribuované uložení dat jsou k dispozici především tři hlavní modely uložení:

- blokové úložiště (Amazon EBS, OpenStack Cinder),
- souborové úložiště (sítové souborové systémy – NFS, distribuovaný souborový systém Hadoop – HDFS),
- objektové úložiště (Amazon S3, EMC Atmos, OpenStack Swift).

Další popis jednotlivých modelů distribuovaného uložení dat, který je obsažen v tomto a následujících dvou odstavcích vychází ze zdroje (Wu, a další, 2017b, str. 4-15). Bloková úložiště jsou poskytována ve formě zařízení blokového úložiště sestávajících z bloků, v jejichž rámci jsou data ukládána. Moderní distribuovaná bloková úložiště mají formu clusteru zahrnujícího servery a uzly blokového úložiště, kdy server zajišťuje mapování a indexování dat na konkrétní bloky, zatímco uzly blokového úložiště zajišťují vlastní uložení dat v rámci bloků o fixní velikosti. Bloková úložiště jsou navržena pro zajištění vysoké výkonnosti a škálovatelnosti.

⁶) V dalším textu kapitoly jsou u jednotlivých typů a kategorií technologií uvedeny příklady implementací. V žádném případě se nejedná o úplné výčty, ale pouze vybrané zástupce. Některý software (zejména databázové systémy) lze zařadit do více kategorií. Za účelem větší přehlednosti je nicméně uveden pouze v rámci jediné kategorie.

Distribuovaná souborová úložiště podobně jako tradiční souborové systémy (FS) ukládají data v rámci souborů organizovaných v rámci hierarchie adresářů a souborů. Naproti tomu distribuované souborové systémy (DFS) jsou rozprostřeny napříč více uzly, kdy na každém uzlu je uložena určitá část dat. Cluster distribuovaného souborového systému je tvořen jmenným uzly (name node) a datovými uzly (data node), kdy jmenný uzel uchovává veškerá metadata, především o tom, na kterých datových uzlech se v clusteru nachází jaká data. Datové uzly zajišťují vlastní uložení dat. Klient v případě čtení i zápisu komunikuje se jmenným uzlem, od něhož získá údaje o příslušném datovém uzlu či uzlech, v jejichž rámci jsou následně jednotlivé požadované operace realizovány. Klíčovou vlastnost DFS představuje replikace zajišťující automatické redundantní ukládání dat na více místech v clusteru současně a tím odolnost proti chybám a/nebo selhání.

Objektová úložiště jsou z hlediska architektury tvořena množinou objektových uzlů zajišťujících uložení dat, objektovým serverem poskytujícím funkcionality evidence a správy identifikátorů dat a jejich lokací, a dále metadata serverem uchovávajícím odděleně metadata, která se k datům vztahují. Data jsou tak organizována v rámci objektů, které nemají žádnou hierarchickou strukturu a navíc jsou evidována metadata objektů, uchovávaná a spravovaná odděleně od vlastních dat. Oproti předchozím modelům distribuovaného uložení dat je objektové úložiště charakteristické vyšší úrovní horizontální škálovatelnosti, především díky ploché struktuře jmenného prostoru a současně umožňuje flexibilně specifikovat a aplikovat nad daty politiky na detailnější úrovni granularity. Objektová úložiště uživatele zcela abstrahuje od skutečného umístění dat.

MODELY DISTRIBUOVANÉ ORGANIZACE DAT

Kromě modelů uložení existují (Wu, a další, 2017b, str. 15) různé modely organizace dat, které vyjadřují způsob organizace a strukturování dat a reprezentují vztahy mezi různými prvky dat, přičemž model organizace dat výrazně ovlivňuje nejen uložení, ale především možnosti zpracování a analýzy big data. Na základě modelů organizace dat je možné databázové systémy rozdělit do následujících kategorií (Binani, a další, 2016, str. 43-45):

- relační databázové systémy (MySQL, MariaDB, PostgreSQL),
- nerelační databázové systémy:
 - klíč-hodnota (Redis, Riak, Memcached, DynamoDB),
 - dokumentové (MongoDB, CouchDB),
 - sloupcové (Big Table, HBase, Cassandra),
 - grafové (OrientDB, Neo4j, GraphDB),
- newSQL databázové systémy (VoltDB, MemSQL, ClustrixDB, NuoDB).

Relační databázové systémy jsou založeny především na (Wu, a další, 2017b, str. 16) relačním modelu organizace a strukturace dat a podpoře principu ACID transakcí, zajišťujícího, že při každé transakci, která je v rámci databáze realizována, je zajištěna nedělitelnost (atomicity), konzistence (consistency), izolovanost (isolation) a trvalost (durability). RDBMS po několik dekád (Wu, a další, 2017b, str. 16) dominovaly trhu s databázovými systémy, nicméně s rostoucím objemem dat, nezřídka splňujících charakteristiky big data, se klíčovým úzkým místem RDBMS stala absence možnosti

horizontálního škálování, která je pro rozsáhlé soubory dat a jejich distribuované uložení a zpracování nezbytná.⁷

Na trhu DBMS se tak postupně etablovala nová kategorie databázových systémů řešících problém distribuovaného uložení dat, a to nerelační/nejen relační (NoSQL) databázové systémy. NoSQL databázové systémy se od RDBMS liší v mnoha ohledech. Především se jedná o (Binani, a další, 2016, str. 44) odlišné uspořádání a uchovávání dat, kdy relační model je substituován některou z forem uspořádání a organizace dat klíč-hodnota, dokument, sloupec nebo graf. Záznamy jsou stále organizovány v rámci databáze. Tato však nevykazuje nezbytně podobu tabulky sestávající z řádků reprezentujících jednotlivé záznamy, jejichž hodnoty se nacházejí v jednotlivých sloupcích daného řádku. Pro NoSQL databáze je rovněž charakteristická tzv. (Binani, a další, 2016, str. 44) „no-sharing“ architektura, kdy napříč jednotlivými uzly, na nichž je databázový systém nasazen není sdílena ani paměť ani úložiště, což umožňuje zcela nezávislý provoz jednotlivých uzlů a podporuje masivní škálovatelnost.

Databáze typu klíč-hodnota představují (Holubová, a další, 2015, str. 95) nejjednodušší typ NoSQL databázových systémů, které data ukládají jako kolekci páru klíč-hodnota (key-value), kdy klíč představuje unikátní identifikátor a hodnota je reprezentována nestrukturovaným datovým typem (objekt), v jehož rámci jsou uložena data a případně také metadata. Objekt může (Holubová, a další, 2015, str. 95) uchovávat prakticky jakýkoli typ obsahu (video, text, obrázky, atd.), mezi záznamy nelze evidovat žádné vztahy ani nelze použít cizí klíče pro zajištění integrity, takže tento typ databází je velice rychlý a škálovatelný a tedy vhodný pro základní zpracování dat, nicméně nabízí pouze základní operace.

Dokumentové databázové systémy využívají (Coronel, a další, 2019, str. 674), jak z názvu vyplývá, pro uložení jednotlivých záznamů v databázi formu dokumentu, přičemž hlavní výhodou uložení záznamů v rámci dokumentu je skutečnost, že taková entita je zcela nezávislá, což umožňuje dokumenty rychleji a snadněji zpracovávat a řízeně distribuovat napříč více uzly. Obsahem dokumentu v případě sloupcových databází přitom není (Coronel, a další, 2019, str. 674) nestrukturovaný obsah (video, text, apod.), ale zpravidla semi-strukturovaný obsah, pro který je nejčastěji využíván formát JSON (JavaScript Object Notation) uchovávající obsah ve formě páru klíč-hodnota. Mezi jednotlivými dokumenty (Coronel, a další, 2019, str. 674-675) nelze evidovat vazby a nejsou ani podporovány operace typu sloučení (join), takže veškerá související, respektive provázaná data, například o zákazníkovi, objednávce a položkách objednávky jsou v dokumentové databázi uložena v jednom záznamu, tj. například v rámci jednoho JSON souboru.

Sloupcové NoSQL databázové systémy (Holubová, a další, 2015, str. 127) organizují data v rámci páru klíč-hodnota, přičemž klíč je mapován na množinu sloupců obsahujících hodnotu. Množina sloupců se může (Holubová, a další, 2015, str. 128) v každé řádce lišit,

⁷⁾ Přestože v současné době je možné nasazení některých RDBMS distribuovaně, respektive formou clusteru (např. Galera cluster pro MySQL), jedná se o vlastnost/funkcionalitu, která je určena primárně pro zajištění možnosti nasazení napříč více uzly a zajištění potřebné replikace a ochrany proti výpadkům, nikoli jako základ pro distribuované zpracování.

dále je možné vytvářet tzv. super sloupce (super column) představující skupinu sloupců s určitými logickými vazbami, a pak také rodiny sloupců (column family) reprezentující kolekci sloupců nebo super sloupců s vazbou na kolekci řádků. Sloupcové NoSQL databázové systémy jsou tedy založeny na konceptu ukládání záznamů ve sloupcích (column) a nikoli v řádcích (row), jako je tomu zpravidla v případě RDBMS.

Grafové databázové systémy jsou založeny na (Coronel, a další, 2019, str. 677) teorii grafů, jež modeluje vztahy vyjádřené pomocí hran (edges) mezi objekty, které jsou označovány jako uzly (nodes), přičemž jak uzly i objekty umožňují uchovávat vlastnosti (properties). Grafové NoSQL databáze jsou tudíž (Coronel, a další, 2019, str. 678) zaměřeny na modelování a uchovávání dat o vztazích, spíše než na data samotná a jsou tudíž vhodné především pro data, mezi nimiž existuje množství vazeb, jelikož umožňují velice efektivně procházet vazby mezi objekty.

Tab. 2.2 Srovnání charakteristik databázových systémů – RDBMS, NoSQL, NewSQL

Vlastnost	RDBMS	NoSQL	NewSQL
Relační model	Ano	Ne	Ano
ACID	Ano	Ne	Ano
SQL	Ano	Ne	Ano
OLTP	Částečně	Podpora	Plně
Škálovatelnost	Ne	Ano	Ano
Komplexnost dotazů	Nízká	Střední	Vysoká
Distribuované nasazení	Ne	Ano	Ano

Zdroj: data (Binani, a další, 2016, str. 45), vlastní zpracování

Jako poslední se pak na trhu databázových systémů objevila kategorie tzv. NewSQL systémů, jejichž podstatou je (Coronel, a další, 2019, str. 582) snaha o zajištění nejlepších vlastností z obou kategorií RDBMS a NoSQL, tedy především vysoké škálovatelnosti a současně podporu ACID transakcí. Jak je patrné z Tab. 2.2, NewSQL systémy umožňují využívat relační model, zajistit ACID a přitom distribuovaně ukládat rozsáhlé soubory dat.

CAP TEORÉM

V roce 2000 prezentoval (Chihoub, a další, 2014, str. 328) Eric Brewer na sympóziu o principech distribuovaného programování svou teorii, která byla později dokázána, a jež je označována jako tzv. CAP teorém, který říká, že v případě distribuovaných databázových systémů je možné současně zajistit pouze dvě z vlastností, mezi které patří konzistence (consistency), dostupnost (availability) a toleranci výpadků sítě (partition tolerance). Další popis CAP teorému v tomto odstavci vychází ze zdroje (Chihoub, a další, 2014, str. 328). Vlastnost konzistence v případě CAP zajišťuje, že transakce je realizována nedělitelně a opouští systém v konzistentním stavu nebo selže, což odpovídá vlastnostem nedělitelnosti a konzistence v případě ACID. Dostupnost distribuovaného systému je definována jako stav, kdy každý požadavek přijatý neselhávajícím uzlem zajistí vrácení odpovědi. Odolnost proti výpadkům sítě pak znamená, že systém je provozuschopný i v případě, že dojde

k výpadkům sítě a tedy ztrátě zpráv mezi uzly, kde se jednotlivé komponenty systému nacházejí.

Zatímco pro tradiční relační databázové systémy nasazené na jednom uzlu (Chihoub, a další, 2014, str. 328) není dostupnost problémem (před dostupností upřednostňují konzistenci), pro většinu distribuovaných systémů je klíčová, a tvůrci a architekti distribuovaných systémů musí s ohledem na CAP teorém a potřebu zajištění odolnosti proti výpadkům sítě zpravidla volit mezi zajištěním konzistence nebo dostupnosti, jak ukazuje Tab. 2.3.

Tab. 2.3 Srovnání kategorií databázových systémů z hlediska CAP teorému

Typ DBMS	C	A	P	Transakční model	Kompromis
RDBMS	Vysoká	Střední	Vysoká	ACID	A
NoSQL	Střední	Vysoká	Vysoká	BASE	C
NewSQL	Vysoká	Vysoká	Střední	ACID	P

Pozn: C – konzistence, A – dostupnost, P – odolnost pro výpadkům sítě.

Zdroj: data (Coronel, a další, 2019, str. 582), vlastní zpracování

Distribuované, respektive NoSQL databázové systémy tak (Coronel, a další, 2019, str. 582) namísto transakčního modelu ACID aplikují některou z kombinací CP nebo AP, přičemž zpravidla je upřednostňována dostupnost, před konzistencí. NoSQL využívají (Coronel, a další, 2019, str. 582) transakční model konzistence označovaný jako BASE a splňující vlastnosti, mezi které patří základní dostupnost (basically available), měkký stav (soft state) a konečná konzistence (eventual consistency). Jedná se o transakční model, v jehož případě (Coronel, a další, 2019, str. 582) se provedené změny neprojeví okamžitě, ale jsou pomalu propagovány napříč systémem (jednotlivými uzly), až jsou všechny repliky dat konečně konzistentní (eventually consistent).

NewSQL databázové systémy (Coronel, a další, 2019, str. 582) podporují plnou konzistenci, vysokou dostupnost a určitou úroveň odolnosti proti výpadkům sítě. Kategorie NewSQL tak zajišťuje (Coronel, a další, 2019, str. 582) to nejlepší z obou světů RDBMS a NoSQL, tj. vysokou škálovatelnost a ACID model transakcí, nicméně zpravidla na úkor odolnosti proti výpadkům.

Díky distribuovaným DBMS z kategorie NoSQL a NewSQL je v současnosti možné efektivně ukládat a zpracovávat big data a v závislosti na konkrétních požadavcích zajistit potřebnou úroveň konzistence, dostupnosti a odolnosti vůči výpadkům sítě. NoSQL a NewSQL DBMS přitom nahrazují RDBMS, naopak je doplňují. RDBMS nejsou využívány pro ukládání ani zpracování big data, nicméně nezřídka jsou do celkového workflow zpracování big data určitým způsobem zapojeny. Díky rozvoji na poli NoSQL a NewSQL je možné big data prostřednictvím těchto databázových systémů ukládat a zpracovávat i způsoby a nástroji, které byly dříve dostupné výhradně v rámci RDBMS.

2.4.2 Distribuované zpracování dat

Jak vyplývá z předchozího textu věnovanému operačnímu big data, vyžaduje příjem a ukládání big data aplikaci nových konceptů a technologií pro příjem a distribuované uložení dat. Podobně analytické big data vyžaduje aplikaci nových konceptů a technologií umožňujících zajistit distribuované zpracování dat vykazujících charakteristiky big data. V dalším textu proto budou představeny klíčové poznatky, koncepty a technologie z oblasti distribuovaného zpracování dat.

PROGRAMOVACÍ MODELY PRO DISTRIBUOVANÉ ZPRACOVÁNÍ DAT

Úlohy zpracování a analýzy big data jsou (Hammoud, a další, 2014, str. 2) realizovány prostřednictvím programů, které jsou vytvořeny, sestaveny, nasazeny a následně spuštěny/provedeny. Program může být spuštěn/proven různými způsoby a v závislosti na způsobu provedení programu je lze kategorizovat především na následující typy (Hammoud, a další, 2014, str. 3):

- sekvenční (sequential) program – je proveden v přesné posloupnosti výroků tvořících daný program tak, jak je tvůrce/programátor ve zdrojovém kódu programu specifikoval,
- konkurenční (concurrent) program – představuje množinu sekvenčních programů, které jsou spuštěny současně, společně sdílí jeden procesor počítače, na kterém může být současně prováděn pouze jeden program, takže se postupně střídají ve vykonávání vlastního programu podle toho, jak je jím přidělován procesorový čas,
- paralelní (parallel) program – reprezentuje množinu sekvenčních programů, které jsou spuštěny a vykonávány současně (paralelně) nad různými procesory, respektive jádry procesorů,
- distribuovaný (distributed) program – představuje paralelní program sestávající z množiny sekvenčních a/nebo konkurenčních a/nebo paralelních programů spuštěných/provedených v rámci různých strojů propojených prostřednictvím sítě.

Moderní operační systémy (OS) podporují (Hammoud, a další, 2014, str. 4) multitasking, tj. paralelní provádění výpočtů několika programů současně, které je řízeno s využitím plánovače (scheduler). Programovací jazyky (Hammoud, a další, 2014, str. 4) umožňují multitasking prostřednictvím podpory více vláknových aplikací (multithreading), kdy vlákno (thread) představuje nejmenší sekvenci instrukcí, kterou lze řídit v rámci OS prostřednictvím plánovače a jednotlivá vlákna jsou zastřešována procesy (process) s přiřazeným vlastním adresním prostorem, v jehož rámci dochází ke spuštění vláken. V kontextu paralelního, respektive distribuovaného programování je (Hammoud, a další, 2014, str. 4) pro proces, který může zastřešovat více vláken, používáno označení úkol (task), zatímco skupina úkolů asociovaných k identické aplikaci je označována jako úloha (job) a aplikace pak může zahrnovat jednu či více úloh.

Mezi hlavní motivy pro paralelizaci programů (na lokální i distribuované úrovni) patří především (Hammoud, a další, 2014, str. 5) zajištění rychlejšího provedení programu a/nebo vyšší propustnosti (throughput). Paralelizace je často využívána v mnoha oblastech vědy a umožňuje mnohem rychleji realizovat výpočty a řešit problémy. Například simulace složení proteinu (Hammoud, a další, 2014, str. 5) by sekvenčním způsobem mohla trvat i

několik roků, zatímco formou paralelního, respektive distribuovaného způsobu může být dokončena v řádu několika dní. V jiných případech je rychlosť zpracování (Hammoud, a další, 2014, str. 5) klíčová s ohledem na typ řešeného problému, kdy například předpovědi výskytu tornád pozbydou významu, pokud nebudou realizována dostatečně rychle, nebo například algoritmy vyhledávače Google, jejichž provedení by bez masivní paralelizace vůbec nebylo možné/efektivní. Ke slovu tudíž přicházejí programovací modely zajišťující distribuované zpracování v (tématu) reálném čase. Využití paralelizace rovněž umožňuje zajistit (Han, a další, 2017, str. 11) efektivní horizontální škálování (scale-out), takže programy lze rychleji realizovat na vyšším počtu komoditních strojů propojených v clusteru, bez potřeby vertikálního škálování (scale-up), tj. navyšování kapacity zdrojů jediného stroje.

Pro zajištění masivní paralelizace v distribuovaném prostředí jsou tedy využívány (Hammoud, a další, 2014, str. 7) distribuované programovací modely umožňující programátorem abstrahovat paralelizaci, přeložit algoritmy do distribuovaných programů, a zajistit provedení v distribuovaném prostředí. Mezi distribuované programovací modely patří (Hammoud, a další, 2014, str. 7) tradiční distribuované programovací modely, které nezajišťují automatickou paralelizaci ani odolnost proti chybám (sdílení paměti – shared memory, předávání zpráv – message passing) a distribuované analytické enginy zajišťující automatickou paralelizaci, odolnost proti selhání, možnost efektivního využití v cloudu a další funkce a/nebo vlastnosti (MapReduce, Pregel, GraphLab).⁸

Zatímco MapReduce a GraphLab představují distribuované programovací modely využívající (Hammoud, a další, 2014, str. 9) koncept sdílení paměti, Pregel reprezentuje programovací model založený na konceptu předávání zpráv. Uvedené programovací modely se oproti starším modelům liší mimo jiné především tím, že (Hammoud, a další, 2014, str. 9) jednotlivé výpočty, respektive úkoly aplikace jsou zpravidla realizovány v místě/lokalitě, kde se zpracovávaná data nacházejí, takže je není nutné odesílat, což naprosto zásadně snižuje zátěž sítě. Rozdíl spočívá v tom, že distribuované programovací modely založené na sdílení paměti zajistí abstrakci programátora od umístění souborů, zatímco modely využívající koncept předávání zpráv vyžadují, aby programátoři data před provedením programu vhodným způsobem rozmištili a/nebo uspořádali.

V rámci distribuovaného programovacího modelu sdílení paměti paralelně realizované úkoly (Hammoud, a další, 2014, str. 7) komunikují prostřednictvím sdíleného adresního prostoru, který může být umístěn v paměti nebo na disku. Data jsou v rámci daného sdíleného prostoru (Hammoud, a další, 2014, str. 7) sdílena, není třeba je odesílat, ale je nutné zajistit synchronizaci, podporující řízení pořadí realizovaných operací čtení a zápisu napříč různými úkoly. Tím se zajistí (Hammoud, a další, 2014, str. 7) eliminace potenciálního simultánního zápisu do sdílených dat, který by způsobil nekonzistence dat, přičemž pro tyto účely jsou zpravidla využívány mechanismy semaforů, zámků a/nebo

⁸⁾ Podrobnější charakteristika všech programovacích modelů pro distribuované zpracování rozsáhlých souborů dat přesahuje rámec této práce. V dalším textu proto budou představeny pouze základní koncepty distribuovaného zpracování rozsáhlých souborů dat. S ohledem na zaměření práce bude podrobně charakterizován programovací model MapReduce, a to v rámci příslušné části kapitoly 3.

bariéry. Programovací model sdílení paměti je charakteristický především tím, že vývojáři nekódují funkce pro odesílání ani přijímání zpráv, a že vrstva úložiště poskytuje sdílený přístup všem úkolům aplikace.

V kontrastu k distribuovanému programovacímu modelu založenému na sdílení paměti je model předávání zpráv, v jehož případě (Hammoud, a další, 2014, str. 10) distribuované úkoly nesdílejí žádný prostor a naopak komunikují prostřednictvím odesílání a přijímání zpráv, což determinuje určitou zátěž a náklady na komunikaci, kdy prostřednictvím zpráv jsou odesílána data a současně zajišťována synchronizace. V případě distribuovaného programovacího modelu typu předávání zpráv (Hammoud, a další, 2014, str. 11) není potřeba zajišťovat sdílení vrstvy distribuovaného souborového systému a současně, jelikož pro každou operaci odeslání zprávy existuje odpovídající operace přijetí zprávy, není nutné zajišťovat explicitní synchronizaci.

Distribuované programy lze kromě toho, zda využívají model sdílení paměti nebo předávání zpráv, koncipovat jako (Hammoud, a další, 2014, str. 12-13) synchronní nebo asynchronní. Distribuované synchronní programy jsou prováděny (Hammoud, a další, 2014, str. 12-13) v přesně specifikovaném sledu operací, které prováděných sekvenčně (dokončení určitého úkolu podmiňuje možnost zahájení realizace některého dalšího úkolu). Naproti tomu úkoly tvořící asynchronní distribuované programy jsou realizovány (Hammoud, a další, 2014, str. 12-13) simultánně, respektive paralelně (zahájení žádného úkolu není podmíněno dokončením jiného úkolu). Z tohoto hlediska lze distribuované programovací modely MapReduce a Pregel klasifikovat (Hammoud, a další, 2014, str. 13) jako synchronní a GraphLab jako asynchronní. Jedním z klíčových modelů synchronizace, který je (Hammoud, a další, 2014, str. 13) využíván pro efektivní realizaci distribuovaných programů, je masivně synchronní paralelizace (BSP), kterou využívá například programovací model Pregel.

V závislosti na typu paralelizace lze dále rozlišit (Hammoud, a další, 2014, str. 14-17) datově a grafově paralelní programy, přičemž podstatou datové paralelizace je paralelní provedení výpočetních úkolů v rámci příslušného stroje, kde jsou umístěna relevantní data nebo jejich část, zatímco grafová paralelizace spočívá především v distribuci grafu a nikoli dat jako takových, kdy jednotlivé části grafu reprezentujícího určitý řešený problém, jsou řešeny v rámci jednotlivých uzlů clusteru paralelně. Z tohoto hlediska lze distribuovaný programovací model MapReduce (Hammoud, a další, 2014, str. 14-17) klasifikovat jako datově paralelní a modely Pregel a GraphLab jako grafově paralelní.

Konečně z hlediska architektury, respektive řízení programu je možné distribuované programy kategorizovat (Hammoud, a další, 2014, str. 18) na symetrické (peer-to-peer), asymetrické (master/slave) a případně hybridní. V případě asymetrické architektury (Hammoud, a další, 2014, str. 19) existuje centrální proces (master) řídící logiku provedení programů, jejichž exekuci zajišťují pracovní (slave) procesy a komunikace probíhá pouze na úrovni řídícího a pracovního procesu, nikoli mezi pracovními procesy. Pro distribuci úkolů pracovním procesům lze využít koncept (Hammoud, a další, 2014, str. 19) tlaku (push), kdy řídící proces přiděluje úkoly jednotlivým pracovním procesům dle vlastních potřeb/strategií nebo principu tahu (pull), kdy řídící proces přiřazuje úkoly pracovním procesům v okamžiku, kdy o to požádají. Naopak v rámci symetrické

architektury (Hammoud, a další, 2014, str. 20) žádný centrální proces neexistuje a organizace, řízení a práce jsou rovnoměrně rozděleny mezi jednotlivé úkoly, které jsou si prakticky rovny a mohou libovolně vzájemně komunikovat. Zatímco komponenta řídícího procesu (Hammoud, a další, 2014, str. 20) představuje v asymetrické architektuře distribuovaných programů potenciální kritické místo determinující selhání celého systému (SPOF), v případě symetrické architektury je nutné pro přijetí rozhodnutí aplikovat určitý rozhodovací mechanismus, do nějž se musí zapojit všechny zúčastněné úkoly. Z hlediska architektury řízení lze distribuované programovací modely MapReduce a Pregel zařadit do kategorie synchronních, respektive master/slave řešení a GraphLab do peer-to-peer, tedy synchronní kategorie.

Tab. 2.4 Srovnání charakteristik základních distribuovaných programovacích modelů

Charakteristika	MapReduce	Pregel	GraphLab
Základní model	Sdílení paměti	Předávání zpráv	Sdílení paměti
Synchronizace	Synchronní	Synchronní	Asynchronní
Typ paralelizace	Datová	Grafová	Grafová
Architektura řízení	Master/slave	Master/slave	Peer-to-peer

Zdroj: data (Hammoud, a další, 2014, str. 7-20), vlastní zpracování

Základní „moderní“ distribuované programovací modely, tedy především MapReduce, Pregel a GraphLab, jejichž klíčové charakteristiky ukazuje Tab. 2.4, lze využít pro realizaci distribuovaného paralelního zpracování rozsáhlých souborů, při zajištění efektivního řešení výzev distribuovaného zpracování rozsáhlých souborů dat, mezi něž patří implementace programovacího a výpočetního modelu, automatická paralelizace a synchronizace, plánování a řízení realizace, komunikace, škálovatelnost a heterogenita prostředí.

Kromě uvedených základních programovacích modelů pro distribuované zpracování rozsáhlých souborů dat je dále v oblasti big data využívána (Wu, a další, 2017a, str. 32) řada dalších programovacích modelů, jež lze využít prostřednictvím příslušné podpory ze strany relevantního software a/nebo programovacího/jiného jazyka, a mezi něž kromě MapReduce patří především funkční programování, SQL a jeho alternativy/dialekty z oblasti big data, aktorový model výpočtu (actor model), statistické a analytické modely, řízení datového toku (dataflow), masivně synchronní paralelizace a vysokoúrovňové doménově specifické jazyky.

PLATFORMY PRO DISTRIBUOVANÉ ZPRACOVÁNÍ DAT

Mezi hlavní platformy, respektive technologická řešení pro distribuované zpracování dat, respektive BDA patří především (Buyya, a další, 2016, str. 18) framework pro distribuované zpracování rozsáhlých souborů dat Hadoop, open source implementace distribuovaného souborového systému Hadoop (HDFS), implementace programovacího modelu pro distribuované zpracování rozsáhlých souborů dat MapReduce, analytický engine pro zpracování rozsáhlých souborů dat Spark a engine pro distribuované zpracování proudů dat Flink.

Všechny uvedené technologie pro distribuované zpracování rozsáhlých souborů dat jsou součástí frameworku Hadoop, s výjimkou Flink a Spark. Podrobný popis těchto nástrojů bude předmětem příslušných částí kapitoly 3, a zde proto nebudou dále charakterizovány. S ohledem na skutečnost, že Flink není součástí žádné z distribucí frameworku Apache Hadoop, jež jsou předmětem komparace v rámci této práce, nebude dále uveden ani popis dané technologie. Uvedené platformy a technologie představují (Mayo, 2016) klíčové frameworky pro zpracování big data. Kromě zmíněných technologií samozřejmě existují další, nicméně s ohledem na zaměření práce zde nebudou blíže specifikovány ani charakterizovány.

PROGRAMOVACÍ MODELY PRO ZPRACOVÁNÍ DAT V REÁLNÉM ČASE

Programovací modely uvedené v předchozí části této podkapitoly poskytují relevantní abstrakci především v oblasti tzv. dávkového (batch) distribuovaného zpracování rozsáhlých souborů dat. Tato řešení plně dostačovala v době, kdy klíčovou charakteristikou big data byl objem (volume) dat, nicméně s příchodem nových zdrojů dat a rostoucím významem charakteristiky rychlosti vzniku a zpracování dat (velocity) nabývaly na významu možnosti efektivního zpracování rozsáhlých souborů dat v téměř reálném (near real-time) a reálném (real-time) čase. V dalším textu proto budou představeny programovací modely z této oblasti.

Programovací modely pro zpracování dat v reálném čase pochopitelně řeší odlišné výzvy než tradiční modely pro dávkové distribuované zpracování rozsáhlých souborů dat, jmenovitě především (Buyya, a další, 2016, str. 42-43) nízkou latenci (low latency), tj. časový interval prodlevy mezi výskytem události a zahájením jejího zpracování, jež zahrnuje především zpoždění na úrovni sítě a na úrovni zpracování, dále potřebu zajištění vysoké dostupnosti a v neposlední řadě nezbytné masivní horizontální škálovatelnosti.

Ústřední entitu programovacích modelů pro zpracování rozsáhlých souborů dat v reálném čase představuje (Buyya, a další, 2016, str. 43) událost (event), která je definována jako „*výskyt určitého jevu v prostředí systému*“, přičemž aplikace zpracování dat v reálném čase jsou realizovány s odlišnými motivy, mezi které může spadat rychlá reakce na výskyt událostí, detekce a reporting událostí, analýza a detekce příležitostí/hrozeb v reálném čase, detekce vzorů v chování uživatelů, paralelní distribuované zpracování v reálném čase, atd.

Zpracování událostí zajišťuje (Buyya, a další, 2016, str. 44) „*... realizaci operací nad událostmi, které jsou reportovány systémem zajišťujícím jejich příjem z určitého prostředí*“, přičemž tyto operace nejčastěji zahrnují čtení, vytváření, transformace a zpracování událostí. Provedení výpočtu nebo akce na základě výskytu určité události je zajišťováno s využitím (Buyya, a další, 2016, str. 44) programovacích modelů založených na událostech (event-based programming) prostřednictvím architektur a aplikací řízených událostmi (event-driven architectures and applications).

Další entitu, která je klíčová z pohledu zpracování rozsáhlých souborů dat v reálném čase, představuje (Buyya, a další, 2016, str. 44) proud (stream), reprezentující „*... kontinuální tok informací a/nebo dat ...*“, přičemž se může jednat například o video či audio data, sekvence dat spojených s událostmi emitovanými ze strany senzorů, zařízení, aplikací, atd.

Pro zpracování entit daného typu se používá označení (Buyya, a další, 2016, str. 44) zpracování proudů událostí (event stream processing) nebo také zpracování proudů dat (data stream processing).

V případě (Buyya, a další, 2016, str. 44) „... realizace komplexních operací nad kombinací různých událostí z jednoho či více prostředí a vazeb mezi těmito událostmi ...“ se jedná o tzv. komplexní zpracování událostí (CEP). Komplexní zpracování událostí představuje (Buyya, a další, 2016, str. 44) programovací model pro zpracování dat v reálném čase, který poskytuje abstrakci logiky zpracování událostí pro operace nad událostmi oddělených od aplikační logiky (zdrojů a příjemců událostí). Pro systémy zajišťující zpracování událostí (EPS) je klíčová (Buyya, a další, 2016, str. 45) definice typu událostí (event type), jež se mohou v daném prostředí vyskytnout. Definice typu události zpravidla zahrnuje (Buyya, a další, 2016, str. 45) časovou značku (timestamp) a další datové typy (formát dat, další metadata, vlastní data, aj.), a dále definice vzorů událostí (event pattern), které „... zachycují vztahy mezi událostmi v reálném světě“, a na jejichž základě lze detektovat výskyt vzorů událostí v rámci daného prostředí.

Primární programovací model využívaný pro zpracování rozsáhlých souborů dat v reálném čase tak představuje komplexní zpracování událostí. Při aplikaci daného modelu je nutné rozlišovat, zda se jedná o zpracování dat skutečně v reálném čase (real-time) nebo v téměř reálném čase (near-real time). Příkladem zpracování rozsáhlých souborů dat v téměř reálném čase je (Scott, 2015, str. 35) tzv. mikro-dávkové (micro-batching) zpracování proudů dat, které realizuje například Spark. Tento systém (Scott, 2015, str. 35) rozděluje kontinuální proud dat (stream) na části (mikro-dávky) zpracovávané dávkově. Příkladem systémů zajišťujících zpracování dat v reálném čase, které (Scott, 2015, str. 35) zpracovávají události postupně (event-by-event) tak, jak se vyskytují a/nebo přicházejí, jsou především enginy pro zpracování proudů dat, např. Storm nebo Flink.

V případě zpracování dat v téměř reálném čase formou mikro-dávek je tedy zpracování realizováno (Scott, 2015, str. 41) nad jednotlivými mikro-dávkami představujícími kolekci událostí z určitého intervalu (zpravidla od 500 do 5000 nebo více milisekund), přičemž platí, že čím kratší je tento interval dávky, tím více se zpracování přiblíží reálnému času. Systémy pro zpracování dat v reálném čase naproti tomu umožňují v reálném čase reagovat prakticky okamžitě na výskyt jednotlivých událostí a rovněž realizovat tzv. kontinuální dotazy (continuous querries). Tato kategorie systémů rovněž umožňuje (Hueske, a další, 2019, str. 10) efektivně vytvářet a využívat tzv. stavové (stateful) aplikace zpracování proudů dat, které uchovávají stav a při každém přijetí události mohou realizovat komplexní výpočty/operace nad danou událostí, včetně čtení/zápisu z/do stavu, jež může být uchováván v paměti, databázi či jinde.

Programovací modely pro zpracování proudů dat v (téměř) reálném čase tak poskytují vývojářům abstrakci pro zajištění komplexního zpracování událostí a umožňují efektivně vytvářet a nasazovat aplikace pro zpracování proudů dat, mezi které patří především (Hueske, a další, 2019, str. 12) aplikace řízení událostmi, aplikace pro zpracování proudů dat a aplikace pro analýzu proudů dat. Nové implementace těchto modelů umožňují (Hueske, a další, 2019, str. 18) dosahovat skutečně zpracování v reálném čase při zajištění vysoké propustnosti, nízké latence a vysoké přesnosti výsledků a na rozdíl od modelů

založených na mikro-dávkách, umožňují efektivně a přesně zpracovávat i události, které jsou přijaty mimo pořadí, v jakém vznikly.

PLATFORMY PRO ZPRACOVÁNÍ DAT V REÁLNÉM ČASE

Jak již bylo uvedeno, mezi platformy pro zpracování dat v reálném čase patří především enginy pro distribuované zpracování proudů dat, konkrétně (Buyya, a další, 2016, str. 45) Strom, Spark a Flink. Zatímco první generace těchto technologií se objevila (Hueske, a další, 2019, str. 21) již v roce 2011, skutečně efektivní a přesné zpracování proudů dat v reálném čase umožnila až třetí generace těchto nástrojů, která byla na trh uvedena v roce 2015.

Kromě enginů pro distribuované zpracování proudů dat je pak v oblasti distribuovaného zpracování rozsáhlých souborů dat využívána řada dalších technologií z kategorií, mezi které patří například (Hueske, a další, 2019, str. 10) NoSQL DBMS, databázové systémy pro masivně paralelní zpracování (MPP), middleware (především různé frontové a messaging systémy), systémy zaměřené na analýzu dat a jiné komponenty pro příjem, zpracování a/nebo analýzu proudů dat.

Podrobný popis nástrojů a technologií z kategorie zpracování proudů dat a zpracování big data v reálném čase bude předmětem příslušné části kapitoly 3 (v případě, že daná technologie bude součástí projektu frameworku nebo ekosystému Hadoop) nebo následně kapitoly 4 (jestliže daná komponenta nebude součástí frameworku ani ekosystému Hadoop, ale bude součástí určité distribuce frameworku Hadoop). Podobně jako v případě platforem pro distribuované zpracování rozsáhlých souborů dat, také v případě platforem pro distribuované zpracování proudů dat v reálném čase existuje množství dalších technologií a řešení, nicméně s ohledem na zaměření práce zde nebudou dále specifikovány ani charakterizovány.

2.4.3 Machine learning

Strojové učení a cloud computing (který bude představen v rámci následující podkapitoly) představují (Buyya, a další, 2016, str. 34) dvě nejdůležitější komponenty BDA. Strojové učení (Bell, 2015, str. 2) „... je podoblastí umělé inteligence. Prostřednictvím výpočetních prostředků navrhuje systémy, které jsou schopny se na základě trénování učit z dat. V čase a s postupným získáváním dalších zkušeností se tyto systémy mohou učit a zlepšovat a zdokonalovat model, který lze, na základě předchozího učení, využít pro predikci odpovědí na otázky.“

Z předchozího textu je patrné, že strojové učení (ML) je klíčové především pro fáze životního cyklu BDA, které jsou označovány jako plánování a tvorba modelu. ML tak představuje (Buyya, a další, 2016, str. 17) ústřední koncept BDA, bez kterého nelze BDA realizovat a k jehož podpoře všechny ostatní komponenty big data frameworku směřují. Pokud jde o výpočetní podporu ML v rovině BDA, existují v současné době čtyři hlavní architektonické modely, které umožňují zpracovávat big data, respektive rozsáhlé soubory dat v rozumném čase, a mezi které patří (Buyya, a další, 2016, str. 17):

- databázové systémy pro masivně paralelní zpracování (MPP),

- in-memory databázové systémy,
- programovací model MapReduce pro zpracování rozsáhlých souborů dat a platformy jako například Hadoop a Google File System (GFS),
- systémy pro masivně synchronní paralelizaci.

Oblast ML tudíž nejenže se zpracováním big data souvisí, ale představuje ústřední komponentu, která umožňuje dosahovat BI.

2.4.4 Cloud computing

Cloud computing je definován jako (Mell, a další, 2011, str. 2) „... model zajišťující na základě potřeb zákazníka prostřednictvím sítě všudypřítomný, pohodlný přístup ke sdílenému poolu konfigurovatelných výpočetních zdrojů (např. sítí, serverů, úložiště, aplikací a služeb), které lze rychle zřizovat a rušit pouze s minimálním úsilím pokud jde o správu těchto zdrojů nebo interakci s poskytovatelem služeb.“ Pro cloud computing je symptomatických pět charakteristik, mezi které patří (Mell, a další, 2011, str. 2):

- samoobslužná služba dle potřeb – zákazník je schopen zajistit si zpřístupnění zdrojů v okamžiku potřeby automatizovaně bez potřeby interakce s poskytovateli služeb,
- široký síťový přístup – zdroje a služby jsou přístupné přes síť a lze k nim přistupovat prostřednictvím standardních mechanismů a heterogenních platform,
- sdružování zdrojů – zdroje, které poskytovatel nabízí, jsou sdružovány (pooled), takže mohou sloužit více zákazníkům s využitím multi-tenantního modelu poskytování služeb, kdy fyzické a/nebo virtuální zdroje jsou přiřazovány dynamicky dle potřeby zákazníků, kteří neznají skutečné umístění základní fyzické infrastruktury, od níž jsou odstíněni,
- rychlá elasticita – zdroje jsou pružně (elasticky) zřizovány a rušeny, někdy automatizovaně, což zajišťuje škálování dle poptávky zákazníků, pro zákazníka se dostupné zdroje jeví jako neomezené a může je kdykoli přizpůsobit jak v čase, tak v množství,
- měřená služba – cloudové systémy automaticky řídí a optimalizují využití zdrojů prostřednictvím měření a účtování zdrojů a služeb (zpravidla na základě skutečného využití) na určité úrovni abstrakce odpovídající typu služby, přičemž využití zdrojů lze transparentně monitorovat, řídit a reportovat na straně poskytovatele i zákazníka.

Pro cloud computing jsou charakteristické následující modely nasazení služeb⁹ (Mell, a další, 2011, str. 2-3):

⁹⁾ Kromě uvedených základních modelů nasazení a poskytování služeb existuje značné množství dalších, například databáze jako služba, kontejner jako služba, business intelligence jako služba, Hadoop jako služba (HaaS), atd. Pro zachování přehlednosti na tomto místě uvádíme pouze základní modely uvedené v rámci definice cloud computingu.

- infrastruktura jako služba (IaaS) – zdroje, s nimiž zákazník interaguje, zahrnují výpočetní zdroje (virtuální stroje, kontejnery, aj.), úložiště, sítě, atd., a zákazník má tak kontrolu nad OS, úložišti, aplikacemi a případně určitými síťovými prvky,
- platforma jako služba (PaaS) – zdroje, s nimiž zákazník interaguje, zahrnují prostředí a nástroje pro nasazení aplikací, a zákazník má tak kontrolu nad aplikacemi, knihovnami, službami a konfigurací běhového prostředí, v jehož rámci jsou aplikace hostovány,
- software jako služba (SaaS) – zdroje, s nimiž zákazník interaguje, jsou připravené aplikace a jejich funkcionalita a zákazník tak má kontrolu pouze nad funkcemi a případně nastavením aplikace a nikoli clouдовou infrastrukturou ani prostředím pro běh aplikací.

V závislosti na tom, v jakém prostředí a komu jsou clouдовé služby poskytovány a zpřístupněny pak lze rozlišit (Mell, a další, 2011, str. 3) modely nasazení, mezi které patří privátní cloud (soukromý vyhrazený pro firmu nebo obchodní jednotku), veřejný cloud (veřejně dostupný potenciálně pro všechny), komunitní cloud (zajišťovaný a dostupný pro určitou komunitu uživatelů) a hybridní cloud (kombinující například privátní a veřejný cloud).

Pro oblast big data je koncept cloud computingu klíčový především proto, že (Buyya, a další, 2016, str. 17) i malým a středním firmám umožňuje zajišťovat realizaci BDA nákladově efektivním způsobem. Lídrem v oblasti služeb veřejného cloudu stále představuje (Smith, a další, 2018, str. 25) Amazon Web Services společnosti Amazon, a dále do této kategorie spadají veřejné cloudy Microsoft Azure a Google Cloud Platform. V rámci všech těchto a dalších cloudu lze nalézt a flexibilně samoobslužně využít nástroje pro zajištění jednotlivých fází BDA a také službu typu HaaS.

Cloud computing, respektive využití služeb cloud computing umožňuje (Buyya, a další, 2016, str. 18) značně snížit náklady díky elastické povaze využívaných zdrojů, které lze flexibilně škálovat oběma směry a také díky snadnému přizpůsobení skladby využívaných zdrojů konkrétním úlohám zpracování rozsáhlých souborů dat (dávkové, mikro-dávkové, interaktivní, téměř v reálném čase, v reálném čase). Díky cloudu lze tudíž začít big data zpracovávat velice rychle (oproti přípravě vlastního prostředí) flexibilně a bez dlouhodobých závazků či hrozby proprietárního uzamčení (vendor-lock-in).

2.5 Praktické aplikace analýzy big data

2.5.1 Zdroje big data

Zdroje big data nezřídka úzce souvisí s oblastmi praktické aplikace big data. Mezi zdroje big data, včetně tzv. proudů dat patří především (EMC, 2015):

- data z oboru lékařství – lékařské záznamy, dokumentace, sekvence genomu, atd.,
- obrazové materiály z internetu – obrázky a video soubory,
- kamerový dohled – videozáznamy ze stále rostoucího množství sledovacích kamer a dalších zařízení rozmisťovaných ve městech,

- mobilní zařízení – poskytující data o poloze, metadata o telefonních hovorech, využití aplikací, atd.
- chytré zařízení – poskytující informace shromažďované ze senzorových chytrých sítí, chytrých budov a dalších veřejných či průmyslových infrastruktur,
- internet věcí (IoT) – informace z rostoucího počtu senzorů a dalších prvků IoT,
- netradiční IT zařízení – RFID čtečky, navigační zařízení, zařízení pro detekci seismické aktivity, atd.,
- operační systémy a aplikace – rostoucí role telemetrie a shromažďování údajů o chování uživatelů,
- provozní systémy – serverové, síťové, aplikační, webové a jiné logy a informace,
- sociální sítě – rychle rostoucí objem obsahu, který uživatelé publikují na sociálních sítích.

2.5.2 Praktické aplikace analýzy big data

V praxi jsou big data a BDA široce využívány napříč množstvím oborů, a to především v následujících oblastech (Jain, 2017, str. 18-22; Sangeetha, a další, 2015, str. 3273-3274):

- detekce podvodů – platformy pro big data umožňují detektovat podvody a identifikovat anomálie v (téměř) reálném čase s využitím masivně rozsáhlé identifikace vzorů v rámci obrovského množství dat a transakcí a tím efektivněji předcházet podvodům a rychleji reagovat na jejich výskyt,
- analýza IT logů – IT software a oddělení generují enormní množství záznamů vykazujících charakter logů (webové servery, síťový provoz, atd.), jež bez technologií pro analýzu big data prakticky nelze v rozumném čase analyzovat, což ponechává značné množství dat nevyužité; s využitím BDA je možné logy zpracovávat zajišťovat podporu pro kvalitnější diagnostiku a prevenci problémů v systémech emitujících dané logy,
- analýza call center – call centra představují významný zdroj informací o náladě trhu a současně významnou sílu, která má možnost postoje zákazníků ovlivnit, přičemž BDA umožňuje rychleji objevit a získat potřebné vhledy a podpořit zlepšení zákaznické péče, vyřešení opakujících se problémů, apod.,
- analýza sociálních médií – sociální sítě představují jedny z nejrozsáhlejších zdrojů big data a BDA poskytuje potřebné nástroje pro zajištění analýzy sociálních sítí, která by na dané škále nebyla prostřednictvím tradičních nástrojů pro zpracování a analýzu dat možná; díky BDA je možné v reálném získávat vhledy týkající se chování zákazníků na sociálních sítích, reakcí na produkty, kampaně, atd.,
- zlepšování zdravotní péče a veřejného zdraví – BDA umožňuje dekódovat genetické řetězce v řádu minut a BDA je zaměřena mimo jiné zejména na podporu nalézání nových léků, predikci vzorů nemocí, apod.; lidé stále častěji využívají zařízení umožňující sledovat a evidovat údaje týkající se zdravotního stavu a BDA poskytuje nástroje pro efektivní analýzu těchto masivních kolekcí dat; dále BDA zajišťuje možnost monitorovat vývoj a predikovat rozvoj epidemií, například prostřednictvím kombinace lékařských záznamů s informacemi o pocitech a stavech, které lidé prezentují na sociálních sítích,

- lékařství – lékařská péče a lékařský výzkum generují obrovské množství dat, která je nezbytné relevantním způsobem zpracovávat a analyzovat, aby bylo možné podpořit prostřednictvím BDA rychlejší rozvoj výzkumu, efektivněji zpracovávat převážně nestrukturovanou lékařskou dokumentaci, vytvářet kvalitnější zdravotní profily a prediktivní modely pacientů, lépe diagnostikovat a léčit,
- věda a technologie – BDA nachází uplatnění v širokém množství vědních a technologických oborů, kde otevírá nové možnosti analýzy a získávání potřebných vhledů z rozsáhlých souborů dat; dobrým příkladem je velký urychlovač častic, který je osazen více než 150 miliony senzorů generujících stovky petabajtů dat, která není prostřednictvím tradičních metod analýzy dat možné efektivně zpracovat ani analyzovat,
- vládní organizace, státní instituce – vládním organizacím a státním institucím umožňuje BDA analyzovat a efektivně využít stále rostoucí množství dat a především taková data, kterými disponují, ale jež nejsou aktivně využívána; pokračující digitalizace služeb a komunikace determinují další růst objemu dat a BDA bude hrát významnou roli při zefektivňování služeb pro zákazníky,
- zemědělství – aplikace BDA v oblasti zemědělství zatím není tak rozsáhlá jako v jiných oblastech, spíše se čeká na vytvoření a získání big data z oblasti zemědělství, nicméně pro udržitelný rozvoj zemědělství bude aplikace BDA klíčová,
- smart cities – aplikace BDA v oblastech tzv. chytrých měst se zaměřuje především na podporu udržitelného rozvoje a zkvalitnění života prostřednictvím chytrého managementu; big data, respektive BDA na tomto poli mimo jiné přináší vhledy týkající se způsobu fungování měst a optimalizace interakce v rámci měst.

V rovině zpracování proudů dat a analýzy big data v reálném čase lze identifikovat především následující případy použití (Hueske, a další, 2019, str. 12-16):

- doporučení v reálném čase – například prezentace produktu zákazníkovi na základě údajů o zákazníkovi a jeho aktuálním chování,
- detekce vzorů nebo komplexní zpracování událostí – například identifikace podvodů v rámci transakcí realizovaných prostřednictvím kreditních karet,
- detekce anomalií – například detekce pokusů o narušení sítě,
- řízení toku dat – příjem, transformace a ukládání/odesílání dat s nízkou latencí,
- analýza proudů dat – monitoring kvality telefonních sítí, analýza chování uživatelů v mobilních aplikacích, ad-hoc analýza dat týkajících se chování zákazníků, atd.,
- reakce na výskyt událostí – situační odpověď, rozhodnutí na základě analýzy v reálném čase, orchestrovaná odpověď na výskyt události, atd.

2.6 Shrnutí

V rámci této kapitoly byl nejprve vymezen pojem big data, dále popsány charakteristiky big data, specifikován proces analýzy big data a vztah BDA k ostatním oblastem analýzy dat, popsány základní koncepty a technologie pro distribuované uložení big data, distribuované zpracování a analýzu big data a také distribuované zpracování a analýzu

proudů dat v reálném čase. V závěru kapitoly byly prezentovány hlavní zdroje big data a praktické aplikace analýzy big data.

Po přečtení kapitoly je k dispozici potřebný vhled do oblasti big data a analýzy big data, pro které je symptomatický značně interdisciplinární charakter, kdy framework Apache Hadoop, jakkoli komplexní, představuje pouze dílčí technologickou podoblast big data a BDA. Na základě sjednocení pojmu a základní množiny poznatků z problémové oblasti big data bude v rámci následující kapitoly v potřebném kontextu oblastí big data a BDA podrobně charakterizován právě framework Apache Hadoop.

3 Framework Apache Hadoop

Předmětem této kapitoly je popis architektury a funkcionality frameworku Apache Hadoop pro distribuované zpracování big data. Jelikož je tato práce zaměřena na komparaci distribucí frameworku Apache Hadoop, tato kapitola je realizována s cílem představit projekt a ekosystém Hadoop a zajistit tak dostatek poznatků pro pochopení podstaty architektury a funkcionality platformy Hadoop a její role v rámci BDA, aby bylo možné na daném základě následně v rámci relevantního kontextu provést analýzu současného stavu trhu distribucí frameworku Apache Hadoop a vlastní komparaci jednotlivých distribucí Hadoop.

V rámci kapitoly jsou nejprve charakterizovány okolnosti, které předcházely vzniku projektu Hadoop, dále je popsán vznik frameworku Hadoop, jeho rozvoj a současný stav. Stěžejní část kapitoly obsahuje charakteristiku architektury a především funkcionality jednotlivých projektů tvořících framework a ekosystém Apache Hadoop. Role Apache Hadoop v rámci procesu BDA ve firemní praxi je nastíněna v rámci závěrečné části kapitoly dokumentující praktické využití frameworku Apache Hadoop v rámci vybraných společností.

3.1 Vznik a řešení potřeb efektivního uložení a zpracování big data

3.1.1 Vznik a řešení potřeby efektivního uložení big data

Jedním z prvních subjektů, který musel v praxi čelit problému efektivního uložení a zpracování rozsáhlých souborů dat, respektive big data, byla společnost Google provozující stejnojmenný internetový vyhledávač, jež v důsledku globální indexace webových stránek řešila problém se spolehlivým a efektivním ukládáním dat nikoli na lokální škále, ale prakticky na úrovni internetu jako celku.

Společnost Google potřebu efektivního uložení dat, jež de facto splňovala charakteristiky big data, vyřešila vlastní implementací speciálního distribuovaného souborového systému, přičemž v roce 2003 tuto implementaci prezentovala (Ghemawat, a další, 2003, str. 20) jako Google File System, „... škálovatelný distribuovaný souborový systém pro rozsáhlé distribuované datově intenzivní aplikace.“ Návrh GFS (Ghemawat, a další, 2003, str. 20) řešil některé stejné problémy jako jiné existující implementace distribuovaného souborového systému (výkonnost, škálovatelnost, spolehlivost a dostupnost) a kromě toho následující architektonické aspekty (Ghemawat, a další, 2003, str. 20-21) :

- selhání komponent úložiště – s ohledem na rozsah distribuovaného úložiště (stovky až tisíce uzlů úložiště) je nutné potenciální selhání některých komponent vnímat jako samozřejmé a nikoli pouze potenciální, a proto je nutné, aby se kontinuální monitoring, detekce chyb, zajištění odolnosti proti chybám a automatické zotavení z chyb staly integrální vlastností navrhovaného distribuovaného souborového systému,

- velikost ukládaných souborů – zpracovávané soubory jsou z hlediska tradičních měřítek obrovské a vzhledem k tomu, že společnost běžně pracuje se (z hlediska objemu) stále rostoucími soubory dat v řádu terabajtů obsahujících biliony objektů, je značně neefektivní tyto datové soubory ukládat v rámci masivního počtu malých souborů o velikosti v řádu kilabajtů, a proto je nutné přehodnotit způsob ukládání souborů především z hlediska velikosti bloků a propustnosti,
- většina souborů je modifikována pouze formou připojování nových dat, zatímco k přepisu a/nebo náhodnému zápisu v rámci stávajících uložených dat nedochází prakticky vůbec, a využívány jsou zcela majoritně operace čtení, které mají často sekvenční charakter, a proto je nutné způsob ukládání optimalizovat se zaměřením pouze na operaci připojení (append) obsahu,
- společný návrh aplikací a rozhraní pro programování aplikací (API) zvyšuje flexibilitu a podporuje další výhody navrhovaného distribuovaného souborového systému.

Architekturu GFS tvoří (Ghemawat, a další, 2003, str. 21) jeden řídící uzel a množství datových uzlů, v jejichž rámci jsou data ukládána do částí (chunk) o defaultní velikosti 64 megabajtů složených z bloků o fixní velikosti defaultně 64 kilabajtů, přičemž tyto části jsou v rámci datových uzlů ukládány jako soubory v rámci tradičního lokálního souborového systému a současně jsou pro zajištění odolnosti proti výpadkům replikovány napříč clusterem, defaultně tříkrát. Řídící uzel zajišťuje (Ghemawat, a další, 2003, str. 21) správu veškerých metadat, především informací o tom, kde se data v clusteru nacházejí. Aplikace/uživatelé (Ghemawat, a další, 2003, str. 21) interagují s GFS prostřednictvím klienta, a to s řídícím uzlem ohledně metadat a přímo s příslušnými datovými uzly ohledně operací zápisu a čtení. S cílem (Ghemawat, a další, 2003, str. 28) eliminace potenciálního SPOF, které řídící uzel představuje, je také tato komponenta, respektive její stav replikován napříč clusterem.

Společnost Google tak implementovala vlastní proprietární distribuovaný systém, který řešil problémy spojené s efektivním ukládáním rozsáhlých souborů dat nad komoditním hardware, a to vysoce škálovatelným, proti výpadkům a chybám odolným a vysoce výkonným způsobem.

V roce 1997 softwarový návrhář Doug Cutting (Singh, a další, 2019, str. 9) začal pracovat na knihovně pro full-textové vyhledávání nesoucí název Lucene, kterou později uvolnil jako open source a jež se stala úspěšným samostatným projektem, který zastřešuje Apache Software Foundation (ASF). V roce 2002 Doug Cutting a Mike Cafarella, profesor na Michiganské univerzitě, (Singh, a další, 2019, str. 9) začali společně pracovat na projektu enginu pro indexaci webových stránek, který dostal název Nutch, přičemž tento software byl později uvolněn jako open source (v současnosti je rovněž projektem ASF) a v podstatě představoval konkurenci proprietárního software využívaného pro indexování internetu vyhledávačem Google.

Při rozvoji prohledávače webu Nutch (Singh, a další, 2019, str. 9-10) narazili jeho autoři v oblasti ukládání dat na podobné problémy jako společnost Google při snaze o implementaci GFS, tedy především potřebu zajištění automatické odolnosti proti chybám/selhání strojů/komponent, vyvažování zátěže (load balancing) v distribuovaném

prostředí a prevence ztráty dat i v případě selhání některého ze strojů uchovávajících data. Tyto problémy vyřešili tak, že (White, 2012, str. 9) s využitím poznatků ze studie o GFS zveřejněné společností Google, respektive autory GFS, v roce 2004 vytvořili vlastní open source implementaci DFS nesoucí název distribuovaný souborový systém Nutch (NDNS). Architektura NDNS tudíž v podstatě kopírovala architekturu GFS, která byla stručně charakterizována v předchozím textu.

3.1.2 Vznik a řešení potřeby efektivního zpracování big data

Vyřešení problému efektivního ukládání rozsáhlých souborů dat na škále odpovídající internetu s využitím GFS prakticky ihned determinovalo potřebu řešení dalšího problému, a to efektivního zpracování takto distribuovaně ukládaných dat. Řešení této výzvy představila i v tomto případě jako první společnost Google, a to v roce 2004, když prezentovala vlastní proprietární implementaci MapReduce, (Dean, a další, 2004, str. 137) „*... programovacího modelu a související implementace pro zpracování a generování rozsáhlých souborů dat.*“ Jedná se o (Dean, a další, 2004, str. 137) programovací model, v jehož rámci uživatelé specifikují funkci map zajišťující zpracování páru klíč-hodnota a generující množinu pracovních páru klíč-hodnota, a dále funkci reduce, která zajišťuje sloučení všech vygenerovaných pracovních páru klíč-hodnota s identickým klíčem. Programovací model MapReduce a jeho implementace pak zajišťuje, že (Dean, a další, 2004, str. 137) programy vytvořené ve funkcionálním MapReduce stylu jsou automaticky paralelizovány a realizovány v rámci rozsáhlého clusteru komoditních strojů, přičemž běhové prostředí zabezpečuje rozdělení vstupních dat, naplánování a řízení provedení programu napříč množinou strojů tvořících cluster, ošetření chyb a selhání strojů a také řízení komunikace mezi jednotlivými stroji.

Architektura MapReduce se z hlediska nasazení do značné míry (Dean, a další, 2004, str. 139) podobá řešení GFS, jelikož se také jedná o cluster zahrnující řídící (master) uzel a množství pracovních uzlů, v jejichž rámci dochází k provedení vlastních dílčích výpočtů. Každá (Dean, a další, 2004, str. 139) úloha (job) sestává z množiny úkolů (tasks), přičemž uživatelé úlohy zadávají do plánovacího systému (scheduler), který běží na řídícím uzlu a zajišťuje rozdělení úkolů mezi pracovní uzly clusteru. Vstupní soubory jsou rovněž (Dean, a další, 2004, str. 140) rozdělovány do částí o defaultní velikosti 64 megabajtů, řídící program pak zajišťuje management a realizaci jednotlivých map a reduce úkolů, přičemž výstup MapReduce je uložen v souborech, jejichž množství odpovídá počtu realizovaných reduce úkolů.

Jednu z hlavních předností této implementace MapReduce představuje (Dean, a další, 2004, str. 141) tzv. lokalita výpočtů, tj. funkcionality, která na úrovni plánovače na řídícím uzlu zajišťuje, že jednotlivé úkoly jsou distribuovány na uzly, kde se nacházejí data, jež jsou předmětem zpracování, což eliminuje/minimalizuje potřebu přesouvat data v clusteru a výrazně snižuje zatížení sítě. MapReduce tak pro zefektivnění běhu aplikací (Dean, a další, 2004, str. 141) využívá vlastnosti GFS, především ukládání dat do částí o defaultní velikosti 64 megabajtů a replikace podporující možnost realizace úkolu map/reduce z repliky i v případě, že primární data nejsou dostupná.

Společnost Google tak vyřešila potřebu efektivního zpracování rozsáhlých souborů distribuovaně uložených dat splňujících charakteristiky big data vytvořením vlastního proprietárního programovacího modelu a implementace MapReduce umožňující i programátorům bez znalostí paralelních a distribuovaných systémů snadno využít zdroje rozsáhlého distribuovaného systému pro distribuované zpracování big data a zajistit potřebnou výkonnost, odolnost proti chybám a škálovatelnost.

Jelikož autoři projektu Nutch rovněž potřebovali vyřešit problémy spojené s potřebou zajistit efektivní distribuované zpracování rozsáhlých souborů dat, také v tomto případě (Singh, a další, 2019, str. 10) se inspirovali řešením, které představila společnost Google a následně vytvořili vlastní open source implementaci MapReduce, která v podstatě kopírovala architekturu MapReduce navrženou společností Google. Na začátku roku 2005 tak měli Cutting a Cafarella k dispozici (White, 2012, str. 10) funkční open source implementaci MapReduce a do poloviny téhož roku modifikovali všechny algoritmy projektu Nutch tak, aby využívaly NDFS a MapReduce. Vytvořili tak open source programovací model a jeho implementaci umožňující efektivně realizovat distribuované zpracování rozsáhlých souborů dat, při zajištění abstrakce od mechanismů zajišťujících mimo jiné výkonnost, škálovatelnost, paralelizaci a odolnost proti výpadkům.

3.2 Vznik a rozvoj frameworku Apache Hadoop

3.2.1 Vznik projektu Hadoop

Projekt Hadoop vznikl původně (White, 2012, str. 10) jako podprojekt v rámci projektu Lucene, když v únoru roku 2006 došlo k přesunutí open source implementace distribuovaného souborového systému NDFS a programovacího modelu a implementace pro distribuované zpracování rozsáhlých souborů dat MapReduce z projektu Nutch do samostatného podprojektu Hadoop¹⁰. Jako zakladatelé projektu Hadoop jsou tudíž označováni Doug Cutting a Mike Cafarella.

Prvním oficiálním vydáním (Bhushan, 2018, str. 9) se stala minoritní verze Hadoop 0.10.0, přičemž toto vydání zahrnovalo (Bonaci, 2015) knihovny představující jádro (Hadoop Common), distribuovaný souborový systém Hadoop (HDFS) a programovací model a implementaci pro distribuované zpracování rozsáhlých souborů (MapReduce).

Problémy s ukládáním a zpracováním rozsáhlých souborů dat v souvislosti s vlastním internetovým vyhledávacím enginem (Bonaci, 2015) řešila rovněž společnost Yahoo!. V roce 2006 bylo v rámci této společnosti (Bonaci, 2015) přijato rozhodnutí o tom, že pro dané účely bude využit právě vznikající projekt Hadoop, na jehož využití, rozvoj a další optimalizaci byl následně alokován prakticky celý tým zabývající se technologií pro ukládání a zpracování rozsáhlých souborů dat pro internetový vyhledávací engine Yahoo! a

¹⁰) Doug Cutting si název (White, 2012, str. 9) „vypůjčil“ od svého syna, který slovem „hadoop“ označoval hračku žlutého plyšového slona. Logo projektu Apache Hadoop rovněž představuje žlutý slon, který je integrován také do loga některých dalších projektů z ekosystému Apache Hadoop, jako například Apache Hive.

současně společnost najala jednoho ze zakladatelů projektu Hadoop Douga Cuttinga. Zájem společnosti Yahoo! na využití Hadoop tak determinoval etablování a další rozvoj projektu. Ještě v roce 2006 přitom tým pod vedením Douga Cuttinga dokázal (Singh, a další, 2019, str. 11) v rámci clusteru Hadoop o velikosti 188 uzlů během necelých 48 hodin provést setřídění dat o objemu 1,8 terabajtů, a následně nasadit cluster o velikosti 300 uzlů, který byl posléze rozšířen na 600 uzlů.

V průběhu roku 2007 již tým ve společnosti Yahoo! (Singh, a další, 2019, str. 11) úspěšně provozoval dva clustery Hadoop o velikosti 1000 uzlů. V téžem roce byla vydána verze Hadoop (Bhushan, 2018, str. 9) zahrnující HBase, tj. open source distribuovaný škálovatelný big data databázový systém. Současně tým ze společnosti Yahoo! pod názvem Pig (Bhushan, 2018, str. 9) vytvořil vysokoúrovňový jazyk pro řízení toku dat a framework pro zajištění realizace paralelních výpočtů, což přineslo další formu abstrakce umožňující jednodušším způsobem vytvářet a efektivně realizovat programy pro škálovatelné distribuované zpracování a analýzu big data v rámci Hadoop clusteru.

V lednu 2008 se framework Hadoop stal (White, 2012, str. 10) samostatným projektem zastřešovaným ASF. V dubnu téhož roku byl Hadoop (White, 2012, str. 10) nasazený v clusteru sestávajícím z 910 uzlů použit pro seřazení jednoho terabajtu dat v rekordním čase 209 vteřin, přičemž společnost Google tento rekord pokořila, když se jí v listopadu podařilo danou úlohu realizovat v čase 68 vteřin. V roce 2008 začala (Singh, a další, 2019, str. 11) společnost Yahoo! využívat Hadoop v ostrém provozu pro indexaci internetu a do clusteru Hadoop denně nahrávala data o objemu 10 terabajtů. Rok 2008 se stal významným také (Singh, a další, 2019, str. 11) z hlediska šíření povědomí a praktického využití Hadoop, jelikož se konal první summit Hadoop, na stránce prezentující firmy využívající Hadoop již figurovalo 20 subjektů a vznikla distribuce Hadoop nesoucí název Cloudera. Kromě Yahoo! již v této době využívaly Hadoop (White, 2012, str. 10) například společnosti Last.fm, Facebook, LinkedIn či New York Times.

V roce 2009 již (Singh, a další, 2019, str. 12) společnost Yahoo! provozovala 7 clusterů zahrnujících 24 tisíc strojů. V tomto roce doznala struktura projektu Hadoop jistých změn, když (Singh, a další, 2019, str. 12) byly HDFS a MapReduce vyčleněny jako samostatné podprojekty. Na trh byla v roce 2009 uvedena (Singh, a další, 2019, str. 12) další distribuce frameworku Apache Hadoop nesoucí název MapR. Spoluzakladatel Hadoop Doug Cutting v srpnu daného roku (Bonaci, 2015) přešel ze společnosti Yahoo! do společnosti Cloudera, která uvedla na trh první distribuci Hadoop mimo rámec projektu Hadoop, a v níž začal působit jako hlavní architekt.

V následujícím roce byla zvýšena (Singh, a další, 2019, str. 12) bezpečnost Hadoop, když byla přidána podpora autentizace s využitím Kerberos. Současně byla vydána (Bonaci, 2015) první stabilní verze dalšího významného podprojektu, na kterém společnost Facebook začala pracovat již o dva roky dříve, a sice Hive poskytujícího funkcionality datového skladu pro distribuovaně uložená big data a umožňující prakticky poprvé v historii pro dotazování big data využít jazyk podobný SQL.

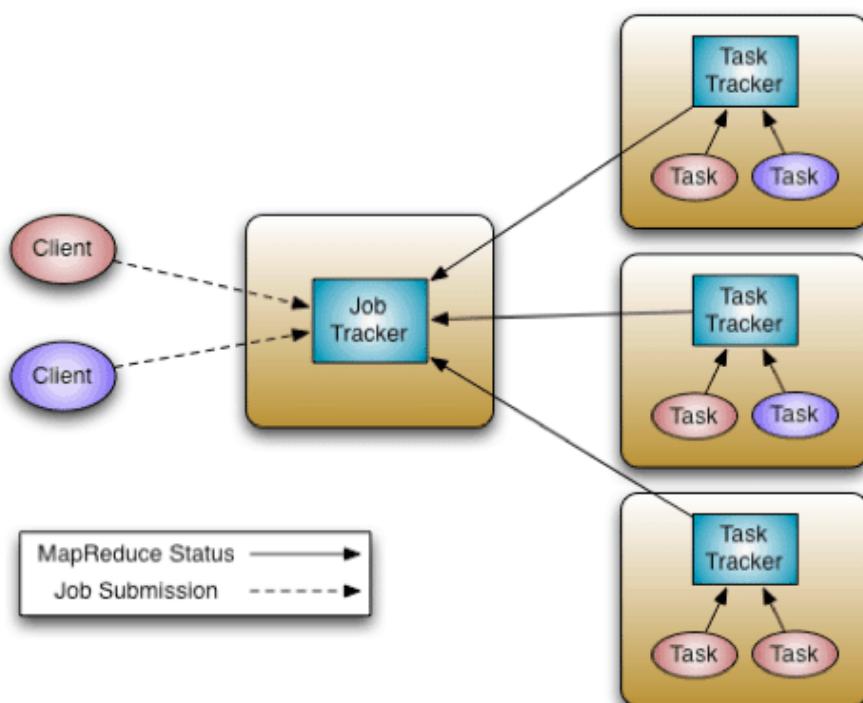
Rok 2011 přinesl další významný rozvoj projektu Hadoop, když (Singh, a další, 2019, str. 12) vývojáři společností Facebook, LinkedIn, eBay a IBM odeslali do repositáře Hadoop 200 tisíc řádků kódu. V téžem roce vznikla třetí významná distribuce Hadoop (Bonaci,

2015) zastřešovaná společností Hortonworks, kterou založilo osm vývojářů pracujících na projektu Hadoop v rámci společnosti Yahoo! v čele s Ericem Baldeschwielerem, a to s podporou společnosti Yahoo!, která v daném roce (Singh, a další, 2019, str. 12) provozovala cluster Hadoop o velikosti dosahující 42 tisíc uzlů.

V roce 2012, šest let po vzniku projektu Hadoop, byla (Singh, a další, 2019, str. 12) vydána první produkční verze Hadoop 1.0. Počet přispěvatelů do projektu Hadoop (Bonaci, 2015) dosáhl v tomto roce 1200. V též roce nicméně bylo přijato zásadní rozhodnutí týkající se architektury projektu Hadoop. Jelikož byla tehdejší architektura podprojektu MapReduce (Bonaci, 2015) vyhodnocena jako úzké místo projektu Hadoop, rozhodla komunita Hadoop o vyčlenění některých funkcí z podprojektu MapReduce a zajištění této a další funkcionality prostřednictvím nového samostatného podprojektu správce zdrojů (YARN). Nadále tak byla využívána architektura označovaná jako Hadoop 1 (verze 1.x) a současně byly zahájeny práce na vývoji a testování architektury nové. Následuje stručný popis původní architektury Hadoop 1 (někdy také označované jako MapReduce 1).

Jak již bylo uvedeno, mezi hlavní komponenty původního projektu Hadoop patřily celky reprezentované podprojekty Hadoop Common, HDFS a MapReduce. Modul Hadoop Common zahrnoval sdílené knihovny využívané ostatními moduly, modul HDFS zajišťoval funkcionality distribuovaného souborového systému, která již byla v předchozím textu stručně popsána. Modul MapReduce pak poskytoval (Fasale, a další, 2015, str. 42) funkcionality API pro koncové uživatele, vlastní framework MapReduce poskytující implementaci a běhové prostředí pro programovací model MapReduce a současně systém MapReduce zajišťující podpůrnou infrastrukturu potřebnou pro běh aplikací MapReduce, včetně správy zdrojů clusteru, plánování úloh, atd.

Obr. 3-1 Architektura MapReduce v rámci Hadoop 1



Zdroj: data a zpracování (Murthy, 2012)

Architektura Hadoop 1 je v rozsahu MapReduce znázorněna na Obr. 3-1, z něhož je patrné, že v případě MapReduce 1 vždy klient úlohu zadával řídící komponentě MapReduce (MR) označované jako JobTracker, která zajišťovala plánování a řízení realizace dané úlohy, tedy jednotlivých úkolů, jejichž provedení řídily komponenty nesoucí označení TaskTracker na příslušných pracovních uzlech, kde se předmětná data nacházela. Přestože na obrázku není uvedena komponenta HDFS, znázorňuje toto vyobrazení v podstatě architekturu Hadoop 1 i v rozsahu HDFS, jelikož v rámci první majoritní verze Hadoop se řídící komponenta MapReduce (JobTracker) nacházela na řídícím (master) uzlu clusteru, společně se jmenným uzlem HDFS a MapReduce komponenty zajišťující vlastní realizaci (TaskTracker) úkolů tvořících úlohu distribuovaného zpracování dat se nacházely na jednotlivých pracovních (worker) uzlech clusteru hostujících rovněž datové uzly HDFS.

Komponenta MapReduce nesoucí název JobTracker zajišťovala (Murthy, 2012) řízení všech přijatých úloh, včetně řízení zdrojů (pracovních uzlů, komponent TaskTracker), sledování využití a dostupnosti zdrojů a také řízení životního cyklu úloh (plánování úkolů tvořících úlohy, monitorování realizace, zajišťování odolnosti proti chybám v úkolech, atd.). Architektura MapReduce 1 (Fasale, a další, 2015, str. 42) postrádala potřebnou škálovatelnost, výkon a současně možnost efektivně realizovat úlohy jiné než typu MapReduce, respektive dávkového zpracování big data.

Uvedené problémy týkající se modulu MapReduce projektu Hadoop, představovaly hlavní důvody, které přiměly komunitu Hadoop přijmout rozhodnutí o restrukturalizaci stávající architektury projektu Hadoop a modulu MapReduce a iniciovat zahájení prací na nové řadě Hadoop 2.x. Následuje popis vývoje, který vedl k vydání nové majoritní verze Hadoop a charakteristika dalšího využití a rozvoje nové verze frameworku Apache Hadoop.

3.2.2 Rozvoj projektu Hadoop

Přestože práce na samostatném správci zdrojů typu YARN byly zahájeny (Bonaci, 2015) již v roce 2006, o vyčlenění dané funkcionality z MapReduce a zařazení do samostatného podprojektu Hadoop, bylo rozhodnuto až v roce 2012. V tomtéž roce byly zahájeny intenzivní práce na přípravě nové majoritní verze Hadoop zahrnující YARN, přičemž (ASF, 2019m) od května 2012 do srpna 2013 byly realizovány práce na alfa verzích Hadoop 2 (2.0.0 až 2.0.6) následované dvěma beta verzemi Hadoop 2 (2.1.0, 2.1.1), jež byly vydány v srpnu a září roku 2013.

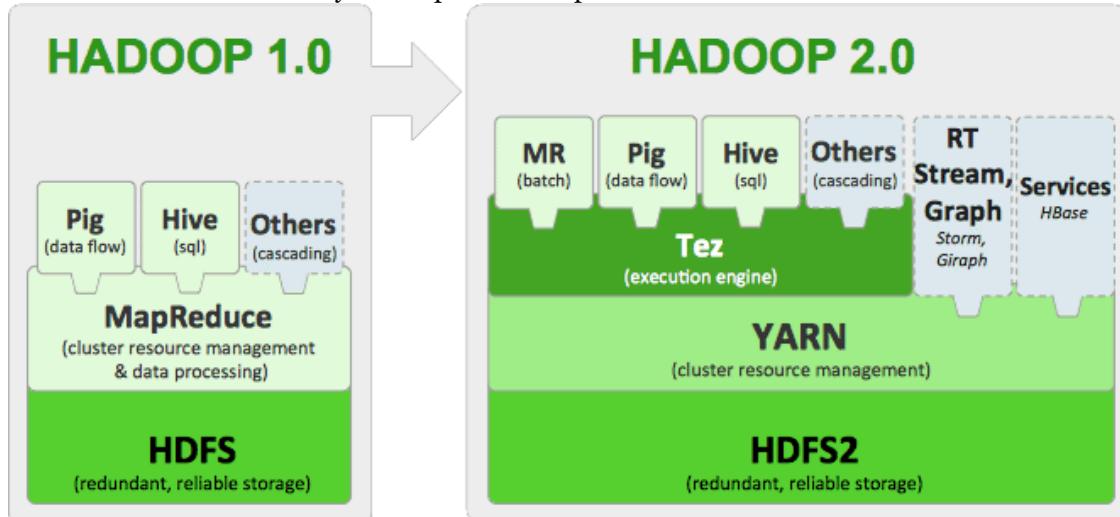
První produkční verze Hadoop 2 (2.2.0) byla vydána (ASF, 2019n) 13. října roku 2013. Klíčový posun oproti předchozí majoritní verzi Hadoop 1 představovalo právě využití samostatného podprojektu správce zdrojů YARN pro zajištění řízení realizovaných úloh distribuovaného zpracování big data, včetně řízení zdrojů, sledování využití a dostupnosti zdrojů, řízení životního cyklu úkolů tvořících úlohy, atd. Společnost Yahoo! začala tuto verzi (Singh, a další, 2019, str. 12) v produkčním prostředí využívat již v roce 2013.

V dalších letech přestala být verze Hadoop 1.x prakticky využívána na úkor modernizované architektury Hadoop 2. V letech 2014 až 2016 byly (ASF, 2019n) vydány další minoritní verze Hadoop 2 (2.3.x až 2.6.x), které většinou přinesly rozšíření a/nebo vylepšení/přidání funkcionality v podprojektech HDFS a YARN, spíše než MapReduce. Největší množinu změn zahrnovala verze (ASF, 2019n) Hadoop 2.6.0 přinášející v případě HDFS podporu

koncepcí heterogenního úložiště a vylepšení podpory šifrování, v případě YARN funkcionality tzv. rolling upgrades podporující minimalizaci výpadků při realizaci aktualizací komponent, dále rezervační substituce YARN, automatické sdílené mezipaměti pro aplikační artefakty, podporu dlouhodobě provozovaných služeb, podporu využití Docker kontejnerů v rámci YARN, atd.

Posun v architektuře Hadoop 1 a Hadoop 2 je dobře patrný z Obr. 3-2. Základním modulem Hadoop je v obou případech distribuovaný souborový systém HDFS, přičemž ačkoli to není z Obr. 3-2 zcela zřejmé, v obou případech je HDFS nasazován ve složení komponent jmenného uzlu (primárního a případně sekundárního) a datových uzlů. Mezi MapReduce a HDFS nicméně v případě Hadoop 2 figuruje nová vrstva, a to právě správce zdrojů YARN zastávající množství funkcí, které dříve zajišťoval podprojekt MapReduce. YARN je někdy označován jako operační systém Hadoop, který v rámci Hadoop 2 zajišťuje správu zdrojů a další podporu realizace distribuovaných úloh. Enginy zajišťující realizaci úloh distribuovaného zpracování big data pak nově využívají pro získání potřebných zdrojů a jejich správu právě YARN. Podprojekt MapReduce, od Hadoop 2 označovaný jako MapReduce 2, je tak v rámci druhé verze Hadoop zaměřen výhradně na zajištění programovacího modelu MapReduce a jeho implementace. Z Obr. 3-2 je rovněž patrná jedna z hlavních předností architektury Hadoop využívající YARN, a to možnost provozovat v clusteru Hadoop s využitím YARN další enginy (nad rámec MR) umožňující realizovat i jiné úlohy než pouze dávkové zpracování big data.

Obr. 3-2 Posun od architektury Hadoop 1 k Hadoop 2



Zdroj: data a zpracování (Murthy, 2013)

Architektura správce zdrojů YARN (Fasale, a další, 2015, str. 13) zajišťuje oddělení správy zdrojů a plánování a monitorování realizace jednotlivých úloh (aplikací) zpracování dat. Mezi hlavní komponenty YARN patří (Fasale, a další, 2015, str. 13):

- správce zdrojů (ResourceManager) – nasazen na řídícím uzlu zajišťuje management zdrojů na úrovni celého clusteru,
- správce uzlu (NodeManager) – nasazen na každém pracovním uzlu, kde zajišťuje management daného pracovního uzlu a zdrojů, které jsou v rámci daného uzlu

využívány a rovněž monitoring životního cyklu kontejnerů, v jejichž rámci jsou na daném uzlu realizovány jednotlivé úkoly (task) úloh (job) zpracování big data,

- správce aplikace (ApplicationMaster) – nasazen samostatně pro každou jednotlivou aplikaci, která je do YARN odeslána; zajišťuje získání zdrojů pro aplikace od správce zdrojů, a dále realizaci a monitoring jednotlivých úkolů napříč clusterem ve spolupráci s příslušnými správci uzlu.

Díky využití YARN zajišťuje architektura Hadoop pro distribuované zpracování big data (Fasale, a další, 2015, str. 10-11) masivní škálovatelnost, efektivnější využití zdrojů clusteru, možnost využití více uživateli při zajištění jejich izolace (multitenance), flexibilitu z hlediska modelu správy zdrojů, prakticky neomezenou podporu programovacích modelů pro zpracování big data (nejen MR), bezpečný a transparentní provoz, spolehlivost a dostupnost a v neposlední řadě zpětnou kompatibilitu.

Vydání verzí Hadoop řady 2 přinesla množství nových funkcí v rámci jednotlivých podprojektů Hadoop, mezi které patřily (Murthy, 2013) mimo jiné vysoká dostupnost HDFS (mimo jiné prostřednictvím konceptu sekundárního jmenného uzlu), funkcionality federovaných HDFS úložišť, možnost vytváření a využívání snímků (snapshot) v rámci HDFS, podpora provozu Hadoop na platformě Windows, vysoká dostupnost komponent YARN, atd.

3.2.3 Současný stav projektu Hadoop

Již v září 2016 (ASF, 2019m) byla vydána první alfa verze další majoritní verze Hadoop 3. Souběžně byly dále uvolňovány (ASF, 2019m) nové minoritní verze Hadoop 2, až do Hadoop 2.9.2 v listopadu roku 2018. Téměř dva roky probíhalo testování alfa a beta verzí Hadoop 3, přičemž první produkční verze Hadoop 3.0.0 byla na trh uvedena (ASF, 2019m) v březnu 2018 a zatím poslední stabilní verzi představuje Hadoop 3.2.1 ze září roku 2019.

Projekt Hadoop je dále aktivně rozvíjen, minoritní verze jsou uvolňovány (ASF, 2019n) pravidelně v intervalu několika měsíců, přičemž majoritní verze determinují potřebu změn kódu uživatelského, respektive aplikačního kódu z důvodu nekompatibilních změn v rámci API. Rovněž komunitu Hadoop lze označit za aktivní, prakticky každý rok (Singh, a další, 2019, str. 12) se koná Hadoop summit, jehož se účastní množství odborníků nejen z úzkého okruhu projektu Hadoop, ale také z oblasti big data a dalších disciplín, jejichž poznatky a/nebo praktiky BDA využívají a/nebo v jejichž rámci je využíván Hadoop.

Na stránce prezentující firmy, které aktivně využívají Hadoop v produkčním prostředí je v současné době uvedeno (ASF, 2019t) téměř 200 firem, mezi nimiž se vyskytují taková jména jako například Amazon, Adobe, Alibaba, AOL, eBay, Facebook, Google, IBM, LinkedIn, LiveBet, Microsoft, Rackspace, Spotify, Yahoo! nebo Twitter.

Posun v architektuře mezi verzemi Hadoop 2.x a 3.x nepředstavoval tak zásadní změnu architektonického paradigmatu, jako v případě přechodu z Hadoop 1 na Hadoop 2, jelikož stále platí, že základ Hadoop tvoří HDFS a YARN, nad nímž operují jednotlivé enginy zajišťující zpracování big data dávkově, interaktivně, v reálném čase, atd. Ze strany komunity a vedení projektu Hadoop byla nicméně (Singh, a další, 2019, str. 12-13) naplánována realizace změn a/nebo nových funkcionalit představujících z hlediska

rozsahu a dopadu v podstatě skokový posun rozsahu funkcionality Hadoop, a proto bylo rozhodnuto o jejich implementaci v rámci nové verze Hadoop 3 zaměřené na poskytnutí takzvaně plně produkčního/firemního (enterprise) frameworku Hadoop. Mezi hlavní faktory determinující vydání nové majoritní verze Hadoop 3 patřily především následující oblasti změn (Singh, a další, 2019, str. 13-14):

- rozsáhlá množina oprav a výkonného zlepšení – objem majoritních a/nebo minoritních změn odesílaných vývojáři do repositáře Hadoop se zvýšil natolik, že bylo nutné je začlenit do nové majoritní verze Hadoop,
- snížení nákladů (overhead) na zajištění odolnosti HDFS proti chybám – replikace byla doplněna konceptem označovaným jako erasure coding (EC),
- skokové zlepšení služby YARN zajišťující správu běžících a dokončených aplikací – služba YARN Timeline Server byla optimalizována z hlediska škálovatelnosti, výkonnosti a spolehlivosti,
- optimalizace výstupního kolektoru map – nativní implementace výstupního kolektoru map umožňuje v určitých případech dosahovat dvojnásobně až trojnásobně vyšší rychlosti realizace map úkolů,
- zvýšení dostupnosti jmenného uzlu – vysoká dostupnost jmenného uzlu je ještě více podpořena možností nasazení více aktivních (standby) záložních jmenných uzlů oproti jednomu pasivnímu (secondary) v rámci Hadoop 2,
- eliminace závislosti na rozsahu dočasných (ephemeral) portů v rámci OS – defaultní porty služeb HDFS (primární a sekundární jmenné uzly, datové uzly) byly v rámci Hadoop 3 vyčleněny mimo rozsah dočasných portů (32768-61000) hostitelské platformy Linux (nově zpravidla na portech 98xx),
- zajištění vyvažování zátěže mezi disky v rámci datových uzlů – implementace utility pro zajištění vyvážení zátěže mezi disky v případě, že datový uzel HDFS využívá více disků, což podporuje efektivnější rozložení dat v rámci datového uzlu.

Spíše než modifikací vysokoúrovňové architektury doznala v rámci Hadoop 3 rozsáhlých změn či restrukturalizace architektura vybraných komponent/podprojektů Hadoop, především HDFS a YARN. Skutečný rozsah změn, které nová majoritní verze frameworku Hadoop přinesla, ilustruje například skutečnost, že (Tan, a další, 2018a) mezi verzemi 2.8.3 a 3.1.0 bylo v rámci projektu Hadoop realizováno celkem 3714 požadavků, přidáno 954885 řádků kódu, odstraněno 157883 řádků kódu a modifikováno 7477 souborů.

Hlavní změny v rámci Hadoop 3 jsou uvedeny v Tab. 3.1. Naprosto zásadní změnu představovalo doplnění strategie replikace dat v clusteru HDFS o alternativu v podobě konceptu erasure coding. Tento způsob uložení dat nevyužívá ochranu proti ztrátě dat formou replik, ale (Fontaine, 2018) dodatečných erasure code bloků představujících paritu nad rámcem jediné kopie dat, která umožňuje v případě ztráty dat provést obnovu. Oproti replikaci dochází ke značné úspoře místa a zásadnímu snížení provozních nákladů při zajištění vysoké úrovně ochrany proti výpadkům (fault tolerance). Nevýhodu EC představuje vysoká náročnost a potenciálně delší doba obnovy v případě, že k výpadku/ztrátě dat skutečně dojde.

Množství nových funkcí a vlastností bylo implementováno v případě správce zdrojů YARN a většina těchto změn vykazovala přímý vliv na další enginy a dílčí frameworky, které jej v rámci Hadoop využívají pro realizaci aplikací a nově také dlouhodobě běžících služeb.

Tab. 3.1 Srovnání vlastností architektury Hadoop 2 a Hadoop 3

Funkcionalita/vlastnost	Hadoop 2	Hadoop 3
HDFS – způsob zajištění odolnosti proti chybám	Replikace	Erasure coding
HDFS – provozní náklady na zajištění odolnosti proti chybám	Až 200%	Až 150%
HDFS – počet záložních jmenných uzlů	Max 1	2 a více
HDFS – vyvažování zátěže a rozložení dat	Mezi jednotlivými datovými uzly	Mezi disky každého uzlu
YARN – YARN Timeline server	Problémy se škálovatelností	Vysoce škálovatelný a spolehlivý
YARN – podpora využití GPU	Omezená	Plně nativní
YARN – podpora izolace GPU	Žádná	Per GPU zařízení
YARN – podpora Docker kontejnerů	Minimální	Plnohodnotná
YARN – podpora dlouhodobě běžících služeb	Omezená	Plnohodnotná
Správa paměti (heap) aplikací a služeb	Ruční konfigurace	Automatické ladění
Minimální verze Java	Java 7	Java 8

Zdroj: data (DataFlair, 2019; Tan, a další, 2018b; He, a další, 2018), vlastní zpracování

Zajištění plné podpory a izolace grafických procesorů (GPU) ze strany YARN umožňuje využívat daný typ procesorů pro realizaci náročnějších úloh ML, hlubokého učení (DL), atd. Obecně tak lze konstatovat, že architektura Hadoop 3 přinesla řadu vylepšení a eliminaci dřívějších omezení hlavních podprojektů a celkově podpořila efektivní nasazení a provoz Hadoop plně produkčního charakteru.

3.3 Architektura a funkcionalita frameworku Hadoop

Jak již bylo uvedeno, framework Hadoop řeší potřeby distribuovaného uložení a zpracování big data. V obecné rovině lze konstatovat, že Hadoop pro tyto účely využívá architekturu (Alapati, 2017, str. 12-13) clusteru komoditních strojů/serverů, které jsou organizovány v rámci jednotlivých stojanů (rack) a vzájemně propojeny prostřednictvím sítě, a na nichž je instalován OS hostující procesy Hadoop komponent zajišťující uchování a/nebo zpracování dat. Architektura clusteru Hadoop je tak zpravidla tvořena některou z kombinací následujících entit (Alapati, 2017, str. 13):

- množina řídících (master) uzlů (serverů), v jejichž rámci běží/jsou provozovány procesy podporující klíčové frameworky/enginy projektu Hadoop,
- množina pracovních (worker) uzlů hostujících úložiště (HDFS) a výpočetní zdroje (YARN),

- jeden nebo více hraničních (edge) uzlů, v jejichž rámci neběží žádné procesy frameworku Hadoop, a jež zajišťují pouze přístup do Hadoop clusteru za účelem spuštění aplikací zpracování dat,
- jedna nebo více relačních databází uchovávající repositáře metadat pro frameworky/enginy projektu/ekosystému Hadoop,
- uzly dedikované pro speciální frameworky/enginy projektu/ekosystému Hadoop (např. Kafka, Storm, aj.).

Uvedenou množinu entit tvořících architekturu Hadoop je v praxi možné nasadit a provozovat v různém rozsahu, především pokud jde o projekty/komponenty ekosystému Hadoop a případně další z ekosystému big data a současně v některém z následujících režimů (Alapati, 2017, str. 61):

- samostatná instalace – všechny služby Hadoop běží v rámci jediného virtuálního stroje Java (JVM), žádná neběží jako proces na pozadí (daemon), data jsou ukládána v rámci lokálního FS, nikoli HDFS, úlohy MapReduce běží pouze s jednou funkcí map a jednou funkcí reduce,
- pseudo-distribuovaná instalace – simulace clusteru, všechny služby běží jako procesy na pozadí, ale pouze v rámci jediného uzlu,
- plně distribuovaná instalace – reálný cluster tvořený více uzly, každá služba běží jako proces na pozadí, a to zpravidla na několika uzlech současně; v závislosti na konkrétních podmínkách a požadavech jsou případně zajištěny další aspekty produkčního clusteru (zajištění ochrany dat proti selhání – replikace/erasure coding, vysoká dostupnost relevantních prvků a komponent architektury, zajištění řízení přístupu a případně audit přístupu ke clusteru, komponentám a datům, eliminace potenciálních SPOF, atd.).

Framework Apache Hadoop byl napsán majoritně (Lublinsky, a další, 2013, str. 143) v programovacím jazyku Java. Zatímco pro administrátory frameworku Hadoop se znalost tohoto programovacího jazyka jeví jako prakticky nezbytná, vývojáři aplikací využívajících některý z projektů frameworku a/nebo ekosystému Hadoop (Alapati, 2017, str. 21) mohou zpravidla kromě programovacího jazyka Java použít Python či Scala. Vývojáři a uživatelé pak mohou většinou bez problémů využít znalostí SQL, jelikož (Alapati, 2017, str. 21) množství projektů pro zpracování big data z ekosystému Hadoop poskytuje abstrakci použitých programovacích modelů, umožňující využít SQL a/nebo jeho podmnožinu či specifický dialekt.

Hadoop lze provozovat nad platformami Windows i Linux, nicméně (Alapati, 2017, str. 21) většina instalací je provozována výhradně nad OS Linux.¹¹ Pro administraci Hadoop je tudíž klíčová znalost základních konceptů platformy Linux a administrace relevantního

¹¹⁾ Společnost Microsoft poskytuje v rámci vlastní cloudové výpočetní platformy Microsoft Azure službu clusteru frameworku Apache Hadoop pod názvem HDInsight. V minulosti bylo možné v rámci této služby provozovat vybrané distribuce frameworku Apache Hadoop nejen nad OS Linux, ale také nad Windows, resp. speciální cloudovou verzí Windows Server. Není bez zajímavosti, že společnost Microsoft (Microsoft, 2019b) tuto možnost zrušila a od verze HDInsight 3.4 je možné v rámci HDInsight provozovat clustery Hadoop pouze nad OS Linux.

Linux OS (Red Hat Enterprise Linux, Oracle Linux, SUSE Linux Enterprise Server, CentOS, Debian, Ubuntu, aj.). Současně platí, že ačkoli je možné s využitím projektů frameworku a ekosystému Hadoop vlastními silami vytvořit plně distribuované produkční prostředí, organizace v praxi zpravidla majoritně (Alapati, 2017, str. 4) využívají některou z dostupných distribucí frameworku Apache Hadoop.

Distribuce frameworku Apache Hadoop a jejich komparace představují hlavní téma této práce a podrobně se jim věnují následující kapitoly. Před provedením podrobného popisu a komparace jednotlivých distribucí je nicméně nezbytné získat relevantní znalosti týkající se samotného projektu frameworku Apache Hadoop a dalších projektů z ekosystému Apache Hadoop. V dalším textu této podkapitoly proto následuje základní charakteristika architektury a funkcionality jednotlivých komponent projektu a ekosystému frameworku Apache Hadoop.¹²

3.3.1 Projekt Hadoop

Apache Hadoop představuje (ASF, 2019b) „... framework zajišťující distribuované zpracování rozsáhlých souborů dat napříč clustery počítačů s využitím jednoduchých programovacích modelů.“ Architektura Hadoop je koncipována jako (ASF, 2019b) distribuovaná masivně škálovatelná umožňující efektivně a rychle škálovat cluster na úroveň tisíců strojů/uzlů a současně způsobem zabezpečujícím odolnost proti chybám zajistit spolehlivé, škálovatelné a výkonné distribuované zpracování dat. V současné době¹³ sestává projekt Hadoop (3.1.2) z následujících podprojektů/modulů (ASF, 2019b):

- Hadoop Common – společné komponenty podporující ostatní moduly Hadoop,
- Hadoop Distributed File System (HDFS) – distribuovaný souborový systém poskytující vysoce propustný přístup k datům,
- Hadoop YARN – framework pro plánování úloh a správu zdrojů clusteru,
- Hadoop MapReduce – systém pro paralelní zpracování rozsáhlých souborů dat založený na YARN,
- Hadoop Ozone – objektové úložiště pro Hadoop.

Hadoop Common

Podprojekt Hadoop Common obsahuje (ASF, 2019b) společné/sdílené komponenty, které zajišťují podporu pro ostatní moduly Hadoop. Modul Hadoop Common je někdy označován jako (Antony, a další, 2016, str. 2) „Hadoop Stack“ a zahrnuje primární a základní služby, procesy a sdílené knihovny poskytují abstrakci hostitelského OS a FS, dále

¹²⁾ Další text podkapitoly není zaměřen na strohou reprodukci architektury jednotlivých projektů frameworku a ekosystému Apache Hadoop z dokumentace, ale naopak představení základních principů architektury a funkcionality těchto komponent a jejich role při zajištění procesu BDA s využitím Hadoop. Podrobné informace týkající se architektonických nuancí jednotlivých komponent frameworku a ekosystému Apache Hadoop přesahují rámec této práce a je možné je nalézt například v rámci dokumentace každého jednotlivého projektu.

¹³⁾ Veškerý popis, který následuje v rámci této kapitoly, se vztahuje k aktuálně předposlední nejvyšší dostupné stabilní produkční (generally available) verzi Hadoop 3.1.2 vydané 6. února roku 2019.

potřebné JAR (Java archive) soubory, skripty nezbytné pro spuštění Hadoop a také zdrojový kód a dokumentaci.

V rámci Hadoop 3.1.2 podsložka modulu Hadoop Common obsahující sdílené knihovny zahrnuje celkem 86 souborů reprezentujících knihovny modulu Hadoop common, dále servlet enginu a webového serveru Jetty, knihovny Gson pro serializaci a deserializaci Java objektů do/z JSON, sady nástrojů Jackson pro zpracování dat v Java, frameworku pro REST (Representational State Transfer) webové služby Jersey, knihovny pro autentizaci distribuovaných služeb prostřednictvím Kerberos, frameworku log4j pro zajištění logování, knihovny slf4j pro abstrakci frameworků zajišťujících logování, atd.

Pro všechny produkční moduly frameworku Apache Hadoop je k dispozici klient v podobě rozhraní příkazové řádky (CLI) umožňující interagovat s API projektu/příslušného modulu Hadoop a využívat funkcionality daného modulu Hadoop. Tento klient má podobu programu, který mohou uživatelé spustit z příkazového řádku na platformě Linux (shell/bash) i Windows (cmd/PowerShell). Jedná se o program, jehož název odpovídá modulu Common (`hadoop`), HDFS (`hdfs`), MapReduce (`mapred`), YARN (`yarn`). Základním klientem pro Hadoop je tak program `hadoop`. Všechna uvedená CLI rozhraní (ASF, 2019l) uplatňují jednotnou syntaxi příkazů ve tvaru `shellcommand [SHELL_OPTIONS] [COMMAND] [GENERIC_OPTIONS] [COMMAND_OPTIONS]`. Pro zobrazení obsahu kořenového adresáře distribuovaného souborového systému Hadoop je tak možné použít například příkaz `hadoop fs -ls /`, zatímco pro zobrazení distribuovaně realizovaných aplikací spravovaných ze strany YARN příkaz `yarn application -list`, apod.

Modul Hadoop Common tak představuje esenciální modul sdílený ostatními moduly Hadoop, bez něhož není možné Hadoop využívat.

Hadoop Distributed File System

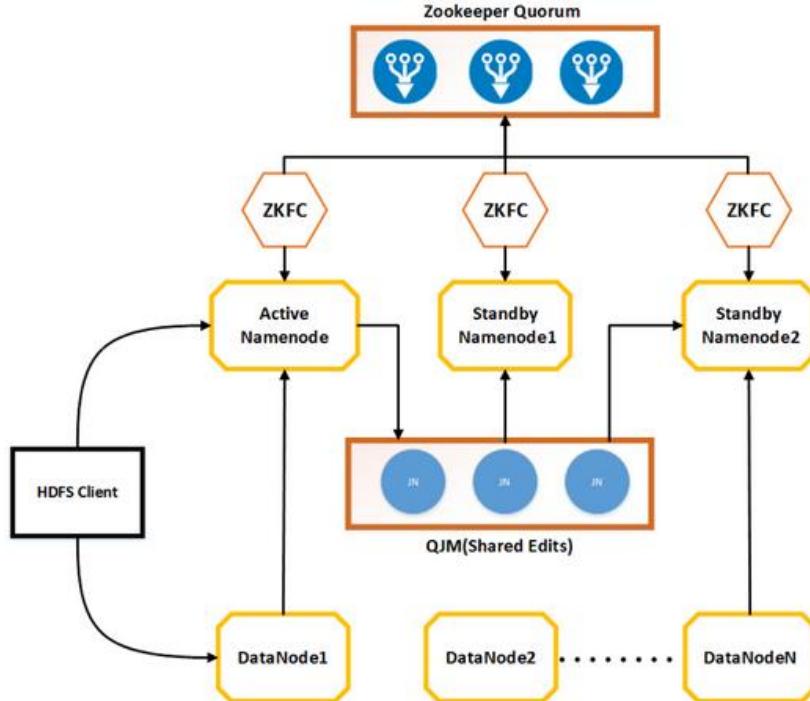
HDFS představuje (ASF, 2019o) „... *distribuovaný souborový systém navržený pro provoz nad komoditním hardware ... je vysoce odolný proti chybám a je navržen pro nasazení na nízkonákladovém hardware ... poskytuje přístup k datům aplikací s vysokou propustností a je vhodný pro aplikace zpracovávající rozsáhlé soubory dat.*“

Architektura HDFS (ASF, 2019o), jak je zřejmé z Obr. 3-3, odpovídá vzoru master/slave, jelikož sestává z jedné primární a případně několika sekundárních záložních (secondary/standby) instancí řídící komponenty jmenného uzlu (NameNode) a množství pracovních instancí pracovní komponenty datového uzlu (DataNode). Jmenný uzel (ASF, 2019o) spravuje metadata týkající se uložených dat (především informace o tom, kde se která data v clusteru nacházejí) a současně řídí přístup klientů k datům, respektive souborům uloženým v rámci HDFS, tj. jednotlivých datových uzlů.

V případě konfigurace pro vysokou dostupnost HDFS je součástí architektury (Singh, a další, 2019, str. 28) jeden nebo více záložních (standby) jmenných uzlů, přičemž tyto jsou kontinuálně monitorovány ze strany služby ZooKeeper (na Obr. 3-3 komponenty ZooKeeper Quorum a ZKFC (ZooKeeper failover controller)), která v případě, že dojde k selhání primárního jmenného uzlu, zajistí jeho okamžité nahrazení (failover) některým ze záložních jmenných uzlů, který se tak stane primárním jmenným uzlem. V rámci

zajištění vysoké dostupnosti (HA) pro HDFS je rovněž využívána (Singh, a další, 2019, str. 37) komponenta QJM (Quorum Journal Manager), která v podstatě zajišťuje orchestraci synchronizace stavu mezi jednotlivými jmennými uzly s využitím takzvaných žurnálových uzlů (JournalNode), v jejichž rámci dochází k logování změn v rámci jmenných uzlů.

Obr. 3-3 Architektura HDFS



Zdroj: data a zpracování (Singh, a další, 2019, str. 26)

Data jsou tak rozprostřena napříč datovými uzly v rámci Hadoop clusteru, přičemž HDFS představuje v podstatě vrstvu nad lokálním FS, kdy data uložená v rámci daného datového uzlu se nacházejí v rámci adresáře `/hdfs/`. HDFS poskytuje (ASF, 2019o) jmenný prostor (namespace) souborového systému, což umožňuje data v rámci HDFS ukládat a organizovat formou tradiční hierarchické struktury složek a souborů, přičemž soubory jsou interně rozděleny do jednotlivých bloků o stejné velikosti, která defaultně činí 128 megabajtů. Díky tomu lze v rámci HDFS ukládat obrovské soubory, které jsou rozděleny do těchto bloků. Současně uživatel, který se dotáže na obsah adresáře přes klienta nebo prostřednictvím grafického rozhraní jmenného uzlu HDFS (zpravidla na portu 50070) v rámci daného Hadoop clusteru získá v odpovědi výpis daného adresáře obsahující seznam složek a souborů, včetně údajů o jednotlivých souborech, jak je patrné z Obr. 3-4, a to přestože se jednotlivé soubory mohou nacházet na různých uzlech clusteru.

Jmenný uzel (ASF, 2019o) zajišťuje operace nad jmenným prostorem HDFS (otevření, zavření, přejmenování souborů a složek) a také mapování bloků napříč jednotlivými datovými uzly, datové uzly zajišťují realizaci jednotlivých požadavků na operace čtení a zápisu pro klienty HDFS a také operace nad bloky (vytvoření, odstranění, replikace, apod.). HDFS podporuje (ASF, 2019o) tzv. model jednoho uložení a mnoha přístupů (write-once-read-many) a neumožňuje tudíž modifikace souborů, pouze připojení dat na konec souboru (append) nebo ořez souborů (truncate). Mimo jiné právě díky tomu pak HDFS poskytuje (ASF, 2019o) tzv. proudový přístup k datům (streaming data access).

s vysokou propustností a je optimalizován pro rozsáhlé soubory dat, zpravidla o velikosti v řádu gigabajtů a vyšším, nikoli pro vysoký počet malých souborů, které jsou standardně ukládány v rámci tradičních nedistribuovaných FS.

Obr. 3-4 Grafické rozhraní HDFS – výpis obsahu adresáře HDFS

	Name	Block Size	Replication	Last Modified	Size	Group	Owner	Permission	
□	ambari-qa	0 B	0	Mar 26 15:42	0 B	hdfs	ambari-qa	drwx-----	Trash
□	cust.csv	128 MB	3	Mar 29 12:48	380.45 KB	hdfs	spark	-rW-r--r--	Trash
□	druid-indexing	0 B	0	Mar 27 10:55	0 B	hadoop	druid	drwxrwxr-x	Trash
□	ebay.csv	128 MB	3	Mar 29 09:49	561.54 KB	hdfs	spark	-rW-r--r--	Trash
□	ida8c06f00_date312619	128 MB	3	Mar 26 15:31	1 KB	hdfs	hdfs	-rW-r--r--	Trash
□	spark	0 B	0	Mar 29 10:09	0 B	hdfs	Spark	drwx-WX-WX	Trash
□	tezsmokeinput	0 B	0	Mar 26 15:41	0 B	hdfs	ambari-qa	drwxr-Xr-X	Trash
□	tezsmokeoutput	0 B	0	Mar 26 15:50	0 B	hdfs	ambari-qa	drwxr-Xr-X	Trash
□	tracks.csv	128 MB	3	Mar 29 10:21	45.04 MB	hdfs	spark	-rW-r--r--	Trash

Showing 1 to 9 of 9 entries (filtered from 11 total entries)

Previous 1 Next

Hadoop, 2018.

Zdroj: vlastní zpracování

Apache Hadoop je kompatibilní také s dalšími úložišti, konkrétně s (ASF, 2019o) cloudovým objektovým úložištěm veřejných cloudů AWS (S3), Alibaba Cloud (Object Storage Service) a Microsoft Azure (Azure Blob Storage), cloudů založených na OpenStack (Swift Object Store) a také službě cloudového data lake veřejného clodu Microsoft (Azure Data Lake), který je založen právě na HDFS. Uvedená úložiště je tak možné v rámci Hadoop snadno integrovat a využívat dle potřeby společně s aplikacemi pro zpracování big data v kombinaci s HDFS.

Pokud jde o samotný přístup do HDFS, pro uživatele i aplikace je k dispozici řada možností jak k HDFS přistupovat, nahrávat do nebo číst z HDFS soubory/data. Nativně poskytuje HDFS (ASF, 2019o) tzv. FileSystem Java API, které mohou využít aplikace interagující s HDFS, ale k dispozici je také API v programovacím jazyce C, které obaluje nativní Java API HDFS a také REST API (WebHDFS).

Pro administraci HDFS je k dispozici (ASF, 2019o) CLI v podobě programu `hdfs`. Pro vlastní práci se soubory v rámci HDFS je možné využít (ASF, 2019o) příkaz `dfs` tohoto programu (v podobě `hdfs dfs`) nebo příkaz `fs` programu `hadoop` (v podobě `hadoop fs`). Oba uvedené příkazy využívají CLI označované jako (ASF, 2019o) FS shell. Příkaz `dfs` je starší, v rámci vyšších verzí Hadoop je preferován příkaz `fs`. Jednotlivé operace nad soubory/adresáři jsou zadávány jako přepínač (-cat, -ls, atp.), za kterým případně následují parametry, například cesta k souboru v rámci lokálního FS nebo HDFS, apod.

Tato CLI podporují (ASF, 2019o) množství operací nad soubory v rámci HDFS podobných příkazům, které jsou dostupné v rámci FS na platformě Linux (například `cat`, `chmod`, `chgrp`, `cp`, `df`, `find`, `ls`, `mkdir`, `mv`, `rm`, `rmdir`, `tail`, aj.), ale také specifické operace v rámci HDFS (například `appendToFile`, `copyFromLocal`, `copyToLocal`, `createSnapshot`, `get`, `moveFromLocal`, `moveToLocal`, `put`, a další). Výpis kořenového adresáře HDFS tak lze získat například prostřednictvím příkazu `hadoop fs -ls /`.

V případě využití HDFS ze strany enginů zajišťujících zpracování big data jsou přitom vzhledem k objemu a charakteru zpracovávaných dat typické operace spíše nad adresáři, nikoli jednotlivými soubory, kdy vstupem pro zpracování dat je zpravidla obsah celého adresáře a nikoli pouze vybraný soubor.

S využitím (ASF, 2019o) NFS brány a/nebo nástrojů založených na implementaci souborového systému v rámci uživatelského prostoru (FUSE) lze HDFS také připojit (mount) a využívat jako lokální FS, včetně možnosti použití Linuxových nástrojů pro práci se soubory, což poskytuje řádově vyšší efektivitu při práci se soubory, než v případě použití výše uvedených CLI. K dispozici je také (ASF, 2019o) grafické rozhraní HDFS, které poskytuje jmenný uzel, a jež bylo znázorněno na výše uvedeném Obr. 3-4. Toto rozhraní je nicméně spíše informativní.

Některé distribuce frameworku Apache Hadoop nicméně poskytují v podstatě plnohodnotnou alternativu typu grafického rozhraní správce souborů pro HDFS. Ta je vhodná především pro méně technické uživatele bez znalostí konceptů a operací práce se soubory v rámci platformy Linux. Uživatelé této kategorie (ale i techničtí uživatelé) také mohou využít některý z frameworků a/nebo aplikací vyšší úrovni představujících nadstavbu nad HDFS a poskytujících abstrakci HDFS (přičemž na pozadí samozřejmě využívají některé ze zmíněných API HDFS). Uživatelé tedy s HDFS nemusí interagovat přímo, ale mohou využít některou z entit poskytujících abstrakci, například typu zdroj (source) HDFS nebo cílového umístění (sink) HDFS v rámci flow zpracování big data, apod.

Ochrana dat proti selhání (fault tolerance) je v rámci HDFS zajišťována (ASF, 2019o) prostřednictvím replikace bloků, v jejichž rámci jsou data uložena, napříč jednotlivými datovými uzly, a to v počtu odpovídajícím konfigurovanému replikačnímu faktoru (defaultně 3). V případě, že dojde k selhání některého z datových uzlů, a tedy ztrátě některé z replik, zajistí (ASF, 2019o) HDFS automaticky opětovnou replikaci (replication), jejímž výsledkem je opětovné dosažení požadovaného počtu replik.

Pokud jsou uzly tvořící Hadoop cluster, respektive datové uzly HDFS rozmístěny v rámci různých rackových stojanů, využívá (ASF, 2019o) proces replikace tzv. rack-awareness vlastnosti HDFS, což determinuje uložení minimálně jedné repliky každého bloku do jiného rackového stojanu za účelem zajištění vyšší úrovni ochrany dat. Replikace poskytuje vysoký stupeň ochrany dat, ale (Singh, a další, 2019, str. 60) s relativně vysokými provozními náklady (overhead) na úrovni 200%, což představuje jeden z hlavních důvodů, proč byl v rámci Hadoop 3 implementován koncept erasure coding. EC nicméně replikaci nahrazuje, ale doplňuje, a proto následuje stručný popis využití tohoto konceptu v rámci HDFS.

Tab. 3.2 Kategorizace dat v rámci HDFS z hlediska erasure coding

Charakteristika dat v HDFS	Hot data	Warm data	Cold data
Stáří dat (dnů)	< 7	=> 7 && < 30	=> 30
Frekvence přístupu	=> 20x za den	< 20x za den	< 1x za den
Počet replik na diskové vrstvě	3	1	0
Počet replik v archivační vrstvě	0	2	1 + EC
Erasure coding	Ne	Ne	Ano
Utilizace HDFS v případě erasure coding	200%	200%	150%

Pozn: údaje pro replikační faktor 3.

Zdroj: data (Singh, a další, 2019, str. 60), vlastní zpracování

Jak již bylo uvedeno, v rámci Hadoop 3 byla zavedena nová forma ochrany dat, a to erasure coding, která oproti replikaci bloků napříč jednotlivými datovými uzly zajíšťuje řádově nižší režijní náklady při srovnatelné úrovni ochrany a spolehlivosti. V případě erasure coding jsou data (Singh, a další, 2019, str. 60) označena na základě stáří a frekvence přístupu ze strany uživatelů, a tím rozdělena do kategorií, které jsou uvedeny v Tab. 3.2. Defauletně jsou přitom všechna data (Singh, a další, 2019, str. 61) zařazena do kategorie „hot data“ a bloky dat jsou replikovány napříč datovými uzly v počtu odpovídajícímu konfigurovanému replikačnímu faktoru (defauletně 3) a pouze v případě, že data splní kritéria kategorie „cold data“ je proveden erasure coding, který zajistí snížení nákladů na uložení dat v rámci HDFS o 50% oproti replikaci. Minimální počet datových uzlů HDFS pro zajištění možnosti využití EC činí 9. Efekt EC se tak samozřejmě projeví především v případě Hadoop clusterů, které jsou provozovány střednědobě až dlouhodobě a využívány například pro účely archivace dat, data lake, apod.

V rámci Hadoop 3 poskytuje HDFS řadu dalších pokročilých funkcí, jejichž podrobný popis přesahuje rámec této práce, a mezi které patří například (ASF, 2019o) federace (federation) umožňující využívat v clusteru více jmenných prostorů a/nebo zařízení blokového úložiště, funkctionalita snímků (snapshot) HDFS, využití uzelů připojených k více síťovými rozhraním (mulithoming), podpora různých typů úložišť (archivační úložiště, SSD (solid-state drive), operační paměť) a další.

Distribuovaný souborový systém tak představuje naprostě základní komponentu Hadoop poskytující funkcionality pro vysoce škálovatelné spolehlivé distribuované ukládání rozsáhlých souborů dat se zajištěním proudového přístupu k těmto datům.

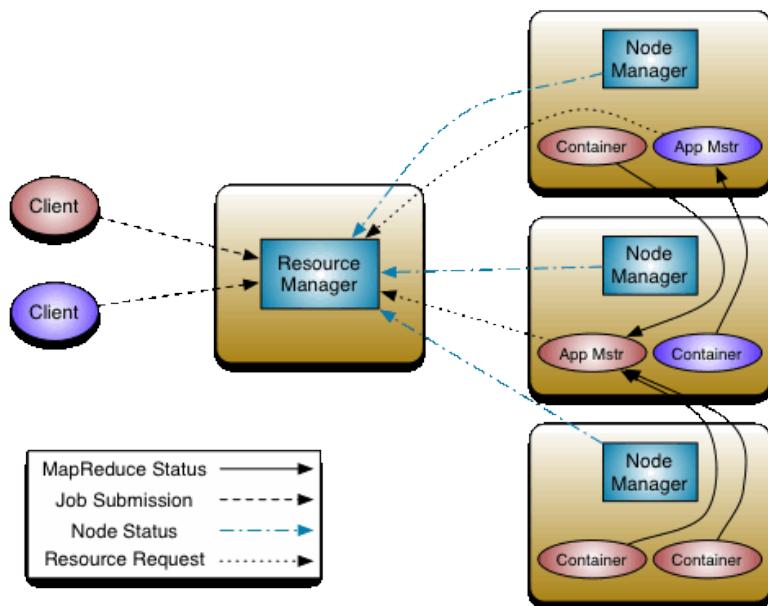
Hadoop YARN

Jak již bylo uvedeno, představuje YARN (ASF, 2019b) „... framework pro plánování úloh a správu zdrojů clusteru.“, který byl implementován v rámci restrukturalizace architektury první majoritní verze Hadoop, jejímž výsledkem byla verze Hadoop 2. Díky YARN bylo v rámci Hadoop 3 možné zajistit především (Singh, a další, 2019, str. 75) vysokou škálovatelnost a potenciálně vysokou dostupnost zpracování rozsáhlých souborů dat v rámci Hadoop, efektivnější utilizaci paměti při zpracování big data a v neposlední

řadě také možnost realizace zpracování big data nejen dávkově, ale také v (téémř) reálném čase.

Klíčový koncept YARN představuje (ASF, 2019c) rozdělení funkcionality správy zdrojů a plánování/monitorování úloh do samostatných procesů běžících na pozadí (daemons). Tomuto konceptu pak odpovídá architektura Hadoop YARN, která je rámcově znázorněna na Obr. 3-5. V rámci této architektury je v clusteru Hadoop nasazena (ASF, 2019c) globální komponenta správce zdrojů (ResourceManager) zajišťující centrální management zdrojů nezbytných pro realizaci jednotlivých úloh zpracování big data (primárně procesor (CPU), operační paměť, disk, síť). V případě HA konfigurace je tato komponenta nasazena redundantně a podobně jako komponenta ZooKeeper v případě distribuovaného souborového systému, zajišťuje automatický failover na záložní (standby) instanci správce zdrojů v případě, že dojde k selhání primárního správce zdrojů.

Obr. 3-5 Architektura YARN



Zdroj: data a zpracování (ASF, 2019c)

Dále jsou součástí architektury YARN (ASF, 2019c) instance komponenty správce uzlu (NodeManager), která je zpravidla nasazena na všech datových uzlech HDFS, aby bylo možné realizovat zpracování dat v místě, kde se nacházejí, a nebylo nutné data přesouvat. Jestliže správce zdrojů představuje (ASF, 2019c) globální autoritu zajišťující alokaci zdrojů mezi jednotlivé aplikace zpracování big data v rámci Hadoop clusteru, komponenta správce uzlu představuje agenta, který je nasazen na každém pracovním, respektive datovém uzlu clusteru. Správce uzlu zajišťuje (ASF, 2019c) v rámci každého pracovního/datového uzlu orchestraci kontejnerů¹⁴, které představují základní jednotku alokace zdrojů, a v jejichž rámci jsou s využitím daných alokovaných zdrojů realizovány jednotlivé úkoly aplikací/úloh zpracování big data. Kromě správy YARN kontejnerů pak

¹⁴⁾ V tomto případě se nejedná o kontejner typu Docker, ale pouze logickou abstraktní entitu využívanou pro alokaci zdrojů určených pro realizaci jednotlivých úkolů tvořících úlohy/aplikace distribuovaného zpracování big data.

správce uzlu také (ASF, 2019c) monitoruje využití jednotlivých zdrojů v rámci těchto kontejnerů a reportuje centrální komponentě správce zdrojů.

Další komponentu YARN představuje (ASF, 2019c) správa aplikace (ApplicationMaster), jejíž instance je nasazena pro každou jednotlivou aplikaci odeslanou za účelem distribuované realizace do YARN. Jedná se o specifickou komponentu, která zajišťuje (ASF, 2019c) získání/přidělení zdrojů od správce zdrojů a dále prostřednictvím komunikace se správcem uzlu vytvoření kontejneru, v jehož rámci jsou dané zdroje alokovány, a pak především vlastní realizaci a monitoring průběhu realizace jednotlivých úkolů v rámci těchto kontejnerů. Pro každou aplikaci tak komponenta správce aplikace zajišťuje správu realizace jednotlivých úkolů tvořících aplikaci distribuovaného zpracování big data, napříč clusterem pracovních/datových uzlů Hadoop.

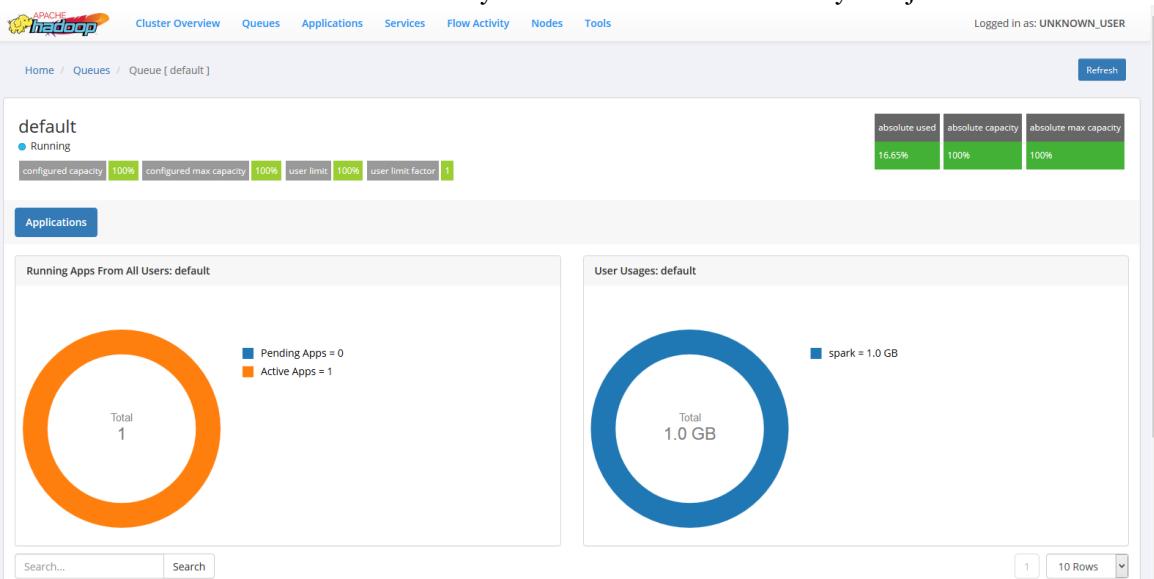
Interně je komponenta správce zdrojů rozdělena do dvou modulů, mezi které patří (ASF, 2019c) plánovač (scheduler) a správce aplikací (ApplicationManager). Plánovač představuje (ASF, 2019c) jednoduchou komponentu, která na základě požadavků jednotlivých aplikací pouze alokuje zdroje, přičemž reflektuje kvóty, fronty a případně další omezující podmínky (constraints) a zdroje alokuje prostřednictvím abstraktní entity označované termínem kontejner, která v rámci/kontextu YARN zapouzdřuje zdroje, jako je operační paměť, CPU, disk, síť, atp. Plánovač YARN může zdroje přidělovat dle různých (ASF, 2019c) politik, respektive priorit, které lze aplikovat formou zásuvného modulu (plug-in) YARN, přičemž aktuálně jsou k dispozici primárně plánovače CapacityScheduler, FairScheduler a FIFO (first in, first out). Defaultně je využíván první z uvedených plánovačů orientovaný na podporu využití clusteru Hadoop jako multi-tenantního prostředí, sdíleného více klienty, kteří do YARN odesílají k realizaci různé aplikace zpracování big data.

Defaultní plánovač přitom využívá koncept (Alapati, 2017, str. 412) front (queue) zdrojů, které lze hierarchicky vytvářet a rozdělovat mezi ně procentuálně zdroje dostupné v clusteru. Aplikace odeslané do YARN k vykonání v rámci určité fronty zdrojů pak (Alapati, 2017, str. 412) mají k dispozici podíl zdrojů, který byl přiřazen dané frontě, aplikace jsou v rámci YARN v jednotlivých frontách realizovány postupně v pořadí odpovídajícím FIFO. Defaultně je v rámci YARN pouze (viz Obr. 3-6) jediná fronta s názvem root a defaultním oddílem a aplikace jsou prováděny v pořadí, v němž byly zadány, nicméně je možné vytvářet specifickou hierarchii front zdrojů a využít tento mechanismus pro základní rozdělení zdrojů Hadoop clusteru mezi uživatele, podniková oddělení/jednotky, tenanty, apod. Kromě toho je možné v rámci YARN také (ASF, 2019c) zadávat rezervace zdrojů v čase, které musí plánovače rovněž respektovat a zajistit jejich dodržení.

Modul správce aplikací (ApplicationsManager) pak (ASF, 2019c) přijímá úlohy/aplikace zpracování big data odeslané (submitted) do YARN, zajišťuje získání prvního kontejneru, v jehož rámci je spuštěna instance komponenty správce aplikace (ApplicationMaster), která řídí další realizaci aplikace, a dále také restart kontejneru správce aplikace v případě, že dojde k jeho selhání. V současné době jsou podporovány čtyři typy kontejnerů YARN, v jejichž rámci je možné úkoly jednotlivých aplikací

distribuovaného zpracování big data realizovat, a mezi něž patří (Singh, a další, 2019, str. 102) defaultní, Linuxový, Windows a Docker kontejner.

Obr. 3-6 Grafické rozhraní YARN – stav využití defaultního oddílu fronty zdrojů



Zdroj: vlastní zpracování

YARN v Hadoop 3 také poskytuje (Singh, a další, 2019, str. 98) novou verzi grafického rozhraní správce zdrojů (defaultně na portu 8088/ui2 oproti původní verzi na portu 8088) v clusteru Hadoop, které je znázorněno na Obr. 3-7 a poskytuje přehled o clusteru dostupných zdrojů, komponentách YARN, frontách zdrojů, jejich využití ze strany aplikací zpracování big data, všech realizovaných aplikacích a službách a jejich stavu a průběhu, využití zdrojů (kontejnery, paměť, CPU), konfiguraci YARN a také logy z průběhu realizace aplikací a služeb, logy komponent YARN, a to včetně údajů, které poskytuje nová verze komponenty YARN Timeline server.

Pro interakci s YARN je možné, stejně jako v případě HDFS, využít API, nicméně ta jsou určena (ASF, 2019c) primárně pro vývojáře aplikací a zpracování big data. Uživatelé využijí spíše CLI ve formě příkazu `yarn`, jehož prostřednictvím (ASF, 2019c) mohou odeslat ke zpracování (submit) požadovanou aplikaci zpracování big data (např. `yarn jar /usr/hdp/current/hadoop-mapreduce-client/hadoop-mapreduce-examples.jar pi 16 1000`), zobrazit seznam běžících aplikací (`yarn application -list`), zobrazit logy běžící aplikace (`yarn logs -applicationId <app Id>`), zastavit a odstranit aplikaci (`yarn application -kill <app Id>`), atd.

Spuštění aplikace přímo prostřednictvím YARN tak zpravidla zahrnuje odeslání aplikace kódu pomocí příkazu `yarn jar` (analogicky lze alternativně také použít příkaz `hadoop jar`), se specifikací umístění daného souboru a dále třídy, která realizuje zpracování dat a případně dalších parametrů. Běžící aplikace lze monitorovat prostřednictvím zmíněných příkazů v rámci CLI, ale také s využitím popsaného grafického rozhraní. Pro ladění uživatel využije především logy aplikace a/nebo kontejnerů, v jejichž rámci je aplikace realizována.

Vytvoření kódu YARN aplikace může být značně náročné, nicméně zpravidla existuje aplikace, kterou lze využít a/nebo přizpůsobit pro realizaci případu užití (use case) zpracování big data, který je nutné zajistit a realizovat. Aplikace může být reprezentována (ASF, 2019c) jednoduchou úlohou nebo více komplexním orientovaným acyklickým grafem (DAG) sestávajícím z více úloh, nebo také (Singh, a další, 2019, str. 98) tokem (flow) zahrnujícím množinu vzájemně navazujících a/nebo provázaných aplikací, případně také (ASF, 2019c) službou, která na rozdíl od aplikace není spuštěna jednorázově, ale běží v Hadoop clusteru dlouhodobě.

Obr. 3-7 Grafické rozhraní YARN – přehled stavu všech aplikací v clusteru Hadoop

The screenshot shows the Apache Hadoop YARN Application Overview interface. At the top, there are tabs for Cluster Overview, Queues, Applications (which is selected), Services, Flow Activity, Nodes, and Tools. The Applications tab shows a table of running applications. The table has columns for Application ID, Application Type, Application Name, User, State, Queue, Progress, Start Time, and Elapsed. There are also filters for User and State on the left, and pagination controls at the top right. The bottom of the page includes a license notice, version information (v3.1.1.3.1.0.0-78), and a start date (Started at 2019/05/03 14:21:00).

User (3)	Application ID	Application Type	Application Name	User	State	Queue	Progress	Start Time	Elapsed
<input checked="" type="checkbox"/> spark 11	application_1556529345245_0001	SPARK	Thrift JDBC/ODB...	spark	Running	default	<div style="width: 10%;">10%</div>	2019/04/29 11:5...	4D 2h 3
<input checked="" type="checkbox"/> hive 6	application_1554986144669_0003	SPARK	Thrift JDBC/ODB...	spark	Failed	default	<div style="width: 100%;">100%</div>	2019/04/15 09:3...	16h 24r
<input checked="" type="checkbox"/> ambari-qa 3	application_1554986144669_0002	SPARK	Thrift JDBC/ODB...	spark	Failed	default	<div style="width: 100%;">100%</div>	2019/04/11 15:5...	23h 35r
More	application_1554986144669_0001	TEZ	HIVE-67270c81-f...	hive	Finished	default	<div style="width: 100%;">100%</div>	2019/04/11 14:5...	10m 25
State (4)	application_1554816173806_0003	SPARK	Thrift JDBC/ODB...	spark	Failed	default	<div style="width: 100%;">100%</div>	2019/04/10 09:4...	8h 8m
<input checked="" type="checkbox"/> FINISHED 11	application_1554816173806_0002	SPARK	Thrift JDBC/ODB...	spark	Failed	default	<div style="width: 100%;">100%</div>	2019/04/09 15:3...	14h 45r
<input checked="" type="checkbox"/> FAILED 7	application_1554816173806_0001	TEZ	HIVE-8f5206d4...	hive	Finished	default	<div style="width: 100%;">100%</div>	2019/04/09 15:2...	10m 16
<input checked="" type="checkbox"/> RUNNING 1	application_1554816173806_0000	SPARK	Thrift JDBC/ODB...	spark	Failed	default	<div style="width: 100%;">100%</div>	2019/04/09 14:2...	31m 49
<input checked="" type="checkbox"/> KILLED 1	application_1554802195081_0002	SPARK	Thrift JDBC/ODB...	spark	Finished	default	<div style="width: 100%;">100%</div>	2019/04/09 11:4...	15m 33
More	application_1554802195081_0001	TEZ	HIVE-55c0eb7a...	hive	Finished	default	<div style="width: 100%;">100%</div>	2019/04/09 11:4...	15m 33
Apply Clear	application_1553611706649_0007	SPARK	Thrift JDBC/ODB...	spark	Failed	default	<div style="width: 100%;">100%</div>	2019/03/29 15:5...	20h 49r

Zdroj: vlastní zpracování

Hlavní přidaná hodnota YARN tudíž spočívá v podstatě jednotného správce zdrojů pro Hadoop, umožňujícího v clusteru Hadoop realizovat aplikace distribuovaného zpracování big data dávkově i v (témař) reálném čase. V rámci jednotlivých enginů projektu a ekosystému Apache Hadoop díky tomu není nutné řešit správu zdrojů a plánování a zajištění realizace úloh zpracování data, jelikož toto zastřešuje právě YARN. Enginy a aplikace pro zpracování big data se tak mohou zaměřit na vlastní proces zpracování big data, jehož realizaci zajišťuje YARN (např. aplikaci frameworku/enginu Spark lze s využitím YARN spustit takto: `spark-submit --master yarn --deploy-mode cluster /usr/hdp/2.5.3.0-37/spark/examples/src/main/python/pi.py 10`).

Samozřejmě, podobně jako v případě HDFS, i YARN poskytuje řadu dalších pokročilých funkcí, jejichž podrobný popis přesahuje rámec této práce. Jedná se například o (ASF, 2019c) využití označených (labeled) uzlů, oportunistické YARN kontejnery, Docker kontejnery, registr služeb YARN, atd.

Správce zdrojů YARN tedy v rámci clusteru Hadoop zajišťuje jednotnou správu zdrojů a management distribuované škálovatelné a spolehlivé realizace aplikací v rámci YARN kontejnerů prakticky pro všechny frameworky/enginy zajišťující zpracování big data v Hadoop clusteru a podporuje zpracování big data nejen dávkově, ale také interaktivně či v (témař) reálném čase.

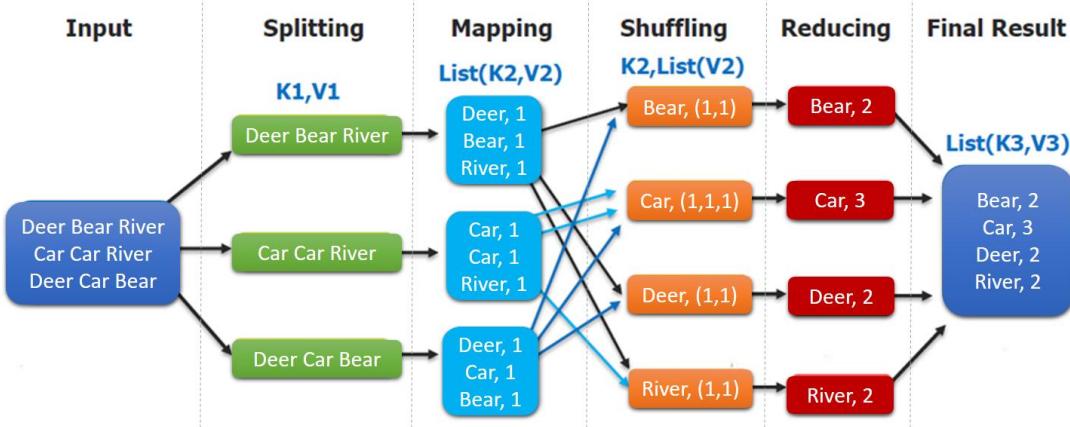
Hadoop MapReduce

Hadoop MapReduce představuje (ASF, 2019b) „... systém pro paralelní zpracování rozsáhlých souborů dat založený na YARN.“, který pochopitelně využívá programovací model MapReduce. Jedná se tedy o (ASF, 2019p) „... framework pro snadnou tvorbu a realizaci aplikací zajišťujících zpracování rozsáhlých souborů dat (v řádu několika terabajtů) paralelně v rámci rozsáhlých clusterů (v řádu tisíců uzlů) komoditního hardware, a to spolehlivě při zajištění odolnosti proti chybám.“

Programovací model MapReduce a úlohy založené na tomto paradigmatu, reprezentují typickou úlohu dávkového zpracování rozsáhlých souborů dat. Názvy Map a Reduce jsou odvozeny od funkcí map a reduce, které jsou realizovány nad vstupními daty a jejichž prostřednictvím je provedeno masivně paralelní dávkové zpracování vstupních dat.

Jednotlivé fáze realizace vzorové aplikace MapReduce zajišťující výpočet výskytu slov v textu (WordCount) je znázorněn na Obr. 3-8. V první fázi aplikace (ASF, 2019p) přijímá na vstupu vstupní soubor dat (soubor nebo celý adresář obsahující vstupní soubory), zpravidla z HDFS, přičemž v tomto ukázkovém případě se jedná o soubor obsahující určitý text. V další fázi dochází k (ASF, 2019p) rozdelení (split) vstupu na nezávislé části (chunks). Tyto jsou následně v další fázi (ASF, 2019p) zcela paralelně zpracovány prostřednictvím úkolů (task) realizujících nad danou částí vstupu funkci map, jejíž podstatou je v tomto příkladu rozdelení vstupní části na jednotlivá slova. Výsledkem funkce map není pouze slovo, respektive hodnota, ale pár klíč-hodnota, přičemž v rámci ukázkového příkladu WordCount je klíčem dané slovo a hodnota (např. ve fázi mapping číslo 1) reprezentuje počet výskytů daného slova v textu.

Obr. 3-8 Schéma průběhu jednotlivých fází MapReduce aplikace (WordCount)



Zdroj: data a zpracování (Pattamsetti, 2017, str. 232)

Ještě před provedením funkce reduce následně (ASF, 2019p) dojde k operaci seskupení (shuffling), v jejímž rámci jsou vždy sloučeny všechny výstupní páry klíč-hodnota z předchozí fáze, které mají stejnou hodnotu klíče (v tomto případě slovo/řetězec), přičemž výstupem fáze shuffle je opět množina párů klíč-hodnota, v jejichž případě zůstává klíč identický. Hodnota zahrnuje množinu všech hodnot sloučených párů, v tomto příkladu tedy všechny hodnoty 1 reprezentující každý jeden výskyt daného slova (klíče) v rámci vstupního textu. Podstatným rozdílem mezi fázemi map a shuffle je fakt, že výstupem fáze map je seznam párů klíč-hodnota v počtu odpovídajícím počtu částí, na něž

byl vstup rozdělen, zatímco v případě funkce shuffle je výstupem počet párů klíč-hodnota odpovídající počtu klíčů, které se vyskytují v rámci celého vstupu.

V poslední fází je (ASF, 2019p) na každý jednotlivý výstup operace shuffle aplikována funkce map, jejímž výsledkem je opět množina párů klíč-hodnota, přičemž v rámci uvedeného příkladu představuje klíč slovo z textu a hodnotou počet výskytů daného slova v rámci vstupního textu. Výstup realizované úlohy MapReduce je opět uložen do HDFS. Pro zobrazení obsahu výstupního souboru lze využít například příkaz `hadoop fs -cat`.

Framework Hadoop MapReduce pro distribuovanou realizaci aplikace MapReduce využívá správce zdrojů Hadoop YARN. Současně v průběhu realizace úlohy MapReduce (ASF, 2019p) zajišťuje framework Hadoop MapReduce plánování realizace jednotlivých úkolů tvořících danou MapReduce aplikaci, monitoring jejich realizace a opětovné provedení v případě, že dojde k selhání některého z úkolů.

Tvorba MapReduce programů/aplikací (EMC, 2015) zahrnuje vytvoření kódu v programovacím jazyku Java. Přestože v rámci tutoriálů a ukázkových souborů jsou zpravidla (ASF, 2019p) všechny relevantní třídy (mapper, reducer, driver a případně další) součástí jednoho souboru, v praxi (EMC, 2015) jsou třídy, a tedy také MapReduce program, rozděleny do tří souborů obsahujících kód pro funkce map, reduce a driver, a případně ještě dalších souborů (combiner, partitioner, aj.). Takto vytvořené soubory obsahující kód Java jsou posléze (EMC, 2015) zkompilovány, čímž vznikne JAR soubor obsahující program MapReduce, který je následně možné spustit nad vstupními soubory umístěnými v rámci HDFS. Výstup je uložen do HDFS, a to v rozsahu počtu souborů, který je roven počtu souborů předaných aplikaci MapReduce na vstupu. Program MapReduce lze spustit prostřednictvím (ASF, 2019p) příslušného API nebo pomocí programu `yarn` nebo případně `hadoop` (například `hadoop jar WordCount /user/joe/wordcount/input /user/joe/wordcount/output`). Pro účely monitoringu MapReduce aplikací lze využít již zmíněné příkazy CLI `yarn` a také grafické rozhraní YARN nebo vlastní grafické rozhraní MapReduce 2 nesoucí název JobHistory UI (zpravidla na portu 19888). Nová verze grafického rozhraní YARN poskytuje ve srovnání s JobHistory UI větší množinu funkcí pro monitoring realizovaných MapReduce aplikací při zajištění lepší přehlednosti.

Vývojář programu MapReduce tak musí ovládat (EMC, 2015) nejen programovací jazyk Java a způsob spuštění a monitoringu realizace úlohy v rámci clusteru Hadoop prostřednictvím YARN, ale především podstatu a koncepty MapReduce, zejména způsob definice logiky kódu v souladu s paradigmatem MapReduce, včetně využití relevantních Java tříd, metod a rozhraní, a pak také implementace funkcí driver, map a reduce v programovacím jazyku Java. Kromě programovacího jazyka Java je možné úlohy MapReduce realizovat také (EMC, 2015) prostřednictvím Hadoop Streaming API, které kromě Java podporuje také Python, C a Ruby nebo využít tzv. Hadoop Pipes, tj. mechanismus využívající pro funkce map a reduce komplikovaný kód C++.

Hadoop MapReduce tak představuje programovací model a framework pro distribuované masivně paralelní dávkové zpracování big data, přičemž aplikace MapReduce jsou realizovány prostřednictvím YARN, na vstupu přijímají soubory zpravidla z HDFS, realizují nad nimi postupně operace split, map, shuffle a reduce, které operují s páry typu klíč-hodnota a výstup ukládají opět do HDFS.

Význam úloh distribuovaného dávkového zpracování big data typu MapReduce byl klíčový především v době vzniku Hadoop 1, kdy v podstatě ani nebylo možné provádět v rámci Hadoop jiný typ úloh. Tato situace se výrazně změnila s příchodem architektury Hadoop 2 a nezávislého správce zdrojů Hadoop YARN, umožňujícího nasazovat a provozovat nad YARN a HDFS i další typy frameworků/enginů a realizovat i jiné typy úloh zpracování big data.¹⁵

Hadoop Ozone

Modul Hadoop Ozone představuje (ASF, 2019s) „... škálovatelné, redundantní a distribuované objektové úložiště pro Hadoop.“ Toto objektové úložiště (ASF, 2019s) poskytuje masivní škálovatelnost na úrovni bilionů objektů různé velikosti a je možné jej efektivně využívat v rámci kontejnerizovaných prostředí jako například clusteru kontejnerů Docker orchestrovaných ze strany Kubernetes nebo správce zdrojů YARN. Aplikace využívající Spark, YARN či Hive (ASF, 2019s) mohou Ozone využívat nativně bez potřeby jakýchkoli modifikací aplikačního kódu (s výjimkou specifikace cesty k úložišti). Projekt objektového úložiště Ozone v současné době (ASF, 2019j) není ve fázi produkční verze, jelikož zatím byly vydány pouze alfa verze a jedna (ASF, 2020b) beta verze v březnu roku 2020.

Ozone (ASF, 2019s) operuje nad vrstvou vysoce dostupného replikovaného blokového úložiště nesoucího název distribuované datové úložiště Hadoop (HDDS). Hadoop Ozone v podstatě představuje (ASF, 2019k) alternativu ke službě objektového úložiště S3, kterou poskytuje společnost Amazon v rámci AWS, a protože Ozone podporuje protokol S3, je možné aplikace využívající S3 provozovat právě nad Ozone. V rámci on-premise instalace Hadoop je tak prostřednictvím Ozone možné (ASF, 2019k) využívat v rámci big data aplikací a BDA objektové úložiště, které je často vhodnější než tradiční souborový systém a/nebo HDFS. V případně hybridního cloutu nebo kombinace on-premise instalace Hadoop a HaaS je možné aplikace využívající objektové úložiště nasazovat v obou prostředích pouze s minimálními modifikacemi a využít předností objektového úložiště.

V rámci architektury Ozone je (ASF, 2019h) jsou oblasti správy jmenného prostoru a správy blokového prostoru odděleny. Řídící komponenta Ozone Manager (ASF, 2019h) zajišťuje správu jmenného prostoru, respektive metadat objektového úložiště, kdy mezi základní entity objektového úložiště Ozone spadají svazky (volume), kyblíky (bucket) a klíče (keys). Svazky vytvářejí (ASF, 2019h) pouze administrátoři pro jednotlivé uživatele objektového úložiště, kteří v rámci přiděleného svazku mohou vytvářet, spravovat a využívat libovolný počet kyblíků, v jejichž rámci ukládají objekty označované jako klíče. Další komponenta, Storage Container Manager (ASF, 2019h) zajišťuje správu bloků objektového úložiště napříč clusterem datových uzlů. Objekty jsou (ASF, 2019h) ukládány do bloků podobných blokům v rámci HDFS a jsou replikovány pro zajištění odolnosti proti chybám, přičemž jednotlivé bloky jsou organizovány v rámci kontejnerů a ukládány

¹⁵⁾ Tato skutečnost je důvodem proč v rámci textu ani příloh neuvádíme žádný příklad kódu aplikace MapReduce. Čtenář jej nicméně může nalézt například v rámci dokumentace Hadoop MapReduce, tedy v rámci zdroje (ASF, 2019p). Další podrobné informace lze nalézt rovněž v rámci dokumentace Hadoop MapReduce (ASF, 2019p) nebo například v rámci zdroje (White, 2012).

v rámci datových uzlů. Součástí architektury Ozone je rovněž (ASF, 2019h) souborový systém Ozone File Systém kompatibilní s Hadoop.

API Ozone podporuje (ASF, 2019h) protokoly vzdáleného volání procedur (RPC) a REST a pro uživatele je k dispozici CLI klient, jehož prostřednictvím mohou, podobně jako v případě ostatních podprojektů Hadoop, interagovat s API a využívat funkcionalitu Ozone. Zápis a čtení objektů v rámci objektového úložiště Ozone probíhá tak, že (ASF, 2019h) komponenta Ozone Manager komunikuje s komponentou Storage Container Manager ohledně umístění čtených/zapisovaných dat a klient Ozone zajišťuje vlastní operace čtení/zápisu v rámci příslušných datových uzlů, zatímco Ozone Manager zajišťuje uchování/aktualizaci/odstranění metadat objektů, které jsou předmětem dané operace.

3.3.2 Ekosystém Hadoop

Jednotné vymezení množiny projektů tvořících ekosystém Hadoop v literatuře neexistuje. Pojem „ekosystém Hadoop“ je v této práci používán ve významu kolekce projektů, které jsou v rámci domovské stránky Apache Hadoop (ASF, 2019b) zařazeny do množiny „Hadoop-related projects at Apache“.¹⁶

Jedná se o projekty, mezi které patří (ASF, 2019b) Ambari, Avro, Cassandra, Chukwa, HBase, Hive, Mahout, Pig, Spark, Submarine, Tez a ZooKeeper. Jak již bylo uvedeno, jedná se o projekty, které zastřešuje ASF. Následuje základní charakteristika architektury a funkcionality uvedených projektů ekosystému Hadoop, se zaměřením na vymezení role těchto komponent v rámci clusteru Hadoop a procesu BDA.

Ambari

Komponenta Ambari představuje projekt, (ASF, 2019q) „... jehož cílem je zjednodušit správu Hadoop prostřednictvím vývoje software pro provisioning, management a monitoring clusterů Apache Hadoop. Ambari poskytuje intuitivní, snadno použitelné grafické rozhraní pro správu Hadoop, které využívá REST API.“ Apache Ambari poskytuje funkcionality především v následujících oblastech (ASF, 2019q):

- provisioning clusteru Hadoop – Ambari poskytuje instalátor a grafického průvodce instalací clusteru Hadoop, respektive jednotlivých softwarových služeb a komponent clusteru Hadoop v rámci libovolné množiny uzlů a také následné nasazení, konfiguraci a spuštění,
- správa clusteru Hadoop – Ambari představuje centrální místo správy a konfigurace uzlů a služeb napříč clusterem Hadoop,

¹⁶) Kromě těchto projektů existuje značné množství dalších projektů (zastřešovaných ze strany ASF nebo jiných subjektů), které jsou v praxi společně s Hadoop a/nebo v rámci některé z distribucí Apache Hadoop používány. Takové projekty jsou v rámci této práce chápány jako součást ekosystému big data. V rámci dalšího textu této podkapitoly jsou blíže charakterizovány pouze projekty z ekosystému Hadoop. Kromě těchto již budou v rámci této práce blíže představeny jen projekty, respektive komponenty, které představují komponentu jednotlivých distribucí Apache Hadoop, ale nejsou součástí projektu frameworku ani ekosystému Apache Hadoop. Takové projekty budou popsány v rámci charakteristiky dané distribuce Hadoop v rámci následující kapitoly.

- monitoring clusteru Hadoop – Ambari poskytuje dashboard zajišťující základní pohled na aktuální stav clusteru Hadoop, a dále funkcionality pro sběr metrik a notifikaci událostí v clusteru.

Z předchozího textu je zřejmé, že projekt Hadoop sestává ze čtyř klíčových podprojektů (Common, HDFS, YARN, MapReduce), které jsou nasazeny distribuovaně napříč celým clusterem, přičemž některé komponenty těchto podprojektů jsou nasazeny na řídících uzlech clusteru, jiné na pracovních, další například na obou těchto typech a v clusteru se mohou nacházet ještě další typy uzlů. V praxi lze samozřejmě využívat clustery, které zahrnují jen projekty frameworku Apache Hadoop nebo samostatně pouze jednotlivé komponenty ekosystému Hadoop, ale častěji a především v případě větších clusterů a distribucí frameworku Apache Hadoop jsou podprojekty frameworku Apache Hadoop využívány společně s řadou projektů z ekosystémů Apache Hadoop a/nebo big data.

Komplexita nasazení a správy takových clusterů Hadoop se pochopitelně zvyšuje s rostoucím počtem uzlů v clusteru a nasazených služeb Hadoop. S ohledem na distribuovaný charakter prostředí Hadoop a nasazení jednotlivých služeb tak uživatel, respektive administrátor clusteru Hadoop nasazuje služby, respektive jejich komponenty na všech uzlech clusteru manuálně nebo s využitím některých automatizačních nástrojů. Apache Ambari pomáhá náročnost procesů nasazení a správy clusterů Hadoop snížit tím, že poskytuje centrální rozhraní pro nasazení a správu služeb clusteru Hadoop. Přitom není bez zajímavosti, že původním autorem projektu Ambari je společnost Hortonworks, producent stejnojmenné distribuce frameworku Apache Hadoop, která později Ambari předala ASF.

Architektura Ambari je tvořena sadou REST API a grafickým rozhraním, mezi hlavní komponenty patří (ASF, 2019a) ambari-server a ambari-agent, přičemž Ambari server je nasazen na některém z řídících uzlů, zatímco na všech ostatních uzlech clusteru je nasazen Ambari agent. Centrální server (ASF, 2019a) zajišťuje koordinace jednotlivých agentů a orchestraci jednotlivých akcí nasazování a správy uzlů a služeb clusteru Hadoop, a to právě prostřednictvím agentů běžících na všech uzlech clusteru a také poskytuje API a grafické rozhraní (zpravidla na portu 8080), jehož prostřednictvím mohou administrátoři clusteru efektivně spravovat. Komponenta agenta (ASF, 2019a) zajišťuje na každém z uzlů clusteru realizaci aktivit, které zadává k realizaci Ambari server, přičemž neustále komunikuje aktuální stav daného uzlu a jednotlivých služeb a realizovaných akcí směrem k Ambari serveru.

Dalším prvek architektury Ambari reprezentuje Ambari Metrics System (ASF, 2019q) zajišťující sběr metrik jednotlivých služeb v rámci všech uzlů clusteru. Tato služba (Hortonworks, 2019l) využívá komponentu metrics-monitor shromažďující metriky v rámci každého jednotlivého uzlu clusteru a dále metrics-collector zajišťující sběr a agregaci metrik od metrics-monitor a jejich předání Ambari serveru a monitorovací platformě Grafana (zpravidla na portu 3000). V rámci Ambari serveru jsou metriky vizualizovány na dashboardu, dále na detailu jednotlivých služeb a případně zobrazeny formou upozornění v rámci grafického rozhraní Ambari serveru nebo případně notifikovány. Vizualizační a monitorovací platforma Grafana (Hortonworks, 2019l)

poskytuje přehledné vizualizace metrik, přičemž v rámci Ambari je k dispozici množství předdefinovaných vizualizací metrik napříč službami frameworku a ekosystému Hadoop.

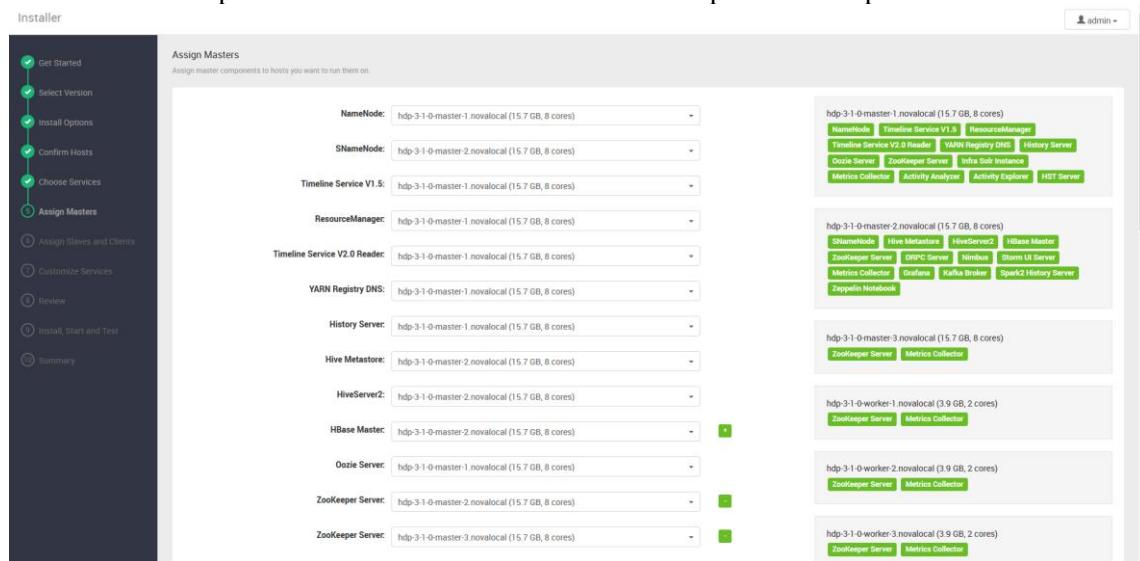
Ambari Alert Framework (ASF, 2019q) umožnuje definovat upozornění (alert/alarm), která jsou notifikována v případě, že dojde k dosažení stanovené prahové hodnoty, a to na vícestupňové škále (ok, warn, critical). Za tímto účelem jsou využívány (ASF, 2019q) metriky shromažďované prostřednictvím Ambari Metrics System. Dále je součástí architektury Ambari ještě také interní komponenta Ambari Infra, Ambari Log Search poskytující grafické rozhraní pro správu logů jednotlivých uzlů a služeb clusteru Hadoop a také Ambari Views, jehož smyslem bylo formou zásuvných modulů zajistit možnost nasazovat v rámci Ambari grafická rozhraní nad relevantními službami clusteru Hadoop (průzkumník HDFS, rozhraní pro správu a konfiguraci front zdrojů YARN, aj.).

První oblastí funkcionality Ambari je tedy, jak bylo uvedeno, nasazení clusteru Hadoop. Jestliže administrátor využívá pro nasazení clusteru Ambari, pak po zajištění provisioningu jednotlivých uzlů a jejich iniciální konfigurace následně na jeden řídící uzel nasadí Ambari, a po jeho spuštění využije pro nasazení clusteru Hadoop průvodce, který je znázorněn na Obr. 3-9 a zahrnuje následující kroky:

0. zahájení – specifikace názvu nasazovaného clusteru Hadoop,
1. výběr verze – výběr verze instalované distribuce frameworku Apache Hadoop, volba repositářů (veřejné/lokální), které budou použity pro nasazení clusteru Hadoop,
2. registrace uzlů clusteru – konfigurace přístupu k hostům představujícím uzly cílového clusteru a provedení jejich automatické/manuální registrace,
3. potvrzení uzlů clusteru – ověření množiny registrovaných uzlů, včetně výsledků automatické kontroly jejich konfigurace, a výběr cílové množiny uzlů, které budou tvořit cluster Hadoop,
4. volba služeb clusteru Hadoop – výběr služeb frameworku a ekosystému Apache Hadoop a/nebo big data, které jsou součástí zvolené verze vybrané distribuce Apache Hadoop, a jež mají být v rámci cílového clusteru Hadoop nasazeny,
5. přiřazení řídících (master) komponent – specifikace rozmístění řídících komponent vybraných služeb na jednotlivé (řídící) uzly clusteru Hadoop,
6. přiřazení pracovních (slave) a klientských (client) komponent – specifikace rozmístění pracovních a klientských komponent vybraných služeb na jednotlivé (pracovní) uzly clusteru Hadoop,
7. konfigurace služeb clusteru Hadoop – konfigurace a/nebo přizpůsobení jednotlivých vybraných služeb clusteru Hadoop, především pokud jde o databáze, databázová spojení, adresáře, uživatelské účty, aj.,
8. kontrola a potvrzení nasazení – vizuální kontrola a potvrzení nasazení clusteru Hadoop specifikovaného v rámci předchozích kroků, včetně možnosti exportu šablony (Ambari blueprint) clusteru Hadoop,
9. instalace, spuštění a otestování – instalace jednotlivých komponent vybraných služeb Hadoop napříč celým clusterem, včetně následného spuštění a otestování,
10. shrnutí – informace o výsledku (úspěch, neúspěch, varování) nasazení clusteru Hadoop týkající se nasazených uzlů a služeb.

Pokud je prostředí uzlů budoucího clusteru Hadoop vhodně připraveno, pak konfigurace zabere administrátorovi několik minut a nasazení clusteru (primárně v závislosti na velikosti clusteru a počtu nasazovaných služeb Hadoop) několik desítek minut až hodin, oproti řádu hodin až dnů v případě manuálního nasazení. Hlavní výhodou použití Ambari pro nasazení clusteru je přitom skutečnost, že jednotlivé služby jsou nasazovány v relevantním pořadí a dále skutečnost, že v případě selhání nebo pouze částečné úspěšné instalace clusteru je k dispozici grafické rozhraní Ambari pro správu clusteru jehož prostřednictvím (a případně s využitím manuálních zásahů) lze jednotlivé služby clusteru Hadoop postupně doladit a spustit.

Obr. 3-9 Ambari – průvodce nasazením clusteru distribuce Apache Hadoop



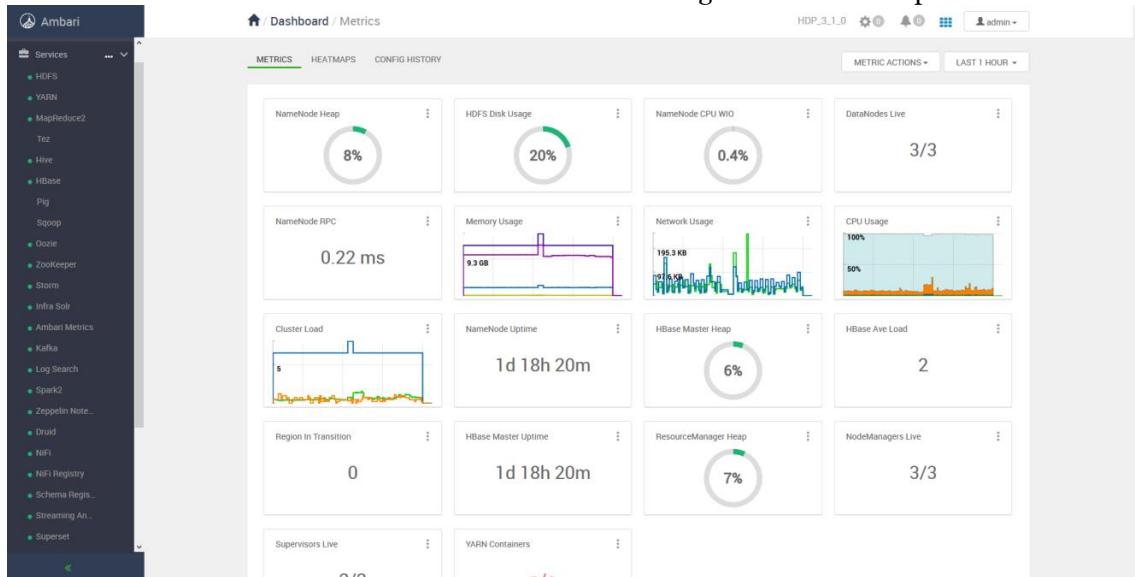
Zdroj: vlastní zpracování

Pokud je cluster Hadoop nasazen, poskytuje Ambari množství užitečné funkcionality pro jeho další správu a monitoring. Po přihlášení do Ambari v rámci již nasazeného clusteru Hadoop, se nejprve zobrazí dashboard, který je znázorněn na Obr. 3-10 a ilustruje aktuální stav daného clusteru. V oblasti správy clusteru Hadoop poskytuje Ambari především funkcionality zahrnující správu nastavení a konfigurace clusteru, správu uzlů tvořících cluster (instalace/škálování, správa nasazených služeb a komponent, evidence logů, monitoring, atd.), správu jednotlivých služeb (instalace/škálování, spuštění, zastavení, restart, odstranění, specifické funkce správy jednotlivých služeb, odkazy na grafická rozhraní služeb, atd.), správu komponent jednotlivých služeb (instalace/škálování, spuštění, zastavení, restart, odstranění, atd.), správu konfigurací jednotlivých služeb (automatická správa verzí konfigurací služeb s možností rollbacku, správa a nasazení konfigurací služeb) a rovněž evidence a vizualizace metrik jednotlivých služeb.

Pokud jde o konfiguraci jednotlivých služeb frameworku Apache Hadoop, ta je zajišťována prostřednictvím (Alapati, 2017, str. 72) množiny XML (extensible markup language) souborů (a případně dalších souborů jiného typu), umístěných v rámci adresáře, v němž je daná služba nasazena. Tyto soubory se nicméně (Alapati, 2017, str. 72) nenacházejí pouze v jednom umístění, ale naopak zpravidla na všech uzlech clusteru Hadoop. V případě některých konfiguračních souborů je tudíž (Alapati, 2017, str. 73) nezbytné provést

požadované změny v rámci všech uzlů a současně provést restart relevantních komponent a/nebo služeb, aby se změny v konfiguraci projevily.

Obr. 3-10 Ambari – dashboard základních metrik monitoringu clusteru Hadoop



Zdroj: vlastní zpracování

Služby projektu a ekosystému Hadoop využívají především následující kategorie konfiguračních souborů (Alapati, 2017, str. 72-75):

- konfigurační soubory prostředí – konfigurace prostředí, v jehož rámci jsou procesy služeb provozovány, formou proměnných prostředí (environment variables), jako jsou různá umístění, nastavení paměti, apod. (např. soubory `hadoop-env.sh`, `mapred-env.sh`, `yarn-env.sh`),
- defaultní konfigurační soubory – konfigurační soubory, které nelze měnit (read-only), a jež jsou aplikovány, pokud neexistují specifická nastavení v rámci specifických konfiguračních souborů (např. soubory `core-default.xml`, `hdfs-default.xml`, `mapred-default.xml`, `yarn-default.xml`),
- specifické konfigurační soubory clusteru (site-specific) – konfigurační soubory jednotlivých služeb, v jejichž rámci je možné (pře)definovat defaultní nastavení služeb (např. soubory `core-site.xml`, `hdfs-site.xml`, `mapred-site.xml`, `yarn-site.xml`),
- další konfigurační soubory – množina ostatních konfiguračních souborů, které se vztahují k prostředí Hadoop (např. soubory `log4j.properties`, `allocations.xml`, `hadoop-metric.properties`, atd.).

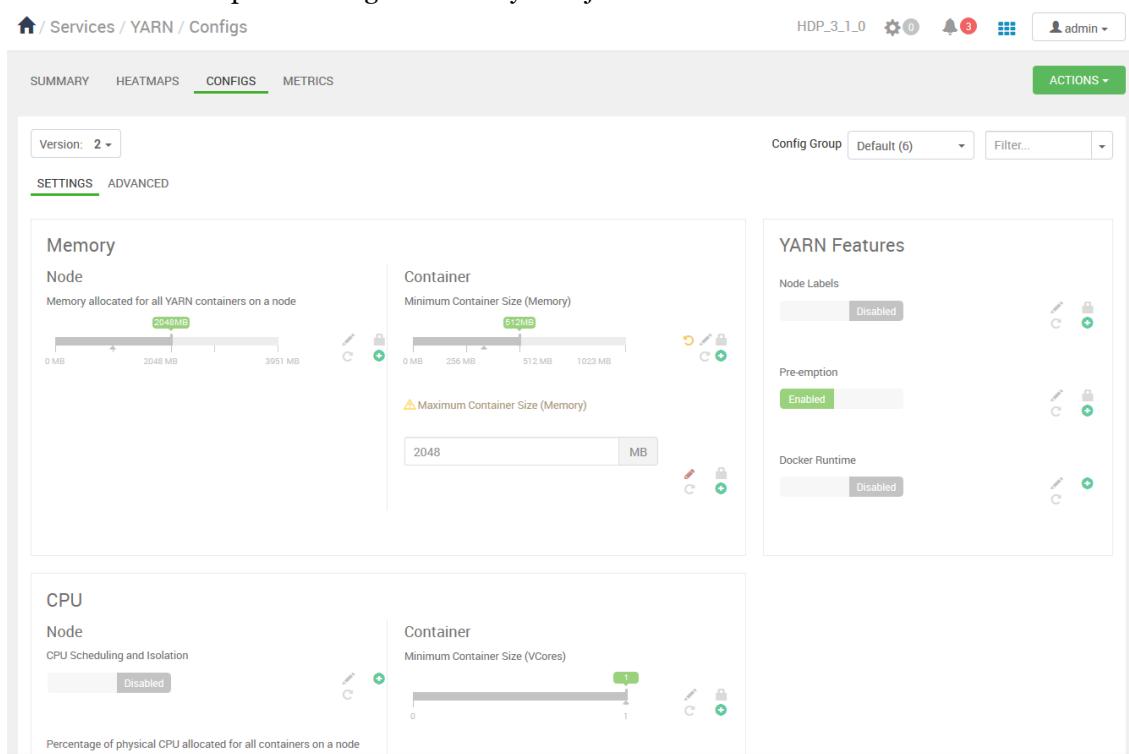
Přednost při uplatnění nastavení v rámci prostředí clusteru Hadoop je přitom následující: (Alapati, 2017, str. 77) 1. aplikační kód (nastavení v kódu úlohy), 2. kód z příkazové řádky (přepínače, parametry, apod., zadávané v rámci CLI nebo volání API, 3. konfigurační soubory na straně klienta, 4. konfigurační soubory na straně uzlu, kde je kód prováděn, 5. defaultní konfigurační soubory.

Z uvedené charakteristiky je zřejmé, že manuální správa konfigurace Hadoop clusteru a jednotlivých služeb Hadoop je značně komplexní. Ambari přitom poskytuje centrální grafické rozhraní prezentující na jednom místě veškeré konfigurační parametry ze všech

relevantních souborů, a to včetně možnosti vyhledávání a filtrování, jak je to patrné z Obr. 3-11. Ambari navíc po uložení změn v konfiguraci indikuje potřebu restartu u všech relevantních komponent a/nebo služeb a tuto funkcionalitu také poskytuje. Z jednoho místa je tak možné rychle a efektivně provést konfigurační změny včetně restartu příslušných služeb napříč celým clusterem. V případě, že restart selže, v Ambari jsou k dispozici logy umožňující identifikovat příčinu chyby. Jednou z hlavních přidaných hodnot správy konfigurace v rámci Ambari je také automatické verzování umožňující v případě problémů provést návrat (rollback) k vybrané předchozí funkční konfiguraci.

Pokud jde o monitoring jednotlivých služeb clusteru Hadoop, poskytuje Ambari řadu předdefinovaných upozornění (alert), která s využitím komponent Ambari Metrics System a Ambari Alert Framework neustále monitoruje a v případě splnění definovaných podmínek okamžitě notifikuje v rámci grafického rozhraní Ambari a/nebo prostřednictvím emailu. Administrátor clusteru Hadoop může prostřednictvím Ambari spravovat jednotlivá upozornění napříč skupinami upozornění, která zpravidla odpovídají jednotlivým službám, a to od vytvoření, přes konfiguraci, až po pozastavení/spuštění a případně odstranění.

Obr. 3-11 Ambari – správa konfigurace služby se zajištěním automatického verzování



Zdroj: vlastní zpracování

Komponenta Ambari představuje zcela zásadní nástroj ekosystému Apache Hadoop, jelikož poskytuje grafické rozhraní pro nasazení clusteru Hadoop pomocí intuitivního průvodce, centrální místo správy služeb nasazovaných a provozovaných napříč clusterem Hadoop, včetně jejich konfigurace, kontinuální monitoring celého clusteru, včetně jednotlivých fyzických a/nebo virtuálních uzlů a služeb projektu a ekosystému Hadoop. Tím eliminuje potřebu manuální správy a konfigurace klíčových služeb a komponent clusteru Hadoop.

Avro

Apache Avro představuje (ASF, 2018e) framework a souborový formát pro serializaci a deserializaci dat do/z formátu Avro. Avro poskytuje (ASF, 2018e) komplexní datové struktury, kompaktní rychlý binární datový formát, soubor kontejneru pro persistentní uložení dat, RPC volání pro komunikaci nad Avro schématy a možnost snadné integrace s dynamickými jazyky, přičemž k dispozici jsou implementace Avro pro programovací jazyky C, C++, C#, Java, PHP, Python a Ruby.

S využitím Avro vývojář/datový analytik může (ASF, 2018a) v JSON specifikovat schéma dat sestávající z primitivních a/nebo komplexních datových typů. Na základě schématu je následně možné (ASF, 2018a) zajistit automatické vytvoření kódu relevantních tříd, což vygeneruje zdrojové *.avsc soubory obsahující dané třídy. Následně pak lze s využitím schématu a vygenerovaného kódu objekty a/nebo data efektivně (ASF, 2018a) serializovat a uložit do binárního souboru *.avro a později provést deserializaci, jejímž výsledkem jsou opět data/objekty.

Uživatelé mohou specifikovat vlastní schéma a zajistit potřebnou serializaci a deserializaci big data v rámci Hadoop právě s využitím Avro. Například (ASF, 2016d) v případě datového skladu pro big data Hive (viz dále) je Avro serializér/deserializér (serde) využíván pro zajištění možnosti čtení dat z tabulek datového skladu Hive a následného zápisu do HDFS v prakticky jakémkoli formátu. Uživatelé pak mohou serde využívat v kombinaci s nástroji pro příjem a/nebo zpracování big data, nad nimiž by jinak s ohledem na jejich strukturu nebyli schopni operovat.

Avro tedy představuje framework pro serializaci a deserializaci dat, který je podobný například Protocol Buffers společnosti Google, nicméně poskytuje možnost dynamické práce s datovými typy a nabízí formát méně náročný z hlediska velikosti souborů.

Cassandra

Apache Cassandra reprezentuje (ASF, 2016a) distribuovaný vysoce škálovatelný proti chybám odolný decentralizovaný NoSQL databázový systém. Tento databázový systém (Alapati, 2018, str. 3) vyvinula společnost Facebook, jež se při návrhu inspirovala existujícími řešeními Amazon Dynamo a Google BigTable. Apache Cassandra představuje databázový systém, který je navržen pro (Alapati, 2018, str. 3) vysokou škálovatelnost a odolnost proti chybám, při zajištění velice výkonného a rychlého uložení a přístupu k rozsáhlým souborům dat. NoSQL DBMS Cassandra byl později (Alapati, 2018, str. 4) uvolněn jako open source, přičemž od roku 2010 se jedná o samostatný projekt zastřešovaný ze strany ASF.

Architektura Apache Cassandra odpovídá (Alapati, 2018, str. 26) clusteru vzájemně komunikujících uzlů, které jsou prakticky stejné, kdy pro účely replikace lze Cassandra uzly organizovat v rámci skupin. Data jsou tudíž (Alapati, 2018, str. 26) rozprostřena napříč jednotlivými uzly a replikována, což společně se skutečností, že prakticky neexistuje žádná řídící komponenta, podporuje vysokou dostupnost, spolehlivost a ochranu proti selhání. Jednotlivé uzly (Alapati, 2018, str. 29) kontinuálně vzájemně komunikují umístění a stav prostřednictvím protokolu, který je označován názvem Gossip. Všechny

uzly v Cassandra clusteru jsou tak prakticky (Alapati, 2018, str. 28) identické (peer) a klient může komunikovat s kterýmkoli uzlem a uzly současně plní funkci koordinátora (coordinator) neboli proxy mezi klientem a uzly, na nichž se nacházejí požadovaná data a/nebo mají být realizovány operace čtení/zápisu.

Cassandra je zástupcem sloupcových NoSQL databázových systémů, který (Alapati, 2018, str. 10) nevyžaduje předdefinované schéma, a v jehož případě jsou hodnoty tvořící záznam uloženy v jednotlivých sloupcích řádku, napříč řádky tabulky se mohou data v jednotlivých sloupcích lišit a záznam objektu se nachází v rámci jednoho širokého řádku, který může být tvořen až biliony sloupců.

Apache Cassandra tak představuje sloupcově orientovaný NoSQL databázový systém, vhodný pro vysoko škálovatelné a spolehlivé uchovávání rozsáhlých souborů dat splňujících charakteristiky big data.

Chukwa

Projekt Apache Chukwa je vymezen jako (ASF, 2016c) „... systém pro sběr dat za účelem monitoringu rozsáhlých distribuovaných systémů, který je postaven nad HDFS a MapReduce, od nichž dědí škálovatelnost a robustnost a také zahrnuje flexibilní a mocné nástroje pro vizualizaci, monitoring a analýzu shromažďovaných dat.“

Architektura Apache Chukwa zahrnuje především komponentu (ASF, 2016b) agenta, nasazovanou na všechny uzly clusteru Hadoop, jež mají být monitorovány. Chukwa Agent zajišťuje sběr dat v rámci daného uzlu clusteru Hadoop prostřednictvím (ASF, 2016b) tzv. adapterů (adaptor) zajišťujících shromažďování dat z určitého typu zdroje (soubor, UDP (User Datagram Protocol), socket, systémové metriky, apod.). Další komponenta, která je označována jako pipeline zajišťuje (ASF, 2016b) ETL nad shromažďovanými daty, tedy extrakci dat emitovaných agenty, případně jejich parsování/transformaci a následně nahrání do distribuovaného souborového systému Hadoop HDFS, distribuovaného databázového systému HBase, distribuované vyhledávací platformy Solr, aj. Za účelem agregace dat lze aplikovat na shromážděná data tzv. datové analytické skripty, které lze vytvářet v jazyce PigLatin (viz dále). Pro vizualizaci dat monitoringu lze využít komponentu webového rozhraní s názvem Hadoop Infrastructure Care Center. S ohledem na skutečnost, že aktuálně poslední verze Chukwa byla vydána před téměř třemi roky, další bližší popis ani snímky obrazovek webového rozhraní Chukwa nebudou uvedeny.

Apache Chukwa představuje službu pro zajištění kontinuálního shromažďování a vizualizace údajů monitoringu Hadoop Clusteru, a to od logů, přes různé metriky, až po síťový provoz, atd.

HBase

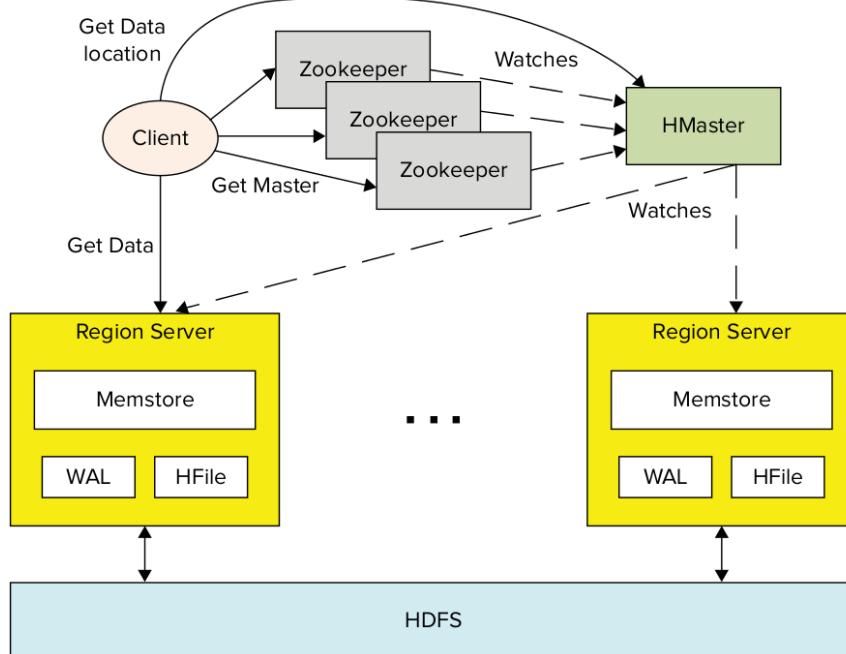
Projekt HBase je charakterizován jako (ASF, 2019d) „... databáze Hadoop, která je distribuovaným, škálovatelným úložištěm pro big data.“ NoSQL databáze HBase je určena pro (ASF, 2019d) náhodný přístup k operacím čtení/zápisu nad big data v reálném čase. Big data NoSQL databáze HBase byla navržena dle sloupcové databáze Google BigTable, přičemž podobně jako BigTable využívá GFS, HBase využívá pro uložení dat

HDFS, a nad clusterem komoditního hardware s instalovaným HDFS umožňuje využívat rozsáhlé tabulky v řádu bilionů řádků a milionů sloupců.

Architektura HBase odpovídá, podobně jako architektura HDFS a některých dalších komponent projektu a/nebo ekosystému Hadoop, (Lublinsky, a další, 2013, str. 34) vzoru master/slave a je znázorněna na Obr. 3-12. Řídící komponentou (master) je v případě HBase HMaster, zatímco pracovní komponenty (worker/slave), které jsou v clusteru Hadoop nasazeny na všech uzlech, kde se nachází datové uzly HDFS, jsou označovány jako Region Server. Analogicky k architektuře HDFS klient HBase komunikuje s řídící komponentou, především pokud jde o umístění dat a metadata, zatímco operace čtení i zápisu jsou realizovány přímo v rámci příslušných Region Serverů. Z Obr. 3-12 je navíc patrné, že i v případě HBase je pro zajištění vysoké dostupnosti využívána komponenta ZooKeeper, která bude ještě v rámci této podkapitoly blíže představena.

Součástí Region Serveru je (Lublinsky, a další, 2013, str. 35) tzv. memstore představující mezipaměť (cache) operující v rámci operační paměti daného uzlu/stroje, v jejímž rámci jsou operace čtení i zápisu řádově rychlejší než v případě disku. Využití této cache umožňuje zajistit (Lublinsky, a další, 2013, str. 35) rychlejší obsluhu požadavků na operace čtení i zápisu díky tomu, že maximální možný počet požadavků je obsluhován s využitím cache, kde je načtena určitá část dat uložených v HBase. Všechny provedené operace zápisu/modifikace jsou (Lublinsky, a další, 2013, str. 35) ukládány do logu databázových operací (WAL), který v případě selhání úložiště umožňuje všechny operace provést dle logu znova a ztracená data obnovit. HFile představuje (Lublinsky, a další, 2013, str. 35) specializovaný HDFS souborový formát HBase podporující možnost čtení, zápisu i modifikací (resp. operací get, put, scan, delete) v rámci HDFS podporujícího pouze operace čtení a zápisu (append).

Obr. 3-12 Architektura distribuovaného big data NoSQL DBMS HBase



Zdroj: data a zpracování (Lublinsky, a další, 2013, str. 35)

Není bez zajímavosti, že HBase lze využívat také v kombinaci s projekty umožňujícími nad distribuovaným NoSQL DBMS využívat SQL(-like) funkcionalitu. Konkrétně se jedná o projekty (ASF, 2019e) Apache Phoenix poskytující nad HBase OLTP a Apache Trafodion představující transakční SQL DBMS nad HBase. Uživatelé nicméně mohou pro účely manipulace s big data prostřednictvím SQL využít také další z projektů ekosystému Hadoop, a to Apache Hive.

Pro využití funkcionality HBase jsou k dispozici Java, REST a Thrift API a také CLI a grafické rozhraní řídící komponenty HBase Master poskytující základní přehledové informace týkající se jednotlivých komponent HBase, logů a metrik stavu HBase a samozřejmě tabulek, procedur, atd.

V předchozím textu byla popsána služba Cassandra představující rovněž distribuovaný NoSQL DBMS pro big data. Přestože oba projekty jsou na trhu po téměř stejně dlouhou dobu a spadají do stejné kategorie, značně se liší. HBase (Bekker, 2018) vznikl dříve, přičemž využívá vlastnosti HDFS, a tedy architekturu master/slave, zatímco v případě Cassandra architektura de facto žádnou řídící komponentu nezahrnuje, všechny uzly jsou si rovny, takže je eliminováno SPOF, které řídící komponenta představuje. Rovněž lze konstatovat, že Cassandra poskytuje (Bekker, 2018) vyšší výkon, a to především, pokud jde o operace čtení. Oba systémy (Bekker, 2018) jsou vhodné pro ukládání rozsáhlých souborů časových řad (time-series), HBase je vhodnější pro případy užití zahrnující analýzu textu a Cassandra je vhodnější pro kontinuálně běžící aplikace a/nebo systémy a pro případy užití zahrnující analýzu v reálném čase.

HBase představuje, podobně jako Cassandra, distribuovaný NoSQL databázový systém pro big data. Přestože v případě Cassandra lze identifikovat řadu předností oproti HBase, je HBase stále aktivně využívan a v rámci ekosystému Hadoop využíván, v některých distribucích je HBase rovněž využíván interně ze strany některých komponent a/nebo služeb tvořících danou distribuci frameworku Apache Hadoop.

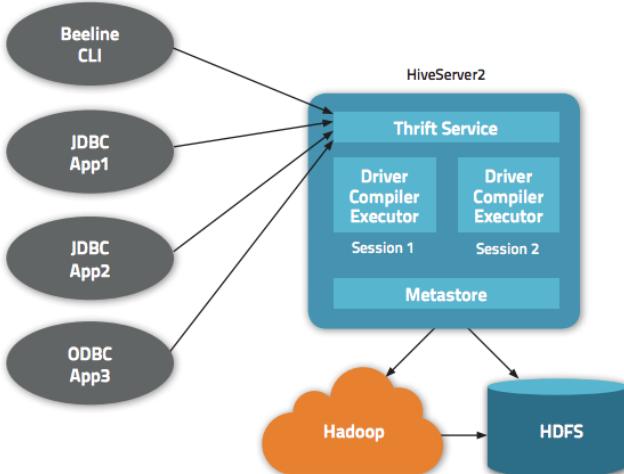
Hive

Apache Hive je software (ASF, 2014) „... datového skladu zajišťujícího čtení, zápis a správu rozsáhlých souborů dat, které se nacházejí v distribuovaném úložišti, prostřednictvím SQL.“ Hive umožňuje (ASF, 2014) aplikovat strukturu na data, která se již nacházejí v rámci distribuovaného úložiště a postrádají jakékoli schéma. Přitom je ale nezbytné zdůraznit, že (Alapati, 2017, str. 141) na rozdíl od HBase není Hive databázovým systémem, ale pouze poskytuje mechanismy pro aplikaci struktury na data uložená v rámci HDFS a jejich dotazování prostřednictvím SQL-like dotazovacího jazyka nesoucího název HiveQL.

Projekt Apache Hive byl původně vytvořen (White, 2012, str. 471) ve společnosti Facebook s cílem umožnit analytikům s dobrou znalostí SQL, ale zanedbatelnou znalostí Java realizovat SQL dotazy nad big data, která společnost Facebook uchovávala v rámci HDFS. SQL samozřejmě (White, 2012, str. 471) není vhodné pro každý problém (např. ML), ale díky tomu, že poskytuje standardizované softwarové API pro přístup k DBMS ODBC (Open Database Connectivity), zajišťuje možnost integrace s nástroji BI, v jejichž případě je SQL de facto klíčovým jazykem.

Architektura Hive je znázorněna na Obr. 3-13 a zahrnuje především komponenty (Mujumdar, 2013) klientských rozhraní (beeline, JDBC (Java Database Connectivity), ODBC), řídící komponentu HiveServer2 a Hive Metastore. Uživatelé a/nebo aplikace odesílají prostřednictvím klientských aplikací dotazy nebo aplikace zpracování big data, přičemž v clusteru Hadoop může být nasazeno více instancí HiveServer2 i Metastore. Klíčovou komponentou HiveServer 2 je tzv. Thrift služba, která zajišťuje (Shaw, a další, 2016, str. 41) příjem požadavků a/nebo dotazů od klientských rozhraní a pro každý jeden požadavek zřizuje tzv. sezení (session) sestávající z dílčích komponent ovladače (driver), komplikátoru (compiler) a vykonavatele (executor). Přijatý dotaz přebírá (DataFlair, 2018) driver, který jej předává komplikátoru zajišťujícímu vytvoření optimálního plánu realizace daného dotazu v podobě DAG sestávajícího z množiny úkolů v rámci příslušného enginu a HDFS a executor následně zajišťuje realizaci těchto úkolů dle stanoveného plánu. Výsledek realizace je vrácen zpět klientovi, přičemž v případě neúspěšné autentizace vůbec nedojde ke zkompilování přijatého dotazu.

Obr. 3-13 Architektura big data datového skladu Hive



Zdroj: data a zpracování (Mujumdar, 2013)

Hive Metastore pak představuje (Shaw, a další, 2016, str. 50) centrální repositář metadat Hive, která zahrnují jmenné prostory, definice objektů (tabulky, jejich schémata a umístění, oddíly, aj.) a další metadata týkající se dat, přičemž tato metadata nejsou ukládána v rámci HDFS, ale naopak ve zvoleném RDBMS. Hive podporuje dva typy tabulek, a to (Shaw, a další, 2016, str. 56) interní a externí tabulky, přičemž častěji jsou využívány právě externí tabulky, v jejichž případě jsou metadata uložena v rámci Metastore a vlastní data jsou uložena v HDFS.

Koncept externích tabulek umožňuje prostřednictvím SQL dotazovat gigabajty až terabajty dat uložených v rámci HDFS. Pro čtení a zápis je často využíván generický nebo specifický SerDe, takže například biliony příspěvků ze sociální sítě Twitter mohou být ukládány do souborů v rámci HDFS ve formátu JSON a s využitím komplexního datového typu struct a relevantního SerDe z tohoto JSON přímo načteny prakticky jako tradiční tabulka.

Přestože na Obr. 3-13 není uvedena, je součástí architektury Hive také komponenta označovaná jako HCatalog, která v podstatě představuje (Shaw, a další, 2016, str. 38-39) rozhraní nad úložištěm Metastore, do něhož zprostředkovává přístup klientům a

umožňuje různým enginům s využitím příslušného SerDe číst a zapisovat různé souborové formáty (SequenceFile, ORC (optimized row columnar), RCFfile, JSON, CSV (comma-separated values), aj.), kdy například v případě vytváření tabulky Hive pomocí klíčového slova `STORE AS`, lze definovat použitý souborový formát.

Zatímco v rámci prvních verzí Hive byly dotazy realizovány prostřednictvím enginu MapReduce, což determinovalo určité zpoždění vlivem režie ze strany realizace MapReduce úloh, od Hive 1.2.1 (Shaw, a další, 2016) je možné volit mezi enginy MapReduce, defaultním Tez a Spark.

Pokud jde o klientská rozhraní, pak je nutné zdůraznit, že zatímco v rámci původního HiveServer byl používán CLI klient `hive`, kterým se uživatel připojoval k instanci HiveServer. V případě HiveServer2 je nutné využít klientské CLI `beeline`, pro připojení k instanci HiveServer2. Zatímco dříve bylo k dispozici také grafické rozhraní Hive, od Hive 2.2.0 již není dostupné a HiveServer2 poskytuje pouze informativní Hive webové rozhraní (zpravidla na portu 10002), které neumožňuje zadávat SQL dotazy, apod.

Apache Hive tedy představuje službu datového skladu pro big data, umožňující nad rozsáhlými soubory dat uloženými v rámci HDFS aplikovat strukturu a dotazovat je prostřednictvím dotazovacího jazyka SQL, respektive HiveQL, který je SQL velice podobný. Data mohou být v HDFS uložena v některém z formátů vhodných pro přenos a/nebo uchování rozsáhlých souborů dat, s využitím Hive je pak lze dotazovat, jako by se jednalo téměř o tradiční RDBMS tabulky. Na rozdíl od HBase, Cassandra a dalších NoSQL DBMS, Hive není distribuovaným databázovým systémem, ale pouze umožňuje definovat nad daty uloženými v rámci Hadoop, respektive HDFS požadovanou strukturu a dotazovat je s využitím SQL, bez potřeby znalosti a/nebo tvorby kódu Java, MapReduce, apod. Nová verze HiveServer2 navíc prostřednictvím JDBC a ODBC podporuje možnost snadné integrace nástrojů BI a dokonce Microsoft Excel, aj.

Mahout

Apache Mahout představuje (ASF, 2017a) „*... distribuovaný framework pro lineární algebru a matematicky expresivní doménově specifický jazyk (DSL) Scala, navržený tak, aby matematikům, statistikům a datovým vědcům umožnil rychle implementovat vlastní algoritmy.*“

Mahout původně vznikl v rámci ekosystému Hadoop jako škálovatelná knihovna pro strojové učení, přičemž aplikace strojového učení vytvářené s využitím Mahout byly realizovány především prostřednictvím MapReduce. V současné době již nicméně (ASF, 2017c) použití Mahout jako knihovny ML není mandatorní a využití MapReduce pro realizaci aplikací vytvořených s použitím Mahout není podporováno (deprecated), jelikož primárním enginem pro realizaci Mahout aplikací je Apache Spark (viz dále).

Apache Mahout tak umožňuje (ASF, 2017c) využívat a vytvářet matematické funkce a ML algoritmy přímo v rámci kódu aplikace s využitím Scala-DSL, přičemž podporuje optimalizaci realizace výpočtů tím, že logický DAG (zapsaný v Scala-DSL) transformuje na fyzický DAG (tj. na úroveň enginu, který výpočty provede) a také s využitím tzv. native solver, umožňujících kromě JVM využít také CPU a/nebo GPU pro efektivnější realizaci

ML algoritmů. Mahout poskytuje algoritmy z oblastí (ASF, 2017c) distribuované lineární algebry, preprocesorů, regrese, shlukování a doporučení a podporuje realizaci aplikací ML nad big data analytickým enginem Apache Spark, enginem pro distribuované zpracování proudů dat Apache Flink a ML a AI platformy H2O.

Apache Mahout umožňuje v rámci Hadoop vytvářet a nasazovat aplikace distribuovaného zpracování a/nebo analýzy dat využívající algoritmy strojového učení a realizovat je distribuovaně škálovatelně a spolehlivě napříč clusterem Hadoop.

Pig

Apache Pig reprezentuje (ASF, 2018c) „... platformu pro analýzu rozsáhlých souborů dat sestávající z vysokoúrovňového jazyka pro tvorbu programu analýzy dat a infrastruktury pro realizaci těchto programů.“

Vysokoúrovňový jazyk nesoucí název (ASF, 2018c) PigLatin umožňuje snadno vytvářet programy analýzy dat i komplexní vícenásobné transformace formou sekvencí toku dat, automaticky tyto programy realizovat při zajištění relevantní optimalizace a přitom je rozšířitelný, takže uživatelé mohou v případě potřeby definovat vlastní funkce. Zmíněná infrastruktura projektu Pig je reprezentována (ASF, 2018c) kompilátorem, podporujícím sekvence MapReduce programů realizované prostřednictvím MR frameworku/enginu.

Projekt Pig vznikl s cílem zajištění abstrakce komplexity tvorby MapReduce programů. V současné době (ASF, 2017b) je možné programy Pig, jejichž kód je zpravidla obsahem *.pig souborů, provádět prostřednictvím MapReduce, Tez, Spark, ve všech případech lokálně nebo distribuovaně. Tvůrce programu musí pochopitelně ovládat jazyk PigLatin a další konstrukty daného frameworku, jako jsou vestavěné funkce, řídící struktury, atd. V současné době je poslední vydání frameworku Pig téměř dva roky staré, a proto další popis nebude uveden.

Apache Pig představuje framework zahrnující jazyk PigLatin poskytující abstrakci od komplexity MapReduce programů a kompilátor, jež z kódu PigLatin vygeneruje sekvenci úloh, které lze následně automatizovaně a paralelizovaně realizovat prostřednictvím některého z frameworků/enginů MapReduce, Tez, Spark.

Spark

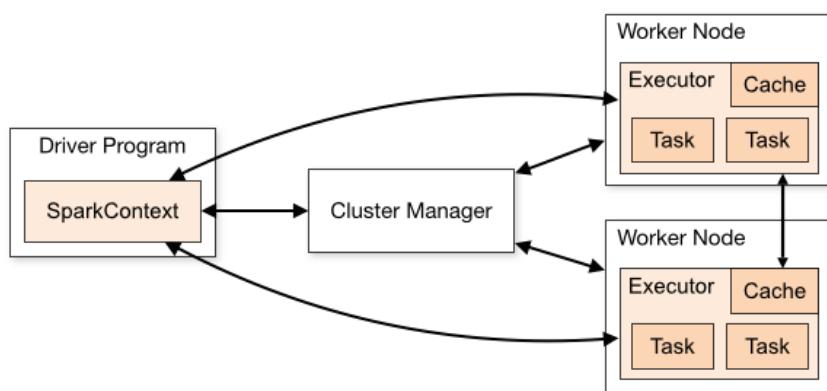
Projekt Apache Spark představuje (ASF, 2019v) „... rychlý a obecný systém pro distribuované zpracování dat.“ Aplikace a výpočty realizované prostřednictvím Apache Spark jsou velice rychlé především díky (ASF, 2019f) unikátnímu DAG plánovači, optimalizaci dotazů a také obecnému enginu pro zpracování rozsáhlých souborů dat. Jedná se tedy o (ASF, 2019f) „... jednotný analytický engine pro zpracování rozsáhlých souborů dat.“ Kromě jádra jsou součástí projektu Spark také knihovny, mezi které patří (ASF, 2019f):

- SparkSQL – umožňuje v rámci Spark programů dotazovat data pomocí SQL,
- Spark Streaming – zajišťuje v rámci Spark programů škálovatelné proti chybám odolné zpracování proudů dat,

- MLlib – knihovna ML algoritmů a utilit umožňující v rámci Spark programů realizovat úlohy strojového učení (statistika, klasifikace, regrese, kolaborativní filtrování, shlukování, dolování častých vzorů, aj.),
- GraphX – knihovna grafových algoritmů poskytující v rámci Spark programů funkcionalitu pro grafové a grafově-paralelní výpočty.

Hlavní výhoda využití těchto knihoven Apache Spark spočívá v (ASF, 2019f) možnosti jejich aplikace a kombinace v rámci každé jednotlivé aplikace. Na rozdíl od většiny ostatních enginů/frameworků pro distribuované zpracování big data, které jsou součástí projektu a/nebo ekosystému Apache Hadoop, je možné (ASF, 2019v) Spark kromě YARN provozovat v kombinaci s jinými správci zdrojů, například Mesos, Kubernetes nebo samostatně (standalone). Jako jeden z mála enginů ekosystému Hadoop je přitom Spark nezřídka využíván také individuálně, mimo cluster Hadoop, samostatně, v kombinaci pouze s projektem Apache Hadoop, atd.

Obr. 3-14 Schéma průběhu realizace Spark aplikace



Zdroj: data a zpracování (ASF, 2019i)

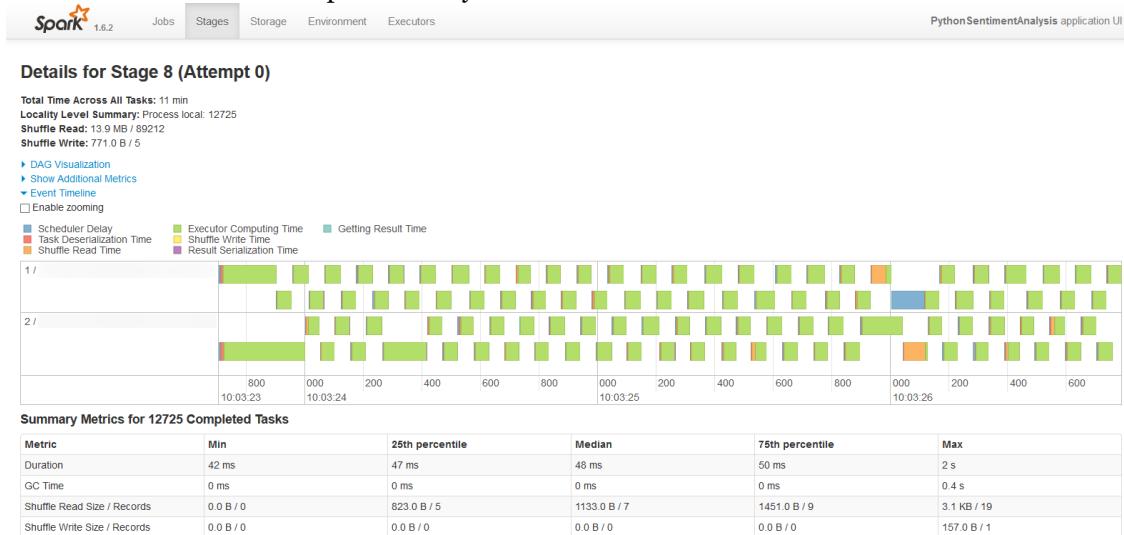
Na Obr. 3-14 je znázorněn průběh realizace Spark aplikace, kdy (ASF, 2019i) vývojář specifikuje kód programu v podobě Java, Python nebo Scala souborů (nebo případně kód zadává interaktivně do interpreteru `spark-shell`), které jsou následně odeslány (submit) prostřednictvím programu `spark-submit` správci zdrojů (Spark, YARN, Mesos nebo Kubernetes) k realizaci. Takto vytvořená aplikace je označována jako (ASF, 2019i) tzv. Driver Program, přičemž vstupním bodem každé aplikace je vytvořený `SparkContext` (nebo `StreamingContext` v případě Spark Streaming) představující objekt, který v rámci aplikace zajišťuje koordinaci množiny nezávislých procesů, jejichž formou je Spark aplikace v rámci daného clusteru realizována.

Řídící objekt Spark aplikace se tedy (ASF, 2019i) připojí k příslušnému správci zdrojů, který zajistí pro realizaci dané aplikace potřebné zdroje v podobě tzv. executors neboli procesů zajišťujících realizaci výpočtů a uchování dat aplikace, v rámci jednotlivých pracovních uzlů daného clusteru. Kód aplikace v podobě (ASF, 2019i) souborů v příslušném programovacím jazyku odeslaných objektu `SparkContext` je následně předán jednotlivým executors, stejně jako následně příslušné úkoly (tasks) respektive fáze (stage). Pro každou Spark aplikaci přitom Spark spustí (ASF, 2019i) instanci grafického rozhraní pro monitoring průběhu dané úlohy (zpravidla na portu 4040, 4041, atd.).

Součástí architektury Apache Spark 2.x, je kromě klientských rozhraní také (Frampton, 2015, str. 24) Spark2 History Server poskytující mimo jiné grafické rozhraní pro monitoring realizovaných Spark úloh/aplikací a Spark prostředí (zpravidla na portu 18081), jehož část je znázorněna na Obr. 3-15, a dále také Spark2 Thrift Server umožňující externím nástrojům snadné připojení k Spark2 prostřednictvím ODBC, a případně ještě také Apache Livy poskytující REST rozhraní pro Apache Spark.

Apache Spark poskytuje vývojářům/uživatelům (Luu, 2018, str. 51-53) abstrakce, nad nimiž mohou snadno využít dostupné operace transformace, zpracování a/nebo analýzy, a to distribuovaně, paralelně a zpravidla se zajištěním automatického zotavení v případě výskytu chyb, respektive selhání uzlu, apod. V případě Spark 1.x byla klíčovou abstrakcí, kterou Spark poskytoval (Luu, 2018, str. 51) entita tzv. resilient distributed dataset (RDD) představující „... neměnnou, proti chybám odolnou, paralelní datovou strukturu umožňující uživatelům explicitně ukládat meziví sledky do paměti, řídit jejich rozdelení pro zajištění optimálního rozmístění dat a manipulovat s nimi prostřednictvím široké množiny operátorů/funkcí.“ Nad RDD bylo možné snadno realizovat především (Luu, 2018, str. 54-58) dva typy operací, a to transformace (např. map, filter, flatMap, sample, union, groupByKey, sortByKey, join, pipe, aj.), jejichž prostřednictvím byla z kolekce dat vytvořena množina nová a akce (např. reduce, collect, count, first, take, saveAsTextFile, saveAsSequenceFile, saveAsObjectFile, countByKey, aj.) vracející řídícímu programu Spark výsledky výpočtu realizovaného nad danou množinou dat.

Obr. 3-15 Grafické rozhraní Spark History Server



Zdroj: vlastní zpracování

V rámci Spark 2.x jsou RDD i nadále k dispozici, a to společně s dalšími abstrakcemi, mezi které patří především (McDonald, a další, 2018, str. 13) množiny dat (Dataset), rámce dat (Dataframe) a další specifické abstrakce napříč jednotlivými výše zmíněnými knihovnami Apache Spark. Podrobná charakteristika všech těchto konceptů přesahuje rámec této práce, a proto bude dále uvedena pouze stručná charakteristika klíčových abstrakcí Spark 2, a to Dataset a DataFrame.

Dataset představuje (McDonald, a další, 2018, str. 13) „... distribuovanou kolekci objektů s definovaným datovým typem, které jsou rozděleny (partitioned) do jednotlivých oddílů

napříč uzly clusteru a nad nimiž je možné provádět paralelní operace.“ DataFrame je (McDonald, a další, 2018, str. 13) „ ... Dataset tvořený řádky (Row) objektů a reprezentuje tabulkou dat tvořenou řádky (rows) a sloupci (columns). DataFrame sestává z oddílů (partitions), z nichž každý je tvořen rozsahem řádků v mezipaměti daného uzlu.“ DataFrame tak představuje (McDonald, a další, 2018, str. 20) tabulkou a Dataset jeden řádek této tabulky, který je objektem reprezentujícím tabulkou, přičemž DataFrame je objekt bez pevného datového typu (dynamicky typovaný), zatímco Dataset je objektem s pevným datovým typem (staticky typovaný). Pro použití těchto abstrakcí je nutné (McDonald, a další, 2018, str. 14) vytvořit instanci objektu `SparkSession`. Nad Dataset a DataFrame lze přitom provádět (McDonald, a další, 2018, str. 20-21) podobné operace jako v případě RDD. Významný rozdíl oproti RDD ale spočívá v tom, že v případě nových abstrakcí DataFrame a DataSet jsou data organizována v rámci pojmenovaných sloupců, což umožňuje na data aplikovat strukturu a snadněji je dotazovat prostřednictvím SQL.

Knihovna Spark Structured Streaming umožňuje prostřednictvím Apache Spark realizovat (ASF, 2019w) zpracování proudů dat (stream) z mnoha různých zdrojů, jako například Kafka, Flume, Amazon Kinesis, TCP (Transmission Control Protocol) sockets, realizovat nad nimi komplexní algoritmy zpracování (map, reduce, join, window, atd.), a následně zpracovaná data ukládat do souborových systémů, včetně HDFS, databázových systémů či dashboardů. Nejedná se nicméně o zpracování proudů dat v reálném čase (real-time), ale v téměř reálném čase (near real-time), protože Spark Structured Streaming (ASF, 2019w) interně proud dat rozděluje do jednotlivých dávek, které jsou zpracovány enginem Spark, a výstupem jsou individuální dávky výsledků zpracování proudu dat. Na rozdíl od systémů, které zpracovávají jednotlivé události (one-at-time), jako například Storm či Flink, Spark zpracovává mikro dávky proudu dat (micro-batching), což determinuje stav, kdy není možné zajistit garanci zpracování dat v pořadí, v jakém se vyskytnou a/nebo jsou přijaty.

Hlavní předností Apache Spark je tudíž (McDonald, a další, 2018, str. 12) možnost využití nástrojů a funkcionality z různých oblastí v rámci jednoho koherentního celku. Zatímco dříve bylo nutné (McDonald, a další, 2018, str. 12) úlohy výběru dat, transformace a analýzy realizovat zpravidla s využitím různých frameworků, Apache Spark umožňuje s využitím jednotlivých výše uvedených knihoven pokrýt v podstatě celý proces BDA v rámci jedné aplikace¹⁷, a to i v kombinaci dávkového a interaktivního zpracování dat a/nebo zpracování dat v téměř reálném čase.

Apache Spark je aktivně využíván mnoha společnostmi, mezi něž patří například (McDonald, a další, 2018, str. 11) DataBricks, IBM, Huawei, Baidu, Taobao nebo čínský provozovatel sociálních sítí Tencent, využívající cluster více než 8000 uzlů pro analýzu 700 terabajtů dat, jež více než 800 milionů uživatelů generuje každý den. Jelikož je Spark součástí všech hlavních distribucí frameworku Apache Hadoop a umožňuje pokrýt celý

¹⁷⁾ Výjimku představuje vizualizace, nieméně pro tyto účely lze využít některý z interaktivních notebooků typu Apache Zeppelin, Jupyter, aj., poskytujících Spark interpreter a další funkcionality umožňující vizualizovat výsledky zpracování, respektive transformace a analýzy dat s využitím Spark. Kód aplikace Spark lze zpravidla definovat přímo v rámci instance daného notebooku a data a/nebo výsledky zpracování vizualizovat.

proces BDA, představuje zpravidla první volbu uživatelů pro zpracování a/nebo analýzu dat.

Apache Spark představuje distribuovaný analytický engine, umožňující efektivně distribuovaně zpracovávat rozsáhlé soubory dat, a to různou formou (dávkově, interaktivně, v téměř reálném čase), při zajištění možnosti kombinace algoritmů z různých oblastí (SQL, ML, grafové algoritmy, algoritmy pro zpracování proudů dat) v rámci jedné aplikace a současně v rozsahu téměř celého procesu BDA (příjem, zpracování, analýza).

Hadoop Submarine

Projekt Submarine byl v době vzniku zařazen (ASF, 2019x) přímo jako jeden z podprojektů frameworku Hadoop. V současné době nicméně představuje projekt ekosystému Hadoop, který je zatím ve fázi počátečního vývoje. Hadoop Submarine je projektem, který (ASF, 2020d) „... umožňuje správcům infrastruktury a datovým vědcům realizovat nad platformou pro správu zdrojů (jako například YARN) aplikace hlubokého učení (např. typu Tensorflow, Pytorch, etc.).“

V rámci Hadoop (ASF, 2020d) využívá Submarine správce zdrojů YARN, přičemž pro zajištění efektivní realizace distribuovaného DL podporuje využití GPU. Aplikace ML/DL jsou spouštěny zpravidla s využitím obrazu Docker kontejneru prostřednictvím YARN (například `yarn jar hadoop-yarn-applications-submarine-<version>.jar job run --name tf-job-001 --docker_image <your docker image>`). Pro monitoring Submarine aplikací je možné využít nové grafické rozhraní YARN.

Hadoop Submarine podporuje ML/DL frameworky, mezi které patří především (ASF, 2020d) TensorFlow, Pytorch, MxNet, atd. Submarine umožňuje v rámci clusteru Hadoop bez potřeby modifikace realizovat aplikace ML/DL vytvořené pro uvedené frameworky, a to (ASF, 2020d) v rozsahu celého procesu ML, od vývoje algoritmu, přes dávkové a inkrementální trénování modelu, až po správu a aplikaci modelu.

Projekt Submarine zajišťuje možnost spouštět rámci frameworku Hadoop aplikace strojového a/nebo hlubokého učení vytvořené v rámci specializovaných frameworků pro zajištění procesu ML/DL.

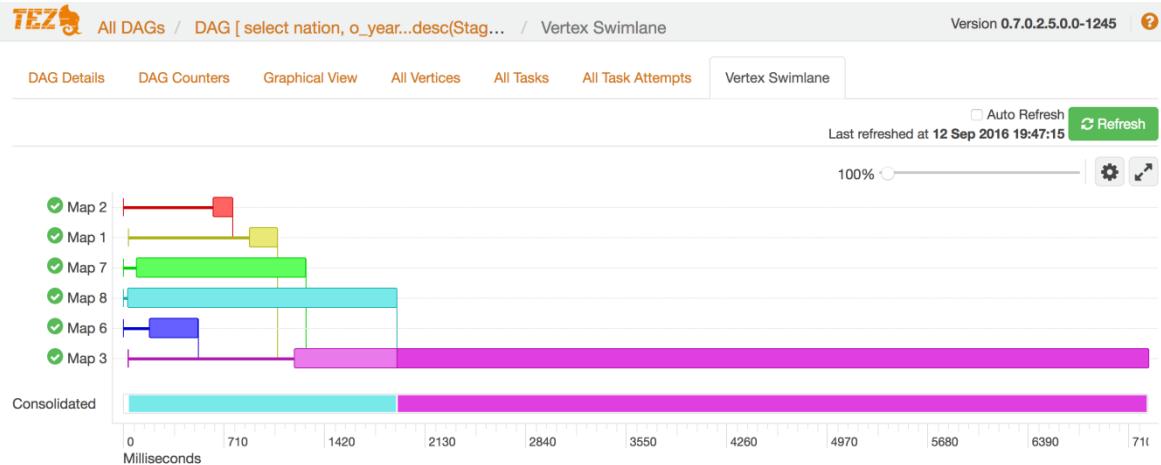
Tez

Tato komponenta reprezentuje (ASF, 2019g) „... aplikaci framework umožňující využívat pro zpracování dat komplexní množiny úkolů specifikované v rámci DAG.“ Podstatou projektu Tez je především skutečnost, že již zmíněným projektům Hive a Pig umožňuje (ASF, 2019g) realizovat komplexní DAG úkoly a v rámci jednoho DAG, respektive jedné Tez úlohy, zajistit zpracování dat, které by dříve bylo nutné provést formou více úloh MapReduce.

Tez poskytuje (ASF, 2019g) expresivní API pro definici toku dat v procesu zpracování, běhové prostředí s flexibilním modelem typu vstup-zpracování-výstup, nezávislost na typu zpracovávaných dat, možnost rekonfigurace za běhu, aj. Kromě CLI a API je k dispozici také grafické rozhraní Tez pro monitoring realizovaných úloh zpracování dat, které je znázorněno na Obr. 3-16 a je vhodné především pro ladění úloh Tez. Ačkoli je možné Tez

využít přímo, uživatelé jej zpravidla použijí spíše zprostředkovaně, kdy například v případě Hive představuje Tez defaultní engine, jehož prostřednictvím jsou dotazy do Hive v rámci clusteru Hadoop realizovány.

Obr. 3-16 Grafické rozhraní frameworku Tez



Zdroj: data a zpracování (Hortonworks, 2016e)

Apache Tez je frameworkm, jehož prostřednictvím je možné efektivně zpracovat komplexní množiny úloh vyjádřených formou DAG a zajistit tak rychlejší realizaci takových množin úloh a efektivnější využití zdrojů.

ZooKeeper

Apache ZooKeeper představuje službu zajišťující (ASF, 2019r) „... *vysoce spolehlivou koordinaci distribuovaných aplikací.*“ Cílem ZooKeeper je (ASF, 2019r) zajistit pro distribuované aplikace abstrakci služeb synchronizace, konfigurace, údržby, apod. Služba ZooKeeper již byla v předchozím textu práce několikrát zmíněna, a to především v souvislosti se zajišťováním HA.

Nasazení architektury ZooKeeper je (ASF, 2019r) distribuované, stejně jako v případě aplikací, které ZooKeeper využívají. ZooKeeper server se (ASF, 2019r) nasazuje na všech relevantních uzlech clusteru Hadoop (zpravidla minimálně na třech za účelem zajištění odolnosti proti výpadku), přičemž všechny instance ZooKeeper jsou si prakticky rovny a neustále komunikují prostřednictvím atomické vrstvy pro předávání zpráv.

Všechny ZooKeeper uzly mohou (ASF, 2019r) obsluhovat požadavky klientů, přičemž požadavky na čtení jsou realizovány ze strany toho ZooKeeper serveru, který požadavek na čtení přijal. Naopak pro realizaci (ASF, 2019r) požadavků na zápis, které v podstatě modifikují stav služby ZooKeeper, je využíván tzv. agreement protokol, jehož prostřednictvím servery stanoví, který ze serverů ZooKeeper je v rámci daného clusteru tzv. vůdce (leader), přičemž ostatní plní roli následovníků (follower). V rámci tohoto protokolu jsou pak (ASF, 2019r) všechny požadavky na operace zápisu předávány ZooKeeper lídrovi zajišťujícímu jejich realizaci v rámci daného uzlu. Nahrazení lídra v případě, že dojde k selhání a rovněž synchronizaci následovníků s lídrem zajišťuje (ASF, 2019r) již zmíněná vrstva pro předávání zpráv v rámci clusteru ZooKeeper, přičemž tento

proces probíhá s využitím algoritmu pro volbu nového lídra a za optimálních podmínek netrvá déle než 200 milisekund.

V rámci ZooKeeper clusteru jsou distribuované služby koordinovány prostřednictvím (ASF, 2019r) sdíleného hierarchického jmenného prostoru, který je organizován podobně jako tradiční souborové systémy, a je tvořen registry dat nesoucími označení znodes, které jsou podobné hierarchicky uspořádaným adresářům a souborům, ale na rozdíl od tradičních FS jsou uloženy v paměti, nikoli na disku. Změny jsou (ASF, 2019r) zapisovány na disk do logu a následně synchronizovány, přičemž všechny ZooKeeper uzly replikují svůj stav napříč clusterem.

ZooKeeper představuje distribuovanou, vysoce výkonnou spolehlivou a dostupnou službu pro koordinaci distribuovaných aplikací/služeb, která je v rámci Hadoop využívána prakticky na pozadí pro účely řešení problémů spojených s koordinací a synchronizací v distribuovaném prostředí jako jsou například (ASF, 2019r) tzv. race condition (pokusy jednoho stroje o provedení dvou nebo více operací současně) nebo tzv. deadlocks (pokusy dvou či více strojů o přístup k jednomu zdroji současně). Dále lze ZooKeeper v rámci Hadoop nebo v jiném distribuovaném prostředí využít pro zajištění centralizovaného repositáře konfiguračních dat, jako frontu zpráv (message queue) zajišťující asynchronní komunikaci, monitorovací nástroj zajišťující synchronizaci propagace a notifikace změn v clusteru, apod. V prostředí Hadoop je ZooKeeper nicméně využíván především pro koordinaci a synchronizaci jednotlivých uzelů clusteru a služeb a komponent, které jsou v tomto distribuovaném prostředí provozovány.

3.4 Praktické aplikace frameworku Apache Hadoop

Využití frameworku a ekosystému Apache Hadoop v praxi samozřejmě koresponduje s oblastmi praktického využití BDA, které byly charakterizovány v rámci podkapitoly 2.5.2. V rámci této podkapitoly budou nicméně prezentovány vybraná nasazení a způsoby využití Hadoop pro řešení konkrétních problémů. Nebude zde přitom popsáno využití frameworku Apache Hadoop a ekosystému Hadoop/big data pro poskytování Hadoop formou služby či produktu. Blíže charakterizovány budou pouze tři vybrané společnosti z různých oborů, které Hadoop v praxi využívají, přičemž kritériem výběru bylo především produkční využití orientované na podporu businessu dané společnosti a dostupnost informací o reálném využití Hadoop v průběhu existence daného businessu, respektive projektu Hadoop.

3.4.1 Twitter

Mezi společnosti, která extenzivně využívají Hadoop, patří provozovatel sociální sítě Twitter. Společnost začala Hadoop, konkrétně distribuci Cloudera, využívat (Hashemi, 2017) v roce 2010, původně se záměrem využití pro uchování záloh MySQL, zatímco dnes jej využívá majoritně pro BDA.

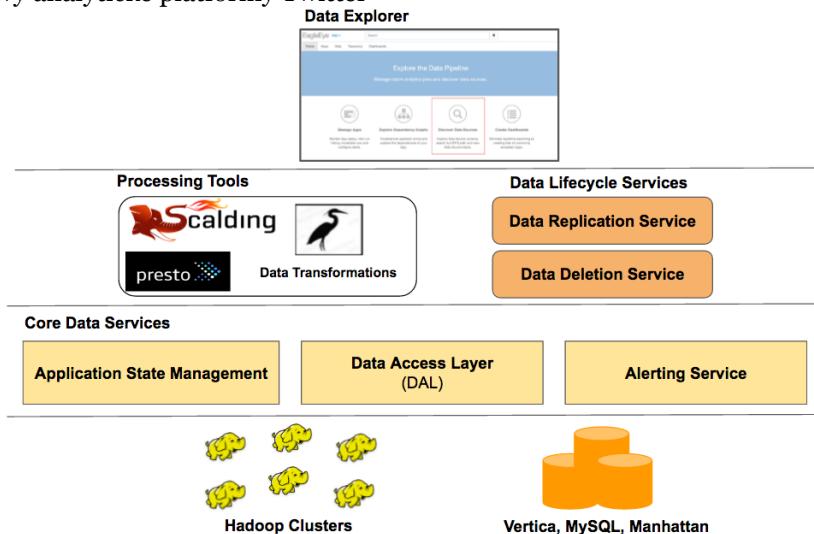
V roce 2017 bylo ve společnosti Twitter pro Hadoop dedikováno (Hashemi, 2017) 19,6% veškerého hardware a 40,8% z veškerého úložiště. Twitter provozoval v roce 2017

simultánně (Hashemi, 2017) několik clusterů Hadoop uchovávajících téměř 500 petabajtů dat přičemž největší z clusterů Hadoop zahrnoval více než 10 tisíc uzlů. Společnost Twitter v roce 2017 v rámci těchto prostředí provozovala (Hashemi, 2017) až 150 tisíc aplikací v rámci 130 milionů kontejnerů denně. Twitter v praxi využívá následující kategorie clusterů Hadoop (Rottinghuis, 2019):

- real-time – zde jsou ukládána příchozí data generovaná koncovými uživateli v rámci využívání sociální sítě Twitter,
- processing – produkční clustery, v jejichž rámci jsou pravidelně spouštěny plánované produkční úlohy s dedikovanou výpočetní kapacitou,
- „ad-hoc“ – využívané pro jednorázové dotazy a příležitostné analytické úlohy, jež jsou méně předvídatelné než produkční úlohy,
- cold storage – v jejich rámci jsou uchovávána/archivována data s nižší frekvencí přístupu.

Není bez zajímavosti, že společnost Twitter (Rottinghuis, 2019) v době vzniku využívala externí hostovaná prostředí, která později nahradila vlastní infrastrukturou a v posledních letech se zaměřuje na možnosti přesunu veškeré infrastruktury do cloudu, především ve spolupráci s Google, přičemž do cloutu plánuje migrovat „ad-hoc“ a cold storage clustery Hadoop, zatímco produkční real-time a processing clustery ponechá na on-premise Twitter infrastruktuře.

Obr. 3-17 Vrstvy analytické platformy Twitter



Zdroj: data a zpracování (Krishnan, 2016)

Twitter využívá Hadoop (Hashemi, 2017) v kombinaci se službou Flume pro zajištění distribuovaného, spolehlivého a dostupného shromažďování, agregace a přesunu rozsáhlých objemů logů. Hadoop je v rámci Twitter také využíván (Hashemi, 2017) v procesu zpracování více než trilionu zpráv denně, zahrnujícího roztrídění do více než 500 kategorií, konsolidaci a selektivní distribuci napříč všemi clustery. Z hlediska podpory businessu je Hadoop (Krishnan, 2016) jakožto součást analytické platformy Twitter, která je znázorněna na Obr. 3-17 využíván pro veřejně publikované metriky, doporučení, A/B testování, cílení reklam, atd.

Společnost Twitter lze také označit za významného přispěvatele do open source projektu a/nebo ekosystému Hadoop a/nebo big data. V rámci Twitteru vznikl např. projekt (Fu, 2018) enginu pro zpracování proudů dat v reálném čase nesoucí název Heron, který byl později uvolněn jako open source a nyní jej zastřešuje ASF.

3.4.2 Spotify

Dalším zástupcem společností využívajících Hadoop v praxi je Spotify, poskytovatel stejnojmenné služby pro online streamování hudby. Tato společnost provozovala v roce 2017 cluster Hadoop (Li, 2017) tvořený 2500 uzly, v jehož rámci denně spouštěla více než 20 tisíc úloh. Podobně jako Twitter, také společnost Spotify zahájila cestu (Li, 2017) migrace infrastruktury z on-premise prostředí do cloutu společnosti Google.

Ve Spotify byl framework Hadoop využíván prakticky již v době zprovoznění služby Spotify, tj. od roku 2008, přičemž v začátcích byl primárně využíván (Li, 2017) vlastní modul pro tvorbu komplexních flow úloh dávkového zpracování napsaný v programovacím jazyce Python a nesoucí označení Luigi, a to především pro účely orchestrace úloh MapReduce tvořených v programovacím jazyce Python a realizovaných prostřednictvím Hadoop streaming. V této době využívala společnost Spotify distribuci Cloudera.

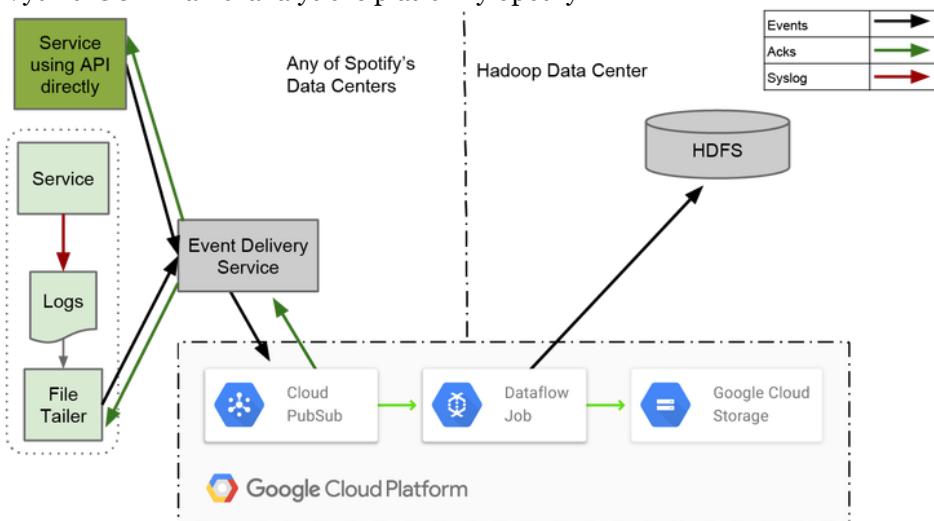
Později začali vývojáři a analytici Spotify pro účely dávkového zpracování big data využívat (Li, 2017) vysokoúrovňovou Java knihovnu pro Hadoop MapReduce nesoucí název Cascading, respektive Scalding, tj. Scala API nad Cascading, vytvořené společností Twitter, čímž dosahovali řádově vyššího výkonu, než v případě využití Python. Pro ML aplikace byl v rámci Spotify využíván (Li, 2017) framework Apache Spark, a to především za účelem efektivní realizace iterativních ML algoritmů s využitím mezipaměti. V roce 2013 došlo v rámci Spotify k migraci z distribuce Cloudera na distribuci Hortonworks.

Za účelem zpracování proudů dat byl využíván (Li, 2017) projekt enginu pro zpracování proudů dat v reálném čase Apache Strom, konkrétně pro účely doporučení pro uživatele služby Spotify, cílení reklam nebo zajištění metrik využití služeb/produktů. Většina procesů BDA realizovaných s využitím Storm přitom zahrnovala (Li, 2017) příjem událostí z distribuované platformy pro příjem, zpracování a uložení proudů dat v Apache Kafka, dále základní zpracování dat prostřednictvím Storm zahrnující filtrování, agregace, vyhledání metadat a uložení výsledků do Apache Cassandra nebo full-textového vyhledávače Elasticsearch. Z důvodu určitých omezení pak byla později komponenta Kafka (Li, 2017) nahrazena analogickou cloudovou službou GCP, a to Cloud Pub/Sub.

Ad-hoc analytické úlohy byly v rámci Spotify realizovány s využitím (Li, 2017) datového skladu Hive umožňujícího business analytikům a produktovým manažerům analyzovat rozsáhlé soubory dat prostřednictvím SQL-like dotazů. Většina dat byla uložena s využitím (Li, 2017) souborového formátu Avro, nicméně tento byl později nahrazen souborovým formátem Apache Parquet, který umožnil dosahovat při čtení dat 5-10x vyšší rychlosti. Vzhledem k (Li, 2017) nízké úrovni adopce nicméně společnost začala pro ad-hoc analýzu rozsáhlých souborů dat využívat další z cloudových big data služeb GCP, jmenovitě BigQuery.

V současné době společnost Spotify pro většinu analytických úloh zahrnujících zpracování proudů dat využívá (Li, 2017) služby GCP, konkrétně především Cloud Dataflow, jak je to patrné z Obr. 3-18, a vlastní Scala API pro Apache Beam nesoucí název Scio. Služba Cloud Dataflow poskytuje (Li, 2017) jednotný model pro dávkové zpracování dat i zpracování proudů dat a je založena na projektu Apache Beam, který společnost Google vyvinula a později uvolnila jako open source, a v současné době je zastřešován ze strany ASF. V případě Apache Beam může vývojář (Li, 2017) vytvořit workflow zpracování big data v programovacím jazyce Java nebo Python a realizovat jej nad některým z podporovaných enginů pro distribuované zpracování dat (Apex, Flink, Spark, Cloud Dataflow). Spotify přitom také využívá (Li, 2017) vlastního vysokoúrovňového Scala API pro Apache Beam a majoritně prostřednictvím Cloud Dataflow zpracovává data z mnoha různých zdrojů, například HDFS, Cassandra, Elasticsearch, PostgreSQL, atd.

Obr. 3-18 Využití GCP v rámci analytické platformy Spotify



Zdroj: data a zpracování (Maravić, 2016)

Hlavní přínos přitom zástupci Spotify spatřují ve skutečnosti, že (Li, 2017) služba Cloud Dataflow je na rozdíl od vlastních clusterů Hadoop se Spark, Storm, Scalding, apod., plně spravovaná, takže není nutné vynakládat zdroje (především lidské) potřebné pro nasazování, správu ani údržbu takových clusterů, ale datový vědec pouze v lokálním prostředí vytvoří aplikaci pro zpracování big data, kterou následně odešle k realizaci do cloutu.

3.4.3 LinkedIn

Společnost LinkedIn provozující profesní sociální síť nesoucí stejný název aktivně využívá Hadoop od samotného vzniku projektu tohoto frameworku. Již od svého vzniku řešila společnost LinkedIn problém (Steinbach, 2016) tvorby a efektivního nasazení algoritmu pro doporučování osob, které mohou členové profesní sítě znát (people you may know), přičemž pro tyto účely využívala software od společnosti Oracle. Toto řešení nicméně nebylo ideální, především s ohledem na (Steinbach, 2016) problémy se škálovatelností, které přetrvávaly v důsledku rostoucí uživatelské základny (8 milionů uživatelů v roce 2006, 50 milionů v roce 2008, atd.) využívající daný algoritmus.

Jakmile vznikl projekt Hadoop, začala společnost LinkedIn (Steinbach, 2016) plánovat jeho využití, přičemž první cluster Hadoop čítající 20 uzlů, nasadila v roce 2009. V roce 2016 již společnost provozovala (Steinbach, 2016) více než 10 clusterů zahrnujících přes 10 tisíc uzlů a pro účely realizace BDA využívaných ze strany více než 1 tisíce uživatelů. Společnost využívala pro BDA projekty (Steinbach, 2016) Hadoop, Pig, Hive, Gobblin (distribuovaný integrační framework pro dávkové i streaming zpracování dat), Cubert (výpočetní engine pro dávkové zpracování a reporting big data v rámci Hadoop), Scalding, Tez, Spark, Presto (distribuovaný engine pro SQL dotazování big data) a další.

V průběhu využití Hadoop narážela společnost LinkedIn na (Steinbach, 2016) potíže se škálováním, především v případě HDFS, již při řádu několika tisíců uzlů, přičemž v roce 2015 přidala do clusteru o velikosti 2 tisíce uzlů jednorázově dalších 500 uzlů, což společně s jistou chybou v HDFS způsobilo výpadek tohoto clusteru.

V roce 2016 společnost LinkedIn denně aplikovala (DeZyre, 2016) komplexní workflow tvořené 82 Hadoop úlohami realizovanými prostřednictvím enginu Hadoop MapReduce nad 16 terabajty dat pro před-výpočty 120 bilionů vazeb mezi entitami v rámci profesní sítě LinkedIn. V roce 2017 vzrostl objem dat uchovávaných a/nebo zpracovávaných v rámci clusterů Hadoop využívaných ze strany LinkedIn až na (Shvachko, a další, 2018) více než 3,7 petabajtů zahrnujících téměř 4 miliony objektů, přičemž počet denně realizovaných úkolů (tasks) dosahoval až 4 milionů.

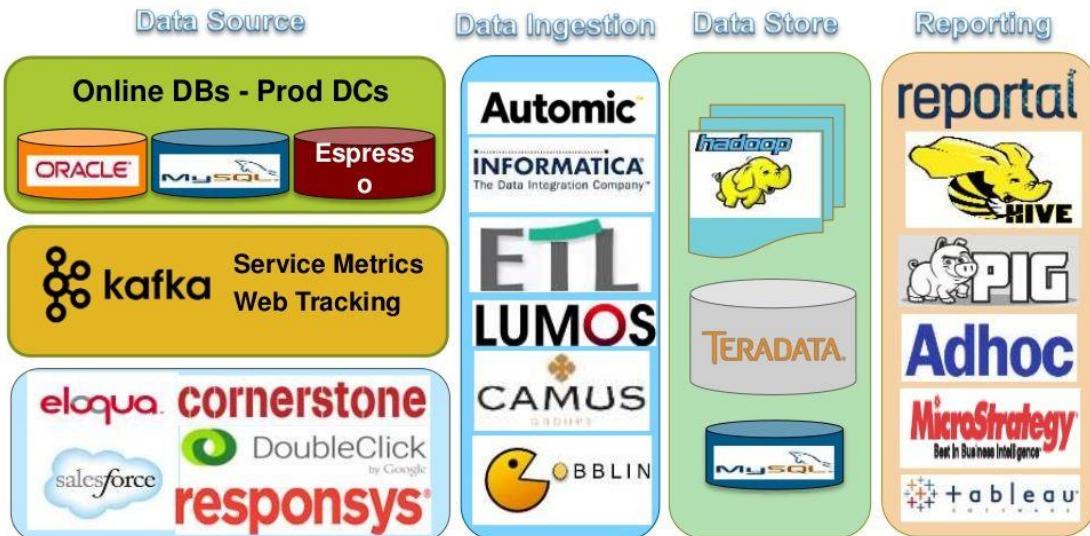
Hadoop a další projekty z ekosystému Hadoop a/nebo big data tak představují součást širší BDA platformy LinkedIn, která je znázorněna na Obr. 3-19. Mezi technologie z projektu a ekosystému Hadoop a/nebo big data, které společnost LinkedIn v této době využívala, patřily systémy (DeZyre, 2016) Hadoop, Pig, Hive, Azkaban (plánovač workflow úloh dávkového zpracování big data), Avro, ZooKeeper, Kafka, Voldemort (NoSQL DBMS), Giraph (systém pro distribuované iterativní zpracování grafů), Avatar (OLAP (analytické zpracování v reálném čase) pro real-time dotazy), Decomposer (knihovna algoritmů pro dekompozici rozsáhlých matic), White Elephant (systém pro parsování logů a vizualizaci provozních metrik clusterů Hadoop) a případně další.

Giraph je využíván pro (DeZyre, 2016) výpočty nad grafy, které vyjadřují vazby mezi členy profesní sítě LinkedIn. Distribuovaná platforma pro příjem, zpracování a uložení proudů dat Kafka je v rámci LinkedIn využívána pro (DeZyre, 2016) příjem a sledování stovek různých typů událostí, jež se v rámci profesní sítě LinkedIn odehrávají (zobrazení stránky, zobrazení profilu, přihlášení, vyhledávání, atd.) v řádu bilionu záznamů denně. Komponentu Avatar poskytující OLAP pro real-time dotazy využívá LinkedIn pro (DeZyre, 2016) zajištění funkcionality „Kdo si zobrazil váš profil“.

Rovněž společnost LinkedIn představuje významného přispěvatele do projektu Hadoop i ekosystému Hadoop a/nebo big data. V rámci projektu Hadoop vývojáři LinkedIn v minulosti významně participovali především na kódu HDFS. V LinkedIn byla vyvinuta řada projektů, z nichž většina byla později vydána jako open source a významnou část z této množiny nyní zastřešuje ASF. Mezi uvedené projekty patří například TonY (komponenta zajišťující nativní podporu TensorFlow v rámci Hadoop), Dynamometer (nástroje pro testování škálování HDFS), Dali Views (funkcionalita virtuálních datasetů

nad abstrakční vrstvou DALi, jejímž tvůrcem je společnost Twitter), Dr. Elephant, Gobblin, Sizr (interaktivní vizualizace využití HDFS), Cubert, Azkaban, Avatar, aj.

Obr. 3-19 Nástroje využívané v rámci BDA společností LinkedIn



Zdroj: data a zpracování (Aruswamy, 2015)

Datová analýza a intelligence je v rámci LinkedIn využívána pro (DeZyre, 2016) zajištění porozumění tomu jaké informace chtějí uživatelé číst, jaké oblasti je nejvíce zajímají, jaké nové funkce nejvíce oceňují a také pro zobrazování agregovaných nejvíce relevantních novinek v reálném čase pro každého jednotlivého uživatele.

Pro zobrazování potenciálních pracovních nabídek na profilu uživatele LinkedIn jsou využívány (DeZyre, 2016) různé algoritmy ML a analýzy textu. Dovednosti a jejich potvrzení jsou zajišťovány prostřednictvím (DeZyre, 2016) algoritmů z kategorie hluboké extrakce informací, zatímco doporučení dovedností jsou zajišťována s využitím NoSQL DBMS Voldemort prostřednictvím mapování id uživatele na seznam ostatních uživatelů, id vlastností a skóre.

Ačkoli byla společnost LinkedIn v roce 2016 předmětem akvizice ze strany Microsoft, v současné době stále provozuje clustery Hadoop v rámci vlastních datových center a migrace do cloudu Microsoft Azure zatím neproběhla.

3.5 Shrnutí

V rámci této kapitoly byl podrobně představen a charakterizován framework pro distribuované zpracování big data Apache Hadoop, a to od vzniku, přes rozvoj, až po současný stav. Dále byla detailně představena architektura a funkcionalita frameworku Apache Hadoop a také jednotlivých projektů ekosystému Apache Hadoop. V závěru kapitoly byly popsány praktické aplikace Hadoop v rámci vybraných společností, konkrétně Twitter, Spotify a LinkedIn.

Z textu kapitoly je patrné, že zatímco projekt Hadoop je svým rozsahem a strukturou relativně malého rozsahu (primárně HDFS, YARN, MapReduce), ekosystém Hadoop a

především ekosystém big data jsou vysoce komplexní. V rámci Tab. 3.3 je proto uvedena kategorie komponent projektu a ekosystému Apache Hadoop dle jednotlivých fází procesu BDA, která charakterizuje, pro jaké účely jsou jednotlivé komponenty využívány. Pro příjem dat ze zdrojových systémů je tak využíván HDFS (jedná se především o nástroje dostupné v rámci API a prostřednictvím klientských rozhraní HDFS, NFS gateway, atd.), dále souborový formát Avro, který lze v rámci příjmu dat využít, a pak také Spark (generické funkce pro nahrání a uložení dat, načítání dat a/nebo ukládání dat z/do souborů Parquet, ORC, JSON, Avro, dále Hive tabulek a také načítání z Kafka, Flume, HDFS/S3, Amazon Kinesis, Twitter a ukládání do HDFS, databázových systémů či dashboardů).

Pro zpracování big data, které může vykazovat různou komplexitu, od prostého přijetí a uložení dat, přes základní transformace až po komplexní ETL, jsou v rámci projektu a ekosystému Apache Hadoop využívány enginy pro distribuované zpracování big data MapReduce, Spark, Pig, Tez a také big data datový sklad Hive, přičemž i v rámci zpracování dat lze využít souborový formát Avro.

Tab. 3.3 Kategorizace komponent projektu a ekosystému frameworku Apache Hadoop z hlediska fází procesu BDA

	Příjem	Zpracování	Uložení	Analýza	Ostatní
Projekt Apache Hadoop	HDFS	MapReduce	HDFS, Ozone*	MapReduce	Hadoop Common, YARN
Ekosystém projektu Apache Hadoop	Avro, Spark	Avro, Hive, Pig, Spark, Tez	Avro, Cassandra, HBase, Hive, Spark	Hive, Mahout, Pig, Spark, Submarine*, Tez	Ambari, Chukwa, ZooKeeper

*Pozn.: * – neprodukční charakter (alfa/beta testování).*

Zdroj: vlastní zpracování

Pro distribuované uložení big data jsou v rámci projektu a ekosystému Hadoop k dispozici HDFS, NoSQL databáze HBase a Cassandra, big data datový sklad Hive, případně také zatím neprodukční distribuované objektové úložiště Ozone. V rámci procesu ukládání lze využít také souborový formát Avro a Spark poskytující řadu funkcí pro ukládání dat především do HDFS a databázových systémů.

Analytické úlohy lze v rámci projektu a ekosystému Hadoop realizovat s využitím enginů pro distribuované zpracování big data MapReduce, Spark, nadstaveb/abstrakcí pro analytické aplikace big data Pig či Mahout, dále prostřednictvím datového big data skladu Hive umožňující dotazovat big data prostřednictvím SQL, resp. HiveQL a také zatím neprodukčního frameworku pro realizaci DL aplikací Submarine. Za účelem optimalizace úloh může být jako součást analýzy přímo nebo zprostředkováně využit také framework Tez.

Kategorie ostatní pak zahrnuje podpůrné/provozní služby clusteru Hadoop, jako je nasazení, správa a monitoring clusteru a/nebo služeb Hadoop (Ambari), správa zdrojů clusteru Hadoop (YARN), monitoring clusteru Hadoop (Chukwa), koordinace distribuovaných služeb v rámci clusteru Hadoop (ZooKeeper) nebo sdílené knihovny a nástroje (Hadoop Common).

Současný stav projektu a ekosystému Hadoop se vyznačuje kombinací aktivně vyvíjených projektů (poslední vydání není starší 12 měsíců), mezi které patří Hadoop Common, HDFS, YARN, MapReduce, Ozone (alfa), Submarine (beta), Ambari, Cassandra, HBase, Hive, Mahout, Spark, Tez, ZooKeeper a také projektů, v jejichž případě vývoj stagnuje (v závorce za názvem je uveden měsíc a rok posledního vydání), mezi které spadá Avro (05/2017), Chukwa (07/2016), Pig (06/2017). V rámci třetí majoritní verze byl do projektu zařazen nový distribuovaného objektového úložiště Ozone, který se zatím nachází ve fázi beta verze. Projekt framework pro realizaci aplikací hlubokého učení nad Hadoop, který nese název Submarine, byl nejprve zařazen mezi podprojekty Hadoop, později byl nicméně přesunut do ekosystému Hadoop a v současné době ještě není v produkčním stavu.

V množině tzv. „Hadoop related projects“ pod záštitou ASF, která je v rámci této práce chápána jako ekosystém projektu Apache Hadoop, je na stránkách projektu Hadoop uvedeno „pouze“ jedenáct projektů. V kategorii big data projektů spravovaných ze strany ASF je nicméně v současné době možné nalézt bezmála 50 různých komponent, přičemž některé jsou součástí distribucí Hadoop a/nebo jsou v kombinaci s Hadoop využívány. Do kategorie ekosystému projektu Hadoop na stránkách frameworku Apache Hadoop by tak bylo možné doplnit některé další nezřídka využívané projekty, jako například enginy pro distribuované zpracování proudů a/nebo dávek dat (Apex, Beam, Flink, Storm), distribuovaný MPP DB (Drill), nástroje z kategorie ETL (Flume, Sqoop) či nástroj pro interaktivní dotazování a vizualizaci dat (Zeppelin).

Z charakterizovaných praktických aplikací frameworku Apache Hadoop v rámci společností Twitter, Spotify a LinkedIn je pak patrné, že Hadoop lze využívat nejen jako data lake pro kontinuálně rostoucí objem dat, která společnost generuje a/nebo využívá, ale především pro pravidelné i ad hoc analytické úlohy realizované dávkově i v (témař) reálném čase a také jako přímou podporu businessu, kdy některé funkce služeb, jež společnosti poskytují, jsou zajišťovány přímo nebo zprostředkováně s využitím projektu Hadoop a/nebo ekosystému frameworku Hadoop/big data (například doporučování informací, produktů či jiných entit pro uživatele, atd.). Větší firmy přitom Hadoop využívají zpravidla jako součást širší BDA platformy.

Firmy, které masivně využívají Hadoop v produkčním prostředí nezřídka narázejí na technické problémy spojené s využitím Hadoop. Jestliže otázka na relevanci využití Hadoop souvisí mimo jiné především s objemem a charakterem dat, která společnost zpracovává („ujistěte se, že máte opravdu big data“), pak určitě překvapí skutečnost, že společnost LinkedIn řešila problémy se škálovatelností HDFS nad rámec několika tisíců uzlů v clusteru Hadoop. Zmíněné firmy pak nezřídka implementovaly vlastní nástroje, s cílem optimalizace a/nebo zaplnění mezer v rámci určitých oblastí BDA realizovaných s využitím Hadoop.

Z uvedených příkladů praktického využití Hadoop je současně zjevné, že produkční nasazení, provoz a využití clusterů Hadoop determinuje řadu provozních nákladů, jejichž významnou složku představují náklady na lidské zdroje (Hadoop vývojáři, architekti, testeři, administrátoři, analytici, datoví vědci, aj.). Tento a další faktory pak determinují trend postupné migrace clusterů Hadoop i realizované BDA do prostředí cloudových platforem poskytujících vyšší míru abstrakce a možnost snížení zmíněných nákladů. V neposlední řadě pak z popisu praktických aplikací vyplývá, že společnosti majoritně nevyužívají pouze projekt Apache Hadoop, ale některou z dostupných distribucí frameworku Apache Hadoop, jejichž podrobná charakteristika je předmětem následující kapitoly.

4 Distribuce frameworku Apache Hadoop

Tato kapitola zahrnuje podrobný popis jednotlivých distribucí frameworku Apache Hadoop, a to včetně popisu okolností vzniku, průběhu rozvoje a charakteristiky současného stavu. Cílem kapitoly je podat především základní informace o aktuálním stavu na trhu poskytovatelů frameworku Apache Hadoop a poskytnout podrobné informace týkající se jednotlivých distribucí Hadoop, které budou předmětem komparace v rámci následujících kapitol této práce.

V úvodu kapitoly jsou nejprve představeny dostupné distribuce frameworku Apache Hadoop, přičemž pro zajištění postupu od obecného ke konkrétnímu je nejprve zařazen popis možností nasazení a provozu Hadoop, následně provedena charakteristika stávajícího trhu poskytovatelů frameworku Apache Hadoop a dále vymezeny dostupné distribuce frameworku Apache Hadoop. Majoritní část kapitoly je věnována podrobnému popisu vybraných distribucí frameworku Apache Hadoop zahrnujícímu charakteristiku vzniku, rozvoje a aktuálního stavu těchto distribucí, architektury a funkcionality a modelu poskytování. Kapitola přináší poznatky týkající se jednotlivých distribucí a jejich srovnání z hlediska architektury, funkcionality a modelů poskytování a představuje první oddíl praktické části této práce.

4.1 Přehled dostupných distribucí frameworku Apache Hadoop

4.1.1 Možnosti nasazení a provozu frameworku Apache Hadoop

Na základě poznatků z předchozí kapitoly lze konstatovat, že framework Apache Hadoop je možné z hlediska provozu nasazovat a spravovat v režimu samostatné, pseudo-distribuované nebo plně distribuované instalace. V případě většiny projektů frameworku a ekosystému Apache Hadoop a rovněž dalších podpůrných služeb, interně/externě využívaných v rámci clusteru Hadoop, je zpravidla možné volit mezi nasazením v rozsahu zajišťujícím základní nebo vysokou dostupnost. Současně je možné nasazovat a provozovat Apache Hadoop v rámci prostředí, mezi něž patří především:

- lokální stroj – typicky pro účely seznámení se s frameworkem, testování komponent frameworku a/nebo ekosystému Hadoop a aplikací pro zpracování a analýzu dat,
- on-premise – provoz na vlastních firemních a/nebo pronajatých dedikovaných strojích umístěných v rámci serverovny a/nebo datového centra,
- cloud – provoz v rámci veřejného nebo privátního (nutně dedikovaného, nikoli nezbytně on-premise) cloudu, v jehož rámci je některá z distribucí Apache Hadoop poskytována formou služby HaaS, někdy také označované jako PaaS (uživatelé využívají funkcionalitu clusteru Hadoop, nestarají se o zajištění jeho provisioningu ani administraci).

Tato kategorizace možností nasazení a provozu je samozřejmě pouze základní, jelikož lze například využívat různé kombinace jak v rovině režimu a provozu nasazení, tak v rovině prostředí, kde je Hadoop provozován a současně je možné clustery Hadoop nasazovat nad fyzickými nebo virtuálními stroji či případně kontejnery nebo v kombinaci uvedených infrastrukturních entit. Rovněž je možné využívat různé typy cloudu (veřejný, privátní, hybridní nebo např. komunitní), přičemž privátní cloud může být provozován v on-premise prostředí. Dále lze v cloudu využívat Hadoop nikoli jako službu, ale vlastními silami zajistit provisioning clusteru infrastruktury a posléze také nasazení a administraci.

Ani výčet uvedený v předchozím odstavci není úplný a jistě existují další možnosti a kombinace. Pro zachování přehlednosti nicméně bude uvažována pouze množina možností nasazení a provozu frameworku, která je uvedena v Tab. 4.1, bez rozlišení typu cloudu, respektive se zaměřením pouze na cloud veřejný, který poskytuje HaaS a případně další služby z kategorie big data.

Tab. 4.1 Základní možnosti nasazení a provozu frameworku Apache Hadoop

		Prostředí		
		Lokální	On-premise	Cloud
Režim nasazení a provozu (instalace)	Samostatná	✓	✓	✓
	Pseudo-distribuovaná	✓	✓	✓
	Plně distribuovaná	✗	✓	✓

Pozn: možnost zajistit nasazení komponent v HA je vyznačena zelenou barvou.

Zdroj: vlastní zpracování

Dalším hlediskem nasazení a provozu frameworku Hadoop představuje způsob, jakým je zajištěn provisioning clusteru infrastruktury, na níž je nasazen cluster Hadoop, a jak je zajištěno vlastní nasazení frameworku a případně dalších projektů ekosystému Apache Hadoop a/nebo big data.

V případě již zmíněného využití Hadoop ve formě služby (HaaS, resp. PaaS) je provisioning, provoz a případně také administrace a/nebo aktualizace infrastruktury i distribuce Hadoop zajištěna ze strany poskytovatele dané služby. Uživatel pouze poskytne parametry požadované konfigurace a po zřízení službu dále již pouze využívá, přičemž má samozřejmě možnost provádět administrační a/nebo konfigurační zásahy do prostředí clusteru, ale v ideálním případě pouze odešle do zřízeného clusteru úlohu zpracování big data ke zpracování. Tato varianta také abstrahuje uživatele od instalátoru dané distribuce, jelikož většinou poskytuje vlastního průvodce, který je zpravidla jednodušší. Druhou možností je z tohoto hlediska zajištění provisioningu clusteru vlastními silami, případně s využitím služeb IaaS a/nebo relevantních automatizačních a/nebo orchestračních strojů.

Pokud jde o způsob nasazení frameworku Apache Hadoop, tento představuje jeden z hlavních faktorů, které determinovaly vznik distribucí frameworku Apache Hadoop. Projekt Hadoop je distribuován ve formě (ASF, 2019m) komprimovaných souborů archivů (*.tar.gz) obsahujících zdrojový kód a již zkomplikované binární soubory jednotlivých podprojektů Apache Hadoop (do verze Hadoop 0.23.0 vydávané společně v rámci jednoho

archivu, od verze 0.23.1 v rámci individuálních archivů pro zdrojový kód a binární soubory) a také jako balíčky pro jednotlivé podporované distribuce OS Linux.

Nasazení tedy může zahrnovat stažení, komplikaci a sestavení balíčku a/nebo binárních souborů ze zdrojového kódu nebo použití dostupných binárních souborů či případně balíčků, které se nasadí v rámci daného prostředí (lokální stroj nebo cluster v rámci on-premise nebo cloudu, aj.). Takto lze manuálně nasadit projekt Apache Hadoop napříč všemi relevantními uzly clusteru. Cluster s nasazeným projektem Apache Hadoop bude nicméně (Alapati, 2017, str. 62) poskytovat pouze služby HDFS, YARN a MapReduce, takže v jeho rámci bude možné pouze uchovávat big data a dávkově je zpracovávat prostřednictvím MapReduce, přičemž pro využití některého z dalších projektů ekosystému Hadoop a/nebo big data by bylo nutné tento do clusteru opět manuálně nasadit. Komplexita a náročnost takového přístupu výrazně poroste především v případě vyššího počtu projektů a uzlů Hadoop clusteru.

Stejně tak v případě škálování clusteru (přidávání dalších uzlů) by bylo nutné manuálně doinstalovat všechny instance relevantních komponent na příslušné uzly. Po nasazení clusteru je nutné jej dále průběžně (re)konfigurovat, administrovat, spravovat, a monitorovat. Zajištění těchto a dalších činností je nicméně v rámci větších clusterů natolik komplexní, že v případě prvních verzí Hadoop se týmy dedikované pro Hadoop věnovaly spíše správě a ladění clusteru než realizaci vlastních analytických úloh. Uvedené a další skutečnosti determinovaly vznik distribucí frameworku Apache Hadoop, které zpravidla disponují grafickým rozhraním pro efektivní nasazení, provoz, administraci a správu clusterů Hadoop a navíc kromě Apache Hadoop poskytují množinu dalších projektů z ekosystému Hadoop a/nebo big data, čímž umožňují pokrýt větší část procesu BDA a více případů užití a/nebo problémových oblastí. Významnou přidanou hodnotu distribucí Apache Hadoop lze také identifikovat ve skutečnosti, že každá verze dané distribuce je zpravidla testována a garantuje vzájemnou kompatibilitu využitých verzí jednotlivých komponent, a potenciálně také zpětnou kompatibilitu s předchozími verzemi dané distribuce Hadoop a určitou úroveň stability. Sám spoluzakladatel projektu Hadoop tyto nedostatky projektu Apache Hadoop identifikoval a relativně brzy se připojil k týmu společnosti, která na trh v roce 2008 uvedla první oficiální distribuci frameworku Apache Hadoop – Cloudera CDH.

Pokud jde o přednosti a nedostatky jednotlivých uvedených možností nasazení a provozu frameworku Apache Hadoop, jejich podrobná charakteristika přesahuje rámec této práce, nicméně klíčová kritéria představují především pořizovací a provozní náklady zdrojů a technologií, náklady na nezbytné lidské zdroje a otázka jejich dostupnosti. Daná kritéria se napříč jednotlivými variantami nasazení a provozu Hadoop liší a ještě výrazněji v závislosti na dalších podmínkách a účelu využití Hadoop. Jako příznivý trend lze hodnotit skutečnost, že zatímco dříve byl Hadoop v produkčním prostředí pro malé a střední firmy spíše nedostupný, díky rozvoji služeb BDA v rámci veřejných cloudů jej možnou prakticky okamžitě využívat i podniky z dané kategorie při zajištění potřebné flexibilité dané možností kdykoli škálovat požadovaným směrem nebo prakticky okamžitě využívání služby ukončit.

4.1.2 Trh poskytovatelů frameworku Apache Hadoop

Framework Apache Hadoop je vydáván jako open source, pod licencí Apache License 2.0, stejně jako ostatní projekty ekosystému Hadoop a/nebo big data, které zastřešuje ASF. Nicméně z předchozí podkapitoly je zřejmé, že pro smysluplné využití Hadoop v oblasti BDA je nutné v kombinaci s frameworkem Apache Hadoop nasadit a použít určitou množinu dalších projektů a současně, že nasazení, provoz, správa, administrace a monitoring clusterů Hadoop představuje značně komplexní a na zdroje náročný proces.

Po vzniku projektu a vydání první verze Hadoop se tudíž okamžitě začal formovat trh poskytovatelů řešení Hadoop a v současnosti lze identifikovat především následující kategorie poskytovatelů Hadoop (Gualtieri, a další, 2014, str. 3-4):

- **framework a ekosystém Apache Hadoop** – projekt frameworku Apache Hadoop a dalších projektů ekosystému Hadoop, které ASF poskytuje jako open source, a jež může kdokoli libovolně kombinovat, vlastními silami nasadit a provozovat bez formální podpory ze strany poskytovatele; hlavním produktem jsou volně dostupné moduly projektu a ekosystému Hadoop pokrývající určitou oblast funkcionality pro zajištění BDA,
- **distribuce Hadoop** – distribuce frameworku Apache Hadoop a určité množiny projektů z ekosystému Hadoop a/nebo big data, které poskytují společnosti, jež se primárně zaměřují na vývoj, poskytování, podporu a propagaci určité distribuce; prodejci prodávají distribuci zákazníkům přímo nebo prostřednictvím sítě partnerů a licence jednotlivých distribucí se mohou lišit; hlavním produkt představuje framework Hadoop a množina projektů, které lze efektivně a rychle nasadit a začít využívat; doplňkové služby mohou zahrnovat poradenství, podporu, atd.,
- **poskytovatelé podnikového software, kteří mají v nabídce mimo jiné Hadoop** – všichni velcí poskytovatelé podnikového software mají definovánu a uplatňují určitou Hadoop strategii, protože Hadoop představuje esenciální technologii pro management dat, přičemž většina z těchto firem uzavírá strategické partnerství s poskytovateli distribucí Hadoop, zatímco jiné vyvíjejí a využívají vlastní distribuci/derivát určité distribuce; distribuce Hadoop není hlavním produktem, naopak představuje pouze jednu složku širokého portfolia podnikového software,
- **Hadoop v cloudu** – Hadoop, respektive určitá distribuce Hadoop je poskytována v cloudu formou služby (HaaS) a kromě toho mohou být k dispozici další služby, které lze v rámci BDA v kombinaci s HaaS využít, např. Kafka jako služba, NoSQL DBMS jako služba, atd.; hlavním produktem je distribuce Hadoop poskytovaná formou služby, včetně infrastruktury, kdy uživatel se nestará o nasazení ani správu infrastruktury a v řádu desítek minut může začít cluster Hadoop využívat a podle potřeby jej flexibilně škálovat,
- **poskytovatelé big data řešení** – firmy poskytující infrastrukturu, Hadoop, další platformy pro management dat a analytické nástroje, jež lze přizpůsobit pro specifické potřeby jednotlivých zákazníků, přičemž tyto komplexní platformy pro zajištění celého procesu BDA zpravidla do určité úrovně abstrahuji uživatele od nižších vrstev a Hadoop v jejich rámci představuje pouze jeden z modulů nižší

úrovně tvořící komplexní BDA platformu; hlavní produkt v tomto případě reprezentuje zpravidla rozhraní pro zadávání a realizaci úloh napříč celým životním cyklem BDA,

- **doplňkové projekty ekosystému Hadoop/big data** – společnosti specializující se na vývoj a/nebo poskytování specifických nástrojů pro Hadoop poskytujících například funkcionality pro správu Hadoop clusterů, integraci dat, modelování dat, realizaci prediktivní analýzy, vizualizaci dat z/v rámci Hadoop, apod., a představují tak další doplňkové projekty ekosystému Hadoop/big data; hlavní produkt spočívá v poskytnutí určité specifické oblasti funkcionality v rámci a/nebo nad Hadoop.

Dílčí charakteristiky řešení z uvedených kategorií, včetně zástupců poskytovatelů, jsou uvedeny v rámci tabulky Tab. 4.2.

Tab. 4.2 Kategorie poskytovatelů Hadoop řešení

Kategorie	Využití Hadoop	Poskytovatelé/produkty	Nasazení	Úroveň abstrakce
Framework a ekosystém Apache Hadoop	Jednotlivé projekty a jejich kombinace	ASF	L, O, C	Nízká
Distribuce Hadoop	Framework Hadoop + určitá množina dalších projektů	Cloudera CDH, Hortonworks HDP, MapR MDP	L, O, C,	Nízká/střední
Velcí tvůrci a poskytovatelé firemního software	Distribuce Hadoop jako součást portfolia poskytovaného SW	Oracle, SAP, IBM, Microsoft, Pivotal, Teradata	O, C	Nízká/střední
Hadoop v cloudu	Distribuce Hadoop jako služba (HaaS)	Amazon EMR, HDInsight, Cloud Dataproc, Altiscale	C	Střední
Poskytovatelé big data řešení	Distribuce Hadoop jako jeden z modulů nižších vrstev BDA platformy	CenturyLink, Datameer, SAS Visual Analytics	O, C	Vysoká
Doplňkové projekty ekosystému Hadoop/big data	Projekty přidávající určitou funkcionality do ekosystému Hadoop/big data	DataTorrent, Pentaho, Patfora, Revelix, Software AG, Talend	L, O, C	Nízká/střední

Pozn: L – lokální, O – on-premise, C – cloud.

Zdroj: data (Gualtieri, a další, 2014, str. 3-4), vlastní zpracování

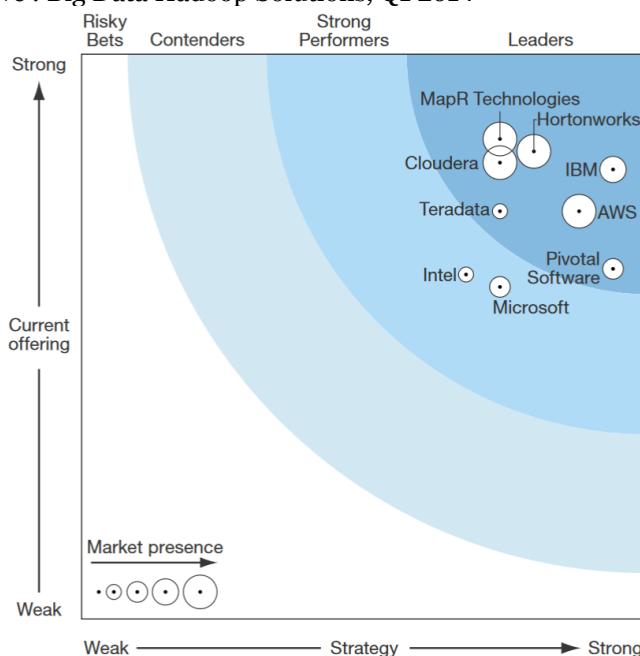
Na trhu samozřejmě figurují i poskytovatelé z jiných kategorií, kteří se zaměřují například na poskytování služeb integrace, poradenství, outsourcingu BDA, atd. Trh poskytovatelů

Hadoop tak lze označit za velmi rozmanitý zahrnující vysoký počet poskytovatelů a řešení vykazujících množství různých forem a současně lze konstatovat, že Hadoop nachází využití i v rámci širší oblasti BDA.

Stav trhu poskytovatelů Hadoop dobře ilustrují především tržní analýzy, které publikují například společnosti Gartner (Gartner Magic Quadrant) nebo Forrester (Forrester Wave), a další. Analýzy trhu zpravidla zahrnují určitou podoblast BDA a nikoli celou oblast big data, protože tato je příliš komplexní.¹⁸

Na základě informací z dostupných studií lze konstatovat, že v roce 2014 společnost Forrester Research (Gualtieri, a další, 2014, str. 6, 8) jako lídry trhu Hadoop big data řešení vyhodnotila poskytovatele (v závorce za názvem společnosti je uveden produkt/služba, které byly hodnoceny) Hortonworks (HDP), IBM (InfoSphere BigInsights), AWS (Amazon EMR), MapR Technologies (MapR Data Platform (MDP) M3, M5, M7), Cloudera (Cloudera Enterprise), Pivotal Software (Pivotal HD) a Teradata (TDH) a mezi silné hráče zařadila společnosti Microsoft a Intel (Intel Distribution for Apache Hadoop). Tato analýza trhu (vizualizace výsledků je znázorněna na Obr. 4-1) byla zaměřena především na poskytovatele distribucí Hadoop, velké poskytovatele firemního software a také poskytovatele Hadoop v cloudu.

Obr. 4-1 Forrester Wave : Big Data Hadoop Solutions, Q1 2014



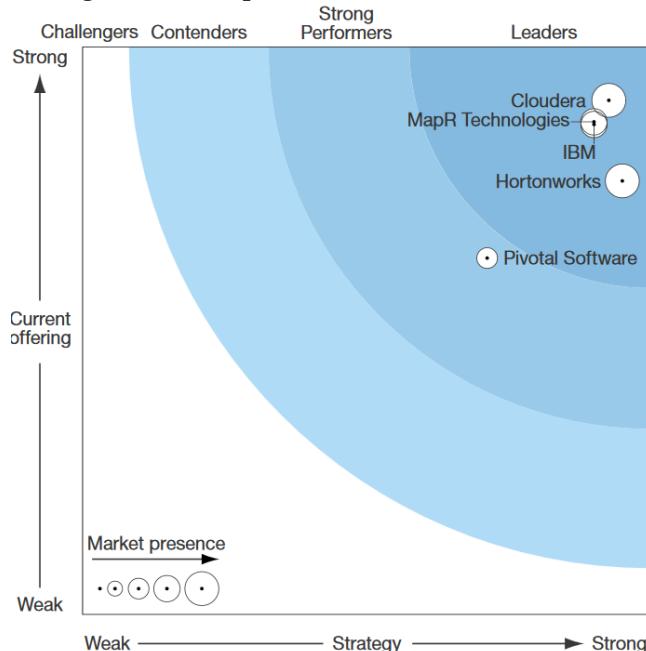
Zdroj: data a zpracování (Gualtieri, a další, 2014, str. 8)

V prvním čtvrtletí roku 2016 vydala společnost Forrester analýzu trhu zaměřenou výhradně na producenty distribucí Hadoop a následně ve druhém čtvrtletí analýzu prezentující stávající situaci na trhu poskytovatelů Hadoop v cloudu. Na základě

¹⁸⁾ Komplexitu množiny technologií oblasti big data dobře ilustruje například infografika nesoucí název „Big data & AI Landscape“, jíž společně s kolegy vytvořil Matt Truck. Vzhledem k rozsahu není v rámci této práce uvedena ani jako příloha. Verzi z roku 2018 lze nalézt na [blogu](#) tohoto investora.

dostupných informací lze uvést, že (Gualtieri, a další, 2016a) v roce 2016 mezi klíčové poskytovatele Hadoop v cloudu patřily společnosti Altiscale, Amazon, Google, IBM, Microsoft, Oracle, Qubole a Rackspace. V tomtéž roce pak analytici společnosti Forrester na trhu poskytovatelů distribucí Hadoop identifikovali (Gualtieri, a další, 2016b, str. 5, 6) čtyři lídry (v závorce je uveden název distribuce, která byla předmětem hodnocení), mezi které patřily společnosti Cloudera (Cloudera Enterprise), MapR Technologies (The MapR Distribution including Apache), IBM (IBM BigInsights for Apache Hadoop) a Hortonworks (Hortonworks Data Platform) a dále jednoho silného hráče, kterým byla společnost Pivotal (Hadoop Pivotal HD). Výsledky této tržní analýzy jsou znázorněny na Obr. 4-2. Autoři dané analýzy přitom zdůraznili, že (Gualtieri, a další, 2016b, str. 5-6) výběr distribuce byl opravdu nesnadný, protože ačkoli lídrem zůstala nadále první z distribucí Hadoop, tj. Cloudera, také ostatní poskytovatelé z kategorie lídrů poskytovali vysoce pokročilou distribuci Hadoop umožňující zajistit plně produkční, resp. enterprise nasazení. Z obrázku je zřejmé, že největší podíl na trhu zaujímaly distribuce Cloudera a Hortonworks a zatímco Cloudera vykazovala vyšší úroveň z hlediska nabídky (konfigurace, škálovatelnost, výkon, HA, disaster recovery, administrace, bezpečnost, funkcionality, vývojové nástroje, atd.), Hortonworks dosahovala lepší pozice v rovině strategie (schopnost konkurovat na trhu distribucí Hadoop, realizovat stanovenou strategii a podporovat zákazníky).

Obr. 4-2 Forrester Wave : Big Data Hadoop Distributions, Q1 2016



Zdroj: data a zpracování (Gualtieri, a další, 2016b, str. 6)

Z roku 2019 je k dispozici také analýza trhu od společnosti Forrester, která (Yuhanna, a další, 2019, str. 1) vyhodnotovala stav trhu cloudových platform Hadoop/Spark. Tato analýza kombinovala kategorie poskytovatelů frameworku a ekosystému Hadoop (v tomto případě projektu Spark) a distribucí Hadoop poskytovaných v cloudu. Výsledky analýzy jsou znázorněny na Obr. 4-3, ze kterého je patrné, že oproti předchozím uvedeným analýzám se poskytovatelé hodnocených řešení nacházeli ve všech vrstvách (také v tomto případě je v závorce za názvem společnosti uveden produkt/služba, které byly hodnoceny). Mezi vyžvývatele společnost Forrester zařadila (Yuhanna, a další, 2019, str. 4, 6) Rackspace

(Managed Big Data), mezi soupeře Qubole (Qubole Data Service), mezi silné hráče Google (Cloud Dataproc), MapR (MDP), Huawei (FusionInsight), SAP (SAP Cloud Platform Big Data Services, SAP Data Hub), Oracle (Oracle Big Data Cloud Services) a konečně mezi lídry trhu Amazon (EMR), Microsoft (Azure HDInsight, Azure Databricks), Hortonworks (HDP, Hortonworks DataPlane) a Cloudera (Cloudera Enterprise (Altus)).

Obr. 4-3 Forrester Wave : Cloud Hadoop/Spark Platforms, Q1 2019



Zdroj: data a zpracování (Yuhanna, a další, 2019, str. 4)

V rovině cloudových řešení poskytovatelé distribucí Hadoop (především Hortonworks, Cloudera, MapR) většinou poněkud zaostávali za lídry v oblasti cloud computingu (AWS, Azure, GCP), kteří ke službě HaaS poskytovali řadu dalších relevantních služeb a možnost využít HaaS v kombinaci s dalšími oblastmi analýzy dat, apod. Z dané analýzy je nicméně zřejmé, že i poskytovatelé distribucí Hadoop význam možnosti nasazení a poskytování Hadoop v rámci cloudu chápali a snažili se jej pro zákazníky zajistit.

Společnost Gartner zatím analýzu trhu, která by se zaměřovala výhradně na distribuce Hadoop, nepublikovala. Všechny lídry poskytovatelů distribucí Hadoop, tedy Cloudera, Hortonworks i MapR nicméně Gartner hodnotil v rámci analýzy trhu řešení managementu dat pro analýzu zveřejněné v letech 2016 a 2018. Přitom je zajímavé, jak je patrné z Obr. 4-4, zatímco v roce 2016 byli všichni zmínění poskytovatelé distribuce Hadoop zařazeni do (Nemschoff, 2016) kvadrantu vizionářů, tj. subjektů s dobrou vizí, ale nižší schopností konkurovat na daném trhu, v roce 2018 byli (Ronthal, a další, 2018, str. 3) všichni uvedení poskytovatelé distribuce Hadoop uvedeni v rámci kvadrantu specifických řešení (niche players), tedy subjektů s nižší úrovní vize i schopnosti konkurovat na daném trhu. Skutečnost, že uvedené subjekty nedosahovaly kvadrantu lídrů trhu není překvapivá vzhledem ke skutečnosti, že se nejedná o trh distribucí Hadoop, nicméně je zvláštní, že v průběhu dvou let došlo ke zhoršení hodnocení v rovině vize a k posunu z kvadrantu

vizionářů do kvadrantu specifických řešení. Tento vývoj může souviseť s trendem růstu významu provozu software a služeb zpracování a analýzy dat v prostředí cloudu tak, jak je zřetelný z předchozích uvedených analýz trhu, ale může mít i množství jiných příčin.

Obr. 4-4 Magic Quadrant for Data Management Solutions for Analytics – 2016, 2018



Zdroj: data a zpracování (Nemschoff, 2016; Ronthal, a další, 2018, str. 3)

Na tomto místě je relevantní uvést, že v praxi jsou také využívány alternativy Hadoop, které množinu projektů frameworku a/nebo ekosystému Hadoop do určité míry substituují, a jež v posledních letech zahrnují především systémy odpovídající tzv. (Kumar, 2018, str. 88-91) architektuře Lambda. V rámci této architektury (Kumar, 2018, str. 88-91) jsou data nejprve přijata prostřednictvím vrstvy pro dávkové zpracování a následně nahrána/předána vrstvě pro zpracování, dotazy a analýzu v reálném čase. Alternativu může představovat také architektura Kappa, která (Kumar, 2018, str. 88-91) přistupuje k veškerým datům jako k proudu dat, zahrnuje pouze vrstvu pro zpracování proudů dat v reálném čase a data pro real-time dotazy tudíž poskytuje ihned po přijetí.

V reakci na dynamický vývoj projektu Apache Hadoop, komponent Apache Hadoop a distribucí Apache Hadoop, byla založena iniciativa otevřené datové platformy (ODPi) představující (ODPi, 2019a) „*... neziskovou organizaci zaměřenou na zjednodušení a standardizaci ekosystému big data prostřednictvím obecných referenčních specifikací a testovacích sad.*“ Pro zajištění vzájemné kompatibility mezi různými distribucemi Hadoop a big data technologií vydává ODPi dvě následující specifikace (ODPi, 2019c):

- ODPi specifikace běhového prostředí – definuje standardní konfiguraci distribuce Hadoop,
 - ODPi specifikace provozu – definuje standardizovanou metodiku pro správu a instalaci aplikací v rámci Hadoop clusteru.

Využitím řešení a/nebo služeb, které jsou v souladu s uvedenými specifikacemi, se tak zákazníci a uživatelé mohou ochránit proti tzv. proprietárnímu zamčení (vendor lock-in) a eliminovat potenciální náklady na migraci k jinému poskytovateli v případě, že by využívali určitou komponentu, která není u jiného poskytovatele dostupná, apod. V rámci

Tab. 4.3 je uveden seznam členů ODPi, včetně toho, zda splňují jednotlivé uvedené specifikace ODPi. Obě specifikace splňuje pouze řešení společnosti IBM. Soulad se specifikacemi je ze strany ODPi potvrzen pro konkrétní produkty specifické verze.

Tab. 4.3 Přehled poskytovatelů Hadoop, kteří jsou v souladu se specifikacemi ODPi

Poskytovatel	ODPi specifikace běhového prostředí	ODPi specifikace provozu
Altiscale	✓	
Arenadata	✓	
AsiaInfo	✓	
DataTorrent		✓
Hortonworks	✓	
IBM	✓	✓
Infosys	✓	
Pivotal		✓
SAS		✓
Syncsort		✓
wanDISCO		✓
Xavient		✓

Zdroj: data (ODPi, 2019b), vlastní zpracování

Z předchozího textu této podkapitoly je zřejmé, že trh poskytovatelů Hadoop je komplexní a zahrnuje řešení a/nebo produkty či služby různých kategorií, přičemž poskytovatelé distribucí Hadoop představují pouze část trhu poskytovatelů Hadoop reprezentující jednu z dílčích kategorií. Většina poskytovatelů z dalších kategorií nicméně produkty poskytovatelů distribucí Hadoop využívá, v některých případech za tímto účelem došlo i k navázání strategického partnerství. V posledních letech pak roste význam schopnosti provozovat Hadoop v cloudu, z čehož samozřejmě těží poskytovatelé Hadoop z kategorie Hadoop v cloudu. Na tento vývoj se snaží reagovat také poskytovatelé distribucí frameworku Apache Hadoop.

4.1.3 Přehled dostupných distribucí frameworku Apache Hadoop

V literatuře v současné době neexistuje jednotná či všeobecně akceptovaná definice pojmu distribuce Hadoop. V rámci analýzy trhu společnosti Forrester lze nalézt vymezení specifikující, že distribuce Hadoop (Gualtieri, a další, 2016b, str. 4) „... je založena na Apache Hadoop a dalších open source projektech ekosystému Hadoop.“ Jiná definice charakterizuje distribuci Hadoop jako (Singh, a další, 2019, str. 18) „... množinu komponent zahrnující projekt frameworku Apache Hadoop a další open source projekty ekosystému Hadoop, jež jsou společně zabaleny do jediného distribučního balíku, který lze snadno použít a spravovat.“

Dále je možné nalézt definici uvádějící, že distribuce Hadoop (Alapati, 2017, str. 60) „... zahrnuje oficiální vydání frameworku Apache Hadoop a další projekty a nástroje, které jsou součástí ekosystému Hadoop a/nebo portfolia daného poskytovatele.“ Ještě jiná definice pak uvádí, že distribuce Hadoop (Achari, 2015, str. 18) „... poskytuje Hadoop společně s dalšími projekty, které jsou vzájemně kompatibilní a pro něž je poskytována komerční podpora ... pro management Hadoop clusteru, poskytuje distribuce Hadoop grafické rozhraní pro nasazení, administraci a monitoring Hadoop clusterů.“

Na základě výše uvedených definic budou v dalším textu této kapitoly blíže představeny dostupné distribuce frameworku Apache Hadoop odpovídající uvedeným vymezením tím, že jsou tvořeny množinou komponent zahrnující projekt Apache Hadoop a další open source projekty ekosystému Hadoop a/nebo big data, přičemž daná množina komponent je součástí jediného distribučního balíku, který lze snadno použít a spravovat. Naopak zde nebudou popsány deriváty distribucí frameworku Apache Hadoop, v jejichž případě tvoří některá z distribucí Hadoop majoritní část výsledného produktu, ani služby, jejichž podstatou je prakticky nasazení určité distribuce Hadoop, ani služby založené majoritně na projektu Apache Spark, apod.

Významným přínosem distribucí Hadoop je (Achari, 2015, str. 18) snížení podílu zdrojů potřebných pro nasazení, provoz i monitoring clusterů Hadoop a také dostupnost nástrojů a utilit podporujících snazší a rychlejší vývoj produktu nebo projektu. Výhody, respektive přednosti, které distribuce Hadoop poskytují, lze rozčlenit především do následujících kategorií (Singh, a další, 2019, str. 18):

- instalace – distribuce Hadoop poskytuje nástroje a rozhraní pro snadnou instalaci jednotlivých komponent napříč clusterem,
- správa balíků – distribuce Hadoop je nad rámec frameworku Apache Hadoop tvořena množinou open source projektů, které jsou v rámci balíku dané distribuce konfigurovány a testovány způsobem zajišťujícím, že dané komponenty společně v rámci clusteru fungují; využitím balíků distribuce je možné efektivně zajistit také aktualizace a instalace nových komponent,
- údržba clusteru – distribuce Hadoop poskytuje přívětivé grafické rozhraní, jehož prostřednictvím lze snadno konfigurovat, administrovat, spravovat a monitorovat jednotlivé služby a komponenty Hadoop clusteru,
- podpora – v případě většiny distribucí Hadoop je možné si zajistit placenou podporu ze strany poskytovatele dané distribuce, která je s ohledem na komplexitu provozu clusterů Hadoop nezřídka v průběhu životního cyklu rozsáhlých produkčních clusterů klíčová.

Přehled distribucí frameworku Apache Hadoop odpovídajících výše uvedeným definicím, je obsažen v Tab. 4.4. Z dané tabulky je patrné, že prakticky první distribuci Apache Hadoop představoval projekt Apache Hadoop. Pro tuto distribuci se používá také označení Vanilla Hadoop.

V roce 2008 se na trhu objevila první distribuce přesahující rámec projektu Apache Hadoop, a to Cloudera's Distribution Including Apache Hadoop (CDH). Tato distribuce jako první (Singh, a další, 2019, str. 19) poskytla uživatelům, respektive administrátorům

grafické rozhraní pro nasazení, správu a monitoring clusteru Hadoop, a to prostřednictvím komponenty nesoucí název Cloudera Manager. Od ostatních distribucí se odlišuje mimo jiné tím, že poskytuje nativní analytický databázový systém pro Hadoop nesoucí název Impala (nyní open source zastřešovaný ze strany ASF).

Tab. 4.4 Přehled distribucí frameworku Apache Hadoop

Název	Poskytovatel	Vznik	Open source
Apache Hadoop (Vanila)	ASF	2006	✓
Cloudera's Distribution Including Apache Hadoop (CDH)	Cloudera	2009	✓
MapR Data Platform	MapR Technologies	2009	✗
Amazon Elastic MapReduce (EMR)*	Amazon	2009	✗
Hortonworks Data Platform (HDP)	Cloudera	2011	✓
IBM BigInsights	IBM	2011	✗
Bigtop	ASF	2011	✓
E-MapReduce*	Alibaba Cloud	2011	✗
DataStax Enterprise Hadoop	DataStax	2011	✗
Pivotal HD	Pivotal Software	2013	✗
WANdisco Distro	WANdisco	2013	✓
Intel Distribution for Apache Hadoop	Intel	2013	✗
HDInsight*	Microsoft	2013	✗
FusionInsight	Huawei	2015	✗
Cloud Dataproc*	Google	2016	✗
Syncfusion Big Data Platform	Syncfusion	2016	✗
Arenadata Hadoop	Arenadata	2017	✗

Pozn.: distribuce, které v současné době nejsou na trhu nebo jsou nabízeny, ale nezahrnují vlastní distribuci daného poskytovatele, ale naopak distribuci některého z jiných poskytovatelů, jsou vyznačeny oranžovou barvou; * - cloudově nativní distribuce frameworku Apache Hadoop; sloupec vznik obsahuje rok, kdy byla uvolněna produkční (generally available) verze.

Zdroj: data (Singh, a další, 2019, str. 19-20; Alapati, 2017, str. 60; Jain, 2017, str. 94-97), vlastní zpracování

Distribuce MapR Data Platform (MDP) byla na trh uvedena v roce 2008. Tato distribuce je charakteristická (Singh, a další, 2019, str. 19) vlastní reimplementací některých komponent projektu a/nebo ekosystému Apache Hadoop, především distribuovaného souborového systému (MapR Filesystem namísto HDFS), NoSQL databázového systému (MapR Database namísto HBase), atd. Také tato distribuce poskytla grafické rozhraní pro nasazení a správu clusteru Hadoop, které v tomto případě nese označení MapR Control System. Od ostatních distribucí se MapR odlišuje především tím, že (Singh, a další, 2019, str. 19) s cílem zajištění vyššího výkonu substituuje některé open source projekty

frameworku Apache Hadoop a ekosystému Hadoop a/nebo big data vlastními implementacemi.

V roce 2009 společnost Amazon v rámci cloudové platformy AWS představila (Singh, a další, 2019, str. 20) vlastní distribuci pod názvem Amazon Elastic MapReduce (EMR), která je nicméně proprietární a je dostupná právě pouze v cloudu v rámci služby EMR. Po určitou dobu bylo v rámci této služby možné využívat distribuci MapR, nicméně v současné době tato možnost k dispozici není a MapR lze v rámci AWS pouze automatizovaně nasadit nad cloudovými virtuálními stroji. V rámci vytvoření nového clusteru EMR uživatel v současné době mimo jiné (Amazon, 2019b, str. 14-15) specifikuje vydání EMR a následně služby, které budou v rámci clusteru nasazeny, přičemž se může jednat o množinu zahrnující například Hadoop, Flink, Ganglia, HBase, Hive, Hue (grafické rozhraní pro SQL dotazování v rámci datových skladů), Jupyter (interaktivní analytický notebook), Livy, Mahout, MXNet, Oozie, Phoenix, Pig, Presto, Spark, Sqoop, TensorFlow, Tez, Zeppelin, ZooKeeper, aj. Amazon EMR představuje (Singh, a další, 2019, str. 20) nejvíce aktivně využívanou cloudovou distribuci Apache Hadoop.

Společnost Hortonworks v roce 2011 na trh uvedla další distribuci frameworku Apache Hadoop, a to Hortonworks Data Platform (HDP). Tato distribuce byla (Alapati, 2017, str. 60) od počátku k dispozici jako open source a společnost Hortonworks získávala tržby především z placené podpory. Společnost Hortonworks představuje průvodního autora projektu Ambari poskytujícího funkcionality pro nasazování, správu a monitoring Hadoop clusteru, který je stále součástí distribuce HDP. Tato distribuce je charakteristická tím, že poskytuje většinu projektů ekosystému Hadoop a ještě další projekty ekosystému big data se zaměřením primárně na open source komponenty.

Z Tab. 4.4 je zřejmé, že v případě distribucí CDH i HDP je jako poskytovatel uvedena společnost Cloudera. Důvodem je skutečnost, že (Cloudera, 2018b) v říjnu 2018 oznámila společnost Cloudera akvizici Hortonworks, která byla (Cloudera, 2019c) úspěšně dokončena v lednu roku 2019. Společnost Cloudera kromě vlastního portfolia nadále poskytuje distribuci HDP a další produkty a služby z produkce společnosti Hortonworks.

Distribuce CDH, MDP i HDP jsou na trhu stále dostupné, a protože jejich podrobný popis bude předmětem příslušných následujících podkapitol, nebudou na tomto místě dále popsány. Představují nicméně tři největší hráče na trhu distribucí frameworku Apache Hadoop. V minulosti byly například distribuce Cloudera, MapR i Hortonworks označeny jako (Gaultieri, a další, 2016b, str. 5) lídři trhu distribucí frameworku Apache Hadoop poskytující best-of-breed enterprise řešení.

V roce 2011 byla na trh uvedena také distribuce Hadoop společnosti IBM, která nesla název (Jain, 2017, str. 95) IBM InfoSphere BigInsights, později pouze IBM BigInsights, přičemž společnost IBM tuto distribuci vyvinula jako součást portfolia poskytovaných big data služeb, na níž byla založena HaaS služba v rámci cloudu společnosti IBM. Druhá verze distribuce IBM BigInsights zahrnovala (Ebbers, a další, 2013, str. 18) projekt Hadoop, Jaql (dotazovací jazyk pro JSON využívaný pro analýzu rozsáhlých semi-strukturovaných dat), Jaql server, BigInsights konzoli poskytující grafické rozhraní pro centrální správu Hadoop clusteru, správu úloh a procházení HDFS, dále workflow plánovač BigInsights, BigSheets – grafický nástroj pro interaktivní analýzu, Avro, Derby

(Java RDBMS), Flume, HBase, Hive, Lucene, Oozie, Pig a ZooKeeper. Součástí distribuce byla také (Lublinsky, a další, 2013, str. 11) komponenta instalátoru (IBM installer).

V dalších letech společnost vydala několik nových majoritních verzí, přičemž jednou z hlavních přidaných hodnot distribuce byl (Adrian, 2017) MPP databázový systém IBM Big SQL nasazovaný přímo nad HDFS a umožňující realizovat výkonné dotazy big data prostřednictvím SQL a efektivně využívat Hive, HBase, Spark, aj. Na konci roku 2017 nicméně společnost IBM vývoj vlastní distribuce Hadoop ukončila, jelikož (Adrian, 2017) navázala partnerství s producentem jedné z hlavních distribucí Apache Hadoop, společnosti Hortonworks. Zákazníci IBM tak mají možnost dále využívat službu HaaS, kterou IBM v rámci vlastního cloutu nabízí, a jež je založena na distribuci HDP.

Distribuce frameworku Apache Hadoop nesoucí název Apache Bigtop se na trhu objevila v roce 2011. První produkční vydání Bigtop bylo uvolněno v roce 2015 a zatím poslední vydání této distribuce (1.3.0) pochází z listopadu 2018. Distribuce Bigtop je v případě zmíněného posledního vydání (ASF, 2018d) založena na Hadoop 2.8.4 a dále zahrnuje komponenty, mezi které patří Alluxio (orchestrace dat pro analýzu a ML v cloutu), Ambari, Apex, Crunch (framework pro tvorbu, testování a realizaci MapReduce pipelines), DataFu (knihovny pro dolování dat a statistiku), Flink, Flume, Giraph, Greenplum Database (GPDB – RDBMS založený na PostgreSQL), Hama (framework pro BSP BDA), HBase, Hive, Ignite (in-memory framework pro SQL dotazy nad RDBMS a NoSQL), Kafka, Mahout, Phoenix, Quantcast File System (QFS – DFS, alternativa k HDFS), Solr, Spark, Sqoop, Tajo (datový sklad pro big data nad Hadoop), Tez, Yahoo! Cloud System Benchmark (YCSB – framework pro benchmarking NoSQL DBMS), Zeppelin, ZooKeeper a další interní Bigtop komponenty.

Apache Bigtop tak představuje (ASF, 2018b) „... projekt pro správce infrastruktury a datové vědce hledající komplexní správu balíku, testování a konfiguraci vedoucích open source big data komponent.“ Distribuce Apache Bigtop poskytuje (ASF, 2018b) připravené distribuční balíky umožňující nasadit, provozovat a spravovat vlastní Hadoop cluster, dále integrovaný testovací framework a také připravené recepty a obrazy umožňující nasazovat BigTop nad různými virtualizačními technologiemi (v cloutu, nad kontejnery Docker, atd.). Hlavním rozdílem oproti ostatním distribucím typu CDH, apod., je skutečnost, že (ASF, 2012):

- zatímco ostatní distribuce nabízejí pro lepší zkušenosť zákazníků záruku funkcionality tím, že pro pečlivě vybrané projekty zajišťují patche, distribuce Bigtop vychází pouze z open source kódu jednotlivých projektů,
- zatímco ostatní distribuce kladou zvláštní důraz na stabilitu a zpětnou kompatibilitu, distribuce Bigtop je agresivnější pokud jde o poskytnutí nejnovějších verzí komponent ekosystému Hadoop.

Důsledkem tohoto přístupu samozřejmě může být horší zkušenosť zákazníků v důsledku nižší úrovně garance funkcionality a/nebo interoperability jednotlivých projektů, nižší stability a nižší úrovně zpětné kompatibility, na něž ostatní distribuce kladou vysoký důraz. Tyto okolnosti jsou důvodem, proč daná distribuce nebude dále blíže popsána a ani nebude předmětem komparace.

Další z cloudových distribucí frameworku Apache Hadoop uvedla na trh (Alibaba, 2019) v roce 2011 společnost Alibaba, provozovatel Alibaba cloud (Aliyun), a to pod názvem E-MapReduce. Společnost vydává novou minoritní verzi této distribuce téměř každý měsíc, přičemž v současné době je k dispozici (Alibaba, 2019) verze 3.2.0, založená na Hadoop 2.8.5 a zahrnující kromě Apache Hadoop komponenty Knox, Spark, Hive, Tez, Pig, Sqoop, Flink, Druid (analytický real-time DBMS), HBase, Phoenix, ZooKeeper, Presto, Storm, Impala, Hue, Oozie, Zeppelin, Ranger, Ganglia, TensorFlow, Kafka, Superset (BI aplikace poskytující grafické rozhraní pro analýzu a tvorbu dashboardů a vizualizací), Jupyter a Analytics Zoo (framework poskytující rozhraní pro integraci Spark, Tensorflow a dalších frameworků prostřednictvím tzv. pipelines). Z uvedeného výčtu je patrné, že oproti ostatním cloudovým distribucím poskytuje Alibaba E-MapReduce širší množinu projektů, včetně nových projektů z oblasti ML, DL a případně AI, jako jsou Tensorflow, Analytics Zoo, aj. Také poskytuje prakticky všechny notebooky pro interaktivní analýzu. Stejně jako ostatní cloudové distribuce zahrnuje konektory a další typy nástrojů podporující snadnou integraci s ostatními službami cloudu dané společnosti, zatímco daná služba maximálně abstrahuje uživatele od nízko úrovňových činností nasazování a správy clusteru, apod.

V roce 2011 byla na trhu ještě k dispozici distribuce (Kobielus, a další, 2012, str. 5) DataStax Brisk. Jednalo se v podstatě o (DataStax, 2013) prototyp integrace Hadoop a NoSQL Cassandra, který nicméně nebyl od té doby aktualizován ani dále vyvíjen. Na tuto distribuci nicméně společnost DataStax navázala produktem (DataStax, 2019) DataStaxEnterprise Hadoop, který substituoval HDFS právě databázovým systémem Cassandra, a dále nad rámec Hadoop poskytoval komponenty Hive, Pig a Mahout. Ani tato distribuce nicméně nebyla dále výrazně rozvíjena a v současné době již není poskytována.

Další distribuci frameworku Apache Hadoop na trh (EMC, 2013) v únoru 2013 uvedla společnost Pivotal (součást EMC a později Dell). Tato distribuce frameworku Apache Hadoop nesla název (EMC, 2013) Pivotal HD a v době vydání představovala významný produkt big data portfolia společnosti doplňující především MPP databázový systém EMC Greenplum a další. Hlavní přednost distribuce Pivotal HD představoval (EMC, 2013) pokročilý dotazovací a analytický engine pro Hadoop nesoucí název HAWQ, který nad Hadoop poskytoval možnost nativního výkonného SQL dotazování (dnes open source zastřešovaný ze strany ASF).

Ještě v roce 2015 byla distribuce Pivotal HD na trhu k dispozici (Pivotal, 2015) v nové verzi, která byla v souladu se specifikací běhového prostředí ODPi, byla založena na Apache Hadoop 2.6 a Ambari a zahrnovala komponenty Hive, HBase, ZooKeeper, Oozie (workflow plánovač pro Hadoop úlohy), Spark, Tez, Ranger (framework pro správu a monitoring bezpečnosti v clusteru Hadoop), Knox (aplikáční brána pro zajištění jednotné interakce s API a grafickými rozhraními v rámci clusteru Hadoop) a také komponenty pro monitoring nad rámec Ambari, konkrétně Nagios a Ganglia.

V roce 2016 nicméně společnost Pivotal (Panettieri, 2016) uzavřela partnerství se společností Hortonworks, ukončila vývoj a vydávání vlastní distribuce Pivotal HD a v rámci big data produktů a řešení začala využívat právě distribuci Hortonworks.

Rovněž v únoru 2013 byla na trh uvedena (Deutscher, 2013) distribuce frameworku Apache Hadoop společnosti WANdisco nesoucí název WANdisco Distro. Vydání této

distribuce následovalo poté, co společnost WANdisco realizovala (Deutscher, 2013) akvizici společnosti AltoStor, která se na Hadoop zaměřovala. Společnost WANdisco do distribuce zařadila (Deutscher, 2013) vlastní technologii pro replikaci. Tato distribuce byla k dispozici ke stažení zdarma. Ještě v dubnu vyšla (WANdisco, 2013) nová verze 3.1.1 založená na Hadoop 2.0.3, kompatibilní a úspěšně testovaná vůči Bigtop a podporující kromě OS platform RedHat a CentOS také SUSE. Toto byla nicméně poslední verze dané distribuce, jejíž doba existence byla ze všech uvedených distribucí nejkratší.

V roce 2013 se na trhu objevila další nová distribuce frameworku Apache Hadoop, kterou vyvinula společnost Intel a vydala pod názvem (Intel, 2013) Intel Distribution for Apache Hadoop (IDH). Společnost Intel tuto distribuci vydala s cílem (Intel, 2013) zajistit optimalizaci pro zpracování big data realizované nad hardwarovými platformami a/nebo produkty využívajícími procesorové čipy Intel, a to prostřednictvím specializované instrukční sady umožňující dosahovat vyššího výkonu při (de)sifrování. Součástí distribuce IDH byla (Intel, 2013) komponenta správce Intel Manager for Hadoop poskytující funkcionality pro nasazování, správu a monitoring Hadoop clusteru a také komponenta Intel Active Tuner for Apache Hadoop zajišťující automatizované ladění výkonu.

Již v roce 2014 nicméně společnost Intel vývoj a vydávání vlastní distribuce (Bell, 2014) zastavila, když uzavřela partnerství se společností Cloudera a začala využívat její distribuci CDH a Cloudera Enterprise. Součástí dohody mezi Intel a Cloudera přitom byla také (Bell, 2014) investice ze strany Intel, za kterou získala ve společnosti Cloudera určitý podíl. Do distribuce Cloudera byly integrovány optimalizace pro čipy Intel (Bell, 2014) a přechod zákazníků na tyto distribuce byl zahájen po vydání poslední verze IDH 3.1 na konci března roku 2014.

V říjnu 2013 představila (Microsoft, 2013) vlastní distribuci Hadoop také společnost Microsoft. Jedná se o distribuci (Microsoft, 2013) HDInsight, která je poskytována v rámci služby stejného názvu v cloudu Microsoft Azure, a jde tedy o proprietární cloudovou distribuci. Společnost Microsoft původně chtěla vytvořit vlastní distribuci (Foley, 2013) zcela od základu v podobě tzv. HDInsight Server for Windows, ale později od tohoto záměru upustila a namísto toho vytvořila vlastní distribuci společně s Hortonworks. Distribuce HDInsight je tudíž založena na HDP. Pro tuto distribuci je charakteristické, že (Foley, 2013) kromě HDP poskytuje nástroje (konektory, aj.) pro integraci dalších produktů Microsoft a/nebo služeb Azure (podobně jako v případě Amazon a služby EMR), jako například Excel, SQL Server, PowerBI, atd.

V současné době jsou podporovány verze (Microsoft, 2019b) HDInsight 3.6 (vydáno 04/17, založeno na HDP 2.6, aktuálně defaultní) a 4.0 (vydáno 9/18, založeno na HDP 3.0), přičemž distribuce zahrnuje HDP a projekty YARN, Tez, Pig, Hive, Ranger, HBase, Sqoop, Oozie, ZooKeeper, Storm (není součástí HDInsight 4.0), Mahout (není součástí HDInsight 4.0), Phoenix, Spark, Livy, Kafka, Ambari, Zeppelin, Mono (open source .NET framework) a Slider (framework pro Hadoop aplikace dlouhodobě běžící nad YARN; není součástí HDInsight 4.0). V rámci služby HDInsight přitom uživatelé mohou volit z typů clusteru, mezi které spadá (Microsoft, 2019a) Hadoop, HBase, Interactive Query, Kafka,

ML Services, Spark, Storm. Kromě HDInsight je také možné v rámci Azure Marketplace objednat klasickou verzi HDP, která je nasazena nad virtuálními stroji v cloudu Azure.

V roce 2015 představila společnost Huawei vlastní distribuci Apache Hadoop, a to pod názvem (Huawei, 2019) FusionInsight. Tato distribuce původně zahrnovala pouze produkt (Huawei, 2019) FusionInsight HD, reprezentující tradiční distribuci frameworku Apache Hadoop, ale později byla rozšířena o tzv. FusionInsight LibrA, tj. vlastní datový sklad pro big data a také o vlastní MPP databázový systém GaussDB. V současné době je k dispozici verze 6.5.0 zahrnující (Huawei, 2019) Hadoop, Spark, Elk (framework pro analýzu logů tvořený nástroji Elasticsearch, Logstash, Kibana), Flink, Storm, Solr, Kafka, Loader (vlastní systém Huawei pro přesun dat mezi FusionInsight a RDBMS a FS), HBase, Flume, GaussDB a FusionInsight LibrA. Přestože je ke stažení zdarma, nejedná se o open source, ale proprietární produkt, přičemž pro využití v produkčním prostředí je vyžadováno zakoupení licence.

Přestože byla již určitou dobu k dispozici v neprodukční verzi, plně produkční verzi vlastní cloudové distribuce uvolnila společnost Google až (Google, 2019b) v roce 2016, a to pod názvem Cloud Dataproc. Nová verze je vydávána přibližně (Google, 2019b) jedenkrát za rok, přičemž poslední dostupnou verzí je v současné době Cloud Dataproc 1.4 zahrnující kromě (Google, 2019a) Hadoop 2.9.2 také Spark, Hive, Pig, Tez, Anaconda (platforma pro Python/R data science a ML), Druid, Jupyter, Kerberos, Presto, Zeppelin a ZooKeeper. Zatímco v případě Alibaba Cloud je distribuce E-MapReduce nasazována (Alibaba, 2019) primárně nad CentOS, v Google Cloud je Cloud Dataproc provozována (Google, 2019b) nad systémy Debian a Ubuntu.

Na trhu je také distribuce (Syncfusion, 2019) Syncfusion Big Data Platform, a to od roku 2016. Tato distribuce společnosti Syncfusion je charakteristická tím, že je (Syncfusion, 2019) navržena pro Windows, Linux a Azure, tedy pro vývoj v rámci platformy Windows a nasazení na platformách Windows (Windows 7, Windows Server 2008 a pozdější, Azure – nikoli v rámci služby HDInsight, ale nad službou virtuálních serverů) nebo Linux (Ubuntu, CentOS). Pro nasazení a správu clusterů je k dispozici (Syncfusion, 2019) grafické rozhraní Syncfusion Big Data cluster Manager, pro správu úloh a práci s frameworky pro zpracování big data pak Syncfusion Big Data Studio. Daná distribuce podporuje (Syncfusion, 2019) nasazení agentů na jednotlivých uzlech clusteru automatizovaně s využitím PowerShell /Active Directory. V současné době je k dispozici (Syncfusion, 2019) verze 3.2.0.20 ze srpna 2017 obsahující kromě Hadoop komponenty Spark, HBase, Oozie, Sqoop, Kerberos, Hive, Pig a další. Uživatelé mohou využít trial verze, ale pro produkční využití je nutné zakoupení licence. Tato distribuce se tak od ostatních liší především zaměřením na platformu Windows a podporou možnosti vytvářet a nasazovat úlohy zpracování dat v programovacím jazyce C#.

Především na (Arenadata, 2019) ruský a/nebo ruskojazyčný trh je zaměřena distribuce, kterou v roce 2017 vydala společnost Arenadata, tj. Arenadata Hadoop. Poslední dostupná verze 1.6.1 obsahuje kromě Hadoop projekty (Arenadata, 2019) Ambari, ZooKeeper, Oozie, NiFi (systém pro zpracování a distribuci rozsáhlých souborů dat), Flink, Sqoop, Flume, Kafka, Atlas (data governance framework pro Hadoop), Knox, Ranger, Hive, Tez, HBase,

Phoenix, Spark, Pig, Mahout, Giraph a Solr. Arenadata Hadoop nepředstavuje open source distribuci, jelikož (Arenadata, 2019) pro stažení balíku je nutné pořídit licenci.

Po vzniku projektu Hadoop představujícího tzv. Vanilla Hadoop distribuci lze sledovat rozvoj trhu distribucí frameworku Apache Hadoop v podstatě ve třech vlnách. V rámci první vlny (2006-2009) se na trhu objevily první významné distribuce frameworku Apache Hadoop (CDH, MDP) a také první cloudová distribuce (EMR). V rámci druhé vlny rozvoje trhu distribucí frameworku Apache Hadoop (2010-2013) došlo nejprve k rapidní saturaci trhu, kdy vzniklo a na trh bylo uvedeno množství nových distribucí Hadoop a posléze k určité konsolidaci daného trhu, kdy někteří poskytovatelé vlastní distribuce frameworku Apache Hadoop upustili od dalšího rozvoje a produkt distribuce frameworku Apache Hadoop přestali poskytovat a/nebo vlastní distribuci substituovali některou z hlavních distribucí CDH, MDP nebo HDP. Tato konsolidace dále pokračovala i v rámci třetí vlny rozvoje trhu distribucí Hadoop (2014-2018/nyní), a to především v důsledku etablování organizace ODPI, která vznikla v roce 2014, a pod jejíž záštitou někteří poskytovatelé distribucí frameworku Apache Hadoop navázali a/nebo prohloubili vzájemnou spolupráci a případně provedli substituci vlastního řešení distribuce Hadoop. V předchozím textu tak byly v rámci vývoje v letech 2006 až 2018 uvedeny a popsány všechny relevantní distribuce Hadoop, které se v minulosti na trhu objevily.¹⁹

Z výše uvedené Tab. 4.4 je tedy patrné, že na trhu je v současné době dostupných 12 distribucí frameworku Apache Hadoop, z nichž pouze 4 (Vanilla, CDH, HDP, Bigtop) vykazují charakter open source, a to v následujícím složení:

- základní distribuce – Vanilla Hadoop,
- klíčoví hráči trhu distribucí – CDH, EMR, HDP,
- cloudové distribuce – EMR, E-MapReduce, HDInsight, Cloud Dataproc,
- ostatní distribuce – Bigtop, FusionInsight, Syncfusion Big Data Platform, Arenadata Hadoop.

Jestliže v případě některých z uvedených cloudových distribucí, respektive služeb HaaS. v jejichž rámci jsou poskytovány, bylo možné dříve nasadit některou z uvedených klíčových distribucí, v současné době to již možné není. Poskytovatelé cloudových distribucí umožňují v rámci příslušné služby typu HaaS nasazovat pouze vlastní distribuci a jiné (zpravidla klíčové) distribuce je případně možné v rámci daného cloudu zprovoznit nad jinými službami typu virtuálních strojů, aj. (tzv. Appliance distribuce Hadoop). Pro cloudové distribuce, respektive odpovídající služby typu HaaS je přitom charakteristické, že umožňují uživatelům volit z řady předpřipravených typů Hadoop clusteru vhodných pro

¹⁹⁾ V literatuře jsou uváděny ještě další distribuce, mezi které patří (Gualtieri, a další, 2014, str. 6) Teradata Open Distribution for Hadoop a také (Kobielus, a další, 2012, str. 5) Zettaset Data Platform, a případně další (Sea Box Big Data Platform, Transwarp Hadoop, Infochimps Platform, Altiscale Data Cloud – HaaS, později akvirováno ze strany společnosti SAP). Protože v současné době není možné v literatuře ani na internetu nalézt dostatek informací k uvedeným distribucím frameworku Apache Hadoop, nejsou v textu této práce uváděny.

různé případy použití a současně, že uživatele do značné míry abstrahují od činností nasazování a správy Hadoop clusterů.

V obecné rovině lze konstatovat, že v rámci distribucí frameworku Apache Hadoop roste význam technologií pro zpracování dat v (témař) reálném čase (Spark, Storm, Flink, Beam, aj.), na což reagují především (kromě jiných) poskytovatelé cloudových distribucí, kteří kromě distribucí frameworku Apache Hadoop a odpovídajících služeb typu HaaS, poskytují specializované služby zaměřené na zpracování proudů dat, apod.

Tab. 4.5 Srovnání komponent tvořících cloudové distribuce frameworku Hadoop

Komponenta	EMR 5.23.0	E-MapReduce	HDInsight 4.0	Cloud Dataproc 1.4.4
Ambari	-	-	2.7.0	-
Anaconda	-	-	-	5.2.0
Analytics Zoo	-	0.2.0	-	-
Druid	-	0.13.0	-	0.13.0
Flink	1.7.1	1.7.2	-	-
Ganglia	3.7.2	3.7.2	-	-
Hadoop	2.8.5	2.8.5	3.1.1	2.9.2
HBase	1.4.9	1.4.9	2.0.1	-
Hive	2.3.4	3.1.1	3.1.0	2.3.4
Hue	4.3.0	4.1.0	-	-
Impala	-	2.12.2	-	-
Jupyter	-	4.4.0	-	4.4.0
Kafka	-	2.11	1.1.1	-
Kerberos	-	-	-	1.15.1
Knox	-	1.1.0	-	-
Livy	0.5.0	-	0.5.0	-
Mahout	0.13.0	-	-	-
MXNet	1.3.1	-	-	-
Oozie	5.1.0	5.1.0	4.3.1	-
Phoenix	4.14.1	4.14.1	5	-
Pig	0.17.0	0.14.0	0.16.0	0.17.0
Presto	0.215	0.213	-	0.215
Ranger	-	1.2.0	1.1.0	-
Spark	2.4.0	2.4.2	2.3.2	2.4.2
Sqoop	1.4.7	1.4.7	1.4.7	-
Storm	-	1.2.2	-	-

Superset	-	0.28.1	-	-
Tensorflow	1.12.0	1.8.0	-	-
Tez	0.9.1	0.9.1	0.9.1	0.90
Zeppelin	0.8.1	0.8.1	0.8.0	0.8.0
ZooKeeper	3.4.13	3.4.13	3.4.6	3.4.13

Zdroj: data (Microsoft, 2019b; Amazon, 2019a; Alibaba, 2019; Google, 2019a), vlastní zpracování

Podobně roste i v rámci distribucí frameworku Apache Hadoop také význam systémů z oblasti ML, DL a AI, což reflektují také poskytovatelé cloudových distribucí, jak je zřejmě mimo jiné z Tab. 4.5.

Podrobný popis cloudových distribucí a příslušných služeb typu HaaS přesahuje rámec této práce a současně neodpovídá stanovenému zaměření. Vzhledem k zaměření této práce, které zahrnuje realizaci typové úlohy zpracování big data v (témař) reálném čase prostřednictvím distribuce frameworku Apache Hadoop nasazované vlastními silami a také s ohledem na skutečnost, že majoritní podíl na trhu tradičních (původně non cloudových) distribucí frameworku Apache Hadoop patří výše uvedeným klíčovým hráčům trhu distribucí Apache Hadoop, následuje dále v rámci této kapitoly podrobný popis a srovnání pouze klíčových distribucí frameworku Apache Hadoop, tj. CDH, MDP a HDP.

4.2 Cloudera's Distribution Including Apache Hadoop

4.2.1 Vznik a rozvoj distribuce

V roce 2008 založili (Bonaci, 2015) Mike Olson ze společnosti Sleepycat Software, Christophe Bisciglia z Google, Jeff Hamerbacher z Facebook a Amr Awadallah z Yahoo! společnost Cloudera. Ve stejném roce (Singh, a další, 2019, str. 19) tato společnost vytvořila (po Hadoop Vanilla) první verzi distribuce frameworku Apache Hadoop. Tato distribuce nesla název (Singh, a další, 2019, str. 19) Cloudera's Distribution Including Apache Hadoop (CDH). První produkční verze CDH byla na trh uvedena (Cloudera, 2009) v březnu roku 2009. Již tato verze distribuce CDH obsahovala (Cloudera, 2009) nástroje pro instalaci a správu Hadoop clusteru a kromě Hadoop dále projekty Hive a Pig.

V srpnu roku 2009 (Bonaci, 2015) přešel do společnosti Cloudera spoluzakladatel projektu Hadoop Doug Cutting a začal zde působit jako hlavní architekt. Na konci června 2010 byla vydána zatím neprodukční (Cloudera, 2010) majoritní verze CDH 3, a to poprvé ve dvou edicích, tradiční open source edici (v současné době označované jako Cloudera Express) a Cloudera Enterprise edici. Edice CDH označovaná jako Cloudera Enterprise nad rámec Cloudera Express zahrnovala (Cloudera, 2010) pokročilé funkce pro nasazení, správu a monitoring Hadoop clusteru a také další projekty cílené na zajištění tzv. enterprise funkcionality (správa a řízení přístupu uživatelů k datům a funkcím v Hadoop clusteru) a rovněž komerční podporu pro uživatele, kteří si licenci pro Cloudera Enterprise formou předplatného zakoupili.

Produkční verze CDH 3 ve zmíněných edicích Cloudera Express a Cloudera Enterprise, byla na trh uvedena (Cloudera, 2011c) v dubnu 2011. Distribuce CDH 3 přinesla (Cloudera, 2011c) podporu 32 i 64 bitové platformy Java napříč OS, mezi které patřily RHEL, CentOS, SUSE a Ubuntu a zahrnovala kromě Hadoop projekty HBase, Hive, Pig, Sqoop, Flume, Hue, Oozie a ZooKeeper. V červnu téhož roku byla vydána (Cloudera, 2011b) minoritní verze CDH 3.5, která poprvé přinesla komponentu a grafické rozhraní pro komplexní nasazení, provoz, správu a monitoring clusteru Hadoop/CDH. Tato komponenta nesla označení (Cloudera, 2011b) Service and Configuration Manager (SCM, dnes Cloudera Manager). Již v době prvního vydání byla množina dostupných funkcí této komponenty determinována (Cloudera, 2011b) využívanou edicí, kdy Cloudera Express umožňovala využít pouze omezenou množinu a/nebo úroveň funkcí komponenty a grafického rozhraní pro nasazení, správu a monitoring Hadoop clusteru s distribucí CDH. Ještě v prosinci 2011 byla vydána (Cloudera, 2011a) verze CDH 3.7, v jejímž rámci byl zmíněný systém dostupný již pod názvem Cloudera Manager.

CDH 4 byla v beta verzi představena (Cloudera, 2012b) v dubnu roku 2012 a mezi novinky, které uživatelům přinesla, patřily především možnost zajištění vysoké dostupnosti jmenného uzlu HDFS, nové funkce HBase pro (near) real-time aplikace, výkonnostní zlepšení komponent Hadoop, HDFS, Flume, aj., Hadoop 2 (tj. především YARN a MapReduce 2 namísto MapReduce 1), nové funkce Cloudera Manager, atd. Množina projektů CDH 4 se od CDH 3.x (Cloudera, 2012b) příliš nelišila (kromě zařazení Avro) a nová majoritní verze distribuce CDH tak přinesla především nové funkce a vlastnosti a také defaultní podporu novějších verzí výše zmíněných podporovaných OS. Produkční verze CDH 4 byla na trh uvedena (Cloudera, 2012c) v červnu 2012. V říjnu 2012 představila společnost Cloudera první beta verzi Impala, významného projektu (Cloudera, 2012a) analytického DBMS pro Hadoop podporujícího BI dotazy nad big data, který se stal součástí CDH a dále ji diferencioval od ostatních distribucí.

V únoru roku 2013 byla vydána (Cloudera, 2013a) minoritní verze CDH 4.5, která v enterprise edici přinesla nové funkce pro zálohování a obnovu z výpadku (BDR). Současně Cloudera vydala nový produkt, který bylo (a stále je) možné používat (Cloudera, 2013a) s příslušnou licencí zakoupenou formou předplatného, a to Cloudera Navigator 1.0 poskytující funkionalitu pro zabezpečení, správu a průzkum rozsáhlých objemů dat uchovávaných v Hadoop clusterech s CDH. CDH 4.5 přinesla také (Cloudera, 2013a) novou verzi Cloudera Manager, mezi jejíž nejvýznamnější nové funkcionality patřila možnost realizace aktualizace CDH bez výpadku (rolling upgrades) a efektivnější vizualizace provozních metrik clusterů CDH. Na konci dubna 2013 společnost Cloudera vydala (Cloudera, 2013f) první produkční verzi Impala 1.0, představující nativní Hadoop MPP DBMS umožňující dotazovat prostřednictvím SQL big data přímo v rámci HDFS a HBase. Pro předplatitele edice Cloudera Enterprise (Cloudera, 2013f) byla k dispozici funkciaonalita relat-time query postavená právě nad Impala.

V květnu 2013 společnost Cloudera vydala pro CDH (Cloudera, 2013b) Cloudera Developer Kit, poskytující vývojářům nástroje pro usnadnění a zefektivnění vývoje aplikací nasazovaných v rámci CDH clusteru. Ještě v prosinci 2013 byl tento software (Sammer, a další, 2013) přejmenován na Kite SDK a poslední vydání (1.1.0) pochází z června 2015, takže vývoj již není aktivní, a proto tento software ani neuvádíme v přehledu milníků

vývoje CDH, který obsahuje Příloha A:. V červnu 2013 vydala společnost Cloudera další projekt dostupný v rámci edice CDH Cloudera Enterprise, a to (Cloudera, 2013c) Cloudera Search, vyhledávací engine pro interaktivní exploraci dat uložených v rámci HDFS a HBase, založený na Solr a integrovaný s CDH, díky kterému mohou uživatelé efektivně indexovat a prohledávat rozsáhlé objemy dat uchovávané v Hadoop clusteru.

V červenci téhož roku byl představený významný projekt z oblasti (Cloudera, 2013g) bezpečnosti dat uchovávaných v Hadoop clusterech CDH, konkrétně komponenta Sentry poskytující pro Hive a Impala funkcionality řízení přístupu k datům formou řízení přístupu založeném na rolích (RBAC). V září byla vydána (Cloudera, 2013e) produkční verze Cloudera Search. Zatím neprodukční nová majoritní verze CDH 5.x byla představena (Cloudera, 2013d) v říjnu 2013. Tato verze (Cloudera, 2013d) integrovala všechny nové projekty s množinou projektů z předchozí verze do jednoho koherentního celku a také poskytla určitá vylepšení v rámci vybraných projektů.

V únoru 2014 představila společnost Cloudera (Cloudera, 2014a) novou cenovou politiku poskytování komerční distribuce CDH, když uvedla tři edice Cloudera Enterprise, a to Cloudera Enterprise Data Hub Edition, Cloudera Enterprise Flex a Cloudera Enterprise Basic. Současně společnost Cloudera oznámila (Cloudera, 2014a) zahájení poskytování komerční podpory pro Spark. V dubnu 2015 byla na trh uvedena (Cloudera, 2014b) produkční verze distribuce CDH 5. Již v říjnu stejného roku byla vydána (Cloudera, 2014c) minoritní verze CDH 5.2, která přinesla především možnost správy bezpečnostních politik Sentry v rámci Hue (pro Impala, Hive a Cloudera Search), funkcionality z oblasti správy bezpečnostních klíčů, a další v rámci Cloudera Navigator a ještě jiná vylepšení pro Cloudera Search, Spark a HBase. Současně byla vydána (Cloudera, 2014c) nová verze nativní analytické databáze pro Hadoop, Cloudera Impala, která byla rovněž součástí distribuce CDH. Cloudera také společně s CDH 5.2 a Impala 2.0 vydala (Cloudera, 2014c) první verzi Cloudera Director, rozhraní pro samoobslužný provisioning clusterů CDH v rámci cloudového a/nebo hybridního prostředí, přičemž jako první bylo podporováno nasazení clusterů CDH v rámci cloudové platformy AWS.

V únoru 2015 oznámila společnost Cloudera (Cloudera, 2015e) dokončení integrace distribuované platformy pro zpracování proudů dat Kafka v rámci distribuce CDH. Nová verze Cloudera Director 1.5 byla vydána (Cloudera, 2015a) v srpnu 2015 a přinesla především kromě stávající podpory nasazení clusterů CDH v rámci cloudové platformy AWS také podporu nasazení v rámci cludu GCP. V září byla veřejnosti zpřístupněna (Cloudera, 2015b) možnost využít cluster CDH ve všech edicích v rámci cludu Microsoft Azure prostřednictvím tržiště Microsoft Azure Marketplace, a to s podporou efektivní integrace CDH s dalšími službami Azure (např. SQL Server, PowerBI, aj.). Ještě v září 2015 společnost Cloudera vydala (Cloudera, 2015c) beta verzi nového DBMS pro Hadoop nesoucího název Kudu a podporujícího analytické dotazy v rámci Hadoop clusteru ze strany Impala, Spark a dalších v reálném čase. V listopadu 2015 byla vydána (Cloudera, 2015d) verze CDH 5.5, jež kromě jiného přinesla především beta verzi Cloudera Navigator Optimizer, tj. modulu pro optimalizaci úloh zpracování a analýzy dat v rámci clusterů Hadoop s CDH.

Na začátku roku 2016 představila společnost Cloudera (Cloudera, 2016a) beta verzi grafického rozhraní pro správu Hadoop úloh a aplikací, Cloudera Desktop, zahrnujícího správce souborů, průvodce tvorbou úloh, správce úloh, a monitoring clusteru. Toto rozhraní nicméně v současné době není dostupné, ani není dále rozvíjeno, a proto ani není uvedeno v přehledu milníků vývoje distribuce CDH, který obsahuje Příloha A:. V lednu byla vydána (Cloudera, 2016b) nová verze Cloudera Director zahrnující nové a/nebo vylepšené funkce pro nasazení a provoz clusterů CDH v rámci cloudových platform AWS a GCP. V dubnu 2016 byla vydána (Cloudera, 2016e) minoritní verze CDH 5.7, přinášející především optimalizace v použití Impala a také Hive nad Spark. Další minoritní verze, CDH 5.8 byla vydána (Cloudera, 2016d) v červenci téhož roku. V září 2016 došlo k vydání (Cloudera, 2016c) první produkční verze úložiště Kudu umožňujícího realizovat v rámci clusterů CDH dotazy v reálném čase.

V lednu 2017 vydala společnost Cloudera (Cloudera, 2017a) další minoritní verzi CDH, a to 5.10, v jejímž rámci byla poprvé zahrnuta produkční verze Kudu. V březnu téhož roku představila společnost Cloudera (Cloudera, 2017g) pod názvem Cloudera Data Science Workbench nové rozhraní poskytující funkcionalitu pro samoobslužné využití nástrojů a funkcí data science a ML nad clustery CDH, přičemž produkční verze byla vydána (Cloudera, 2017b) již v květnu. Rovněž v květnu spustila společnost Cloudera (Cloudera, 2017e) vlastní službu dedikovaného spravovaného clusteru CDH v rámci cloutu, a to pod názvem Cloudera Altus. Tato služba zajistila společnosti Cloudera (Cloudera, 2017e) vstup na trh poskytovatelů cloudově nativní služby HaaS, přičemž v době vzniku dané služby byly clustery CDH v rámci služby Cloudera Altus nasazovány nad infrastrukturou cloutu AWS. V září 2017 společnost uvedla další doplňkovou oblast funkcionality dostupnou v rámci Cloudera Enterprise, konkrétně modul (Cloudera, 2017f) Shared Data Experience poskytující funkcionalitu jednotného datového katalogu, zabezpečení, správy a řízení životního cyklu dat, umožňující efektivnějším způsobem mezi týmy, projekty, clustery a dalšími úrovněmi sdílet data, která jsou v rámci CDH předmětem zpracování, a to při zajištění odpovídající úrovně zabezpečení a transparentnosti využití. V září 2017 společnost oznámila (Cloudera, 2017d) chystanou beta verzi Cloudera Altus v rámci Microsoft Azure, poskytující v cloutu možnost integrace CDH s produkty Microsoft SQL Server, Power BI nebo např. Azure Data Lake Store. Před koncem roku, v listopadu 2017, ještě společnost oznámila (Cloudera, 2017c) vydání beta verze Cloudera Altus Analytic DB, tj. vlastního řešení cloudového datového skladu (v rámci této fáze nad AWS, a mimo jiné tedy také s využitím objektového úložiště S3) umožňujícího využít Impala pro rychlé analytické dotazy nad rozsáhlými soubory dat a také zmíněný modul Cloudera Shared Data Experience.

V březnu roku 2018 oznámila společnost Cloudera (Cloudera, 2018f) plánované rozšíření funkcionality Cloudera Altus o oblast ML prostřednictvím modulu Data Science, nicméně podle dostupných informací k tomu zatím nedošlo. V květnu 2018 byla vydána (Cloudera, 2018a) aktualizovaná beta verze Cloudera Data Science Workbench, Cloudera Altus v rámci Microsoft Azure byl uvolněn do produkčního provozu a současně byla uvedena beta nové majoritní verze distribuce CDH 6.0, která přinesla řádově vyšší efektivitu zpracování big data prostřednictvím integrace nových verzí klíčových komponent, jako například Hadoop, Hive, HBase, Oozie, Solr, Kafka či Spark a také další dílčí inovace a zlepšení. V srpnu byla do produkčního provozu uvolněna (Cloudera, 2018g) služba,

respektive oblast funkcionality Cloudera Altus Data Warehouse, dříve označovaná jako Cloudera Altus Analytic DB, a v tomto okamžiku byla dostupná nejen v rámci AWS, ale také v Microsoft azure. V září 2018 byla (Cloudera, 2018d) distribuce CDH 6.0 uvolněna jako produkční, přičemž současně bylo vydáno další rozšíření pod názvem Cloudera Workload Experience Manager, poskytující funkcionality a nástroje pro optimalizaci úloh a dotazů v rámci clusterů CDH a oblastí DW, BDA i ML. Jak již bylo uvedeno, 3. října 2018 oznámila společnost Cloudera na základě akvizice (Cloudera, 2018b) plánované sloučení se společností Hortonworks poskytující distribuci HDP. V prosinci 2018 společnost Cloudera ještě představila (Cloudera, 2018c) preview verzi cloudové ML platformy nasazované nad clustery Kubernetes a poskytující mimo jiné nasazování ML algoritmů nad více clustery (federated ML) a také využití GPU pro zajištění maximálního výkonu, přičemž vydání tohoto produktu bylo plánováno na rok 2019.

Jak již bylo uvedeno, 3. ledna roku 2019 bylo (Cloudera, 2019c) dokončeno sloučení se společností Hortonworks na základě akvizice ze strany Cloudera. Společnost Cloudera tak fakticky získala na trhu distribucí frameworku Apache Hadoop bezprecedentní většinu. V důsledku tohoto vývoje společnost Cloudera v současné době nabízí (Cloudera, 2019d) jak vlastní distribuci Hadoop, tj. CDH, tak distribuci, kterou původně vytvořila společnost Hortonworks, tedy HDP. S využitím (Cloudera, 2019d) platformy pro škálovatelné zpracování proudů dat v reálném čase, kterou původně vytvořila společnost Hortonworks, a jež nese název Hortonworks DataFlow (HDF), rovněž Cloudera nabízí vlastní derivát této platformy, který nese označení Cloudera DataFlow (CDF). Řešení CDF přitom společnost Cloudera dále aktivně rozvíjí (Cloudera, 2019i) prostřednictvím integrace funkcionality původní HDF s funkcionality Cloudera Manager, Impala nebo např. Kudu. Dále lze v nabídce společnosti Cloudera nalézt dříve zmíněné produkty, mezi něž patří především (Cloudera, 2019d) Cloudera Data Science Workbench (systém a grafické rozhraní pro samoobslužnou data science), Cloudera Navigator (software pro šifrování a správu zabezpečení v clusterech CDH řady Enterprise), databázové ovladače ODBC a JDBC pro Impala a Hive a také konektory pro Teradata a Netezza. Zatímco pro nasazení, správu a monitoring CDH v on-premise prostředí je využíváno především rozhraní (Cloudera, 2019d) Cloudera Manager, pro nasazení v rámci cloutu je k dispozici Altus Director, který je součástí širší množiny řešení Cloudera Altus pro využití distribuce CDH a dalších produktů Cloudera v cloudovém a/nebo hybridním prostředí, a to včetně veřejných cloudů GCP, AWS, Azure, IBM i Oracle. Na konci března roku 2019 byla vydána (Cloudera, 2019e) minoritní verze distribuce CDH 6.2.0.

Krátkce po dokončení sloučení se společností Hortonworks výkonný ředitel společnosti Cloudera Thomas Reilly popsal plány (Vaughan, 2019) na vydání jednotné distribuce kombinující to nejlepší z funkcionality distribucí HDP a CDH. Tato distribuce nese kódové označení (Vaughan, 2019) Unity a oficiální název by měl znít Cloudera Data Platform. Současně nicméně ředitel společnosti Cloudera potvrdil, že (Vaughan, 2019) jak HDP 3, tak i CDH 5 a 6 by měly být podporovány minimálně po dobu dalších tří let. Distribuce Cloudera Data Platform by přitom měla být (Vaughan, 2019) dostupná jak on-premise, tak v cloutu (AWS, Azure, GCP, IBM a Oracle) a umožňovat nasazení napříč více různými

cloudy (multi-cloud). Ani v květnu 2019 nicméně zatím tato plánovaná distribuce vydána nebyla.²⁰ Z distribuce HDP byly pouze v rámci nové majoritní verze (Hortonworks, 2019e) odstraněny projekty Flume, Falcon (framework pro řízení životního cyklu dat v clusterech Hadoop), Mahout a Slider. Současně byly například z Apache Ambari odstraněny některé dílčí funkce, například grafické rozhraní pro interakci s Hive (Hive View). Cloudera nadále poskytuje distribuce CDH i HDP a CDF i HDF současně a pokračuje v integraci produktů Hortonworks do vlastního portfolia a/nebo relevantních oblastí funkcionality v rámci vlastních produktů.

Zatím poslední verze distribuce CDH 6.3.3 byla vydána (Cloudera, 2020) v březnu roku 2020, přičemž se bohužel jednalo o první verzi CDH, jejíž repositáře nebyly veřejně dostupné a pro jejich využití bylo nezbytné mít od společnosti Cloudera zakoupenu příslušnou licenci.

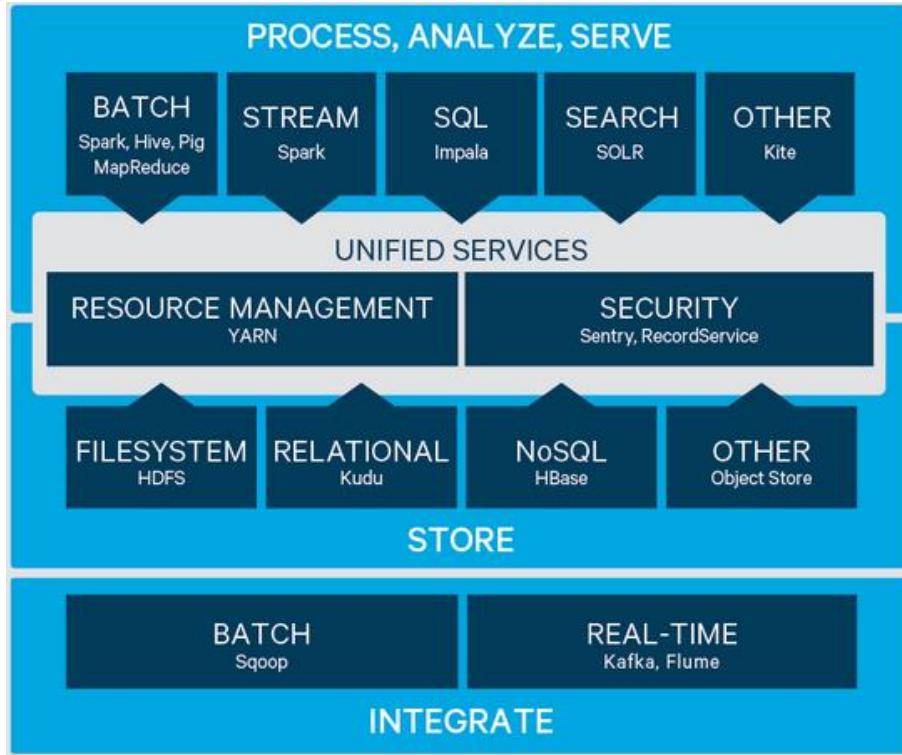
4.2.2 Architektura a funkcionalita distribuce

Jak bylo uvedeno, v březnu 2019 (Cloudera, 2019e) byla vydána distribuce CDH 6.2.0, která byla zvolena pro realizaci praktické úlohy zpracování big data v rámci této práce. Tato verze zahrnovala komponenty, mezi které patřily (Cloudera, 2019a) Avro, Flume, Hadoop, HBase, HBase Indexer (komponenta pro indexaci řádků HBase v rámci Solr), Hive, Hue, Impala, Kafka, Kite SDK, Kudu, Solr, Oozie, Parquet, Parquet-format, Pig, Sentry, Spark, Sqoop a ZooKeeper.

Základní architekturu distribuce tvořené uvedenými projekty znázorňuje Obr. 4-5, který usnadňuje orientaci v oblasti funkcionality jednotlivých komponent distribuce. Jádrem distribuce CDH je pochopitelně Hadoop poskytující především distribuovaný souborový systém HDFS a správce zdrojů pro realizaci úloh zpracování big data YARN a také objektové úložiště Hadoop Ozone. V rovině ukládání big data je součástí CDH NoSQL databázový systém HBase a pro distribuci CDH specifický relační DBMS Kudu umožňující v clusterech CDH provádět dotazy v reálném čase. Pro zajištění transferu a případně transformace big data směrem do a/nebo z clusteru CDH jsou k dispozici nástroje Sqoop, Kafka a Flume. Pokud jde o frameworky pro zpracování a analýzu big data, poskytuje distribuce CDH nástroje pro dávkové zpracování (Spark, Hive, Pig, MapReduce), zpracování proudů dat (Spark) a také specifické nástroje pro SQL dotazování (Impala) a vyhledávání (Solr) a rovněž vysokoúrovňové API pro práci s kolekcemi dat (Kite SDK). Ačkoli na Obr. 4-5 nejsou znázorněny, zahrnuje distribuce CDH také podpůrné systémy Oozie, ZooKeeper, systém pro řízení autorizace Sentry (funkcionalita komponenty RecordService uvedené na Obr. 4-5 je v současné době součástí Sentry), grafické rozhraní pro interaktivní analýzu dat Hue a také podporu big data formátů Avro a Parquet.

²⁰⁾ Společnost MapR se v reakci na akvizici Hortonworks ze strany Cloudera a plánované vydání distribuce Unity kombinující to nejlepší z HDP a CDH vůči tomuto vývoji marketingově vymezila (MapR, 2019d) vyhlášením programu Clarity, v jehož rámci pro uživatele distribucí frameworku Apache Hadoop deklarovala stabilitu řešení pokrývajícího všechny oblasti, na které se Cloudera zaměřuje (AI, ML, hybridní cloud, kontejnery, operativní analytika, IoT), a poskytovaného ze strany MapR již v době dokončení sloučení Cloudera a Hortonworks, se značným předstihem před plánovaným vydáním distribuce Unity.

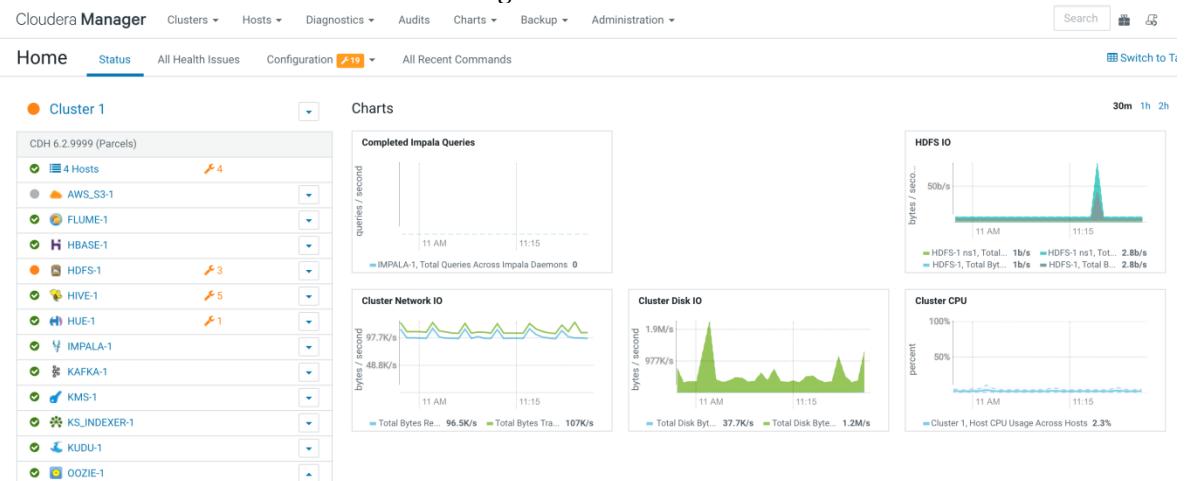
Obr. 4-5 Vysokoúrovňová architektura distribuce CDH



Zdroj: data a zpracování (Cloudera, 2019b)

V závislosti na využívané edici distribuce CDH (detaily týkající se jednotlivých edicí viz v rámci následující podkapitoly) mohou být součástí distribuce CDH ještě další systémy/komponenty. Jedná se především o komponentu pro nasazení, správu a monitoring clusterů CDH Cloudera Manager, jejíž (Cloudera, 2018e) základní funkci je součástí volně dostupné open source edice CDH Cloudera Express. Pokročilé funkce, například (Cloudera, 2018e) detailní správa a řízení uživatelů a přístupu, automatizované zabezpečení prostřednictvím šifrování, multi-tenantní řízení kvót využití zdrojů a/nebo funkcí, aktualizace bez výpadků (rolling updates), automatizovaná záloha a zajištění obnovy z výpadku, atd., jsou k dispozici pouze v rámci některé z edic CDH Cloudera Enterprise. Grafické rozhraní Cloudera Manager je znázorněno na Obr. 4-6.

Obr. 4-6 Grafické rozhraní Cloudera Manager



Zdroj: data a zpracování (Cloudera, 2019g)

Dále se jedná o komponentu Cloudera Altus Director poskytující funkcionality pro nasazení, správu a monitoring CDH v cloudu a/nebo v hybridním prostředí kombinujícím on-premise a cloud, která je k dispozici pouze (Cloudera, 2018e) v rámci Cloudera Enterprise. Mezi komponenty dostupné pouze v rámci některé z edic distribuce CDH Cloudera Enterprise, které je tudíž možné využít pouze v případě zakoupení příslušné licence, ještě patří (Cloudera, 2018e) komponenta Cloudera Navigator zajišťující funkcionality pro řízení a audit životního cyklu dat, šifrování a správu bezpečnostních klíčů nebo například optimalizaci úloh zpracování big data v rámci distribuce CDH.

V rámci procesu přípravy vydání distribuce CDH je využívána (Frampton, 2015, str. 22) již zmíněná distribuce Bigtop a další open source projekty které jsou součástí distribuce CDH. Takto získaná množina komponent projektu a ekosystému Hadoop a/nebo big data, následně (Frampton, 2015, str. 22) prochází integračními testy, jejichž úspěšnost zajišťuje, že lze danou množinu využívat jako jednotný integrovaný funkční a kompatibilní celek (stack). Kód a binární soubory jsou navíc (Frampton, 2015) reorganizovány do struktury služeb a balíčků odpovídajících distribuci CDH, což zajišťuje, že uživatelé naleznou relevantní soubory a logy služeb v rámci daného souborového systému v umístění typickém pro danou distribuci Hadoop. Tento přístup současně (Frampton, 2015) umožňuje s jednotlivými službami distribuce CDH operovat pomocí relevantních příkazů platformy Linux (`service`, `systemctl`, `apt`, aj.). Výstupem daného procesu je konkrétní vydání distribuce CDH obsahující koherentní množinu komponent, které lze společně efektivně využívat za účelem pokrytí celého procesu zpracování a analýzy big data.

Distribuce CDH kromě „standardních“ komponent, které jsou součástí ekosystému projektu Hadoop a/nebo big data, zahrnuje také specifické komponenty, mezi které patří především relační DBMS pro big data Kudu, nativní analytický DBMS pro Hadoop Impala nebo například vysokoúrovňové API a SDK pro práci s kolekcemi dat Kite. Přestože jsou uvedené komponenty k dispozici jako open source a zastřešovány ze strany ASF, jejich absence v rámci ostatních distribucí může v případě potřeby změny využívané distribuce frameworku Apache Hadoop determinovat náklady na zajištění možnosti využití určité funkcionality s využitím odlišných systémů.

4.2.3 Model poskytování distribuce

Společnost Cloudera poskytuje distribuci CDH (Cloudera, 2019h) ve dvou základních řadách, a to Cloudera Express a Cloudera Enterprise. Hlavní rozdíl v těchto edicích spočívá v (Cloudera, 2019h) rovině licencování, kdy Cloudera Express licenci nevyžaduje, zatímco pro Cloudera Enterprise je zakoupení licence nezbytné.

Placená a free verze CDH se dále liší z hlediska velikosti clusteru, pro který je lze využít, jelikož Cloudera Enterprise umožňuje (Cloudera, 2018e) nasazení na neomezeném počtu uzlů, zatímco v případě Cloudera Express je maximální velikost clusteru limitována na maximálně 100 uzlů. Dále se tyto řady CDH liší (Cloudera, 2018e) v rozsahu dostupné funkcionality (především pokročilé, jako je automatické šifrování přístupu, operativní reporting, multi-tenantní řízení kvót, reporting utilizace clusteru, automatizované zálohování a obnova z výpadku, atd.), množině dostupných projektů (placené edice mohou navíc zahrnovat projekty Impala, Kudu, Solr, HBase, Cloudera Navigator) a samozřejmě

také v úrovni poskytované podpory (žádná/komunitní oproti různým úrovním komerční dedikované podpory).

Jak Cloudera Express, tak i Cloudera Enterprise zahrnují (Cloudera, 2019h) CDH i Cloudera Manager, ale v případě placené verze CDH je k dispozici více především pokročilých funkcí Cloudera Manager. Uživatelé mají možnost (Cloudera, 2019h) využít zkušební (trial) verzi edic Cloudera Enterprise, a to po dobu 60 dní a po uplynutí této doby mohou dále používat edici Express nebo si zakoupit licenci pro některou edici Cloudera Enterprise. Využívanou Express verzi Cloudera je tedy kdykoli možné upgradovat na CDH Cloudera Enterprise.

V případě Enterprise verze distribuce CDH, je možné zakoupit různé úrovně licence odpovídající jednotlivým edicím Enterprise CDH (každá vyšší edice obsahuje další funkce a nástroje nad rámec předchozí edice), mezi které patří (Cloudera, 2019h):

- Essentials Edition – poskytuje lepší podporu a pokročilé funkce správy jádra Apache Hadoop,
- Data Science and Engineering Edition – zahrnuje navíc funkcionalitu a nástroje pro automatizovanou přípravu dat a prediktivní modelování,
- Operational Database Edition – obsahuje komponenty a funkce pro online aplikace zajišťující zpracování a/nebo analýzu dat v reálném čase,
- Enterprise Data Hub Edition – kompletní platforma zahrnující veškeré komponenty a funkce předchozích edic.

Všechny uvedené edice distribuce CDH je možné nasadit (Cloudera, 2019h) on-premise, v cloudu nebo v hybridním prostředí kombinujícím on-premise a cloud. V případě on-premise nasazení některé z Enterprise edic CDH jsou licence (Cloudera, 2019h) účtovány ročně za každý jednotlivý uzel. V cloudu umožňujícím flexibilní využití Hadoop clusteru s CDH pouze v požadovaném rozsahu a po nezbytně nutnou dobu, je licence zpravidla (Cloudera, 2019h) připočítána k ceně za jednotku času za každý jednotlivý uzel. Společnost Cloudera také neustále buduje síť partnerů, jejichž počet přesahuje 400, a s nimiž při vývoji a/nebo poskytování či podpoře distribuce CDH i dalších produktů spolupracuje.

4.3 MapR Data Platform

4.3.1 Vznik a rozvoj distribuce

Společnost MapR Technologies (dále jen MapR), která je producentem distribuce MapR Data Platform, založili (Erraissi, a další, 2017) v roce 2009 M. C. Srivas a John Schroeder. Zatímco M. C Srivas se před založením MapR (MapR, 2019c) podílel ve společnosti Google na vývoji distribuovaného souborového systému GFS, NoSQL databáze BigTable a MapReduce, John Schroeder (MapR, 2019b) působil před vstupem do MapR Technologies jako ředitel v subjektech, z nichž později vznikly přední společnosti v sektoru IT, např. Microsoft, EMC, aj.

Z archivu dokumentace k distribuci MapR Data Platform je zřejmé, že (MapR, 2016d) alfa verze MDP byla vydána již březnu roku 2011, beta verze byla uvolněna prvního dubna

téhož roku a první stabilní verze 1.0 byla na trh uvedena 29. června 2011. Vydání první produkční verze distribuce MDP následovalo prakticky ihned po založení společnosti MapR Technologies. První majoritní produkční verze MDP (MapR, 2011) byla založena na Hadoop 1 a z open source projektů z oblasti big data zahrnovala především služby HBase, MapReduce a ZooKeeper a současně obsahovala proprietární komponenty společnosti MapR Technologies, mezi které patřily MapR-FS a optimalizovaná služba HBase. Již od prvních verzí byl v rámci distribuce MDP také k dispozici (Lublinsky, a další, 2013) vlastní systém pro nasazování a správu životního cyklu clusteru distribuce, včetně grafického rozhraní nesoucího název MapR Control System (MCS).

Od samého začátku bylo pro distribuci MapR symptomatické zaměření na (Turkington, a další, 2015) maximální výkon a zajištění vysoké dostupnosti. V reakci na nedostatky Hadoop 1.x tak MDP již v prvních verzích zajišťovala například (Turkington, a další, 2015) vysokou dostupnost komponent NameNode a JobTracker, nativní integraci s NFS, tedy funkce, které ostatní distribuce přinesly až v okamžiku, kdy inkorporovaly Hadoop 2.x. Distribuce MDP se oproti ostatním distribucím frameworku Hadoop diferencovala především substitucí některých klíčových komponent, které byly nahrazeny vlastní proprietární reimplementací. Distribuovaný souborový systém HDFS byl nahrazen (Erraissi, a další, 2017) distribuovaným souborovým systémem MapR-FS (později MapR XD), framework MapReduce společnost substituovala vlastní reimplementací nesoucí název MapR MapReduce a NoSQL databázi pro big data HBase nahradila proprietární optimalizovanou verzí. Společným motivem pro uvedené substituce přitom bylo zajištění vyššího výkonu a spolehlivosti. V případě první produkční verze distribuce MDP společnost v oficiální dokumentaci uváděla (MapR, 2011) až trojnásobný výkon oproti jakékoli jiné distribuci frameworku Apache Hadoop.

V průběhu roku 2011 a také 2012 následovala (MapR, 2016d) řada minoritních vydání 1.x, jež přinesla další optimalizace distribuce MDP a v srpnu roku 2012 byla vydána první produkční verze MDP 2.0. Verze 2.x (MapR, 2012) kromě dříve dostupných komponent MapR-FS, MapReduce, HBase, ZooKeeper a MCS nově zahrnovala také projekty Cascading, Flume, Hive, Mahout, Oozie, Pig, Sqoop a Whirr (kolekce knihoven pro správu cloudových služeb a nasazování clusterů). Kromě nových služeb poskytla nová majoritní verze distribuce (MapR, 2012) další funkce pro zajištění vysoké dostupnosti a vysoké výkonnosti i pro efektivnější monitoring a správu clusteru distribuce MDP a realizaci úloh zpracování big data (MapR Metrics).

Třetí majoritní verze MDP byla na trh uvedena (MapR, 2013) v květnu roku 2013. Množina komponent tvořících distribuci MDP zůstala oproti předchozí verzi prakticky (MapR, 2013) beze změny. Klíčovou novinku představovala v případě třetí verze MDP (MapR, 2013) možnost využití proprietární optimalizované verze HBase (MapR-DB). MDP 3.x současně reprezentovala první verzi distribuce MDP (MapR, 2013) oficiálně dostupnou v rámci veřejného cloutu společnosti Amazon v rámci služby EMR.

V červnu roku 2014 byla po řadě minoritních verzí vydána (MapR, 2014b) první produkční verze čtvrté řady distribuce MDP, která zajistila přechod z Hadoop 1 na Hadoop 2.x a podporu YARN. Nově byly v rámci této verze k dispozici komponenty Hue, Spark a Impala. V rámci čtvrté majoritní verze byly vydány (MapR, 2016d) tři minoritní verze,

poslední v dubnu roku 2015. Společnost MapR Technologies uvedla, že v roce 2014 (MapR, 2014a) dosáhla 700 platících zákazníků využívajících některou z komerčních edic distribuce MDP.

První produkční verze páté majoritní řady distribuce MDP byla na trh uvedena (MapR, 2019i) v červenci roku 2015 a v průběhu roku 2016 následovaly dvě minoritní verze. Mezi hlavní novinky, které pátá řada MDP přinesla, patřily především (MapR, 2016d) funkce pro audit využití dat spravovaných v rámci clusteru MDP (prostřednictvím projektu Sentry), širší množina analytických funkcí (projekt Drill), možnost indexace tabulek MapR-DB v rámci fulltextového vyhledávače Elasticsearch a dále řada dílčích vylepšení a/nebo optimalizací na úrovni vybraných komponent distribuce. Klíčové změny však přinesla především verze MDP 5.1, kdy (MapR, 2016a) byl z množiny komponent tvořících distribuci odstraněn projekt Whirr a nově byly přidány projekty Sqoop2 a Myriad (podpora nasazení YARN v rámci systému MESOS) a byla začleněna vlastní reimplementace Kafka (MapR-Streams, později MapR-ES). Současně daná verze znamenala (MapR, 2015) změnu názvu distribuce na MapR Converged Data Platform a také modifikace v názvech jednotlivých edic (podrobněji viz následující podkapitolu).

Rovněž verze MDP 5.2 a dvě další dílčí aktualizace přinesly řadu změn. Distribuce byla (MapR, 2016b) doplněna o nástroje pro efektivnější monitoring provozu clusteru s využitím komunitních verzí komponent třetích stran, mezi které patřily Grafana a Kibana, dále byla zařazena možnost častějších aktualizací komponent prostřednictvím tzv. MapR Ecosystem Pack (MEP) obsahujících nové verze primárně open source komponent, jež lze v rámci určité majoritní verze MDP nasazovat bez potřeby aktualizace distribuce jako celku. V neposlední řadě byl po řadě předchozích optimalizací (MapR, 2016b) změněn název reimplementace HDFS z MapR-FS na MapR-XD.

V březnu roku 2017 společnost MapR Technologies uvedla na trh (MapR, 2017b) odlehčenou verzi distribuce HDP nesoucí název MapR Edge primárně určenou pro případy užití z oblasti internetu věcí a umožňující nasazení na skutečně komoditním hardware.

Zatím poslední majoritní verzi distribuce MDP představuje (MapR, 2019i) řada 6, jejíž první produkční verze byla vydána v listopadu 2017 a do současné doby doznala pouze jednu minoritní aktualizaci (6.1.0) v srpnu roku 2018. Podporovány jsou již (MapR, 2019i) pouze verze MDP řady 6.x, jelikož podpora verzí řady 5.x oficiálně skončila v dubnu roku 2019. Šestá řada distribuce MDP poskytla kromě nových funkcí a optimalizací klíčových komponent především (MapR, 2018d) redesign grafického rozhraní MCS, vylepšenou podporu integrace s prostředím cloutu (podpora využití objektového úložiště S3 a sdílených souborových systémů v rámci cloudů založených na OpenStack) a podporu nasazení MDP o jednom uzlu v rámci kontejneru pro vývojáře. Minoritní verze 6.1.0 vydaná v září 2018, zajistila (MapR, 2018e) výrazně vyšší možnosti zabezpečení clusteru, především pokud jde o autentizaci (Kerberos) a šifrování dat v clusteru MDP, nové verze vybraných komponent, především ZooKeeper a Kafka a také změnu názvu MapR-Streams na MapR-ES.

Přestože společnost MapR Technologies od svého vzniku v roce 2009 opakovaně vykazovala (MapR, 2016c) (MapR, 2017a) enormní nárůst objemu předplatného za licence

pro distribuci MDP, v roce 2019 se společnost dostala (Gillin, 2019) do finančních potíží. Po (Gillin, 2019) krachu jednání s investory byl spoluzakladatel a výkonný ředitel společnosti John Schroeder nucen, na základě „extrémně špatných výsledků“ za aktuální fiskální čtvrtletí, zaslat 13. května 2019 zaměstnancům dopis, v němž oznámil, že v případě, že nebude nalezen partner, který by další chod společnosti zafinancoval, bude nucen společnost MapR Technologies uzavřít.

Někteří analytici přitom vysvětlují zhoršení finančních výsledků v letech 2018 a především 2019 tím, že (Gillin, 2019) trh s distribucemi frameworku Apache Hadoop je relativně malý, a že řada zákazníků na vlně adopce cloudu migrovala do prostředí veřejných poskytovatelů cloudu (AWS, GCP, Azure, aj.), kde mohou pro své případy použít využít i jiné nástroje, a kde distribuce frameworku nepředstavuje klíčovou platformu, ale zpravidla pouze jedno z řešení dostupných v rámci tržiště služeb/aplikací/řešení daného cloudu.

V důsledku uvedených událostí došlo v srpnu 2019 k (HPE, 2019) akvizici společnosti MapR Technologies, včetně zaměstnanců, technologií a intelektuálního vlastnictví ze strany společnosti Hewlett Packard Enterprise. Skutečnost, zda v roce 2020 bude nebo nebude vydána minoritní verze šesté řady distribuce MDP, tak může být jedním z indikátorů toho, zda bude společnost HPE distribuci dále rozvíjet či nikoli. V roce 2019 žádná nová majoritní ani minoritní verze distribuce MDP vydána nebyla.

4.3.2 Architektura a funkcionalita distribuce

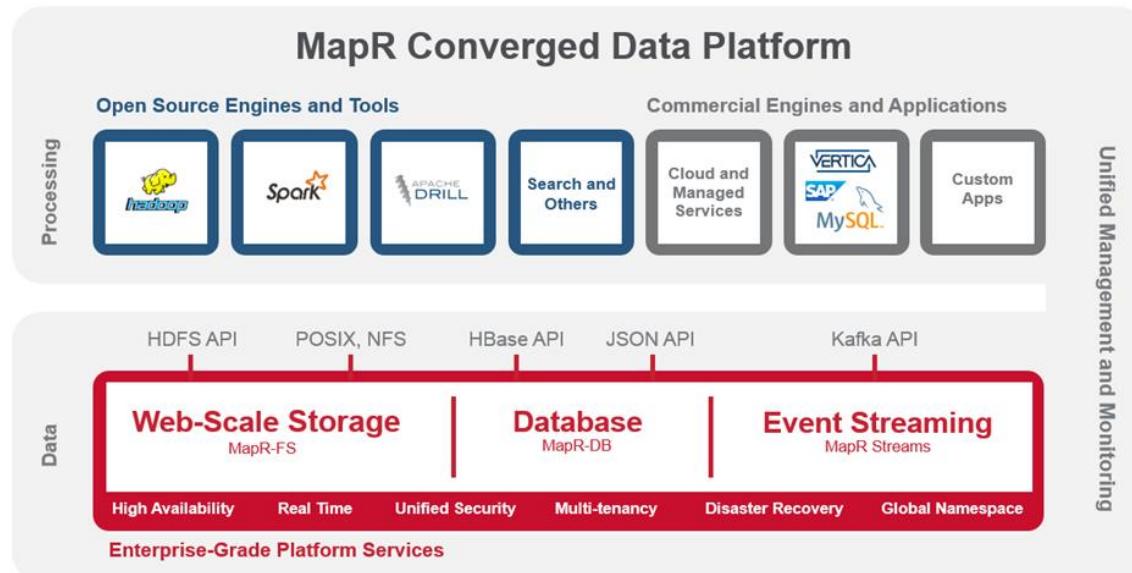
V době vzniku této práce představovala poslední dostupnou verzi distribuce MDP (MapR, 2019i) minoritní verze MDP 6.1.0 vydaná v září roku 2018. Tuto distribuci tvořily dvě hlavní množiny komponent. V první řadě se jednalo o (MapR, 2019i) množinu proprietárních reimplementací a komponent představujících jádro distribuce. Druhou kolekcí reprezentovala sada open source projektů tvořících tzv. (MapR, 2018b) balíčky ekosystému MapR (MEP). Tyto zahrnují (MapR, 2018b) množinu komponent, jejichž verze mohou (jako součást daného MEP) fungovat v rámci jedné nebo více verzí jádra MDP. Jádro distribuce MDP 6.1 tvořily následující komponenty (MapR, 2018c; MapR, 2018a):

- MapR-XD Distributed File and Object Store (dříve MapR-FS) – proprietární reimplementace HDFS,
- Hadoop – komponenty YARN, MapReduce 2,
- MapR Data Fabric for Kubernetes – interface a ovladač pro nasazování aplikací interagujících s platformou MDP v rámci (Kubernetes) clusterů kontejnerů,
- MapR Database (MapR-DB) – reimplementace HBase,
- MapR Event Store for Apache Kafka (MapR-ES) – reimplementace Kafka,
- MapR Control System – grafické rozhraní pro nasazení a správu distribuce MDP,
- MapR Monitoring (Spyglass Initiative) – sada integrovaných open source nástrojů pro monitoring provozu clusteru (Collectd, OpenTSDB, MapR-DB, Grafana pro monitoring metrik, Fluentd, Elasticsearch, Kibana pro monitoring logů).

Množina open source projektů tvořící MEP pro MDP 6.1, pak zahrnovala především projekty (MapR, 2018b) Cascading, Drill, Flume, HBase, Hive, Hue, Impala, Myriad,

Oozie, Pig, Sentry, Spark a Sqoop. Vysokoúrovňovou architekturu²¹ distribuce MDP znázorňuje Obr. 4-7, z nějž je dobře patrné, že distribuce MDP je tvořena jádrem zaměřeným na příjem a uložení „klasických“ i streaming dat a nadstavbou (MEP) poskytující nástroje pro transformace a zpracování big data.

Obr. 4-7 Vysokoúrovňová architektura distribuce MDP



Zdroj: data a zpracování (MapR, 2016b)

Jelikož byla majoritní část zmíněných open source projektů tvořících MEP již v rámci předchozího textu charakterizována, bude další popis architektury a funkcionality distribuce MDP zaměřen primárně na komponenty jádra distribuce.

Ačkoli reimplementace HDFS v současné době oficiálně nese název MapR-XD Distributed File and Object Store, majoritní část této služby stále (MapR, 2018c) představuje a zajišťuje distribuovaný souborový systém MapR neboli MapR-FS. Na rozdíl od HDFS (MapR, 2018c) podporujícího pouze zápis do existujících souborů (append) a čtení z uzavřených souborů, MapR-FS podporuje paralelní zápis i čtení. Faktorem, který determinuje (MapR, 2018c) násobně vyšší výkonnost MapR-FS oproti HDFS je pak skutečnost, že MapR-FS není implementován nad existujícím souborovým systémem operačního systému, ale přímo nad hardwarem, tedy nad rotačními nebo SSD diskami, a že MapR-FS je napsán v programovacím jazyku C/C++, což eliminuje režii spojenou s automatickým uvolňováním paměti (garbage collection) v případě HDFS, který je napsán v programovacím jazyku Java. Současně je klíčovou architektonickou vlastností MapR-FS podpora vysoké dostupnosti, jelikož komponenta zajišťující správu jmenného prostoru distribuovaného souborového systému (MapR, 2018c) nepředstavuje SPOF, na rozdíl od jiných distribucí v případě, pokud by byl nasazen pouze jeden NameNode.

²¹⁾ Schéma znázorňující vysokoúrovňovou architekturu distribuce MDP bylo záměrně použito z dokumentace MDP 5.1 a nikoli 6.1, protože lépe ilustruje reálnou strukturu distribuce, na rozdíl od schématu, které je součástí dokumentace MDP 6.1, a jež je zaměřeno primárně na funkcionality.

V rámci každého datového uzlu clusteru MDP jsou (MapR, 2018c) disky (defaultně po 3) seskupovány do tzv. kolekcí úložišť (storage pools). Data distribuovaného souborového systému jsou organizována (MapR, 2018c) v rámci tzv. kontejnerů (containers)²², které jsou replikovány napříč storage pools, přičemž defaultní velikost kontejneru odpovídá 32 gigabytům. Soubory jsou podobně jako v případě HDFS (MapR, 2018c) rozděleny do dílů (chunks) zařazených do kontejnerů, jež jsou zapsány a replikovány napříč storage pools. Defaultní velikost chunku (MapR, 2018c) odpovídá v případě MapR-FS 256 megabajtů, jednotkou replikace je 32 gigabajtový kontejner a jednotku alokace představuje blok o velikosti 8 kilabajtů. Naproti tomu v případě HDFS odpovídají (MapR, 2018c) chunk, jednotka replikace i jednotka alokace v podstatě velikosti 64/128 megabajtů. Uvedené vlastnosti v případě MapR-FS determinují (společně s dalšími faktory) možnost relativně flexibilní čtení a zápis, vyšší škálovatelnost a řádově vyšší výkon.

Kontejnery MapR-FS jsou dále organizovány v rámci tzv. (MapR, 2018c) svazků (volumes), které mohou dosahovat velikosti 50 až 100 milionů kontejnerů, přičemž MapR-FS poskytuje funkcionality pro vytváření snímků (snapshots) svazků a také automatické zrcadlení (mirroring) mezi různými clustery MDP.

Komponenta, která substituuje NameNode, jak jej známe z HDFS, nese v případě distribuce MDP název (MapR, 2018c) Container Location Database (CLDB) a zajišťuje, analogicky k NameNode, správu jmenného prostoru clusteru MapR-FS, tedy informací o všech kontejnerech, a dále figuruje v rámci replikace a aktualizace kontejnerů. Vždy je nasazována (MapR, 2018c) jedna master instance CLDB, přičemž veškerá CLDB data jsou replikována na více uzelů v clusteru a v případě selhání procesu master CLDB je tento automaticky znova spuštěn, a pokud dojde k selhání uzlu jako takového, lze iniciovat (v případě použití edice enterprise automaticky) tzv. failover, tedy spuštění nového master na základě dostupných dat na jiném uzlu v clusteru.

MapR-FS představuje distribuovaný souborový systém, který reimplementuje HDFS a je klíčový pro zajištění vyššího výkonu a spolehlivosti oproti dalším distribucím frameworku Apache Hadoop a příslušným komponentám ekosystému big data, a na němž staví prakticky téměř všechny další komponenty jádra platformy MDP.

Projekt MapR Data Fabric for Kubernetes umožňuje (MapR, 2018c) nasazovat aplikace zpracování big data v rámci clusterů kontejnerů, a to nikoli kontejnerů MapR-FS, ale naopak aplikačních kontejnerů, které jsou nasazovány prostřednictvím některého z podporovaných orchestrátorů kontejnerů, primárně Kubernetes. Projekt v současné době poskytuje především dvě komponenty, a to (MapR, 2018c) MapR CSI Storage Plugin, zásuvný modul využívající standardní rozhraní kontejnerů pro nasazení aplikací MDP nad clustery kontejnerů, a pak také MapR Data Fabric for Kubernetes FlexVolume Driver, ovladač v podobě množiny Docker kontejnerů poskytujících persistentní úložiště pro Kubernetes objekty v rámci MapR-FS. Komponenta MapR Data Fabric for Kubernetes umožňuje integrovat cluster distribuce MDP s technologiemi pro kontejnerizaci, které jsou v současné době velice rozšířené a oblíbené a zajišťují vyšší flexibilitu oproti „tradičním“

²²) Kontejner zde představuje jednotku abstrakce, nikoli kontejner typu Docker, apod.

prostředím big data clusteru. Nejedná se nicméně o způsob, jakým by se cluster MDP nasazoval nad Kubernetes clusterem, ale naopak o nástroje podporující snadnou integraci těchto prostředí.

MapR Database (MapR-DB) představuje vysoce efektivní proprietární reimplementaci HBase, která (McDonald, 2017) využívá předností MapR-FS a poskytuje oproti ostatním big data NoSQL databázím, jako například Cassandra nebo HBase o minimálně 50% vyšší výkon při zápisu, čtení i aktualizaci záznamů v databázi. Vyšší výkon MapR-DB je determinován především tím, že (McDonald, 2017) využívá MapR-FS, jehož výkonnostní přednosti byly charakterizovány v rámci předchozího textu, a protože se nachází v rámci identického procesního prostoru jako MapR-FS, je stejně jako v případě MapR-FS eliminována režie na provoz dalších vrstev, jako např. FS nebo JVM. Architektura MapR-DB je také oproti HBase odlišná v tom, že (McDonald, 2017) nedochází k tzv. compaction, tj. procesu, kdy jsou soubory, v nichž jsou uložena data (HFiles) seskupovány s cílem minimalizace latence operací čtení. Naopak, aktualizace záznamů jsou zařazeny do tzv. micro logů, které jsou realizovány v rámci operační paměti, což opět determinuje vyšší výkon. MapR-DB tedy reprezentuje substituci za HBase umožňující plně využít všech předností plynoucích z využití MapR-FS a dosahovat maximálního výkonu při práci s daty v rámci big data NoSQL databáze.

MapR Event Store for Apache Kafka (MapR-ES) je (Downard, 2017) proprietární reimplementací Kafka Java API, takže aplikace vytvořené pro Apache Kafka je možné bez jakékoli modifikace provozovat i nad MapR-ES. Společnost MapR Technologies uvádí, že propustnost MapR-ES je až (Downard, 2017) čtyřikrát vyšší než v případě Kafka, a to mimo jiné díky využití více vláken při přenosu dat z klientských aplikací do serveru, a dále díky výkonnostním přednostem MapR-FS. Zatímco Kafka (Downard, 2017) ukládá fronty zpráv (topic) formou adresáře a několika souborů v rámci obecného FS, v případě MapR-ES představují fronty pouze metadata, což opět snižuje režii nezbytnou pro práci s takovou entitou. Z hlediska replikace Kafka v případě clusteru brokerů aplikuje mechanismus volby master instance (leader), přičemž pro koordinaci využívá ZooKeeper, MapR-ES využívá přímo replikaci, kterou zajišťuje MapR-FS, takže při skutečně velkých objemech streamů dat poskytuje znatelně vyšší škálovatelnost. MapR-ES představuje optimalizovanou reimplementaci Kafka, která stejně jako v případě ostatních komponent jádra MDP, přináší oproti komponentám, jež reimplementuje, výkonnostní a provozní benefity.

Poslední část jádra platformy MDP reprezentuje množina nástrojů pro nasazení, provoz a monitoring clusteru MDP. Společnost MapR Technologies vyvinula, podobně jako Cloudera a Hortonworks, vlastní (MapR, 2018a) nástroj pro nasazení, správu a provoz clusteru distribuce frameworku Apache Hadoop, který nese název MapR Control System a poskytuje grafické rozhraní, jež je znázorněno na Obr. 4-8. Pro zajištění monitoringu provozu clusteru využívá distribuce MDP relevantní open source projekty, konkrétně Collectd pro sběr metrik, OpenTSDB pro uchování metrik, které jsou dále synchronizovány do MapR-DB, Grafana pro vizualizaci metrik, Fluentd pro shromažďování logů, Elasticsearch pro agregaci a uchování logů a Kibana pro vizualizaci a reporting logů. Na rozdíl například od společnosti Hortonworks a projektu Ambari tak společnost využila obecně dostupné projekty namísto vlastního systému pro sběr a

monitoring metrik a logů (Ambari sice využívá Solr a produktů Elasticsearch, ale v rámci speciálních balíků pro Ambari, tj. Ambari Infra Solr a Logsearch).

Obr. 4-8 Grafické rozhraní MapR Control System



Zdroj: data a zpracování (MapR, 2019e)

Distribuce MDP poskytuje komponenty, které komplexně pokrývají celý proces zpracování big data, od přijetí big data (MapR-XD, MapR-DB, MapR-ES, Flume, Sqoop), přes transformaci a uložení (MapR-XD, MapR-DB, MapR-ES, Flume, Hive), až po zpracování, analýzu a vizualizaci (YARN, MapReduce 2, Cascading, Drill, Impala, Oozie, Pig, Spark, Myriad, MapR Data Fabric for Kubernetes), a to jak pro úlohy tradičního dávkového/ad hoc zpracování, tak i kontinuálního zpracování proudů dat a při zajištění relevantních nástrojů pro nasazení, správu a monitoring clusteru MDP (MCS, MapR Monitoring).

Jak bylo uvedeno v kapitole věnované distribuci CDH, její tvůrci kromě vlastní distribuce uvedli na trh také další projekty, jimiž obohatili ekosystém projektů z oblasti big data a později, když byly uvolněny jako open source, byly tyto v některých případech zařazeny i do jiných distribucí. Naproti tomu společnost MapR Technologies se zaměřila na vlastní proprietární reimplementaci klíčových komponent projektu Hadoop (HDFS) a vybraných

komponent ekosystému Hadoop (HBase, Kafka) a vývoj vlastních proprietárních driverů pro integraci clusterů (primárně) Kubernetes v procesu zpracování big data. Vytvořila tak jádro distribuce, které oproti ostatním distribucím umožňuje dosahovat řádově vyššího výkonu, především díky eliminaci vrstev JVM a lokálních souborových systémů, a od samého začátku je zaměřeno na zajištění vysoké dostupnosti celého řešení. Toto jádro je navíc velice vhodně doplněno o klíčové projekty ekosystému Hadoop zajišťující dostatečnou flexibilitu při zpracování, analýze a vizualizaci big data.

4.3.3 Model poskytování distribuce

Podobně jako v případě ostatních distribucí frameworku Apache Hadoop se model poskytování MDP postupně vyvíjel. V době uvedení první produkční verze MDP na trh byly k dispozici (MapR, 2011) dvě edice, a sice M3 a M5, přičemž edice M3 byla k dispozici zdarma, na rozdíl od verze M5, kterou bylo nutné hradit a zahrnovala placenou podporu a funkce pro zajištění vysoké dostupnosti clusteru MDP. Uvedený model zůstal v platnosti také v případě (MapR, 2012) druhé majoritní verze distribuce MDP.

S vydáním třetí majoritní verze MDP byl model poskytování distribuce (MapR, 2013) rozšířen o třetí edici, M7, která byla pochopitelně placená a nad rámec nižších edic umožňovala především využití proprietární optimalizované komponenty HBase. Společnosti MapR Technologies se také v případě třetí distribuce (MapR, 2013) podařilo prosadit v prostředí veřejných cloudů, když společnost Amazon začala MDP od roku 2013 oficiálně nabízet v rámci služby Amazon EMR.

Další výrazná změna modelu poskytování distribuce přišla s vydáním minoritní verze MDP 5.1 v únoru 2016, kdy byl změněn nejen název distribuce na MapR Converged Data Platform, ale také názvy jednotlivých edic (MapR, 2016a):

- MapR Community Edition (dříve M3) – volně dostupná komunitní edice,
- MapR Enterprise Edition (dříve M5) – edice s placenou podporou a funkcemi pro zajištění vysoké dostupnosti,
- MapR Enterprise Database Edition (dříve M7) – komerční edice, která nad rámec předchozí edice zahrnuje proprietární reimplementaci HBase, tedy MapR-DB.

V současné době je distribuce MDP k dispozici ve (MapR, 2019f) dvou edicích, a to Converged Community Edition a Converged Enterprise Edition. V obou edicích jsou dostupné prakticky všechny komponenty, nicméně verze Enterprise poskytuje nad rámec komunitní edice především (MapR, 2019f) funkcionalitu pro zajištění pokročilé multitenance, konzistentních snapshotů, vysoké dostupnosti, obnovy z výpadku (disaster recovery), pokročilé replikace MapR-DB a MapR-ES a také placenou podporu v režimu 24x7. Zde je nutné zdůraznit, že ačkoli je komunitní edice distribuce MDP dostupná zdarma, v souladu s licenční smlouvou společnosti MapR (MapR, 2019g) není možné komunitní edici distribuce MDP žádným způsobem použít pro komerční účely.

Také distribuce MDP umožňuje nasazení on-premise i v prostředí cloutu. Ceny za předplatné nejsou veřejně dostupné, nicméně například v případě využití distribuce MDP v rámci služby big data clusteru Amazon EMR byly zákazníkům účtovány (Amazon, 2019c) za hodinu na každém stroji v clusteru 0,070\$ za edici M3, 0,119\$ za M5 a 0,210\$ za edici

M7. V případě clusteru o velikosti 10 uzlů by to znamenalo účet 50,4\$ za každý den využití. Podobně jako producent předchozí distribuce frameworku Apache Hadoop, také společnost MapR Technologies od svého vzniku budovala síť partnerů a v současné době disponuje řadou strategických partnerů na straně zákazníků i v řadách technologických partnerů, systémových integrátorů a dalších.

4.4 Hortonworks Data Platform

4.4.1 Vznik a rozvoj distribuce

Společnost Hortonworks, která vytvořila distribuci frameworku Apache Hadoop pod názvem Hortonworks Data Platform (HDP), založilo v čele s Ericem Baldeschweilerem (Hortonworks, 2019k) v červenci 2011 dvacet čtyři bývalých zaměstnanců společnosti Yahoo!, kteří se věnovali projektu Hadoop.

První oficiálně dostupná verze distribuce HDP (1.0.0.12) byla na trh uvedena (Hortonworks, 2012a) v červnu 2012. Tato byla založena na Hadoop 1.0.3 a zahrnovala tzv. core komponenty (Hadoop – HDFS, MapReduce), základní komponenty (Pig, Hive, HBase, ZooKeeper) a podpůrné projekty (Oozie, Sqoop, Ganglia, Nagios). První vydaná verze distribuce HDP podporovala operační systémy CentOS a RHEL a zahrnovala (Hortonworks, 2012a) grafické rozhraní pro nasazení a správu clusteru nesoucí název Hortonworks Management Center, přičemž pro monitoring byly využity externí nástroje Ganglia a Nagios.

Následovala řada minoritních i údržbových (maintenance) vydání první řady distribuce HDP. V rámci verze HDP 1.2.0, vydané v lednu 2013, byl název grafického rozhraní pro nasazování a správu clusterů HDP (Hortonworks, 2013a) změněn na Ambari, nicméně pro monitoring byly stále ještě využívány externí nástroje Ganglia a Nagios. Tato verze distribuce zahrnovala projekty (Hortonworks, 2013a) Ambari, Hadoop (HDFS, MapReduce), Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Ganglia, Nagios a rozšířila podporu OS kromě CentOS a RHEL také na Oracle Enterprise Linux a SUSE Linux Enterprise Server. Dále následovala ještě minoritní verze HDP 1.3 a řada maintenance vydání HDP 1.3.x.

Již v říjnu roku 2013 byla vydána (Hortonworks, 2013b) první stabilní verze HDP řady 2, tj. HDP 2.0.0, která znamenala přechod distribuce na Hadoop 2 a tedy YARN. Iniciální vydání druhé řady HDP zahrnovalo projekty (Hortonworks, 2013b) Ambari, Hadoop (HDFS, YARN, MapReduce 2), Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume a Mahout. V následujících letech distribuce frameworku Apache Hadoop společnosti Hortonworks prošla intenzivním vývojem zahrnujícím šest minoritních vydání HDP 2 (HDP 2.1 – 2.6) a desítky údržbových vydání napříč jednotlivými minoritními verzemi.

V lednu 2014 byla na trh uvedena verze HDP 2.0.0, která podporovala (Microsoft, 2014) nasazení dané distribuce na OS Windows Server (o několik let později byla podpora nasazení nad Windows Server zrušena). V dubnu 2014 (Hortonworks, 2014c) společnost Hortonworks vydala HDP 2.1.0, kdy do této verze byla zařazena řada dalších projektů ekosystému Hadoop, především Tez, Storm, Falcon, Knox, Accumulo, Phoenix, Avro a

Solr). Uvedená množina komponent tvořila distribuci HDP napříč všemi následujícími maintenance vydáními řady 2.1.x.

Současně s vydáním HDP 2.1.0, které poprvé zahrnovalo Solr, byl vydán zřejmě první produkt, který nebyl k dispozici zdarma v rámci HDP, ale byl dostupný pouze pro zákazníky s placenou podporou. Jednalo se o (OnDataEngineering, 2018) Hortonworks Data Platform Search poskytující funkcionality pro indexování a full-textové vyhledávání v rámci clusteru Hadoop, především pro komponenty HDFS, Hive, HBase, Storm, Spark. Hortonworks Data Platform Search je stále součástí portfolia doplňkových služeb distribuce HDP, přičemž zatím poslední verze 5.0.0 byla vydána (Hortonworks, 2019d) v srpnu roku 2019.

V rámci verze HDP 2.2.0, vydané v prosinci 2014, byly nově přidány komponenty (Hortonworks, 2014b) Ranger, DataFu, Kafka a Slider a současně byly jako deprecated označeny externí nástroje pro monitoring Ganglia a Nagios, takže od dané verze HDP funkcionality monitoringu clusteru HDP v plném rozsahu zajišťovala komponenta Ambari.

V dubnu roku 2015 společnost Hortonworks realizovala (OnDataEngineering, 2019) akvizici společnosti SequenceIQ, která se primárně zabývala vývojem nástroje Cloudbreak, umožňujícím automatizovaně nasazovat a spravovat clustery Hadoop nad cloudovou infrastrukturou běžící na Docker kontejnerech s využitím automatizovaného nasazování prostřednictvím Ambari blueprints (specifikace cílového prostředí clusteru v rámci JSON souboru). Tímto způsobem si společnost Hortonworks zajistila možnost efektivního nasazení HDP v rámci cloudových platforem AWS, Azure, GCP i OpenStack a v (OnDataEngineering, 2019) červenci 2015 nástroj Cloudbreak uvedla na trh ve verzi 1.0 v rámci HDP 2.3.0. Nástroj Cloudbreak je pro zákazníky stále k dispozici, přičemž v současné době je dostupná verze Cloudbreak 2.9.1 vydaná (Hortonworks, 2019b) v květnu 2019 a umožňující nasazovat v cloudu Ambari, tak HDP i HDF.

Nová verze HDP 2.3.0 také znamenala rozšíření portfolia projektů integrovaných v rámci distribuce frameworku Apache Hadoop společnosti Hortonworks o (Hortonworks, 2015d) Calcite, Cascading, Solr, Spark a především eliminaci externích nástrojů Ganglia a Nagios.

V říjnu roku 2015 představila společnost další produkt, který byl dostupný pouze pro zákazníky s placenou podporou ze strany společnosti Hortonworks. Jednalo se o řešení SmartSense, které poskytovalo (Cision PR Newswire, 2015) funkcionality proaktivní údržby a monitoringu clusteru HDP pro eliminaci provozních problémů ještě před jejich vyústěním v incident způsobující výpadek služeb clusteru. Přestože je komponenta SmartSense součástí volně dostupné distribuce HDP a lze ji nasazovat prostřednictvím Ambari, její reálné využití je možné pouze v případě, že si uživatel hradí placenou podporu ze strany Hortonworks. Komponenta SmartSense v dalších letech neprošla žádnými signifikantními změnami, je stále dostupná i v posledních verzích distribuce HDP a zatím poslední vydanou verzí je (Hortonworks, 2018d) SmartSense 1.5.1 z listopadu roku 2018.

V roce 2015 společnost Hortonworks uvedla na trh další produkt, a to Hortonworks DataFlow (HDF) přestavující platformu pro škálovatelné zpracování proudů dat v reálném čase. Již ve verzi HDF 1.1.0, vydané (Hortonworks, 2015c) v prosinci roku 2015, byl

produkt HDF založen na (Hortonworks, 2019i) projektu Apache Ni-Fi, systému pro masivně škálovatelný příjem, zpracování a distribuci dat, MiNiFi, odlehčené verzi Ni-Fi určené pro nasazení blízko/přímo u zdroje emitujícího proudy dat, Schema Registry – sdíleného repositáře schémat pro aplikace a komponenty HDF a Streaming Analytics Manager – rozhraní pro drag and drop návrh, vývoj, nasazení a správu aplikací pro zpracování proudů dat s využitím HDF a enginů pro zpracování dat (Storm, Spark, aj.). Tyto core komponenty byly doplněny komponentami HDP relevantními pro zpracování proudů dat, především Ranger, Kafka, Storm.

Stejně jako v případě HDP i HDF bylo od samého vzniku produktu možné nasazovat prostřednictvím Ambari, a to samostatně nebo jako doplněk k již nasazenému clusteru HDP. Uvedením produktu HDF společnost reagovala na trend internetu věcí a stále častější podíl úloh zpracování big data vykazujících charakter kontinuálního zpracování proudů dat spíše než pravidelného/ad hoc zpracování dávek big data.

Minoritní verze HDP 2.4.0 vydaná (Hortonworks, 2016b) v březnu 2016 přinesla z hlediska složení množiny projektů tvořících HDP pouze jednu výraznější změnu, a to nově zařazený projekt Atlas. Následující maintenance vydání řady HDP 2.4.x, HDP 2.5.0 vydaná v (Hortonworks, 2016c) srpnu 2016, ani údržbová vydání řady HDP 2.5.x nepřinesla žádnou změnu struktury komponent HDP s výjimkou přidání komponenty Zeppelin a řady minoritních upgradů stávajících komponent a dalších dílčích optimalizací. V září roku 2016 společnost vydala (Hortonworks, 2016d) první verzi řady HDF 2.0.0 přinášející především zlepšení výkonu celého řešení, optimalizaci grafického rozhraní a lepší integraci s ekosystémem clusteru HDP.

Naproti tomu HDP 2.6.0, kterou společnost Hortonworks na trh uvedla (Hortonworks, 2017a) v dubnu 2017, reprezentovala společně s následujícími minoritními a maintenance vydáními dané řady významné změny. V rámci první stabilní verze totiž byly (Hortonworks, 2017a) jako deprecated označeny komponenty Falcon, Flume, Mahout, Slider, Cascading, Hue a komponenty Accumulo, Kafka, Storm a Cloudbreak byly označeny jako moving, což indikovalo plánované přesunutí do jiného produktu a/nebo zavedení určité formy zpoplatnění ve formě předplatného. Zatímco všechny komponenty označené jako deprecated byly v rámci vydání HDP 3.0.0 (v dalším textu bude ještě charakterizováno) skutečně z distribuce odstraněny, všechny komponenty označené jako moving, s výjimkou Cloudbreak, v distribuci HDP zůstaly. V neposlední řadě byly v rámci HDP 2.6.0 signifikantně povýšeny verze vybraných komponent distribuce. Jednalo se především o (Hortonworks, 2017a) Spark (2.1.0) a Hive (2.1.0). Maintenance vydání řady 2.6.x přinesla i nové komponenty, konkrétně Livy, Druid, Superset.

Novou majoritní verzi HDF 3.0.0 společnost Hortonworks na trh uvedla (Hortonworks, 2017b) v červnu 2017. V této verzi byly poprvé, i když pouze jako technical preview, dostupné komponenty (Hortonworks, 2017b) Hortonworks Schema Registry, rozhraní pro správu schémat a metadat a Hortonworks Streaming Analytics Manager, grafické rozhraní poskytující funkcionality pro návrh a realizaci flow zpracování proudu dat a především analytické úlohy nad přijímanými daty.

V říjnu 2017 (Hortonworks, 2017c) společnost Hortonworks představila řadu komerčních, respektive pouze v rámci placené podpory dostupných produktů, a to Hortonworks

DataPlane Service 1.0.0 později přejmenovanou na Hortonworks DataPlane Platform. Jednalo se o zastřešující komponentu pro (Hortonworks, 2017c) nasazování aplikace Data Steward Studio, zaměřené na funkce pro lepší porozumění a správu dat v rámci clusteru HDP a aplikace Data Lifecycle poskytující funkcionalitu pro vytváření a správu politik a úloh replikace a disaster recovery mezi clustery HDP. Podobně jako dříve zmíněný produkt SmartSense, neprošla ani sada komponent Hortonworks DataPlane Platform žádnými signifikantními proměnami. Následující minoritní a maintenance vydání byla zaměřena spíše na odstraňování chyb a zajištění podpory nasazení pro clustery s novými verzemi HDP. Zatím poslední verze Hortonworks DataPlane Platform 1.2.3 byla vydána (Hortonworks, 2019c) v březnu 2019.

V červenci roku 2018 byla vydána (Hortonworks, 2018b) první stabilní verze nové řady HDP 3.0.0. Jak již bylo uvedeno, byla dle předem avizovaného plánu (Hortonworks, 2018b) odstraněna z distribuce řada komponent, takže množina projektů HDP 3.1.0 zahrnovala Accumulo, Atlas, Calcite, DataFu, Druid, Hadoop, HBase, Hive, Kafka, Knox, Livy, Oozie, Phoenix, Pig, Ranger, Spark, Sqoop, Storm, Tez, Zeppelin, ZooKeeper, Superset. Třetí řada HDP opět integrovala novou verzi frameworku Hadoop, konkrétně 3.1.0. Výrazné změny doznalo oproti HDP 2.x rovněž grafické rozhraní nástroje Ambari pro nasazení, provoz, správu a monitoring clusteru HDP.

Ještě téhož roku společnost rozšířila portfolio produktů dostupných v rámci placené podpory, když (Hortonworks, 2018a) v srpnu 2018 vydala produkt Data Analytics Studio 1.0.0. Jednalo se v podstatě o grafické rozhraní pro správu a dotazování Hive tabulek, včetně funkcionality pro optimalizaci výkonnosti těchto analytických dotazů. Zatímco v prvních verzích se tato aplikace nasazovala nad již zmíněnou nadstavbou Hortonworks DataPlane Platform, později již bylo možné nasazovat Data Analytics Studio samostatně. Také tento produkt je stále nabízen, přičemž zatím poslední verze, 1.4.4, byla vydána (Hortonworks, 2020) v lednu 2020.

Jak již bylo uvedeno, v říjnu roku 2018 oznámila společnost Cloudera (Cloudera, 2018b) sloučení se společností Hortonworks v reakci na propad hospodářských výsledků způsobený zejména vlivem (Bhatia, 2018) rozmachu objektových úložišť, která determinovala eliminaci závislosti na HDFS a nastupu cloudových platform AWS, GCP, Azure, aj. poskytujících elastická řešení nejen pro zpracování a analýzu big data.

V prosinci roku 2018 (Hortonworks, 2018c) byla na trh uvedena HDP 3.1.0. Daná minoritní verze nepřinesla žádné výraznější změny, spíše se jednalo o očekávané vydání poskytující některé opravy chyb z předchozích vydání HDP 3.0.x. Jednalo se o poslední verzi, kterou společnost Hortonworks vydala ještě před dokončením fúze se společností Cloudera. Právě tato verze byla použita pro realizaci praktické úlohy zpracování big data, která je předmětem následující kapitoly.

V lednu roku 2019 bylo (Cloudera, 2019c) dokončeno sloučení společností Cloudera a Hortonworks. To mělo mimo jiné za důsledek, že webové stránky hortonworks.com byly postupně přesměrovány na doménu cloudera.com a restrukturalizovány. Ačkoli společnost Cloudera avizovala, že distribuci HDP bude podporovat i po dokončení akvizice Hortonworks, nebylo zcela zřejmé, zda bude nadále vydávat nové verze této distribuce. V srpnu 2019 byla nicméně na trh uvedena (Hortonworks, 2019g) HDP 3.1.4, což

rozptýlilo obavy na dané téma. Tato verze nepřinesla žádné změny co do množiny projektů tvořících distribuci, ani signifikantní změny verzí jednotlivých komponent.

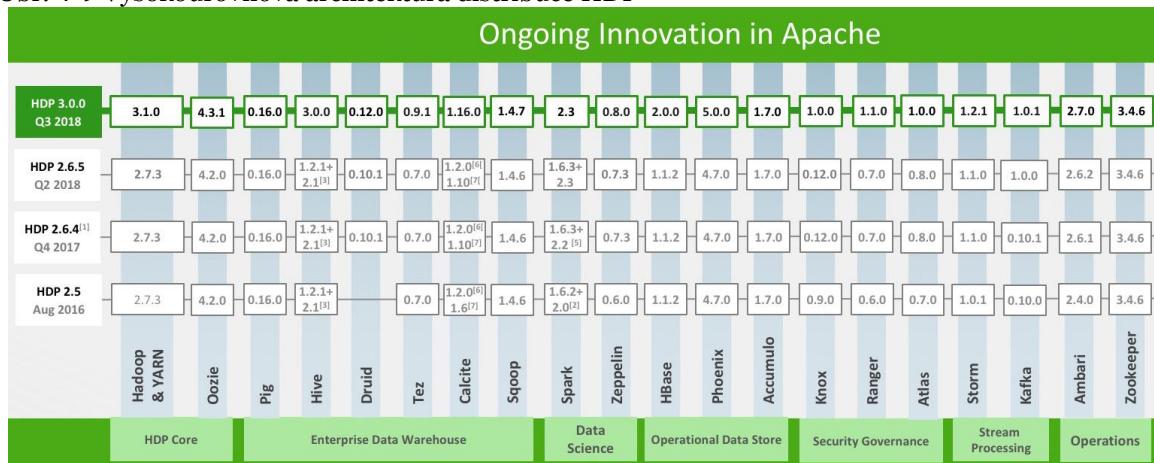
Poslední verzi HDF, která byla zatím vydána, představuje verze HDF 3.4.1.1, kterou společnost Hortonworks na trh uvedla (Hortonworks, 2019j) v květnu roku 2019. V této verzi již byla součástí HDF také podpora SmartSense a prakticky všechny komponenty byly podporovány jako produkční.

Zatím poslední verzí HDP, která byla uvedena na trh ještě před dokončením této práce, byla HDP 3.1.5, jež byla vydána (Hortonworks, 2019h) v prosinci roku 2019. Tato verze přinesla (Hortonworks, 2019h) novou verzi Ambari 2.7.5, novější verzi HBase 2.1.6 a především více než 200 oprav a/nebo optimalizací celé distribuce. Na druhou stranou dané vydání bylo zlomové vydání v tom směru, že se jednalo o první verzi distribuce HDP, jejíž repositáře nejsou veřejně dostupné a mohou je využívat pouze platící zákazníci společnosti.

4.4.2 Architektura a funkcionalita distribuce

Distribuce HDP ve verzi 3.1.0 vydané v prosinci 2018 zahrnovala projekty (Hortonworks, 2018c) Ambari, Accumulo, Atlas, Calcite, DataFu, Druid, Hadoop 3, HBase, Hive, Kafka, Knox, Livy, Oozie, Phoenix, Pig, Ranger, Spark, Sqoop, Storm, Superset, Tez, Zeppelin a ZooKeeper. Základní architektura distribuce HDP 3.x je znázorněna na Obr. 4-9.

Obr. 4-9 Vysokoúrovňová architektura distribuce HDP



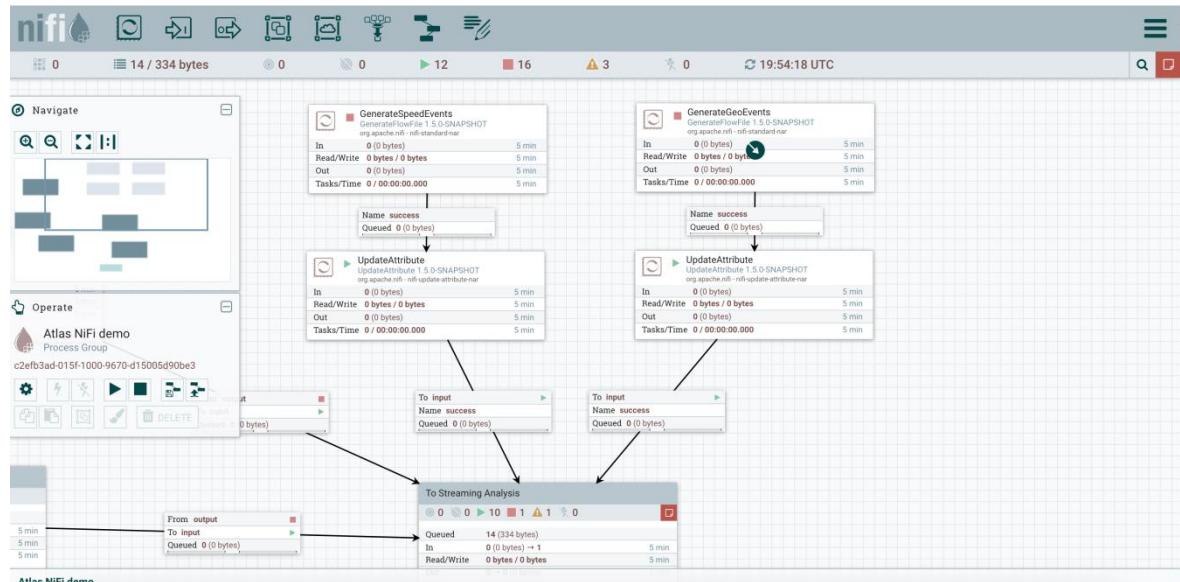
Zdroj: data a zpracování (Woodie, 2018)

Z uvedeného schématu je zřejmé, že jádrem distribuce HDP je (Hortonworks, 2018c) projekt Hadoop a plánovače workflow Hadoop úloh nesoucího název Oozie. Pro zajištění funkcionality big data datového skladu slouží komponenty Pig, Hive, Druid, Tez, Calcite a Sqoop. Core funkcionalitu datového skladu pro big data zajišťuje Hive, ostatní uvedené komponenty jsou spíše podpůrného charakteru s výjimkou provozní (operational) real-time databáze Druid a nástroje Sqoop pro migraci rozsáhlých kolekcí dat mezi relačními databázemi a clustery Hadoop. Jako provozní datové úložiště byly v rámci distribuce HDP 3.1.0 k dispozici NoSQL databáze pro big data HBase, databázový engine pro OLTP nad HBase a Hadoop Apache Phoenix a NoSQL databáze typu klíč-hodnota Accumulo.

Další součástí distribuce byly (Hortonworks, 2018c) Spark a Zeppelin poskytující funkcionality z oblasti data science. Pro zajištění bezpečnosti clusteru HDP a big data, která jsou v jeho rámci uložena a zpracovávána, slouží v rámci distribuce HDP komponenty Knox, Ranger a Atlas. Storm a Kafka lze využít v procesu příjmu a zpracování proudů dat. Pro nasazení, správu, provoz a monitoring clusteru HDP slouží již zmíněný nástroj Ambari, jehož grafické rozhraní bylo znázorněno v kapitole 3.3.2 na Obr. 3-10. Majoritní část komponent distribuce využívá podpůrný nástroj ZooKeeper zajišťující vysoce spolehlivou koordinaci v distribuovaném prostředí. Přestože nejsou uvedeny v předchozím schématu, jsou součástí distribuce HDP také komponenty webového grafického rozhraní poskytujícího funkcionality BI, tj. Superset a ještě podpůrné komponenty Tez, DataFu a Livy.

Stejně jako předchozí distribuce frameworku Apache Hadoop i HDP poskytuje komplexní množinu projektů, jejichž prostřednictvím je možné pokrýt celý proces zpracování big data. Pro první majoritní verzi distribuce přitom bylo typické, že společnost Hortonworks produkovala pouze distribuci HDP sestávající výhradně z open source projektů, které byly veřejně dostupné, stejně jako výsledná distribuce. V rámci druhé majoritní řady nicméně společnost začala postupně vytvářet i produkty, které nebyly součástí HDP a/nebo byly dostupné pouze pro zákazníky s placenou podporou ze strany společnosti Hortonworks. Jednalo se především o produkty (Hortonworks, 2017c) SmartSense poskytující proaktivní monitoring a maintenance clusteru HDP, DataPlane Platform pro nasazení Data Lifecycle Manager umožňujícího zajistit replikaci a disaster recovery big data napříč clustery HDP a Data Steward Studio pro snazší porozumění, správu a zabezpečení dat v big data clusterech, dále také Data Analytics Studio pro optimalizaci využití Hive a ještě také Data Platform Search pro indexování a full-textové vyhledávání nad daty komponent HDFS, Hive, HBase a Spark.

Obr. 4-10 Grafické rozhraní NiFi



Zdroj: data a zpracování (Liu, 2018)

Významný produkt, který společnost na trh uvedla, představuje platforma pro příjem, zpracování a analýzu proudů big data, Hortonworks DataFlow. Nejedená se nicméně o samostatný projekt, který by zastřešovala ASF, ale naopak o ucelenou kolekci projektů,

především (Hortonworks, 2017b) Ambari, NiFi, Kafka, Storm, atd. Význam HDF dobře ilustruje také skutečnost, že společnost Cloudera, jak již bylo uvedeno, tento produkt převzala a dále jej nabízí pod vlastním označením CDF. Obr. 4-10 znázorňuje grafické rozhraní NiFi pro definici, správu a monitoring flow příjmu a zpracování proudů dat, které je součástí HDF.

Také HDP, stejně jako předchozí distribuce, představuje plnohodnotnou distribuci frameworku Apache Hadoop poskytující sadu nástrojů, jejichž prostřednictvím je možné komplexně zabezpečit proces příjmu, zpracování, uložení, analýzy i vizualizace big data. Na rozdíl od předchozích distribucí se tvůrci HDP při vzniku distribuce HDP snažili vytvořit 100% otevřené volně dostupné řešení. V průběhu pozdějšího rozvoje distribuce byli nicméně nutenci nově vytvářené produkty zařazovat do nabídky pro zákazníky s placenou podporou. Kromě HDP tak společnost Hortonworks na trh uvedla řadu produktů, které dále rozšiřují funkcionality distribuce HDP a také samostatný produkt HDF, který do svého portfolia později zařadila společnost Cloudera.

4.4.3 Model poskytování distribuce

Jak již bylo uvedeno, společnost Hortonworks se od svého vzniku soustředila na vytvoření 100% open source distribuce frameworku Apache Hadoop. Tato filozofie se promítla do modelu poskytování distribuce, kdy samotná distribuce byla volně ke stažení bez omezení. Pro zákazníky byla k dispozici placená podpora různých úrovní determinujících příslušnou úroveň poskytovaných služeb. Již v roce 2011 měli zákazníci možnost využít těchto úrovní předplatného podpory od společnosti Hortonworks (Hortonworks, 2011):

- Hortonworks Enablement Support Subscription – podpora v procesu návrhu, vývoje a tvorby prototypu clusteru HDP před vlastním nasazením,
- Hortonworks Production Support – Basic Edition – podpora pro testovací a produkční prostředí clusteru HDP pro nekritická nasazení,
- Hortonworks Production Support – Enterprise Edition – podpora 24x7 a reakční doba do 1 hodiny nad rámec edice předplatného basic, a to pro veškerá i kritická nasazení.

Tento model byl zachován (Hortonworks, 2012b) i v roce 2012, pouze s drobnými změnami v názvech jednotlivých úrovní podpory a byl rozšířen o tzv. Hortonworks Support Starter Pack určený pro krátkodobou asistenci při vývoji, testování a nasazování Hadoop aplikací.

Uvedený model zůstal nezměněn i v následujícím roce, ale v roce 2014 přišla další změna, která restrukturalizovala úrovně předplatného podpory společnosti Hortonworks do dvou edic, a to (Hortonworks, 2014a) Enterprise a Enterprise Plus. Obě dvě úrovně předplatného zahrnovaly (Hortonworks, 2014a) podporu 24x7 s reakční dobou 1 hodina/4 hodiny/1 pracovní den v závislosti na závažnosti problému, přístup do zákaznického portálu, znalostní databáze a k dalším podpůrným materiálům, včetně software. Uvedené úrovně podpory se lišily také (Hortonworks, 2014a) podporou komponent, kdy verze Enterprise zahrnovala v rámci HDP podporu komponent Hadoop, Tez, Hive, Pig, Sqoop, Flume, Mahout, Ambari, Oozie, Falcon, Knox, HBase a Phoenix a verze Enterprise Plus navíc zahrnovala ještě podporu komponent Accumulo, Storm, Ranger, Spark a Kafka.

Na konci roku 2015 byl model poskytování předplatného distribuce HDP opět změněn a nově zahrnoval dokonce čtyři edice předplatného, mezi které patřily (Hortonworks, 2015b) HDP Jumpstart, HDP Enterprise, HDP Enterprise Plus a HDP Enterprise Premier. Jednotlivé úrovně se lišily (Hortonworks, 2015b) dobou trvání kontraktu na předplatné, rozsahem podpory (8x5, 24x7), reakční dobou pro jednotlivé úrovně závažnosti problému a rozsahem podporovaných komponent, podobně jako u předchozího modelu. Současně byly stanoveny (Hortonworks, 2015a) dostupné úrovně předplatného pro HDF, a to HDF Jumpstart a HDF Enterprise, které se lišily podobně jako v případě edic předplatného podpory HDP, s tím rozdílem, že obě úrovně podpory HDF podporovaly celý stack HDF.

V následujícím roce byla (Hortonworks, 2016a) úroveň podpory HDP Enterprise Premier zrušena, zatímco úrovně podpory pro HDF modifikovány nebyly. V dalších letech úrovně předplatného neprošly žádnými výraznými proměnami. I po sloučení Cloudera a Hortonworks jsou stále nabízeny (Cloudera, 2019k) úrovně předplatného HDP Jumpstart, HDP Enterprise, HDP Enterprise Plus a HDF Jumpstart a HDF Enterprise, přičemž nově jsou pro jednotlivé produkty stanovena také data konce podpory, např. prosinec 2021 v případě HDP 3.1 a březen 2022 v případě HDF 3.4.

V předchozím textu již bylo uvedeno, že distribuci HDP je možné nasazovat v prostředí cludu. Ačkoli ceny za předplatné podpory nebyly nikdy na webových stránkách společnosti veřejně dostupné, lze například uvést, že v případě AWS se cena za využití distribuce HDP pohybuje od 0,10\$ do 0,60\$ za hodinu na každém jednom uzlu clusteru v závislosti na velikosti dané instance, tedy od 72\$ do 432\$ za každý jeden uzel clusteru za měsíc, nad rámec nákladů na infrastrukturu jako takovou. Stejně jako ostatní tvůrci distribuce Hadoop, také společnost Hortonworks si během doby existence zajistila rozsáhlou síť partnerů z řad zákazníků i dodavatelů.

4.5 Shrnutí

V předchozím textu byly podrobně charakterizovány vznik, rozvoj, architektura, funkcionalita a model poskytování distribuce frameworku Apache Hadoop CDH, MDP a HDP. Přestože zkoumané distribuce nebyly na trh uvedeny simultánně (CDH v roce 2008, MDP 2001, HDP 2012), všechny se na trhu postupně etablovaly a jejich tvůrcům se dařilo získávat zákazníky a zvyšovat tržní podíl. Každá z těchto distribucí přitom vykazovala určitá specifika, a to nejen z hlediska architektury a funkcionality, ale také modelu poskytování zákazníkům.

Distribuce CDH byla od samého počátku koncipována jako komerční distribuce, jejíž určitá část je k dispozici zdarma a za vybrané komponenty a pokročilé funkce musí zákazníci připlatit. Společnost Cloudera přitom kromě proprietárních komponent platformy vyvinula řadu projektů, které později předala komunitě a zůstala významným přispěvatelem i do jejich open source formy.

Naproti tomu, společnost MapR Technologies vytvořila ryze komerční proprietární jádro distribuce, které díky reimplementaci klíčových komponent (především HDFS -> MapR-FS, HBase -> MapR-DB, Kafka -> MapR-ES) umožnilo dosahovat vyšší výkonnéosti při

zpracování big data, a to zejména vlivem eliminace vrstev, jakou jsou JVM či lokální souborové systémy. K tomuto jádru společnost v rámci MDP přidala množinu zavedených open source projektů. Přestože distribuce MDP byla dostupná pro vyzkoušení zdarma, její použití pro komerční účely bylo vždy možné pouze v případě úhrady licence.

Distribuce HDP byla na trh uvedena jako poslední, nicméně společnosti Hortonworks se dařilo s tímto produktem rychle získávat široké řady zákazníků, a to především díky snadné dostupnosti, jelikož distribuce HDP byla od samého vzniku koncipována jako 100% otevřená a volně dostupná. HDP vždy obsahovala množinu open source projektů. Také společnost Hortonworks se stala významným přispěvatelem do rozvoje open source projektů Hadoop a ekosystému Hadoop. Později však byla společnost Hortonworks nucena, s cílem zlepšit hospodářské výsledky, nové produkty a také některé pokročilé funkce z distribuce HDP izolovat a poskytovat je za úhradu, především v rámci předplatného.

V letech 2017 a 2018 se na trhu distribucí Hadoop odehrála zcela zásadní změna, když všechny zmíněné společnosti zaznamenaly drastický propad v tržbách a vykazovaly velice špatné hospodářské výsledky. Hlavní příčinou byla zřejmě skutečnost, že zpracování big data se do značné míry přesunulo do prostředí cloudových platform AWS, GCP a Azure a producentům distribucí frameworku Hadoop se nejen nedařilo získávat nové zákazníky, ale navíc začaly výrazně ztrácet zákazníky stávající.

Celý trh distribucí Hadoop se nejprve konsolidoval na tři hlavní zmíněné producenty. Následně došlo ke sloučení společností Cloudera a Hortonworks a akvizici MapR Technologies ze strany HPE. Všechny zkoumané distribuce jsou sice stále nabízeny a dostupné, nicméně společnost Cloudera přikročila k omezení veřejného přístupu k repositářům distribucí od verze HDP 3.1.5 a Cloudera 6.3.3. Současně společnost Cloudera propaguje jednotnou platformu Cloudera Data Platform, která má integrovat to nejlepší z distribucí CDH a HDP, nicméně je rovněž orientována komerčně. V budoucnu tedy potenciálně hrozí, distribuce CDH a HDP na úkor řešení Cloudera Data Platform zaniknou a především, že k dispozici nebude žádná aktuální verze plnohodnotné distribuce frameworku Hadoop, která by byla veřejně a bezplatně dostupná a bylo by možné ji použít pro komerční účely. S ohledem na charakterizovanou situaci na trhu distribucí frameworku Hadoop byly pro realizaci praktické úlohy zpracování big data zvoleny distribuce CDH, MDP a HDP.²³

²³⁾ S ohledem na aktuální situaci na trhu distribucí frameworku Apache Hadoop, kdy naprostě majoritní podíl na trhu vykazují tři největší distribuce, mezi které patří Cloudera, Hortonworks a MapR, nebyla pro praktickou část práce provedena selekce distribucí frameworku Apache Hadoop dle předem stanovených kritérií. Naopak, pro realizaci typové úlohy zpracování big data v reálném čase byly bez dalšího přímo zvoleny uvedené distribuce.

Kromě uvedených distribucí je pro účely zpracování big data používána také distribuce společnosti Amazon (Elastic MapReduce – EMR), která má nezanedbatelný podíl na trhu, nicméně je poskytována výlučně v tzv. managed verzi, pouze v rámci veřejného cloutu společnosti Amazon, tj. AWS, zatímco ostatní distribuce lze nasazovat a využívat vlastními silami nebo formou služby zajišťované ze strany množství různých poskytovatelů. Z uvedených důvodů nebyla distribuce EMR do množiny komparovaných distribucí frameworku Apache Hadoop zařazena.

5 Zpracování big data v reálném čase

Tato kapitola obsahuje charakteristiku způsobu, průběhu a výsledků realizace praktické úlohy zpracování big data v reálném čase, konkrétně analýzy sentimentu nad daty získanými ze sociální sítě Twitter v reálném čase. Cílem kapitoly je na konkrétním příkladu ověřit možnosti a vlastnosti jednotlivých distribucí frameworku Apache Hadoop při reálném zpracování dat vykazujících charakteristiky big data a získat podklady pro vlastní komparaci jednotlivých distribucí Hadoop.

V rámci jednotlivých podkapitol bude postupně popsán způsob a průběh jednotlivých fází realizace praktické úlohy zpracování big data, od nasazení jednotlivých distribucí frameworku Apache Hadoop, přes zajištění zdroje big data, příjem získaných big data, až po vlastní analýzu sentimentu nad získanými daty v rámci jednotlivých distribucí frameworku Apache Hadoop, jejichž komparace je předmětem této diplomové práce.

5.1 Nasazení distribuce frameworku Apache Hadoop

5.1.1 Proces nasazení distribuce frameworku Apache Hadoop

Pro zajištění možnosti realizace zpracování big data v reálném čase je nutné v první řadě provést nasazení vlastní distribuce frameworku Apache Hadoop, v jejímž prostředí budou následně data přijata a analyzována.

Proces nasazení jednotlivých komparovaných distribucí frameworku Apache Hadoop se napříč jednotlivými distribucemi v základních rysech v podstatě nelíšil a v souladu s pokyny pro instalaci distribuce (Cloudera, 2019f) v konkrétních podmínkách realizované typové úlohy zahrnoval následující fáze:

1. nasazení strojů tvořících cluster distribuce Hadoop,
2. konfigurace a další příprava strojů clusteru distribuce Hadoop,
3. nasazení a spuštění instalátoru distribuce Hadoop,
4. instalace distribuce Hadoop,
5. validace úspěšného nasazení distribuce Hadoop.

Na podrobnější úrovni detailu se proces instalace napříč jednotlivými distribucemi frameworku Apache Hadoop samozřejmě v určitých nuancích liší.²⁴⁾ V rámci realizace typové úlohy zpracování big data v reálném čase byly jednotlivé distribuce frameworku Apache Hadoop, a tedy také clustery strojů distribuce Hadoop, nasazovány nikoli

²⁴⁾ Podrobný popis nasazení jednotlivých distribucí zde uveden nebude. Příslušné detaily je možné nalézt v dokumentaci jednotlivých distribucí, jež je v případě všech komparovaných distribucí dostatečně podrobná. V dalším textu této podkapitoly budou pouze vyzdvíženy nejdůležitější zjištěné poznatky a/nebo významné rozdíly v procesu nasazení vybraných distribucí Hadoop.

souběžně, ale postupně (vždy byla nasazena pouze jedna distribuce a po realizaci typové úlohy nad danou distribucí byla tato odstraněna a nasazena další), a to z důvodů, které byly popsány v kapitole Předpoklady a omezení práce.

V prvním kroku byla vždy nasazena množina strojů s parametry, které jsou uvedeny v Tab. 5.1, přičemž všechny uzly byly nasazeny paralelně. Před nasazením druhé a třetí instance bylo pochopitelně nutné vždy odstranit předchozí instanci clusteru. Virtuální stroje byly nasazovány v rámci veřejného cloutu, automatizovaně, kdy byly pouze specifikovány požadované parametry. Vlastní nasazení (provisioning) bylo realizováno automatizovaně a zahrnovalo především vytvoření a spuštění virtuálního stroje s definovanými parametry, instalaci operačního systému, konkrétně serverové edice Ubuntu 16.04 LTS (long-term support) Xenial Xerus pro 64-bitové procesory a uložení SSH (SSH – secure shell) klíčů zajišťujících možnost přihlašovat se k serverům pomocí daného SSH klíče bez potřeby zadávat heslo.

Tab. 5.1 Parametry clusteru pro nasazení distribuce frameworku Apache Hadoop²⁵

Typ uzlu	vCPU	RAM (GB)	HDD (GB)	Počet uzlů
Řídicí (master)	8	16	50	2
Pracovní (worker)	2	8	100	3
Celkem	22	44	400	5

Zdroj: vlastní zpracování

Konfigurace připravených virtuálních strojů byla provedena dle oficiální dokumentace pro instalaci jednotlivých distribucí frameworku Apache Hadoop, které byly předmětem komparace. Tento krok zahrnoval především (Cloudera, 2019f; Hortonworks, 2019a; MapR, 2019h) další konfiguraci cílového prostředí clusteru (např. nastavení maximálního počtu paralelně otevřených souborů, který je pro clustery Hadoop doporučován vyšší než defaultní a instalaci specifických závislostí dané distribuce frameworku Apache Hadoop), konfiguraci síťových jmen (více viz v následujícím textu), instalaci a konfiguraci služby pro synchronizaci času pro zajištění minimální odchylky mezi serverovým časem na jednotlivých strojích v clusteru (ve všech případech byla použita služba `ntpd`), povolení/otevření portů pro jednotlivé služby a komponenty v clusteru Hadoop (jedná se o desítky portů), nastavení repositářů distribuce frameworku Apache Hadoop.

V rámci přípravy prostředí je tak nezbytné v případě, že je to vyžadováno, zajistit (Cloudera, 2019f; Hortonworks, 2019a; MapR, 2019h) nasazení DBMS, který je zpravidla vyžadován ze strany systému pro nasazení, správu a monitoring clusteru a případně pro některé z komponent tvořící cluster dané distribuce frameworku Apache Hadoop. V rámci realizace typové úlohy zpracování big data v reálném čase byl DBMS nasazen vždy na prvním z řídících uzlů clusteru (konkrétní použité typy a verze DBMS jsou uvedeny v Tab.

²⁵) V případě distribuce MDP bylo nutné navýšit operační paměť na pracovních uzlech na 16 GB a všem uzlům v clusteru přidat ještě jeden disk o stejné velikosti dedikovaný pro MapR-FS, protože v opačném případě by nebylo možné distribuci nasadit. Zdroje dostupné pro komponenty využité v rámci realizace praktické úlohy byly nicméně v rámci nasazené distribuce MDP omezeny tak, aby byla zachována možnost komparace a vypořídací hodnota zjištěných výsledků.

5.2), na rozdíl od produkčních clusterů, v jejichž případě je doporučeno nasazení dedikovaného DBMS, případně ještě v konfiguraci HA a/nebo zcela mimo cluster Hadoop.

Všechny kroky uvedené v instalačním manuálu jsou důležité, nicméně jako naprosto kritická se jeví konfigurace názvů serverů, kdy pro úspěšnou instalaci a provoz clusteru Hadoop je naprosto nezbytné (Cloudera, 2019f; Hortonworks, 2019a; MapR, 2019h), aby se název hosta (hostname) vždy shodoval s plně kvalifikovaným doménovým jménem (fully qualified domain name – FQDN).²⁶ Pokud je toto zajištěno, pak lze instalaci úspěšně dokončit, dokonce i včetně automatické instalace agentů dané distribuce zajišťujících monitoring a správu jednotlivých uzlů clusteru na jednotlivé uzly clusteru.

V opačném případě, pokud je použito pouze hostname nebo IP adresy (v podobě číselného zápisu formou čtyř oktetů oddělených tečkou, např. 10.1.0.101), se mohou objevit problémy již v průběhu instalace nebo v rámci následujícího provozu a/nebo monitoringu nasazeneho clusteru, kdy i samotný rozdíl ve FQDN je vyhodnocen jako chyba. Validní nastavení doménových jmen tedy představuje jeden ze zásadních předpokladů efektivního provozu clusteru Hadoop. Autor práce má nicméně dobrou zkušenosť i s alternativním přístupem k nastavení resolveru, kdy se i pro loopback rozhraní nastaví doménové jméno (`localhost.localdomain`) a pro všechny uzly clusteru se uvede IP adresa, FQDN a hostname. V takovém případě není nutné nastavovat hostname identické jako FQDN a automatické nasazení agentů i další provoz proběhne z hlediska lokálního systému doménových jmen (domain name system – DNS) bez problémů.

Po dokončení konfigurace a přípravy jednotlivých strojů tvořících cluster bylo následně provedeno nasazení a spuštění instalátoru distribuce frameworku Apache Hadoop. Každá z vybraných distribucí disponuje vlastním instalátorem s grafickým rozhraním, přičemž dostupnost instalátoru, respektive programu, zajišťujícího nasazení, ale také pozdější správu a monitoring clusteru Hadoop, představuje jednu z významných přidaných hodnot distribucí Hadoop. Instalátor byl nasazen vždy na prvním z řídících uzlů clusteru (v případě rozsáhlějších produkčních clusterů není výjimkou, že instalátor bývá nasazován na samostatný dedikovaný stroj). Zatímco v případě distribucí CDH a HDP jsou k dispozici balíky instalátoru, které se manuálně uloží a nainstalují na cílový stroj, v případě MDP je nejprve nutné stáhnout a spustit skript (`mapr-setup.sh`), který zajistí nasazení grafického rozhraní průvodce nasazením clusteru. Nasazený instalátor vždy běží na daném stroji a poslouchá na určitém portu. Spuštěn je zpravidla automaticky, takže následně už stačí pouze do internetového prohlížeče zadat adresu daného serveru, včetně příslušného portu a přihlásit se jako administrátor pomocí nastavených nebo defaultních přihlašovacích údajů.

²⁶⁾ V operačním systému Linux se může název hosta (hostname) od FQDN lišit. Příkladem názvu hosta může být `cdh_6_2_0_master_1`. FQDN, jak název napovídá, pak zpravidla zahrnuje kromě hostname ještě doménové jméno, které je od názvu hosta odděleno tečkou. Příkladem FQDN tedy může být `cdh_6_2_0_master_1.example.com` (host `cdh_6_2_0_master_1` na doméně `example.com`). Plně kvalifikované doménové jméno je možné nastavit i v případě, že žádnou doménu k dispozici nemáme, a to tak, že libovolnou vybranou doménu (unikátní v rámci dané sítě), např. `cdh_6_2_0_master_1.hadoop` nastavíme v lokálním DNS resolveru (např. v případě distribuce Ubuntu v souboru `/etc/hosts`). Shodnou hodnotu hostname a FQDN pak lze, např. v případě distribuce Ubuntu zajistit prostřednictvím příkazu `hostnamectl set-hostname $HOSTNAME.example.com`.

Vlastní instalace/nasazení distribuce Hadoop se napříč zvolenými distribucemi frameworku Apache Hadoop rámcově příliš neliší a zahrnuje ve všech případech především následující kroky nasazení distribuce frameworku Apache Hadoop (Cloudera, 2019f; Hortonworks, 2019a; MapR, 2019h):

1. specifikace názvu clusteru distribuce Hadoop,
2. specifikace množiny uzlů tvořících cluster distribuce Hadoop,
3. volba repositářů, které budou použity pro instalaci distribuce Hadoop,
4. instalace agentů distribuce Hadoop na definované uzly clusteru,
5. výběr služeb (komponent) distribuce Hadoop a specifikace jejich rozmístění v clusteru, včetně konfigurace,
6. specifikace a nastavení přístupových údajů ke službám (komponentám) distribuce Hadoop,
7. instalace zvolených služeb (komponent) distribuce Hadoop ve specifikovaném rozsahu a konfiguraci v clusteru.

Průvodce instalací a proces realizace instalace se napříč komparovanými distribucemi frameworku Apache Hadoop spíše než v rámcovém postupu liší v instalovaných komponentách a případně dalších nuancích. Všechny komparované distribuce poskytují grafického průvodce (wizard) instalací. Jedná se o Apache Ambari v případě distribuce HDP, distribuce CDH je instalována prostřednictvím Cloudera Manager a pro distribuci MDP je k dispozici instalátor MapR Control System. Zatímco Apache Ambari představuje open source systém (ač původně vyvinutý speciálně pro účely distribuce HDP), instalátory distribucí CDH a MDP lze označit za proprietární.

Tab. 5.2 Verze klíčových nasazených komponent clusteru distribuce Hadoop

	CDH	HDP	MDP
Operační systém	Ubuntu 16.04 LTS	Ubuntu 16.04 LTS	Ubuntu 16.04 LTS
Jádro OS	4.4.0-169-generic	4.4.0-169-generic	4.4.0-169-generic
Java	1.8.0	1.8.0	1.8.0
DBMS	PostgreSQL 10. 10	PostgreSQL 9.5	MySQL 5.7
Distribuce Hadoop	6.2.0	3.1.0	6.1.0
Hadoop	3.0.0	3.1.1	-
Kafka	2.1.0	2.0.0	4.1.0
Spark	2.4.0	2.3.2	2.3.2

Pozn: Distribuce CDH byla nasazena v edici Express, MDP v edici Community.

Zdroj: vlastní zpracování

Pro všechny zmíněné instalátory, respektive programy je nicméně symptomatické, že po provedení nasazení clusteru jsou i nadále využívány, jelikož následně v průběhu celého životního cyklu clusteru poskytují velice přínosnou funkcionality pro správu a monitoring clusteru dané distribuce Hadoop. Také jsou využívány v procesu aktualizace (upgrade) jednotlivých komponent a/nebo celé distribuce frameworku Apache Hadoop a případně také při nasazování/odstranění vybraných služeb/komponent clusteru Hadoop. Nejedná se tedy o instalátory v pravém slova smyslu, ale spíše systémy pro správu distribuce

clusteru Hadoop, které z hlediska funkcionality pokrývají i fázi nasazení distribuce frameworku Apache Hadoop.

V rámci realizace typové úlohy zpracování big data v reálném čase byly prostřednictvím zmíněných instalacích nástrojů nasazeny verze distribuce frameworku Apache Hadoop, které jsou uvedeny v Tab. 5.2. Zde jsou také uvedeny verze hlavních komponent použitých v rámci realizace praktické úlohy zpracování big data. Pro úplnost je nezbytné ještě uvést, že služba Kafka byla v rámci clusteru nasazena vždy v počtu jedné instance v rámci druhého řídícího uzlu. Na identickém uzlu byly nasazeny, rovněž v počtu jedné instance, komponenty služby Spark. Klientské komponenty služby Spark byly nasazeny na všech uzlech clusteru. YARN NodeManager byl nasazen na všech pracovních uzlech clusteru, aby bylo možné realizovat úlohy Spark distribuovaně v režimu cluster právě s využitím správce zdrojů YARN.

Uvedené verze distribuce HDP ani CDH nepředstavovaly v době vzniku této práce poslední vydané stabilní verze. K dispozici byla verze HDP 3.1.4, která nicméně byla dodávána defaultně s Ambari 2.7.4 navázáném na komponentu Ambari Infra, založenou na Solr 7.5.7 a způsobující vlivem zdokumentované chyby kontinuální 100% vytížení CPU determinující nestabilitu uzlů, kde byla nasazena instance Ambari Infra. Dále byla k dispozici verze HDP 3.1.5 a CDH 6.3.3, jejichž repositáře ale nebyly veřejně dostupné. Proto byla použita poslední veřejně dostupná stabilní verze HDP 3.1.0, poslední veřejně dostupná stabilní verze CDH 6.2.0 a poslední veřejně dostupná stabilní verze MDP 6.1.0.

Všechny uvedené distribuce frameworku Apache Hadoop poskytují nástroje pro realizaci typové úlohy zpracování big data v reálném čase. Jedná se především o služby Hadoop, Kafka, Spark (v případě distribuce MDP o relevantní substituenty charakterizované v rámci příslušné předchozí podkapitoly věnované dané distribuci Hadoop a nahrazující a/nebo kompatibilní s danými službami). Podrobné srovnání jednotlivých komponent, které tvořily jednotlivé distribuce v průběhu nasazení jednotlivých clusterů, je k dispozici v rámci kapitoly Komparace vybraných distribucí frameworku Apache Hadoop.

5.1.2 Nasazení distribuce Cloudera's Distribution including Apache Hadoop

Jak již bylo uvedeno, v rámci realizace praktické úlohy zpracování big data v reálném čase byla nasazena distribuce CDH ve verzi 6.2.0. Instalátor distribuce CDH je na rozdíl od ostatních komparovaných distribucí Hadoop rozdělen do tří sekcí.

V první části následuje po uvítací obrazovce akceptace licence distribuce, bez níž není možné instalaci dokončit. Poté uživatel volí požadovanou edici (Express, Enterprise Trial, Enterprise). K dispozici je přehledná tabulka poskytující srovnání funkcí dostupných v rámci jednotlivých verzí. V rámci realizace praktické úlohy zpracování big data byla zvolena a nasazena edice CDH Cloudera Express.

V rámci druhé sekce probíhá příprava prostředí clusteru pro distribuci. Také zde je uživateli nejprve prezentována uvítací obrazovka. V dalším kroku uživatel specifikuje název clusteru. Následně je nutné specifikovat uzly, které budou cluster tvořit. Stejně jako v případě ostatních komparovaných distribucí Hadoop je možné zadat konkrétní FQDN

jednotlivých hostů nebo specifikovat rozsah IP adres či jinak definovat vzorec (pattern) pro vyhledání uzlů, což lze využít především v případech, kdy bude cluster tvořen větším počtem uzlů, nikoli pouze v řádu jednotek, ale desítek či stovek až tisíců.

Pokud instalátor specifikované uzly úspěšně kontaktuje, je možné přejít k dalšímu kroku, v jehož rámci uživatel specifikuje repositáře, z nichž bude distribuce nasazena, požadovanou verzi, doplňkové komponenty a způsob instalace. Pro distribuci CDH je totiž symptomatické, že kromě tradičních repositářů s balíky umožňuje nasadit cluster Hadoop také prostřednictvím vlastních balíků (tzv. parcels), které zajišťují mimo jiné možnost snadné aktualizace v rámci dlouhodobého provozu clusteru (tzv. rolling upgrades). V rámci praktické úlohy zpracování big data byla nasazena verze distribuce CDH 6.2.0, a to právě prostřednictvím tzv. parcels.

Dalším krokem specifickým pro distribuci CDH je schválení licence Java Development Kit (JDK). Zatímco v případě ostatních distribucí je nasazení relevantní JDK součástí přípravy jednotlivých uzlů clusteru, instalátor CDH nabízí kromě využití již nainstalované verze JDK také možnost automaticky instalovat Oracle Java SE Development Kit. V rámci realizace byla nicméně pro zajištění srovnatelných podmínek ve všech clusterech Hadoop využita OpenJDK 1.8, instalovaná již v průběhu přípravy prostředí pro clusteru Hadoop.

V dalším kroku uživatel zadá přístupové údaje pro definované uzly clusteru. Stejně jako v ostatních případech byl pro tyto účely využit privátní klíč. Následuje instalace agentů distribuce na jednotlivé uzly. V rámci realizace praktické úlohy tento krok napoprvé selhal, při druhém opakování však proběhl bez problémů. Poté byly v dalším kroku distribuovány balíky (parcels) CDH na jednotlivé uzly clusteru. To zahrnovalo stažení, distribuci na uzel clusteru, rozbalení a aktivaci v rámci daného uzlu. Druhá sekce instalátoru distribuce CDH v posledním kroku ještě zahrnuje validaci síťové konektivity uzlů clusteru a jejich relevantní konfigurace, což v rámci realizace praktické úlohy zpracování big data proběhlo bez chyb a bylo možné přejít k poslední části instalace.

Závěrečná sekce průvodce instalací zahrnuje vlastní nasazení distribuce frameworku Apache Hadoop do připraveného prostředí clusteru. Nejprve uživatel zvolí požadované služby/komponenty distribuce, které mají být nasazeny. V tomto kroku je možné volit z předpřipravených šablon množiny služeb nebo vybrat vlastní specifickou množinu služeb. V rámci realizace praktické úlohy byly nasazeny všechny dostupné komponenty komunitní edice distribuce. V dalším kroku následuje specifikace rozmístění služeb, které jsou v případě distribuce CDH označovány termínem role, napříč jednotlivými uzly clusteru.

Dále uživatel uvede přístupové údaje k podpůrnému DBMS a databázím pro jednotlivé nasazované služby. Pro účely realizace praktické úlohy byl využit v případě distribuce CDH relační DBMS PostgreSQL 10.10. Pak již následovalo schválení nasazení clusteru distribuce CDH ve stanoveném rozsahu, včetně možnosti přizpůsobit konfiguraci jednotlivých služeb. Možnost customizace konfigurace nebyla v rámci realizace praktické úlohy využita. V průběhu instalace distribuce má uživatel možnost sledovat realizaci jednotlivých kroků nasazení clusteru, včetně výsledku. Po úspěšném dokončení instalace se zobrazí shrnutí a uživatel je přesměrován na rozhraní pro správu nasazенного clusteru, tj. Cloudera Manager.

Realizace praktické úlohy zpracování big data zahrnovala nasazení komunitní edice distribuce CDH, což bylo provedeno bez jakýchkoli problémů. Stejně jako v případě distribuce MapR uživatel volí požadovanou edici distribuce. Za specifický lze označit způsob nasazení formou balíčků (parcels), dále odsouhlasení licence používané edice Java (Oracle JDK) a také specifické komponenty/služby, např. Impala nebo Kudu. Celkově je instalátor a celý proces intuitivní a z hlediska struktury lze průvodce instalací označit za nejpodrobnější ze všech komparovaných distribucí frameworku Apache Hadoop.

5.1.3 Nasazení distribuce MapR Data Platform

Jako druhá v pořadí byla nasazena distribuce MDP. V předchozím textu již bylo zmíněno, že na rozdíl od ostatních distribucí se instalátor nenasazuje formou získání a instalace balíčků, ale stažením a spuštěním skriptu (`mapr-setup.sh`). Po přihlášení do instalátoru, respektive MapR Control System, nabídne průvodce instalací uvítací obrazovku, kde jsou shrnuty informace o komponentách distribuce a údaje, které bude uživatel pro instalaci potřebovat.

V druhém kroku instalace uživatel nejprve označí požadovanou verzi distribuce MDP. Dále je, podobně jako v případě distribuce HDP, nutné zvolit instalovanou edici distribuce (Community Edition, Enterprise Edition), přičemž pro instalaci je nezbytné zadat údaje uživatelského účtu u společnosti MapR. V rámci realizace praktické úlohy zpracování big data byla nasazena komunitní edice distribuce MDP ve verzi 6.1.0.

Na rozdíl od ostatních distribucí frameworku Apache Hadoop, které byly předmětem komparace, má uživatel v rámci daného kroku možnost ještě zvolit některou z připravených šablon clusteru MDP, které determinují jak zvolenou množinu komponent, tak jejich rozmístění napříč clusterem uzlů. Šablony jsou koncipovány především s ohledem na určitou oblast využití distribuce (Data Lake, Data Exploration, Operational Analytics). V případě, že uživatel zvolí některou ze šablon, nemusí v rámci dalších kroků volit požadované komponenty a specifikovat jejich požadované rozmístění v clusteru. Naopak jestliže si uživatel z připravených šablon clusteru MDP nevybere, má možnost definovat vlastní množinu požadovaných služeb/komponent a jejich distribuci napříč clusterem. Pro účely praktické úlohy zpracování big data byly požadované komponenty zvoleny manuálně, žádná z připravených šablon využita nebyla.

V rámci třetího kroku uživatel specifikuje, stejně jako v případě ostatních distribucí, přístupové údaje k požadovaným podpůrným službám, především DBMS (defaultně je podporován MySQL). Současně je zde možné zadat heslo pro administrátora clusteru distribuce MDP a nastavit název cílového clusteru. Jako podpůrný DBMS byl zvolen MySQL ve verzi 5.7 (sdílená nikoli embedded instance), jehož nasazení zajistil instalátor.

Následující krok zahrnuje především specifikaci množiny uzlů, které budou tvořit cluster a přístupových údajů k této uzlům. Stejně jako v případě ostatních distribucí bylo použito SSH klíčů pro uživatele `root`. Tento krok se od průběhu nasazení ostatních distribucí prakticky nelišil. Na rozdíl od ostatních distribucí bylo ještě před zahájením instalace, v rámci přípravy jednotlivých uzlů, nutné vytvořit na všech strojích uživatele pro administraci clusteru, tj. uživatele `mapr`, včetně příslušné uživatelské skupiny.

V rámci daného kroku je vyžadována také konfigurace disků, kdy uživatel specifikuje disky, které budou dedikovány pro MapR-FS. Musí se přitom jednat o nepřipojené disky bez jakéhokoli oddílu (partition), např. `/dev/sdb`. Jestliže uživatel tuto konfigurační položku nezadá nebo uvede odkaz na disk, na kterém existuje nějaký oddíl (například oddíl obsahující operační systém), validace v následujícím kroku selže a není možné v dalším nasazení pokračovat. Tato situace při nasazení clusteru MDP nastala a bylo tudíž nutné všem 5 strojům přidat ještě druhý disk o stejné velikosti jako kořenový disk.²⁷ V praxi je možné přidat více disků, vyžadován je minimálně jeden. Distribuce MDP tak z daného hlediska vyžaduje striktní oddělení dat operačního systému, potažmo distribuce a distribuovaného souborového systému, tedy datových oddílů.

V dalším kroku instalátor provedl validaci specifikovaných strojů, přičemž tato v rámci realizace praktické úlohy několikrát selhala a musely být provedeny požadované úpravy v konfiguraci prostředí, aby proběhla bez nedostatků. Následně již byla pouze upřesněna volba požadovaných komponent a jejich rozmístění v clusteru. Pokud jde o konfiguraci jednotlivých služeb, bylo použito připravené defaultní nastavení. V průběhu instalace nebyly zaznamenány žádné chyby, ale instalace trvala přibližně o třetinu déle než v případě ostatních komparovaných distribucí.

Po dokončení instalace byla nabídnuta aplikace zvolené licence v rámci nasazeného clusteru, představující zcela ojedinělý úkon, který v rámci instalace ostatních komparovaných distribucí nebylo nutné realizovat. Jelikož byla zvolena komunitní edice MDP, nebylo nutné licenci zadávat, pouze bylo zobrazeno upozornění, že funkcionalita nelicencované distribuce může být omezena. V rámci shrnutí průběhu instalace nabízí závěrečná obrazovka průvodce instalací odkazy na klíčová rozhraní pro správu nasazeného clusteru (MapR instalátor, MapR Control System, Hue).

V rámci realizace praktické úlohy zpracování big data byla s menšími obtížemi nasazena komunitní editace distribuce frameworku Apache Hadoop společnosti MapR. Zde je nutné konstatovat, že po nasazení se ukázalo, že celé prostředí bylo pro nasazení MDP dostačující pouze částečně a spíše jej lze označit za poddimenzované. Jelikož první řídící uzel maximálně využívala komponenta CLDB nezbytná pro bezproblémový chod celé distribuce, bylo nutné vybrané komponenty, které nebyly zapojeny do realizace praktické úlohy zpracování big data vypnout. Také z hlediska systému pro správu a monitoring provozu distribuce MDP se nepodařilo celé prostředí dostat takzvaně do „zelených“ hodnot monitorovaných provozních metrik ani do 100% vyhovující konfigurace, kterou doporučuje MDP. Pro nasazení MDP lze tedy doporučit prostředí s vyššími parametry, a to především jeden nebo více dedikovaných uzlů pro hostování core služeb (počet řídích uzlů by měl být minimálně 3) a také vyšší úroveň dostupné paměti a případně více CPU, aby byl zajištěn bezproblémový provoz.

²⁷⁾ V případě distribuce MDP bylo nutné poskytnout více diskového místa než pro distribuce CDH a HDP. Bez zajištění dodatečných diskových oddílů by nebylo možné MDP nasadit a v důsledku ani realizovat typovou úlohu zpracování big data. Objem dostupného diskového prostoru neměl žádný vliv zjištěné výkonnostní metriky ani kritéria komparace a srovnatelnost hodnocení jednotlivých komparovaných distribucí byla zajištěna v maximální možné míře.

Průvodce i celý proces nasazení lze označit za vcelku přímočaré. Klíčovými specifiky v porovnání s ostatními komparovanými distribucemi jsou v této rovině především nasazované komponenty, zejména substituenty klíčových služeb Hadoop, respektive jejich proprietární reimplementace a také komponenty nasazované za účelem podpory monitorování provozu clusteru. Do množiny těchto nástrojů spadá v případě distribuce MDP především Collectd, Elasticsearch, Fluentd, Grafana, Kibana, Open STDB. Rovněž je pro MDP symptomatická povinnost využít minimálně 2 disků (alespoň jeden prázdný disk musí být dedikován pro distribuovaný souborový systém), mandatorní minimum 16 gigabajtů paměti na každém z uzlů clusteru a v neposlední řadě nabídka aktivace licence pro cluster.

5.1.4 Nasazení distribuce Hortonworks Data Platform

Poslední distribucí, která byla v rámci realizace praktické úlohy zpracování big data nasazena, byla Hortonworks Data Platform. V prvním kroku vyzve průvodce uživatele k zadání názvu clusteru, který je předmětem nasazení. Dále uživatel specifikuje požadovanou verzi distribuce a repositáře, které budou pro instalaci použity (lokální nebo vzdálené veřejně dostupné). V rámci realizace praktické úlohy byla zvolena verze HDP 3.1.0 instalovaná z veřejně dostupných repositářů. Lokální repositáře jsou vhodné například v situacích, kdy je nasazován větší počet clusterů a/nebo kdy stroje, na které je nasazován Hadoop nemají přístup k internetu (který je nicméně nezbytný pro přípravu strojů – aktualizace, instalace podpůrných komponent/služeb, aj., pokud i v takovém případě není použito lokálních repositářů).

V rámci třetího kroku průvodce instalací uživatel specifikuje množinu strojů, na které bude distribuce nasazena a současně přístupové údaje k těmto strojům. Uživatel má možnost SSH klíč nezadávat a registraci (zahrnující především instalaci agentů na každý stroj) jednotlivých uzlů provést manuálně. V rámci praktické úlohy byl nicméně, stejně jako v případě ostatních distribucí, zadán SSH klíč a registrace strojů clusteru byla zajištěna automatizovaně.

Následně uživatel v rámci průvodce instalací distribuce HDP zvolí požadované služby distribuce. V rámci dalších dvou kroků průvodce specifikuje rozmístění instalovaných služeb, respektive jejich řídících a pracovních komponent a klientů napříč cílovým clusterem distribuce Hadoop. Stejně jako v případě předchozích komparovaných distribucí frameworku Apache Hadoop byly řídící komponenty nasazeny na řídící uzly a ostatní komponenty na uzly pracovní.

V dalším kroku může uživatel ještě přizpůsobit konfiguraci jednotlivých zvolených služeb dle individuálních požadavků, přičemž stejně jako v případě předchozích nasazovaných distribucí Hadoop nebyla tato možnost v rámci realizace praktické úlohy zpracování big data využita. Pouze byly specifikovány přístupové údaje pro DBMS, kterým byl v případě distribuce Hadoop HDP PostgreSQL 9.5. Před zahájením instalace je uživateli zobrazeno shrnutí specifikovaných parametrů nasazovaného clusteru. Po jeho potvrzení dojde k zahájení instalace služeb v clusteru. Po dokončení instalace se zobrazí shrnutí výsledku instalace a uživatel je přesměrován do grafického rozhraní pro správu clusteru HDP, jímž je rovněž Ambari.

V rámci realizace praktické úlohy zpracování big data byla nasazena distribuce HDP ve verzi 3.1.0 v plném rozsahu dostupných komponent. Instalace i v tomto případě proběhla bez komplikací. Již v rovině průvodce procesem instalace dané distribuce Hadoop lze konstatovat, že se liší od ostatních komparovaných distribucí, které nejsou historicky do takové míry otevřené (open source). V případě HDP není potřeba v rámci instalace volit edici ani vyjadřovat souhlas s licencí distribuce nebo služeb či komponent. Také v tomto případě lze proces a průvodce nasazením distribuce označit za přehledný a účelný, přičemž v porovnání s ostatními komparovanými distribucemi se jeví jako nejkratší a nejstručnější (nikoli na úkor přehlednosti procesu ani konfigurace cílového prostředí clusteru Hadoop).

Obecně lze říci, že proces a průvodce instalací jsou v případě všech distribucí, které byly předmětem komparace, vyspělé a naprosto dostačující, jelikož uživatele přehledně provedou instalací při zajištění relevantních možností přizpůsobení clusteru Hadoop. Uživatel, který úspěšně realizoval nasazení kterékoli z komparovaných distribucí frameworku Apache Hadoop by neměl mít problém zajistit nasazení jakékoli jiné distribuce Hadoop.

Pro všechny distribuce jsou naprosto specifické především umístění balíků jednotlivých služeb (`/usr/hdp/...`, `/opt/cloudera/...`, `/opt/mapr/...`) a porty využívané pro jednotlivé služby (např. Kafka na portu 9092 v rámci distribuce CDH, 8082 v MDP a 6667 v případě HDP). Distribuce MDP je charakteristická ovládáním a správou jádra vyplývajícím z architektonické podstaty popsané v rámci předchozí kapitoly (služby `mapr-cldb`, `mapr-warden`, `mapr-zookeeper`, atd., klient `maprcli` aj.). Tato distribuce také klade vyšší nároky na parametry dostupného hardware (HW), jelikož vyžaduje minimálně 16 gigabajtů operační paměti u každého uzlu, minimálně 1 dedikovaný disk pro distribuovaný systém, vyšší parametrická náročnost jádra platformy. Pro všechny zkoumané distribuce Hadoop je symptomatické, že disponují vlastním specifickým instalátorem a systémem pro monitoring a správu nasazeneho clusteru, a že ve všech případech jsou kromě „tradičních“ komponent z oblasti big data dostupné rovněž specifické služby/komponenty, které se v ostatních distribucích nevyskytují.

5.2 Zajištění zdroje big data

V rámci kapitoly 2.5.1 byly představeny vybrané zdroje big data. S cílem přiblížit se reálné úloze zpracování big data byla jako zdroj big data pro typovou úlohu zpracování big data zvolena sociální síť, konkrétně Twitter. Tato sociální síť byla vybrána především s ohledem na možnost získání reálných dat (příspěvků ze sociální sítě Twitter, tj. Tweetů) vykazujících charakteristiky big data (především objem a rychlosť vzniku dat) zdarma bez potřeby úhrady předplatného či jednorázového poplatku za přístup k datům.

5.2.1 Registrace vývojářského účtu na platformě Twitter Developer

Pro přístup k datům ze sociální sítě Twitter je nutné mít na vývojářské platformě Twitter (Twitter Developer Platform) registrován vývojářský účet. Uživatel si musí (Twitter,

2019a) o vývojářský účet zažádat (apply), přičemž je konfrontován s množinou případů užití, pro které je zakázáno vývojářský účet využívat. Před podáním žádosti se musí uživatel přihlásit k sociální síti Twitter, takže bez registrace v rámci Twitter není možné žádost o uživatelský účet odeslat. Po přihlášení k sociální síti Twitter je uživatel přesměrován na průvodce žádostí o vývojářský účet na platformě Twitter Developer.

V rámci prvního kroku uživatel nejprve (Twitter, 2019e) v obecné rovině vymezí účel použití nástrojů Twitter pro vývojáře (tzv. případ užití neboli use case), přičemž má možnost zvolit z kategorií, mezi které patří profesionální komerční použití, osobní nekomerční použití, akademické použití pro výuku nebo výzkum a jiné. Pro účely této práce byla zvolena možnost osobního nekomerčního použití za účelem prozkoumání API.

V dalším kroku uživatel zadá (Twitter, 2019e) osobní údaje, především uživatelský účet Twitter a uvede, zda požaduje individuální nebo týmový účet vývojáře Twitter, zadá email, zemi, ve které žije, požadované oslovení a zda si přeje zasílat emailem novinky týkající se Twitter API.

Třetí krok vyžaduje uvedení (Twitter, 2019e) konkrétního popisu zamýšleného použití Twitter API a dat, která jejich prostřednictvím uživatel získá. Nejprve uživatel slovně (minimálně 200 znaků) popíše zamýšlené použití, dále uvede, zda bude získaná data analyzovat či nikoli (v případě, že ano, je nutné detailně popsát jakým způsobem). Podobně je nutné uvést, zda bude využita funkcionalita pro vytváření Tweetů (Tweet), přeposílání Tweetů (Retweet), označování jako „to se mi líbí“ (like), sledování (follow) nebo posílání zpráv (Direct Message) a také, zda uživatel plánuje získané Tweety zobrazovat a/nebo agregovat získaná data mimo Twitter a ještě zda bude výsledek zpřístupněn nějaké vládní instituci (přičemž školské instituce do této kategorie nespadají). Pro účely realizace typové úlohy zpracování big data v reálném čase bylo v této části průvodce specifikováno zamýšlené použití Twitter API pro získání Tweetů týkajících se zkoumaných distribucí Hadoop a realizace analýzy sentimentu nad takto získanými Tweety za účelem komparace jednotlivých distribucí, včetně uvedení nástrojů pro realizaci analýzy sentimentu. Využití ostatních možností nebylo ve formuláři potvrzeno.

V rámci předposledního kroku uživatel (Twitter, 2019e) provede kontrolu zadaných údajů a následně v posledním kroku odsouhlasí podmínky používání služby Twitter Developer (Developer Agreement) a žádost odešle. Odeslaní žádostí je ještě nutné potvrdit kliknutím na odkaz v potvrzovacím emailu, který je po odeslání žádosti zaslán uživateli na emailovou adresu použitou při registraci.

5.2.2 Vytvoření Twitter aplikace

Po úspěšné validaci prostřednictvím odkazu v emailové zprávě je uživatel přesměrován do portálu Twitter Developer a může jej začít prakticky okamžitě využívat. Základní abstrakce v rámci platformy Twitter je reprezentována aplikací Twitter (Twitter app) představující (Twitter, 2019b) „... bránu poskytující množinu autentizačních klíčů a nastavení oprávnění nezbytných pro volání většiny Twitter API.“ S využitím přístupových údajů, které uživatel získá v rámci konkrétní Twitter aplikace lze volat Twitter API a vyhledávat, respektive přijímat požadované příspěvky ze sociální sítě Twitter.

Ve formuláři pro vytvoření nové Twitter aplikace uživatel (Twitter, 2019b) specifikuje název aplikace, popis, webovou adresu (Uniform Resource Locator – URL), zda má aplikace umožňovat přihlášení k sociální síti Twitter, případně autentizační URL pro zpětná volání (callback), volitelně URL, na kterých se nachází podmínky používání služby, politiku ochrany osobních údajů a případně také název organizace, pro kterou aplikaci vytváří, včetně její webové stránky a povinně ještě popis způsobu použití dané Twitter aplikace. Po kliknutí na tlačítko „Vytvořit aplikaci“ musí ještě vývojář znova potvrdit souhlas s podmínkami použití platformy Twitter Developer.

Pro účely realizace typové úlohy zpracování big data byla vytvořena aplikace „Spark Sentiment Analysis App“. Informace týkající se dané Twitter aplikace jsou v rámci zobrazení detailu aplikace rozčleněny do samostatných sekcí, přičemž na první záložce jsou zobrazeny (Twitter, 2019j) základní atributy aplikace, především informace, které uživatel zadal při vytváření Twitter aplikace a je zde také možnost aplikaci editovat nebo odstranit.

Obr. 5-1 Detail Twitter Developer aplikace – přístupové klíče a tokeny

The screenshot shows the Twitter Developer Apps interface. At the top, there's a purple banner with the text "We've launched Twitter Developer Labs! Test new Twitter API previews, provide feedback, and get the latest news on what we're building. [Learn more](#)". Below the banner, the navigation bar includes links for Developer, Use cases, Products, Docs, More, and Labs. The main content area has a breadcrumb trail: Apps > Spark Sentiment Analysis App. There are three tabs: App details (selected), Keys and tokens (current view), and Permissions. The "Keys and tokens" section contains two main sections: "Consumer API keys" and "Access token & access token secret". Under "Consumer API keys", there are fields for "API key" (KgCZQ) and "API secret key" (x67KltZUG0Ty), with a "Regenerate" button below. Under "Access token & access token secret", there are fields for "Access token" (4IlyQhJprlcGo7) and "Access token secret" (3EzbWgw5WJz1), with "Read and write (Access level)" and "Revoke" and "Regenerate" buttons below.

Zdroj: data a zpracování (Twitter, 2019j)

Z pohledu praktického využití aplikace je významná především druhá záložka obsahující (Twitter, 2019j) klíče a tokeny, které uživatel, respektive aplikace použije pro autentizaci při volání Twitter API. Mezi tyto přístupové údaje, které jsou specifické pro každou jednotlivou Twitter aplikaci, patří (Twitter, 2019j):

- klíče konzumenta API (Consumer API keys):
 - klíč API (API key),
 - heslo klíče API (API secret key),
- přístupové tokeny (Access tokens):
 - přístupový token (Access token),
 - heslo přístupového tokenu (Access token secret).

Zatímco oba klíče konzumenta API jsou vygenerovány v rámci procesu vytvoření aplikace Twitter automaticky, vytvoření přístupových tokenů musí uživatel iniciovat ručně kliknutím na příslušné tlačítko na záložce „Klíče a tokeny“ na detailu dané Twitter aplikace. V případě klíčů konzumenta API má uživatel má možnost tyto (Twitter, 2019j) pouze znova vygenerovat, naproti tomu přístupové tokeny může odstranit nebo opětovně vygenerovat, jak je patrné z Obr. 5-1.

Poslední záložka na detailu Twitter aplikace obsahuje (Twitter, 2019j) nastavení oprávnění dané aplikace (defaultně čtení a zápis), přičemž jakákoli změna se promítne do nově vygenerovaných přístupových tokenů. Pro účely typové úlohy zpracování big data byly v rámci Twitter aplikace „Spark Sentiment Analysis App“ vygenerovány přístupové tokeny a ponechána defaultní oprávnění na čtení a zápis.

5.2.3 Výběr relevantního Twitter API

Společnost Twitter v současné době poskytuje celkem (Twitter, 2019g) pět hlavních API, v jejichž rámci je možné volat různé koncové body (endpoint) zprostředkovávající různou podoblast funkcionality a mezi něž patří (Twitter, 2019g):

- Tweets – poskytuje funkcionality pro publikování a správu Tweetů z vlastního profilu, resp. uživatelského účtu a také získání historických Tweetů dávkově nebo v reálném čase,
- Direct Messages – umožňuje publikovat, získat nebo odstranit zprávy (direct message), které si mohou členové sociální sítě Twitter vzájemně posílat,
- Account and users – zajišťuje podporu čtení a editace určité podmnožiny nastavení uživatelského profilu sociální sítě Twitter,
- Metrics – zprostředkuje aktuální a historické statistické údaje týkající se interakce s příspěvky a dalším typem objektů na sociální síti Twitter,
- Ads API – poskytuje funkcionality pro tvorbu a správu reklamních aplikací na Twitteru.

Kromě uvedených hlavních API, která jsou součástí oficiálního ceníku Twitter API, jsou k dispozici ještě doplňková API, především (Twitter, 2019f) API pro nahrávání mediálních souborů (Media), pro sledování trendů v rámci určité lokality a sledování lokalit s populárními trendy (Trends) nebo pro získání informací o lokalitě a získání umístění v rámci lokality (Geo). Dále jsou k dispozici také podpůrné (Twitter, 2019g) nástroje pro vydavatele a SDK, jež však v pravém slova smyslu nepředstavují API.

V rámci uvedených API je možné volat různé koncové body (endpoint) zprostředkovávající specifickou podoblast funkcionality daného API. Aktuální stav jednotlivých koncových bodů je možné sledovat (Twitter, 2019k) na stránce [Twitter API Status](#).

Všechna API a příslušné koncové body jsou poskytovány (Twitter, 2019g) na některé z úrovní služby Standard (k dispozici zdarma), Premium (k dispozici za úhradu dle veřejného ceníku) a/nebo Enterprise (k dispozici pouze na vyžádání na základě kontraktu uzavřeného prostřednictvím obchodního oddělení společnosti Twitter). Některá API, respektive koncové body jsou přitom dostupné (Twitter, 2019g) pouze na jedné úrovni,

jako například Direct Messages (pouze úroveň Standard) nebo Metrics (pouze úroveň Enterprise), zatímco jiná jsou k dispozici ve dvou či třech úrovních, jako například Filter Tweets (koncový bod, který je součástí API Tweets a je dostupný na úrovni Standard a Enterprise) nebo Accounts and users (API dostupné na všech třech úrovních, tj. Standard, Premium i Enterprise).

Pro účely realizace typové úlohy zpracování big data bylo zvoleno Tweets API. Pokud pomineme již zmíněná doplňková API (Media, Trends, Geo), lze konstatovat, že v rámci Tweets API je k dispozici celkem pět oblastí funkcionality, které jsou, včetně dostupnosti napříč zmíněnými úrovněmi poskytované služby, znázorněny v rámci tabulky Tab. 5.3²⁸, přičemž v rámci každé oblasti je dostupný jeden či více koncových bodů, které je možné volat za účelem publikování či získání dat nebo využití jiné funkcionality nad Tweety.

Tab. 5.3 Přehled funkcionality Tweets API

Název	Endpoint	S	P	E
Publish and engage	/statuses, /favorites	✓		
Search Tweets: 7 days	/search/tweets	✓		
Search Tweets: 30 Days	/tweets/search/30day		✓	✓
Search Tweets: Full-archive	/tweets/search/fullarchive		✓	✓
Filter Tweets	/statuses/filter	✓		✓
Sample Tweets	/statuses/sample	✓		✓
Batch Tweets	/historical/powertrack			✓

Pozn: S – Standard, P – Premium, E – Enterprise.

Zdroj: data (Twitter, 2019g; Twitter, 2019h), vlastní zpracování

Standardní API Twitter jsou v současné době k dispozici (Twitter, 2019i) ve verzi 1.1 (<https://api.twitter.com/1.1/>), přičemž při volání se použije URL, již tvoří základní adresa API (Standard nebo Enterprise – viz poznámku pod čarou na této stránce), dále adresa konkrétního endpointu a případě parametry volání API. Příkladem může být volání <https://api.twitter.com/1.1/search/tweets.json?q=#hadoop>, které zajistí získání posledních Tweetů obsahujících hashtag #hadoop.

Z Tab. 5.3 vyplývá, že na úrovni Standard, která je k dispozici zdarma, jsou dostupná pouze čtyři API, dále na úrovni Premium dvě API (respektive jedno API s různými koncovými body) a na úrovni Enterprise celkem pět API. Pro účely typové úlohy zpracování big data lze tedy ihned eliminovat API, jež jsou dostupná pouze na úrovni Enterprise, tj. API Batch Tweets, jehož využití je možné pouze na základě předchozího

²⁸⁾ V tabulce je uvedena hierarchie koncových bodů (endpointů) pouze pro úroveň poskytované služby Standard, které jsou dostupné na standardní adrese Twitter API, tedy pod URL <https://api.twitter.com/1.1/<path/to/api/endpoint>. Výjimku tvoří API Batch Tweets, které je dostupné pouze na úrovni Enterprise, a tedy pod zcela jinou URL adresou, konkrétně <https://gnip-stream.twitter.com/<path/to/api/endpoint>. Funkcionality pro získání Tweetů v reálném čase (Filter Tweets, Sample Tweets) která je poskytována na úrovni Standard, je k dispozici na standardní adrese Twitter API, zatímco v případě Enterprise úrovně služeb v rámci druhé z uvedených URL.

jednání s obchodním oddělením a uzavření příslušného smluvního konaktu. Pro praktickou část této práce není vhodné ani API Publish and engage, jelikož neslouží k získávání dat, ale k jejich publikování.

API Search Tweets umožňuje (Twitter, 2019i) získávat historické příspěvky ze sociální sítě Twitter v horizontu posledních 7 dní, 30 dní nebo celého archivu, přičemž na úrovni Standard jsou k dispozici pouze data za posledních 7 dní. Jedná se přitom o klasické volání typu požadavek – odpověď, kdy odpověď obsahuje požadovaná data. Naproti tomu, Filter Tweets API umožňuje (Twitter, 2019c) získávat data v reálném čase, formou kontinuálního spojení, jehož prostřednictvím Twitter průběžně odesílá příspěvky v reálném čase. Výstupem volání v případě využití daného API není jednorázová odpověď obsahující požadovaná data, ale proud (stream) dat obsahující Tweety, které v reálném čase vznikají a odpovídají specifikovaným kritériím.

Konečně API Sample Tweets poskytuje funkcionality, která (Twitter, 2019h) zajišťuje možnost v reálném čase získat určitý „nahodilý“ vzorek (sample) Tweetů. V případě tohoto API tedy není možné specifikovat žádná kritéria determinující strukturu/obsah výsledné množiny získaných Tweetů. Dva uživatelé využívající dané API v identickém časovém okně navíc získají identickou množinu Tweetů.²⁹

Pro účely realizace typové úlohy zpracování big data bylo zvoleno API Filter Tweets, a to z několika důvodů. Především, na rozdíl od klasického Search Tweets API, umožňuje získávat Tweety v reálném čase, což lépe koresponduje se zaměřením této diplomové práce. Dalším faktorem je dostupnost daného API na úrovni Standard, tedy zdarma. V neposlední řadě pak pro výběr daného API hovoří možnost testovat dané API v rámci Twitter Labs v tzv. režimu preview umožňujícím využívat vyšší objemy dat než v případě úrovně Standard.

Na základě výběru Filter Tweets API pro účely realizace typové úlohy zpracování big data byla následně provedena registrace do Twitter Labs, která je bez dalších požadavků dostupná pro registrované uživatel Twitter Developer. Dále byla provedena aktivace dostupného Filtered Stream v rámci Twitter Labs a jeho propojení s dříve vytvořenou Twitter aplikací Spark Sentiment Analysis App.

5.3 Příjem big data v reálném čase

5.3.1 Specifikace konkrétního procesu příjmu big data v reálném čase

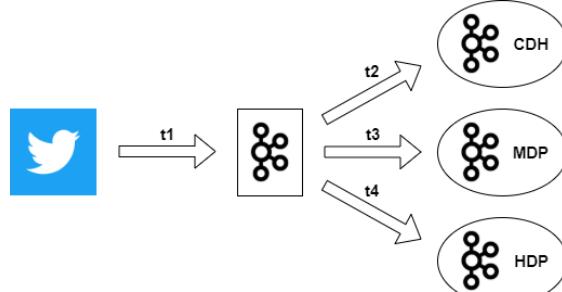
Pro účely realizace typové úlohy zpracování big data v reálném čase bylo nutné vymezit konkrétní specifický proces příjmu big data ze sociální sítě Twitter prostřednictvím zvoleného API.

²⁹) Výrazně zajímavější variantu představuje Sample Tweets API poskytované na Enterprise úrovni, jehož prostřednictvím je možné například (Twitter, 2019h) získat 10% (Decahose) vzorku z aktuální množiny Tweetů (Firehose) vznikající v reálném čase, na rozdíl od 1% v případě úrovně Standard.

Koncový bod Filtered Stream se od ostatních API, respektive koncových bodů, které společnost Twitter poskytuje, liší především v tom, že skutečně (Twitter, 2019d) umožňuje v reálném čase filtrovat proud (stream) veřejných příspěvků neboli Tweetů, které jsou právě publikovány v rámci sociální sítě Twitter, a to na základě definované množiny pravidel (rules) specifikovaných prostřednictvím kombinace operátorů (operators). Po navázání spojení s API Filtered Stream je vytvořeno (Twitter, 2019d) perzistentní HTTP (hypertext transfer protocol) „streaming“ spojení, kdy dané API klientskému programu, který spojení inicioval, neustále kontinuálně odesílá proud Tweetů v JSON formátu splňujících definovaná pravidla, a to až do doby (pokud pomineme ukončení spojení z důvodů, jejichž příčina není na straně daného API ani iniciujícího klientského programu), kdy spojení ukončí klientský program nebo dané Twitter API z důvodu, že klientský program data neodbavuje, respektive nepřijímá dostatečně rychle nebo v okamžiku dosažení limitů využití API či z jiného důvodu.

Jak bylo uvedeno v kapitole Předpoklady a omezení práce, nebylo možné nasadit a spustit cluster všech komparovaných distribucí současně, ale pouze postupně. Realizace příjmu dat opakovaně postupně vždy v rámci každé jednotlivé právě nasazené distribuce frameworku Apache Hadoop, byť s identickými parametry volání zvoleného Twitter API, by nicméně determinovala získání a následné zpracování a analýzu naprostě odlišné množiny Tweetů, což by mohlo značně zkreslit výsledky komparace distribucí frameworku Apache Hadoop, zejména v rovině zajištění analýzy sentimentu na získanými big data. Z tohoto důvodu byl proces příjmu big data v reálném čase koncipován a následně realizován tak, jak je to znázorněno na Obr. 5-2.

Obr. 5-2 Proces příjmu big data v reálném čase



Zdroj: Vlastní zpracování, grafické prvky (ASF, 2020a; Twitter, 2020a)

V rámci realizace praktické úlohy zpracování big data byl proto zprovozněn další samostatný uzel, na který byla nasazena instance platformy pro (ASF, 2020a) distribuované zpracování proudů dat Apache Kafka, a to ve stejné verzi, jaká byla dostupná v rámci komparovaných distribucí frameworku Apache Hadoop, tedy 2.1.0.

Tato komponenta, na rozdíl například od některých message queue systémů, zpravidla přijaté zprávy z fronty po jejich přečtení neodstraňuje, ale naopak defaultně uchovává určitou množinu přijatých ve frontě po specifikovanou dobu nebo do určeného objemu zpráv, a to bez ohledu na to, zda byly zprávy konzumovány či nikoli. Pro zajištění uchování zpráv pro účely zpracování v rámci všech komparovaných distribucí proto bylo nutné pozměnit konfiguraci retence přijatých zpráv (`log.retention.hours`), a to z defaultní hodnoty 168 hodin na -1, což determinuje kontinuální uchování zpráv po neomezenou dobu. Poté již bylo možné přistoupit k vlastnímu příjmu dat ze sociální sítě Twitter.

5.3.2 Příjem big data a analýza sentimentu nad jednotnou množnou big data

V první řadě byla z výše uvedených důvodů nejprve získána jednotná množina dat ze sociální sítě Twitter a uložena do systému pro distribuované zpracování proudů dat Apache Kafka, který se nachází mimo clustery komparovaných distribucí. Big data tedy byla ze sociální sítě přijata jednou a později zpracována jedenkrát v rámci každého clusteru distribuce frameworku Apache Hadoop, které byly předmětem komparace.

Pro zpracování a analýzu získaných big data, které měly být realizovány až po přijetí big data z Kafka, byla zvolena komponenta Spark dostupná v rámci všech komparovaných distribucí frameworku Hadoop. Projekt Spark poskytuje řadu API pro distribuované zpracování big data. Pro kontinuální zpracování proudů dat lze využít především (ASF, 2019v) starší API Spark Streaming pracující s objekty typu DStream nebo novější API Spark Structured Streaming, pracující především s objekty typu DataFrame a DataSet. Zatímco starší API je bližší kontinuálnímu dávkovému zpracování big data, novější API je založeno na podstatně předpracovaném modelu zpracování proudů big data, který se více blíží zpracování v reálném čase (ačkoli se ani v tomto případě o zpracování v reálném čase v pravém smyslu slova nejedná). Pro maximální přiblížení zpracování přijatých big data v reálném čase bylo tedy zvoleno nové API Spark Structured Streaming.

Původně byla tedy realizace analýzy sentimentu plánována s využitím Spark Structured Streaming API, tzn., jak již bylo uvedeno, až po přijetí dat z Kafka v rámci každého jednotlivého nasazeného clusteru distribuce Hadoop. Nicméně Spark Structured Streaming DataFrame API produkuje tzv. (ASF, 2019w) dynamicky typované (untyped) objekty DataFrame, jejichž schéma je validováno pouze v rámci běhového prostředí a nikoli v rámci komplikace. Jestliže tedy v rámci Spark Structured Streaming definujeme schéma zpracovávaných dat, a to včetně datových typů, není možné použít příslušnou knihovnu pro realizaci analýzy sentimentu právě proto, že data v průběhu komplikace v případě streaming DataFrame neodpovídají požadovanému datovému typu. Z těchto důvodů byla analýza sentimentu zařazena v procesu zpracování big data ihned po jejich přijetí, ještě před prvotním uložením do Kafka.

Hlavním cílem analýzy sentimentu je (Sarkar, 2018) „... *analyzovat obsah textu a porozumět názoru, který vyjadřuje.*“ Pro účely analýzy sentimentu lze využít (Sarkar, 2018) řadu slovníků obsahujících kolekci slov, kterým je přiřazeno skóre, číslo, jež může nabývat (Sarkar, 2018):

- zápornou polaritu, tj. např. hodnotu -3, indikující negativní sentiment,
- nulovou polaritu, tj. např. hodnotu 0, indikující neutrální sentiment,
- kladnou polaritu, tj. např. hodnotu 2, indikující pozitivní sentiment.

Analýza sentimentu (Sarkar, 2018) spadá primárně pod obor zpracování přirozeného jazyka (natural language processing – NLP) a jako taková nachází široké uplatnění v mnoha oblastech. Analýza sentimentu představuje komplexní proces, který nezřídka vykazuje interdisciplinární přesah. V kontextu této práce je nutné uvést, že v rámci praktické úlohy zpracování big data se nejedná o plnohodnotnou analýzu sentimentu, ale pouze o „primitivní“ formu analýzy sentimentu, jejímž účelem je především získat data pro

komparaci distribucí frameworku Hadoop v rovině praktické úlohy a nikoli prezentovat komplexní a správný způsob, jakým provádět analýzu sentimentu.³⁰

Jak již bylo uvedeno, pro získání big data ze sociální sítě Twitter bylo využito Twitter Filtered Stream API. K tomuto účelu bylo přitom použito ukázkového kódu z dokumentace Twitter Filtered Stream API (Twitter, 2020b), který znázorňuje Výpis 5-1, přičemž uvedený kód již obsahuje modifikace specifické pro danou praktickou úlohu.

Výpis 5-1 `kafka_producer.py` – kód pro příjem big data z Twitter Filtered Stream API (vlastní zpracování dle (Twitter, 2020b; Nielsen, 2011; Sarkar, 2018))

```
1  import os
2  import requests
3  import json
4  import time
5  from requests.auth import AuthBase
6  from requests.auth import HTTPBasicAuth
7  from tweepy.streaming import StreamListener
8  from tweepy import AppAuthHandler
9  from tweepy import Stream
10 from kafka import SimpleProducer
11 from kafka import KafkaClient
12 from afinn import Afinn
13
14 afinn = Afinn()
15
16 consumer_key = "YourConsumerKeyHere"
17 consumer_secret = "YourConsumerSecretHere"
18
19 stream_url = "https://api.twitter.com/labs/1/tweets/stream/filter"
20 rules_url = "https://api.twitter.com/labs/1/tweets/stream/filter/rules"
21
22 sample_rules = [
23     { 'value': 'Bernie Sanders lang:en', 'tag': 'Bernie Sanders' },
24     { 'value': 'Joe Biden lang:en', 'tag': 'Joe Biden' },
25     { 'value': 'Pete Buttigieg lang:en', 'tag': 'Pete Buttigieg' },
26     { 'value': 'Elizabeth Warren lang:en', 'tag': 'Elizabeth Warren' },
27     { 'value': 'Amy Klobuchar lang:en', 'tag': 'Amy Klobuchar' },
28 ]
29
30 class BearerTokenAuth(AuthBase):
31     def __init__(self, consumer_key, consumer_secret):
```

³⁰⁾ Plnohodnotná analýza sentimentu by zahrnovala například eliminaci interpunkce, vymezení a použití zástupných názvů pro hodnocené termíny (např. v případě Tweetů se často namísto plného jména politiků objevují přezdívky) a řadu dalších aspektů. Pro více informací o analýze sentimentu viz např. [SHEELA, L. J. 2016. A Review of Sentiment Analysis in Twitter Data Using Hadoop.](#)

```

32     self.bearer_token_url = "https://api.twitter.com/oauth2/token"
33     self.consumer_key = consumer_key
34     self.consumer_secret = consumer_secret
35     self.bearer_token = self.get_bearer_token()
36
37     def get_bearer_token(self):
38         response = requests.post(
39             self.bearer_token_url,
40             auth=(self.consumer_key, self.consumer_secret),
41             data={'grant_type': 'client_credentials'},
42             headers={'User-Agent': 'Spark Sentiment Analysis App'})
43
44         if response.status_code is not 200:
45             print('Cannot get a Bearer token rules', response.status_code, response.text)
46
47         body = response.json()
48         return body['access_token']
49
50     def __call__(self, r):
51         r.headers['Authorization'] = 'Bearer ' + self.bearer_token
52         r.headers['User-Agent'] = 'Spark Sentiment Analysis App'
53         return r
54
55     def get_all_rules(auth):
56         response = requests.get(rules_url, auth=auth)
57
58         if response.status_code is not 200:
59             print('Cannot get rules', response.status_code, response.text)
60
61         return response.json()
62
63     def delete_all_rules(rules, auth):
64         if rules is None or 'data' not in rules:
65             return None
66
67         ids = list(map(lambda rule: rule['id'], rules['data']))
68
69         payload = {
70             'delete': {
71                 'ids': ids
72             }
73         }
74
75         response = requests.post(rules_url, auth=auth, json=payload)
76

```

```

77     if response.status_code is not 200:
78         print('Cannot delete rules', response.status_code, response.text)
79
80     def set_rules(rules, auth):
81         if rules is None:
82             return
83
84         payload = {
85             'add': rules
86         }
87
88         response = requests.post(rules_url, auth=auth, json=payload)
89
90         if response.status_code is not 201:
91             print('Cannot create rules', response.status_code, response.text)
92
93     kafka = KafkaClient("localhost:9092")
94     producer = SimpleProducer(kafka)
95
96     def stream_connect(auth):
97         response = requests.get(stream_url, auth=auth, stream=True)
98         for response_line in response.iter_lines():
99             if response_line:
100                 response_line = json.loads(response_line.decode('utf-8'))
101                 if "data" in response_line:
102                     response_line['sentiment_score'] = afinn.score(response_line["data"]["text"])
103                     response_line['sentiment_category'] =
104                         ['positive' if response_line["sentiment_score"] > 0
105                          else 'negative' if response_line["sentiment_score"] < 0
106                          else 'neutral']
107                     producer.send_messages("tweets", json.dumps(response_line).encode('utf-8'))
108                 else:
109                     producer.send_messages(
110                         "tweets_errors", json.dumps(response_line).encode('utf-8'))
111
112     bearer_token = BearerTokenAuth(consumer_key, consumer_secret)
113
114     def setup_rules(auth):
115         current_rules = get_all_rules(auth)
116         delete_all_rules(current_rules, auth)
117         set_rules(sample_rules, auth)
118
119     setup_rules(bearer_token)
120
121     timeout = 0

```

```

120 while True:
121     stream_connect(bearer_token)
122     sleep(2 ** timeout)
123     timeout += 1

```

Z dostupných možností (Python, Ruby, JavaScript) byl pro získání dat z Filtered Stream API zvolen programovací jazyk Python 3. Tento byl v rámci všech clusterů již instalován a bylo nutné doinstalovat pouze relevantní Python knihovny, především Requests pro zajištění HTTP spojení, Tweepy pro přístup k Twitter API, a finn pro realizaci analýzy sentimentu a Kafka pro publikování dat do fronty (topic) v rámci Apache Kafka. Všechny zmíněné knihovny byly instalovány prostřednictvím Python správce balíků pip3.

Kód, který obsahuje Výpis 5-1, již zahrnuje úpravy a změny, jež bylo nutné pro účely praktické úlohy provést. Především se jednalo o modifikace syntaxe chybových hlášení, bez nichž kód nebylo možné úspěšně kompilovat. Dále se jednalo o nahrazení původního výstupu na standardní výstup (tedy obrazovku) provedením analýzy sentimentu. Pro realizaci sentimentu v rámci praktické úlohy zpracování big data byl zvolen slovník AFINN (Nielsen, 2011), a to prostřednictvím modulu, respektive knihovny pro programovací jazyk Python, ve verzi afinn 0.1 obsahující slovník AFINN-en-165.txt zahrnující 3382 slov s přiřazeným skóre sentimentu. Vložený kód (řádky 100 až 105) zajistil dekódování přijatých dat z datového typu `bytes` na typ `dict`, výpočet hodnot `sentiment_score` a `sentiment_category` a jejich uložení do přijatých dat. Další úprava zajistila následné přetyповání zpět na `bytes`.

Apache Kafka rozlišuje v procesu příjmu a distribuce dat dvě základní abstrakce, a to (ASF, 2020c) producenta/zdroj zprávy (producer) a konzumenta/příjemce zprávy (consumer). Další modifikace kódu proto zajistila publikování získaných Tweetů do fronty `tweets` nebo `tweets_errors` (v případě, že volání nevrací Tweet, ale chybu ve spojení) v rámci Kafka prostřednictvím metody `SimpleProducer`. V neposlední řadě byla dle potřeb modifikována pravidla, jejichž splnění determinovalo přijetí Tweetu. Pro úplnost je ještě nezbytné uvést, že v kódu ve Výpis 5-1 byly záměrně skutečné přístupové údaje (`consumer_key`, `consumer_secret`) nahrazeny textovým řetězcem.

Připravený kód byl následně spuštěn příkazem `python3 kafka_producer.py` v rámci nově nasazeného stroje, kde se nacházela samostatná komponenta Apache Kafka, jež nebyla součástí clusteru žádné z komparovaných distribucí Hadoop. Při prvním spuštění byla použita pravidla odpovídající názvům jednotlivých komparovaných distribucí Hadoop. Spuštěná Python úloha nicméně skončila s chybou implikující, že pro názvy distribucí CDH, MDP, HDP (byly použity plné názvy) není v reálném čase k dispozici dostatek Tweetů, což v konečném důsledku determinovalo ukončení spojení ze strany koncového bodu Twitter. Po několika dalších neúspěšných pokusech s jinými klíčovými slovy, respektive pravidly bylo rozhodnuto použít jako filtrovací pravidla jména pěti kandidátů, kteří se účastnili primárních prezidentských voleb ve Spojených státech amerických, tzn. Bernie Sanders, Joe Biden, Pete Buttigieg, Elizabeth Warren, Amy Klobuchar, přičemž byl ještě nastaven operátor `lang` s cílem zajistit pouze Tweety psané v Angličtině. Tato pravidla umožňovala získávat v reálném čase dostatečný objem zpráv tak, aby nedošlo k ukončení spojení ze strany Twitter API.

Příjem big data ze sociální sítě Twitter byl zahájen dne 3. března 2020 ve 14:00 hodin středoevropského času, což odpovídalo času 09:00 hodin ve státě New York. Právě na tento den připadalo tzv. volební superúterý (Super Tuesday), kdy se primární volby konají v největším počtu zemí Spojených států amerických. Aby nedošlo k pádu programu z důvodu ztráty/výpadku spojení se serverem, byl skript `python3 kafka_producer.py` spuštěn v rámci programu `screen`, který tuto možnost eliminuje. Příjem dat byl pravidelně kontrolován dotazem na nejvyšší hodnotu indexu (offset) ve frontě (topic) tweets, a to pomocí příkazu `watch -n 3 kafka-run-class.sh kafka.tools.GetOffsetShell --broker-list localhost:9092 --topic tweets --time -1`.

Obr. 5-3 Údaje prvních získaných Tweetů

Offset	Key	Timestamp	Text
0		1969-12-31 23:59:59.999	empty
1		1969-12-31 23:59:59.999	empty
2		1969-12-31 23:59:59.999	empty
3		1969-12-31 23:59:59.999	empty
4		1969-12-31 23:59:59.999	empty
5		1969-12-31 23:59:59.999	empty
6		1969-12-31 23:59:59.999	empty
7		1969-12-31 23:59:59.999	empty

Zdroj: vlastní zpracování

Jakmile byl příjem dat zahájen, začala se tato hodnota navýšovat. Po celou dobu přijímání dat vracelo Twitter Filtered Stream API přibližně okolo 1000 Tweetů za minutu (plus míinus cca 100), přičemž pro účely této praktické úlohy nebylo rozlišováno, zda se jedná o Tweety nebo Retweety, aby bylo sníženo riziko, že objem zpráv klesne na hodnotu, která by determinovala ukončení spojení ze strany daného Twitter API. Spojení a současně příjem dat, běžely bez přerušení až do 21:26 hodin, kdy bylo spojení přerušeno ze strany Twitter API s chybou typu `OperationalDisconnect` a popisem implikujícím, že spojení bylo ukončeno z provozních důvodů. Tato událost byla zjištěna díky tomu, že pravidelný dotaz na hodnotu indexu (offset) fronty tweets vracel stále stejnou hodnotu. Celý skript byl proto téměř ihned spuštěn znova a příjem dat byl obnoven. Ve 21:46 hodin středoevropského času bylo spojení ukončeno ze strany Twitter API s chybou typu `UsageCapExceeded` a popisem uvádějícím, že bylo dosaženo limitu počtu stažených Tweetů.

Uvedeným způsobem se podařilo v rámci necelých 8 hodin získat 500 821 Tweetů obsahujících hledaná slova, což odpovídá limitu, který byl pro daný koncový bod na neplacené úrovni Twitter Filtered Stream stanoven za měsíc (Twitter API nezastaví odesílání dat ihned při odeslání 500 tisícího Tweetu, ale je zde určitá setrvačnost). Obr. 5-3 znázorňuje základní údaje o množině osmi prvních získaných Tweetů. Na snímku je přitom patrné, že ve frontě (topic) je všech 500 821 získaných Tweetů. Na Obr. 5-4 je znázorněna struktura získaného Tweetu a je patrné, že data již obsahují také hodnoty parametrů `sentiment_score` a `sentiment_category`. Pro úplnost je nezbytné uvést, že v případě Obr. 5-3 i Obr. 5-4 se jedná o snímek obrazovky programu Kafdrop umožňujícího dotazovat Kafka API prostřednictvím grafického rozhraní.

Obr. 5-4 Struktura získaných Tweetů

```

Partition 0 Offset 0 # messages 100 Message format DEFAULT View Messages
Offset: 0 Key: Timestamp: 1969-12-31 23:59:59.999 Headers: empty
{
  "data": {
    "format": "default",
    "text": "RT @redd3451: Joe Biden is one of the most dishonest, salacious, impractical & inexperienced individuals of which of",
    "entities": {
      "annotations": [
        {
          "start": 14,
          "end": 22,
          "normalized_text": "Joe Biden",
          "probability": 0.9974498939010518,
          "type": "Person"
        }
      ],
      "mentions": [
        {
          "start": 3,
          "end": 12,
          "username": "redd3451"
        }
      ]
    },
    "created_at": "2020-03-03T13:01:46.000Z",
    "author_id": "1136310400537042944",
    "id": "1234826298922586112",
    "referenced_tweets": [
      {
        "type": "quoted",
        "id": "1234552989700675840"
      },
      {
        "type": "retweeted",
        "id": "1234689691011756033"
      }
    ],
    "sentiment_category": [
      "negative"
    ],
    "sentiment_score": -2,
    "matching_rules": [
      {
        "tag": "Joe Biden",
        "id": "1234826336000147456"
      }
    ]
  }
}
  
```

Zdroj: vlastní zpracování

5.3.3 Příjem big data z jednotného zdroje do clusteru distribuce Hadoop

V rámci přípravy na příjem získaných big data do clusteru byly nejprve v rámci dedikované instance platformy pro distribuované zpracování proudů dat Apache Kafka pomocí příkazu ve tvaru `kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1`

--partitions 1 --topic <topic_name> vytvořeny tři fronty (topic), jedna pro každou distribuci Hadoop, jež byla předmětem komparace. Takto byly vytvořeny fronty tweets_for_cdh, tweets_for_mdp, tweets_for_hdp. Data z fronty tweets byla následně zkopirována do těchto nově vytvořených front. Pro tyto účely byl nasazen a použit nástroj kafkacat (příkaz ve tvaru kafkacat -C -b localhost:9092 -o beginning -e -t tweets | kafkacat -P -b localhost:9092 -t <topic_name>). V rámci všech front tak byla k dispozici naprostě identická data pro jednotlivé komparované distribuce frameworku Apache Hadoop.

Příjem big data z dedikované instance Kafka do instance Kafka v rámci clusteru obsahujícího komparovanou distribuci Hadoop byl dále realizován s využitím nástroje Kafka Mirror Maker, který je standardně součástí Kafka. Jedná se o program, který zajišťuje (ASF, 2020a) replikaci dat ze zdrojového clusteru Kafka do jiného cílového clusteru Kafka. V případě praktické úlohy, která je zde popisována, přitom zdrojový cluster Kafka představovala dedikovaná instance Kafka, zatímco cílový cluster reprezentovala instance Kafka v rámci každého jednotlivého clusteru dané komparované distribuce frameworku Apache Hadoop. Pro každou jednotlivou distribuci Hadoop, která byla předmětem komparace byly provedeny následující kroky:

1. vytvořit frontu tweets (kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic tweets) v rámci cílového clusteru,
2. odstranit frontu tweets (./kafka-topics.sh --zookeeper localhost:2181 --delete --topic tweets) v rámci zdrojového clusteru,
3. vytvořit frontu tweets (./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic tweets) v rámci zdrojového clusteru,
4. spustit v rámci cílového clusteru periodické zjišťování indexu (offset) poslední zprávy evidované v rámci fronty tweets (watch -n 3 </usr/hdp/3.1.0.0-78>/kafka/bin/kafka-run-class.sh kafka.tools.GetOffsetShell --broker-list destination.kafka.broker.ipaddress:6667 --topic tweets --time -1),
5. spustit v rámci zdrojového clusteru program Kafka Mirror Maker nad frontou tweets (/usr/bin/time -v ./kafka-mirror-maker.sh --consumer.config ..//config/consumer.properties --producer.config ..//config/producer.properties --whitelist="tweets"),
6. zkopiřovat v rámci zdrojového clusteru zprávy z fronty pro danou distribuci do fronty tweets (kafkacat -C -b localhost:9092 -o beginning -e -t <tweets_for_cdh> | kafkacat -P -b localhost:9092 -t tweets),
7. ukončit spuštěný Kafka Mirror Maker v okamžiku, kdy index (offset) poslední evidované zprávy ve frontě tweets v rámci cílového clusteru dosáhne hodnoty 500821.

Pro zahájení replikace mezi clustery Kafka je na straně Kafka Mirror Maker, respektive v místě, kde bude tento program spuštěn, nezbytná konfigurace příjemce zpráv (consumer) a odesílatele (producer) zpráv. Skutečnou konfiguraci těchto komponent, která byla v rámci realizace praktické úlohy zpracování big data použita, obsahuje Výpis 5-2. Tento dobře ilustruje, že Kafka Mirror Maker v podstatě kombinuje komponentu příjemce zpráv (consumer), která v rámci zdrojového clusteru (v tomto případě dedikované instance Apache Kafka) přijímá zprávy ze specifikované fronty (topic) a

komponentu producenta zpráv (producer), která v rámci cílového clusteru (v tomto případě instance Apache Kafka v rámci každé jednotlivé komparované distribuce Hadoop) publikuje zprávy do identické fronty (v tomto výpisu byly skutečné hodnoty IP adres zaměněny ze textový řetězec).

Výpis 5-2 consumer.properties, producer.properties – konfigurace Kafka Mirror Maker (vlastní zpracování)

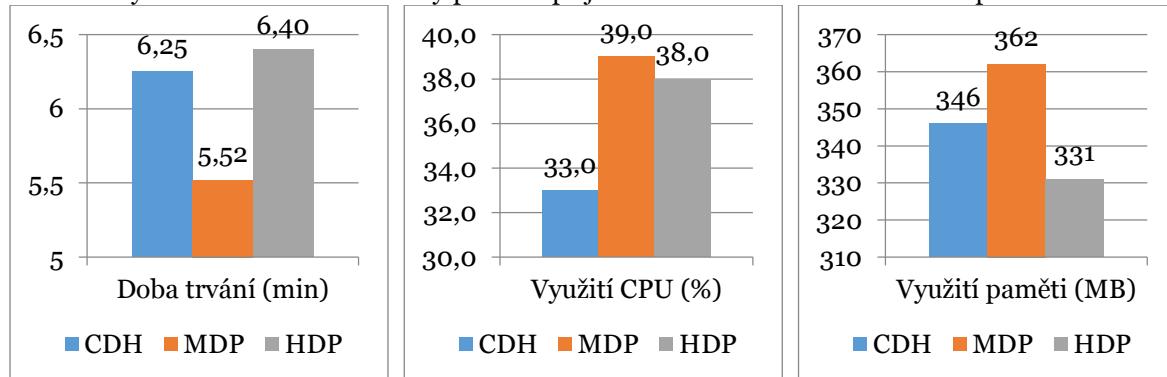
```

1 # soubor consumer.properties:
2 bootstrap.servers=source.kafka.broker.ipaddress:9092
3 client.id=mirror_maker_consumer
4 group.id=mirror_maker_consumer
5
6 # soubor producer.properties:
7 bootstrap.servers=destination.kafka.broker.ipaddress:6667
8 client.id=mirror_maker_producer
9 compression.type=none
10 batch.size=100

```

Předání zpráv mezi těmito komponentami spolehlivým způsobem zajišťuje právě Kafka Mirror Maker, který je možné spustit (s relevantní konfigurací) v rámci zdrojového nebo i cílového clusteru Kafka. Důležitou vlastností Kafka Mirror Maker je skutečnost, že neprovádí replikaci zpráv, které se již v replikované frontě nacházely před spuštěním Mirror Makeru. Právě tato skutečnost byla důvodem pro použití výše uvedeného postupu, který byl realizován v různých časových okamžicích individuálně vždy pro danou komparovanou distribuci Hadoop.

Obr. 5-5 Výkonnostní charakteristiky procesu příjmu dat v rámci distribucí Hadoop



Zdroj: vlastní zpracování

Uvedeným způsobem byl zajištěn příjem big data z jednotného zdroje (dedikovaná instance Apache Kafka) do clusteru distribuce Hadoop, která byla právě nasazena, postupně tedy CDH, HDP a MDP. Zatímco lokální kopírování tweetů z jedné fronty do druhé prostřednictvím nástroje kafkacat bylo dokončeno prakticky okamžitě, replikace tweetů ze zdrojového do cílového clusteru určitou dobu trvala. V případě všech zkoumaných clusterů distribucí Hadoop byly v průběhu procesu příjmu big data z jednotného zdroje do clusteru distribuce Hadoop měřeny výkonnostní charakteristiky, konkrétně doba trvání, využití CPU a využití paměti. Pro tento účel byl využit nástroj `time`,

který je defaultně v rámci OS Ubuntu k dispozici. Výsledky měření jsou k dispozici v rámci Obr. 5-5.

Nejrychleji byla data pomocí nástroje Kafka MirrorMaker zkopirována do clusteru distribuce MDP, dále pak CDH a HDP. V případě rozsáhlějších objemů dat by se tyto hodnoty mohly lišit výrazněji. Na druhou stranu, nástroj Kafka MirrorMaker není určen pro příjem dat, ale pro replikaci dat mezi clustery, respektive brokeru Kafka, takže z hlediska příjmu dat nejsou naměřené hodnoty relevantní. Pro zajištění identické množiny zpracovávaných dat v rámci jednotlivých komparovaných distribucí byla replikace nicméně nezbytná. Z hlediska využití CPU vykazovala nejvyšší nároky distribuce MDP a dále HDP a CDH. Rozdíl se pohyboval v rozmezí do 10%, což nepředstavuje zásadní odchylku. Nejvyšší hodnota maximální přidělené paměti byla naměřena v případě MDP a dále CDH a HDP. Ani v případě této metriky nebyly zaznamenány výraznější odchylky. Celkově lze konstatovat, že využití zdrojů pro replikaci dat do jednotlivých distribucí se příliš nelišilo, nejvíce zdrojů bylo využito pro replikaci v případě distribuce MDP, ale při dosažení nejkratšího časového intervalu potřebného pro zreplikování dat.

5.4 Zpracování a analýza získaných big data

5.4.1 Příprava a transformace získaných big data

Poslední fází praktické úlohy zpracování big data bylo zpracování a analýza získaných big data. Jak již bylo uvedeno, pro tyto účely byla zvolena komponenta Spark, přičemž konkrétní verze dané komponenty v rámci jednotlivých nasazených distribucí frameworku Hadoop byly uvedeny již v Tab. 5.2. Rovněž již bylo uvedeno, že s cílem přiblížit se v maximální možné míře zpracování big data v reálném čase, bylo zvoleno Spark Structured Streaming API.

Mezi základní komponenty, jejichž prostřednictvím jsou realizovány úlohy Spark, patří (ASF, 2019i) řídící program (Spark Driver) zajišťující řízení a orchestraci programů realizujících konkrétní výpočty (Spark Executors). Jestliže jsou Spark úlohy zpracování dat prováděny nad clusterem Hadoop, kde distribuci zdrojů pro realizaci úloh zpracování dat zajišťuje YARN, je možné využít Spark v provozním režimu (ASF, 2019i) částečně distribuovaném nebo plně distribuovaném. Pro účely realizace praktické úlohy zpracování big data, která je předmětem této kapitoly, byl Spark provozován v částečně distribuovaném režimu (client mode), kdy Spark Driver běží na klientu (v tomto případě druhým řídícím uzlem), z něhož je úloha spuštěna a je pro něj vytvořen dedikovaný YARN Application Master, zatímco všechny Spark Executors běží v clusteru.

Všechny výpočetní úlohy byly iniciovány interaktivně v rámci klienta Spark pro programovací jazyk Python spouštěného vždy z druhého řídícího uzlu clusteru Hadoop následujícím příkazem: `pyspark --jars </usr/hdp/3.1.0.0-78>/spark2/aux/spark-sql-kafka-0-10_2.11-2.3.0.cloudera2.jar,</usr/hdp/3.1.0.0-78>/spark2/aux/kafka-clients-2.0.0.jar --executor-memory 4G --num-executors 3 --driver-memory 2g`. Defaultní instalaci Spark není možné pro načítání dat z Kafka s využitím Spark Structured Streaming API použít. Proto byly, jak je zřejmé z uvedeného příkazu, použity dvě doplňkové knihovny, a sice (Weber

Informatics, 2018) kafka-clients pro klienty Kafka a (Cloudera, 2019l) spark-sql-kafka pro Spark Structured Streaming s využitím Kafka od společnosti Cloudera. Z výše uvedeného příkazu je také patrné, že pro řídící program Spark byly vyžádány 2 gigabajty paměti, a že byly vyžádány 3 Spark Executors, přičemž pro každý byly vyhrazeny 4 gigabajty operační paměti, aby bylo zajištěno, že úlohy Spark budou realizovány distribuovaně napříč všemi pracovními uzly v clusteru.

Jelikož byly všechny úlohy provedeny manuálně v rámci jedné interaktivní Spark session, nebude u každé popisované úlohy zpracování dat uváděn celý kód, protože části jako je iniciace `SparkSession`, definice schématu dat, definice parametrů spojení s Kafka a funkce pro přípravu dat by se ve výpisech kódu neustále bez změny opakovaly. Proto bude nejprve prezentován kód zahrnující sekce přípravy závislostí a dat. U každé jednotlivé úlohy zpracování big data pak bude prezentována pouze příslušná navazující část kódu, která vychází z úvodních sekcí, jež obsahuje Výpis 5-3.

Výpis 5-3 spark_sentiment_static_quries.py – příprava dat (vlastní zpracování dle (ASF, 2019w))

```
1 from pyspark.sql import SparkSession
2 from pyspark.sql.types import StructType
3 from pyspark.sql.types import StructField
4 from pyspark.sql.types import StringType
5 from pyspark.sql.types import ArrayType
6 from pyspark.sql.functions import from_json
7 from pyspark.sql.functions import col
8 from pyspark.sql.functions import when
9 from pyspark.sql.functions import sum
10 from pyspark.sql.functions import count
11 from pyspark.sql.functions import from_unixtime
12 from pyspark.sql.functions import unix_timestamp
13 from pyspark.sql.functions import window
14
15 spark = SparkSession\
16     .builder\
17     .appName("Spark Sentiment Analysis")\
18     .getOrCreate()
19
20 schema = StructType([
21     StructField("data", StructType([
22         StructField("id", StringType(), True),
23         StructField("created_at", StringType(), True),
24         StructField("text", StringType(), True),
25         StructField("author_id", StringType(), True),
26         StructField("referenced_tweets", ArrayType(StructType([
27             StructField("type", StringType(), True),
28             StructField("id", StringType(), True)
29         ])), True),
30         StructField("attachments", StructType([
31             StructField("type", StringType(), True),
32             StructField("url", StringType(), True)
33         ])), True),
34         StructField("in_reply_to_status_id", StringType(), True),
35         StructField("in_reply_to_user_id", StringType(), True),
36         StructField("geo", StructType([
37             StructField("lat", DoubleType(), True),
38             StructField("lon", DoubleType(), True)
39         ])), True),
40         StructField("place", StructType([
41             StructField("name", StringType(), True),
42             StructField("full_name", StringType(), True),
43             StructField("country", StringType(), True),
44             StructField("country_code", StringType(), True),
45             StructField("bounding_box", StructType([
46                 StructField("type", StringType(), True),
47                 StructField("coordinates", ArrayType(DoubleType(), True))
48             ])), True),
49             StructField("place_type", StringType(), True)
50         ])), True),
51         StructField("lang", StringType(), True)
52     ]), True)
53 ])
```

```

31     StructField("media_keys", ArrayType(StringType()), True)
32   ], True),
33   StructField("entities", StructType([
34     StructField("urls", ArrayType(StructType([
35       StructField("start", StringType(), True),
36       StructField("end", StringType(), True),
37       StructField("url", StringType(), True),
38     ])), True),
39     StructField("mentions", ArrayType(StructType([
40       StructField("start", StringType(), True),
41       StructField("end", StringType(), True),
42       StructField("username", StringType(), True),
43     ])), True),
44   ]), True),
45   StructField("format", StringType(), True)
46 ], True),
47 StructField("sentiment_category", ArrayType(StringType()), True),
48 StructField("sentiment_score", StringType(), True),
49 StructField("matching_rules", ArrayType(StructType([
50   StructField("id", StringType(), True),
51   StructField("tag", StringType(), True)
52 ])), True)
53 ])
54
55 tweets_df = spark\
56   .readStream\
57   .format("kafka")\
58   .option("kafka.bootstrap.servers", "source.kafka.broker.ipaddress:6667")\
59   .option("subscribe", "tweets")\
60   .option("startingOffsets", "earliest")\
61   .load()
62
63 tweets_df = tweets_df.selectExpr("CAST(value AS STRING)")\
64   .withColumn("value", from_json(col("value"), schema=schema))
65
66 tweets_df = tweets_df\
67   .withColumn("created_at",
68     from_unixtime(unix_timestamp(col("value.data.created_at"),"yyyy-MM-
69     dd'T'HH:mm:ss.sss'Z'")))\\
70   .withColumn("negative_tweets", when((col("value.sentiment_score") < 0),
71     1).otherwise(0))\
72   .withColumn("neutral_tweets", when((col("value.sentiment_score") == 0),
73     1).otherwise(0))\
74   .withColumn("positive_tweets", when((col("value.sentiment_score") > 0),
75     1).otherwise(0))\

```

```

71     .selectExpr("created_at as created_at", "value.data.id as id", "value.data.author_id
        as author_id", "value.data.text as text", "value.matching_rules[0].tag as tag",
        "value.sentiment_category[0] as category", "value.sentiment_score as score",
        "negative_tweets as negative", "neutral_tweets as neutral", "positive_tweets as
        positive")

```

Z předchozího výpisu kódu je patrné, že byly nejprve importovány všechny závislosti nezbytné pro realizaci jednotlivých úloh. Jednalo se především o `SparkSession` představující „vstupní bod“ každé Spark Structured Streaming úlohy, dále relevantní datové typy a funkce. Inicializace aplikace Spark je vždy realizována prostřednictvím objektu `SparkSession`. Dále bylo definováno schéma dat přijímaných z Kafka. Jelikož Kafka defaultně uchovává zprávy v binárním formátu, bylo nezbytné provést jejich dekódování. Ačkoli je Spark do jisté míry schopen provést odvození schémata z dat (data inference), která jsou mu poskytnuta, explicitní definice schémata umožňuje začít data zpracovávat okamžitě s využitím deklarované struktury. Na druhou stranu, odladění schémata zabralo přibližně jednu třetinu veškerého času věnovaného Spark úlohám.

Dále byl definován zdroj proudu dat, a to Kafka broker v rámci clusteru dané distribuce frameworku Hadoop (skutečná IP adresa byla v rámci výpisu nahrazena textovým řetězcem). Následně byla data převedena z binární formy do JSON řetězce a z původních dat Kafka zprávy byla ponechána pouze hodnota `value` obsahující předmětná data. Následně byla provedena finální příprava dat zahrnující vytvoření Spark Structured Streaming DataFrame obsahujícího sloupce `value` (přijatá data Tweetů v JSON formátu dle deklarovaného schématu), `created_at` (transformovaná časová značka reprezentující datum vzniku Tweetu), `negative_tweets`, `neutral_tweets`, `positive_tweets` (hodnoty 1 nebo 0 indikující, zda daný Tweet vykazuje nebo nevykazuje danou polaritu sentimentu). Prostřednictvím funkce `selectExpr` byly vybrány požadované sloupce a nastaveny jejich názvy.

Před popisem realizace praktické úlohy a jednotlivých realizovaných Spark úloh je nutné stručně charakterizovat způsob, jakým použité API funguje. Následující odstavce proto obsahují popis vycházející z oficiální dokumentace Spark Structured Streaming (ASF, 2019w). Základní model Spark Structured Streaming API je založen na skutečnosti, že (ASF, 2019w) proud přijímaných dat je organizován jako tabulka, do které jsou neustále přidávány (append) nové záznamy. Spark Structured Streaming se v pravidelných intervalech dotazuje na nová data. V případě spuštění úlohy zpracování proudu dat Spark neustále iteruje v následujících krocích: získání nových dat, zpracování nových dat a výsledků zpracování dat z předchozí iterace, odeslání výsledku na výstup.³¹

³¹⁾ Interval dotazování na nová data je možné vždy u zdroje proudu dat nastavit. V rámci realizace praktické úlohy zpracování big data tato možnost využita nebyla, jelikož defaultní chování, kdy se Spark dotazuje na nová data vždy ihned po dokončení iterace, tedy po zpracování dat, bylo s ohledem na objem přijatých dat a možnosti vizualizace výsledků v rámci této práce zcela dostačující a nebylo nutné analyzovat data na nižší úrovni časové granularity, např. v intervalu jednotek vteřin, což je také možné.

Pro získání dat lze využít některý ze čtyř podporovaných zdrojů dat, mezi které patří (ASF, 2019w) soubor (text, csv, json, orc, parquet), Kafka, socket (pro testování), rate (pro generování dat). Realizace úloh využívajících Spark Structured Streaming je nejen distribuovaná, což determinuje škálovatelnost a paralelizaci, ale současně také odolná proti selhání (fault tolerant). Všechny podporované zdroje proudů dat, engine pro zpracování dat i cílová umístění pro uložení dat (sink) jsou navrženy tak, aby v případě selhání bylo možné zreproduktovat celý proces zpracování dat a/nebo navázat tam, kde zpracování dat skončilo. Pro tyto účely využívá Spark Structured Streaming API především kontrolní body (checkpoint) a logy operací (write-ahead log) a celá architektura tak garanteuje distribuované proti výpadkům odolné zpracování každé zprávy, respektive jednotky dat, přesně jedenkrát (exactly-once).

Obr. 5-6 Výstup dotazu na metriky sentimentu u kandidátů v primárních volbách

Batch: 35		
tag	Celkový sentiment	Celkový počet Tweetů
Joe Biden	145018.0	1224177
Bernie Sanders	161940.0	150047
Elizabeth Warren	59109.0	182354
Amy Klobuchar	17223.0	17546
Pete Buttigieg	16904.0	123954
Batch: 36		
tag	Celkový sentiment	Celkový počet Tweetů
Joe Biden	147763.0	1229079
Bernie Sanders	163187.0	154261
Elizabeth Warren	59759.0	184597
Amy Klobuchar	17259.0	17692
Pete Buttigieg	17248.0	124300
Batch: 37		
tag	Celkový sentiment	Celkový počet Tweetů
Joe Biden	147914.0	1229463
Bernie Sanders	163283.0	154587
Elizabeth Warren	59746.0	184756
Amy Klobuchar	17268.0	17706
Pete Buttigieg	17259.0	124309

Zdroj: vlastní zpracování

Základní model Spark Structured Streaming charakterizovaný v předchozím textu vraci uživateli za předpokladu, že jsou kontinuálně přijímána nová data, aktualizovanou tabulkou výsledků, jak znázorňuje Obr. 5-6, z něhož je patrné, jak postupně v rámci jednotlivých iterací dochází k aktualizaci hodnot v jednotlivých sloupcích po zpracování nově příchozích dat. Tento model byl v rámci realizace praktické úlohy zpracování big data aplikován za účelem získání základních sumárních metrik sentimentu (kumulativní součty v čase).

Uvedený přístup je podobný pravidelnému dotazu do tradiční relační databáze prostřednictvím SQL. S nárůstem doby trvání opakovaného dotazování by však tradiční SQL dotazy do RDBMS za Spark Structured Streaming API signifikantně zaostávaly.

Důvodem je především skutečnost, že Spark Structured Streaming si v rámci každé iterace (ASF, 2019w) uloží nezbytné mezivýsledky výpočtů a v další iteraci je využije pro aktualizaci výpočtu, takže nepočítá neustále kompletní data, ale pouze mezivýsledky z předchozí iterace kombinované s novými daty.

Kromě základního programovacího modelu Spark Structured Streaming lze využít tzv. (ASF, 2019w) window operací, kdy Spark rozdělí data do časových intervalů neboli oken (window), jejichž rozsah lze uživatelsky stanovit. Spark následně provede výpočty pro jednotlivá okna, přičemž postupně doplňuje další a aktualizuje stávající okna na základě nově přijatých dat. V rámci volání metody `window` je nezbytné definovat zdroj časové značky, podle níž budou data rozdělována do jednotlivých oken a uvést požadovaný rozsah okna, tedy kolik vteřin, minut nebo hodin jedno okno trvá. Volitelně lze také definovat posun okna (slide) vyjadřující, o kolik vteřin, minut nebo hodin se okno v čase vždy posune. Jestliže se okna nepřekrývají, není nutné posun specifikovat (je roven intervalu okna) a data jsou rozdělena do navazujících intervalů (např. okna v intervalu 1 hodina bez definovaného posunu), v jejichž rámci jsou dostupná data. V případě, že se okna překrývají (např. okna v intervalu 10 minut s posunem 5 minut), pak mohou některá data, v závislosti na časové značce, spadat do více oken současně.

Obr. 5-7 Výstup dotazu na metriky sentimentu v intervalu jedné hodiny

tag	window	Celkový sentiment	Celkový počet Tweetů
Elizabeth Warren	[2020-03-03 13:00:00, 2020-03-03 14:00:00)	5621.0	17654
Joe Biden	[2020-03-03 13:00:00, 2020-03-03 14:00:00)	21094.0	29166
Bernie Sanders	[2020-03-03 13:00:00, 2020-03-03 14:00:00)	3307.0	14672
Amy Klobuchar	[2020-03-03 13:00:00, 2020-03-03 14:00:00)	1648.0	1341
Pete Buttigieg	[2020-03-03 13:00:00, 2020-03-03 14:00:00)	13253.0	3207
Joe Biden	[2020-03-03 14:00:00, 2020-03-03 15:00:00)	21072.0	32797
Pete Buttigieg	[2020-03-03 14:00:00, 2020-03-03 15:00:00)	12353.0	3387
Amy Klobuchar	[2020-03-03 14:00:00, 2020-03-03 15:00:00)	11422.0	1328
Elizabeth Warren	[2020-03-03 14:00:00, 2020-03-03 15:00:00)	18278.0	9758
Bernie Sanders	[2020-03-03 14:00:00, 2020-03-03 15:00:00)	18791.0	17910
Amy Klobuchar	[2020-03-03 15:00:00, 2020-03-03 16:00:00)	1393.0	1156
Elizabeth Warren	[2020-03-03 15:00:00, 2020-03-03 16:00:00)	11643.0	10144
Pete Buttigieg	[2020-03-03 15:00:00, 2020-03-03 16:00:00)	2618.0	3345
Joe Biden	[2020-03-03 15:00:00, 2020-03-03 16:00:00)	18688.0	31475
Bernie Sanders	[2020-03-03 15:00:00, 2020-03-03 16:00:00)	10564.0	19679

Zdroj: vlastní zpracování

Jestliže je tedy spuštěna úloha Spark a jsou kontinuálně přijímána nová data, nad nimiž je volána funkce `window`, Spark vrací na výstupu opět tabulkou, aktualizovanou v rámci každé iterace. V tomto případě jsou nicméně aktualizovány pouze výsledky v rámci oken, do nichž spadají nově přijatá data, což indikuje Obr. 5-7 nebo jsou přidávána nová časová okna. Funkcionalita window operací je užitečná především pokud uvážíme, že nad takto organizovanými daty lze provádět agregace. V rámci praktické úlohy zpracování big data v této práci byla využita pro zjištění toho, jak se metriky sentimentu vyvíjely v jednotlivých časových intervalech. Pro zajištění přehlednosti vizualizací byl zvolen interval 1 hodina bez posunu (slide).

Další významnou funkcionalitu, kterou Spark Structured Streaming API poskytuje, představuje mechanismus umožňující (ASF, 2019w) pracovat s daty přijatými s určitým více či méně signifikantním zpožděním v porovnání s časem, kdy data vznikla/byla emitována. Taková data, pokud jsou přijata výrazně později, nemusí být v době zpracování příslušného časového okna dostupná, a protože Spark Structured Streaming se při každé

iteraci dotazuje na nová data a původní již neuchovává a tedy ani nepřečítává, nebudou vůbec zpracována. Spark Structured Streaming poskytuje pro řešení těchto problémů funkcionalitu označovanou jako tzv. watermarking spočívající v tom, že uživatel volá ještě před metodou `window` metodu `withWatermark`, které předá parametry časové značky a intervalu, v jehož rámci mohou zpožděná data přicházet. Spark Structured Streaming eviduje starší mezivýsledky na úrovni umožňující zajistit, že v případě pozdě přijatých dat spadajících do stanoveného rozsahu budou tato zpracována a příslušné časové okno bude aktualizováno. Funkcionalita watermarking nebyla v rámci realizace praktické úlohy zpracování big data aplikována. Důvodem pro toto rozhodnutí byla skutečnost, že funkcionalita je relevantní pro situace, kdy zdroj a/nebo příjemce dat čelí výpadkům sítě/infrastruktury, příjemce není schopen data přijímat s dostatečnou rychlostí, data jsou přijímána z více zdrojů bez synchronizace, apod. Pro realizaci analýzy sentimentu nad příspěvky ze sociální sítě Twitter proto daná funkcionalita nebyla relevantní.

V předchozím textu byly popsány hlavní oblasti funkcionality, které Spark Structured Streaming API poskytuje, včetně uvedení těch, které byly v rámci realizace praktické úlohy zpracování big data aplikovány. Kromě uvedených hlavních oblastí funkcionality poskytuje Spark Structured Streaming množství dalších dílčích funkcí, z nichž řada bude v rámci jednotlivých Spark úloh aplikována a jejich další popis na tomto místě by již nebyl přínosný. V následujících podkapitolách budou charakterizovány jednotlivé realizované Spark úlohy, včetně vizualizace výsledků a srovnání výkonnostních charakteristik při realizaci napříč jednotlivými komparovanými distribucemi frameworku Apache Hadoop.

5.4.2 Aplikace základního programovacího modelu

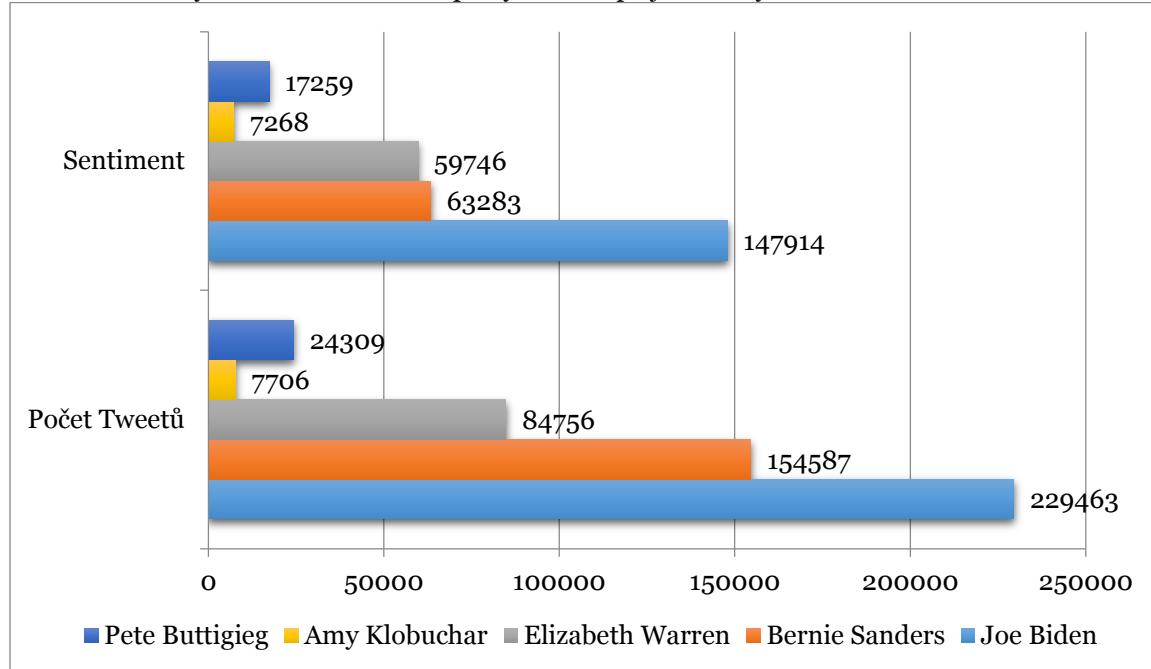
S využitím základního programovacího modelu Spark Structured Streaming byly nejprve v rámci realizace praktické úlohy provedeny nad přijímanými Tweety výpočty základních metrik sentimentu pro jednotlivé kandidáty, a to hodnoty celkového sentimentu (součet hodnot `sentiment_score`, které byly v rámci příjmu dat vypočteny s využitím knihovny afinn), počtu negativních, neutrálních a pozitivních Tweetů (počet Tweetů kandidáta s příslušnou polaritou sentimentu) a celkového počtu Tweetů (suma Tweetů jednotlivých kandidátů).

Výsledky v podobě uvedených „statických“ metrik znázorňují Obr. 5-8 a Obr. 5-9, v jejichž rámci jsou uvedeny hodnoty po realizaci 37. iterace, kdy již bylo prostřednictvím Kafka Mirror Maker do clusteru, respektive do Kafka v rámci clusteru dané distribuce, přesunuto všech 500 821 získaných Tweetů.

Jestliže porovnáme výsledky získané analýzou přijatých Tweetů s výsledky zveřejněnými po volebním superúterý, které jsou znázorněny na Hodnoty všech uvedených základních metrik sentimentu byly získány s využitím dotazu, který obsahuje Výpis 5-4. Z kódu je patrné, že Spark Structured Streaming podporuje agregace, takže není nutné samostatně zadávat jednotlivé dotazy, ale v rámci jediného složeného dotazu je možné s využitím agregace nadefinovat sloupce, v nichž mají být vypočítány aggregované hodnoty pro jednotlivé prvky z analyzované množiny (zde jméno kandidáta ve volbách).

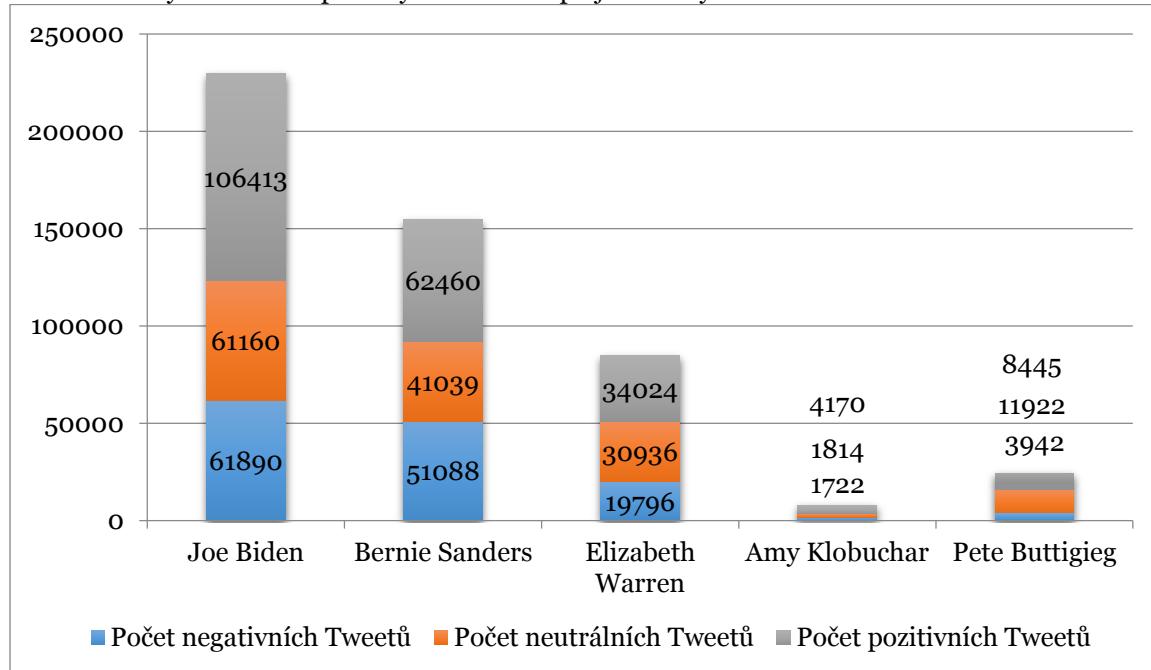
Obr. 5-10, lze konstatovat, že konečné pořadí kandidátů silně koresponduje s hodnotou sentimentu, která byla získána zpracováním přijatých Tweetů v rámci realizace praktické úlohy. Monitorováním sentimentu příspěvků na sociální síti Twitter, například způsobem aplikovaným v rámci realizace praktické úlohy v této práci, tak lze v téměř reálném čase získávat vhled do postojů a názorů uživatelů, monitorovat jejich vývoj v čase a využít tyto informace například pro predikce výsledků voleb.

Obr. 5-8 Celkový sentiment a celkové počty Tweetů po jednotlivých kandidátech



Zdroj: vlastní zpracování

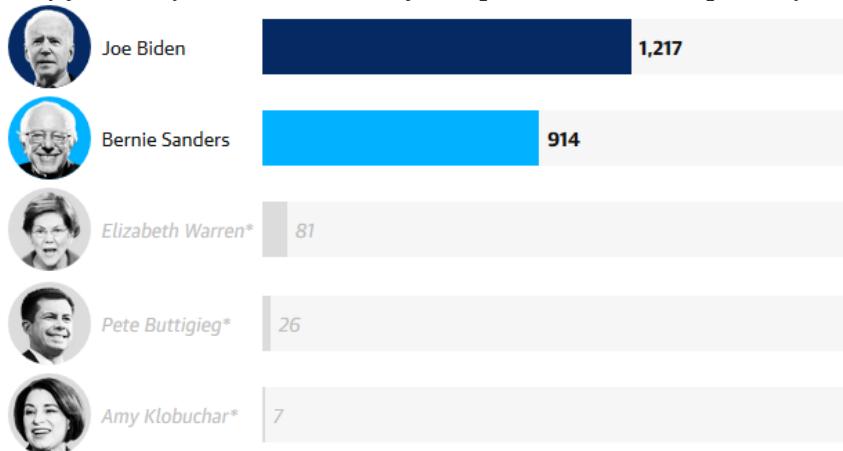
Obr. 5-9 Počty Tweetů dle polarity sentimentu po jednotlivých kandidátech



Zdroj: vlastní zpracování

Hodnoty všech uvedených základních metrik sentimentu byly získány s využitím dotazu, který obsahuje Výpis 5-4. Z kódu je patrné, že Spark Structured Streaming podporuje agregace, takže není nutné samostatně zadávat jednotlivé dotazy, ale v rámci jediného složeného dotazu je možné s využitím agregace na definovat sloupce, v nichž mají být vypočítány aggregované hodnoty pro jednotlivé prvky z analyzované množiny (zde jméno kandidáta ve volbách).

Obr. 5-10 Výsledky jednotlivých kandidátů zveřejněné po tzv. volebním superúterý



Zdroj: zpracování a data (Kommenda, 2020)

Ačkoli to není na první pohled z výpisu kódu nezbytně zřejmé, je nutné uvést, že vlastní výpočet začíná až v okamžiku, kdy je zavolána metoda `writeStream` nad příslušným DataFrame.

Výpis 5-4 `spark_sentiment_static_quries.py` – výpočet základních metrik sentimentu ze získaných Tweetů (vlastní zpracování dle (ASF, 2019w))

```

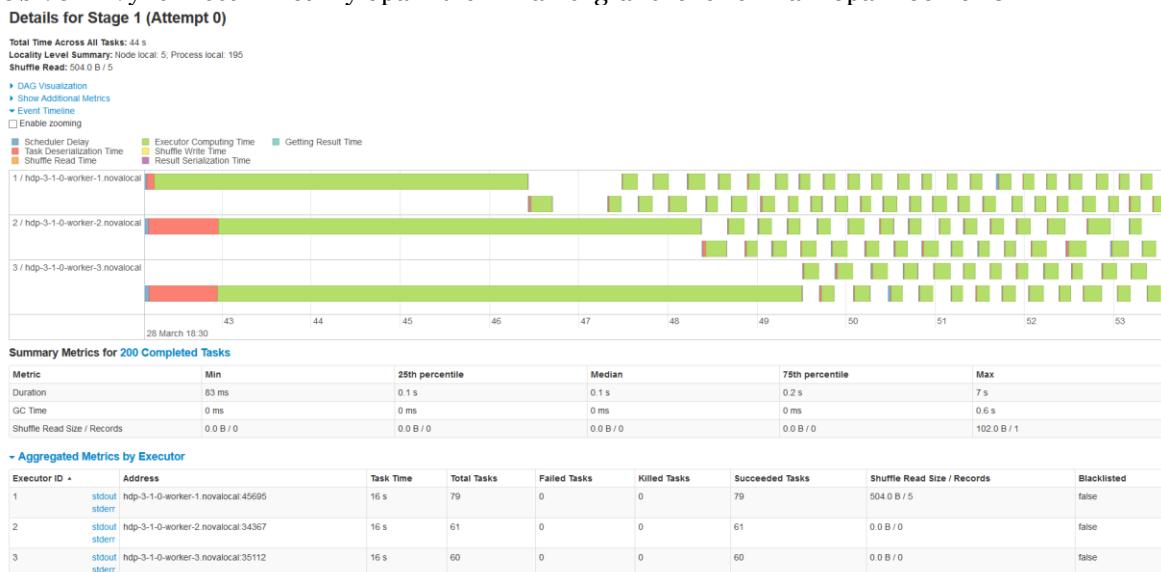
...
72
73 sentiment_metrics_df = tweets_df\
74 .groupBy(tweets_df.tag)\ 
75 .agg(
76     sum(tweets_df.score).alias("Celkový sentiment"),
77     sum(tweets_df.negative).alias("Počet negativních Tweetů"),
78     sum(tweets_df.neutral).alias("Počet neutrálních Tweetů"),
79     sum(tweets_df.positive).alias("Počet pozitivních Tweetů"),
80     count(tweets_df.tag).alias("Celkový počet Tweetů")
81 )
82
83 sentiment_metrics_results = sentiment_metrics_df\
84 .writeStream\
85 .outputMode("complete")\
86 .format("console")\
87 .option("truncate", "false")\
88 .option("numRows", 5) \
89 .queryName("sentiment_metrics_results")\
90 .start()

```

Výstup lze přesměrovat (ASF, 2019w) do souboru (file), do fronty Kafka (kafka), na obrazovku, respektive standardní výstup (console) či do paměti (memory), a to v některém z výstupních režimů (append, complete, update).

Výše uvedená úloha byla nejprve spuštěna ihned po zahájení zrcadlení data z Kafka mimo cluster distribuce do Kafka brokeru distribuce. Zde se však ukázalo, že doba trvání úlohy v podstatě odpovídala době, po kterou Kafka MirrorMaker data přesouval. Spark úloha tedy v podstatě pouze čekala, až budou data přijata a prakticky okamžitě po získání poslední množiny dat byla data zpracována a úloha dokončena. Jinými slovy, velikost zpracovávané množiny byla taková, že bylo v podstatě možné Spark úlohu realizovat nad celou množinou dat, jíž se v rámci této praktické úlohy podařilo získat. Po prvním zpracování postupně přijímaných dat tak byly jednotlivé Spark úlohy dále prováděny nad celou množinou dat a výkonnostní metriky napříč jednotlivými distribucemi frameworku Hadoop byly hodnoceny právě za realizaci nad celou množinou 500 821 Tweetů.

Obr. 5-11 Výkonnostní metriky Spark úloh v rámci grafického rozhraní Spark Server UI



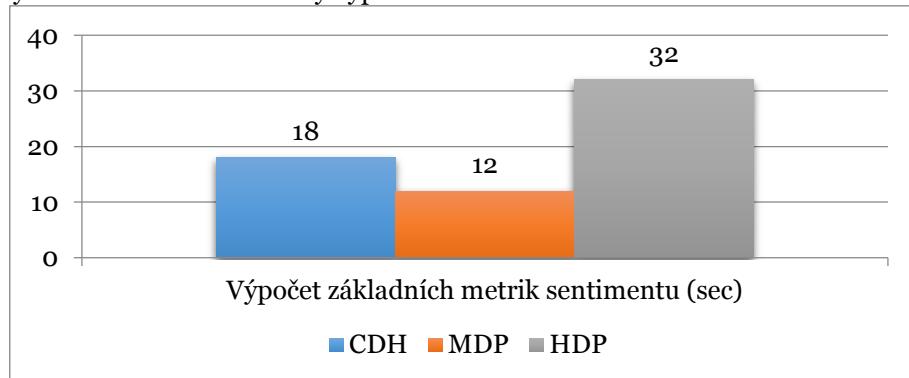
Zdroj: vlastní zpracování

Jak bylo uvedeno, úlohy Spark byly realizovány distribuovaně. Výkonnostní metriky pro jednotlivé úlohy byly získávány z grafického rozhraní Spark Server UI, jehož část znázorňuje Obr. 5-11. Z uvedeného snímku je patrné, že pro Spark úlohu byly skutečně vyčleněny 3 Spark Executors, a že celá úloha byla realizována distribuovaně napříč všemi pracovními uzly v clusteru Hadoop. Obrázek také indikuje zajištěnou úroveň paralelizace, jelikož každý z úkolů realizovaných jednotlivými Spark Executors trval v případě dané Spark úlohy v rámci distribuce HDP 16 vteřin, zatímco celá úloha trvala 32 vteřin. Pro srovnání výkonu v rámci jednotlivých distribucí byla využita metrika doby realizace jednotlivých úloh, která je oproti manuálnímu měření, přesnější, jelikož ji interně monitoruje a eviduje Spark.

Konkrétní naměřené metriky doby trvání výpočtu základních metrik sentimentu v rámci jednotlivých distribucí frameworku Hadoop, které byly předmětem komparace, jsou znázorněny na Obr. 5-12, z něhož je zřejmé, že nejrychleji byla úloha zpracována v rámci distribuce MDP, o třetinu pomaleji v rámci CDH a nejpomaleji v rámci distribuce HDP.

Výpočet agregovaných hodnot 5 metrik sentimentu pro jednotlivé kandidáty z 500 821 Tweetů je tak možné s využitím Spark při načtení dat z Kafka, v případě výše uvedených parametrů clusteru, realizovat v čase od 12 do 32 vteřin.

Obr. 5-12 Výkonnostní charakteristiky výpočtu základních metrik sentimentu

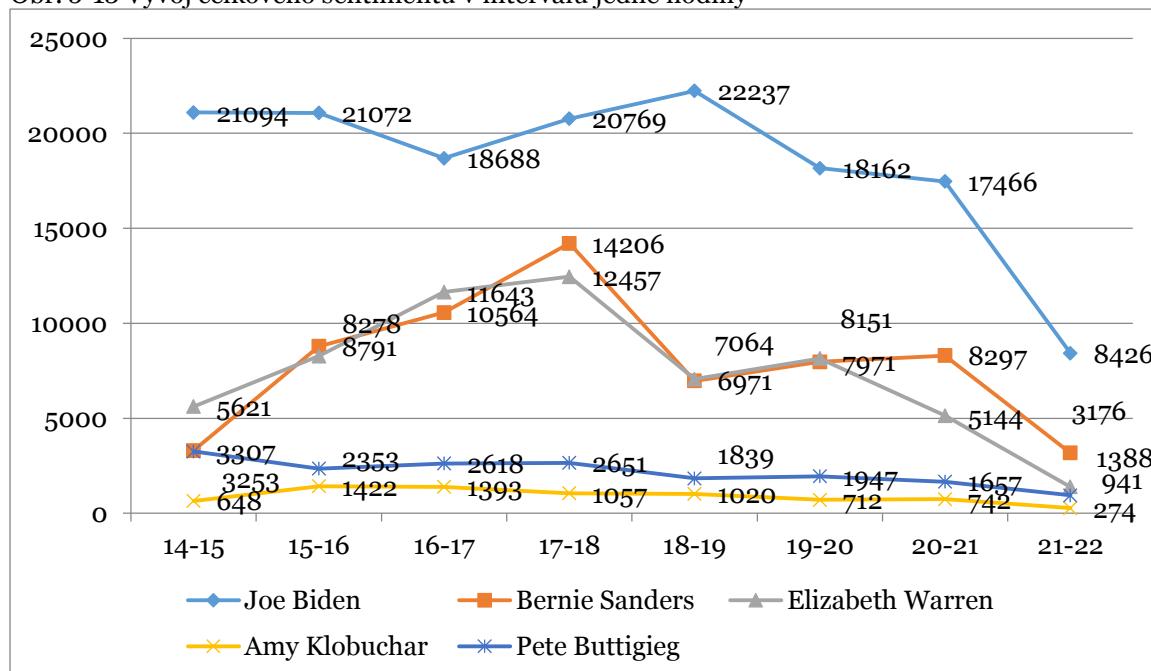


Zdroj: vlastní zpracování

5.4.3 Aplikace window operací

V rámci realizace praktické úlohy zpracování big data byla dále provedena Spark úloha využívající window operací pro výpočet vývoje všech sledovaných metrik sentimentu v čase, respektive v intervalu jedné hodiny po celou dobu, kdy byla data ze sociální sítě Twitter přijímána. Výsledky jsou znázorněny na Obr. 5-13, Obr. 5-14, Obr. 5-15, Obr. 5-16, Obr. 5-17.

Obr. 5-13 Vývoj celkového sentimentu v intervalu jedné hodiny



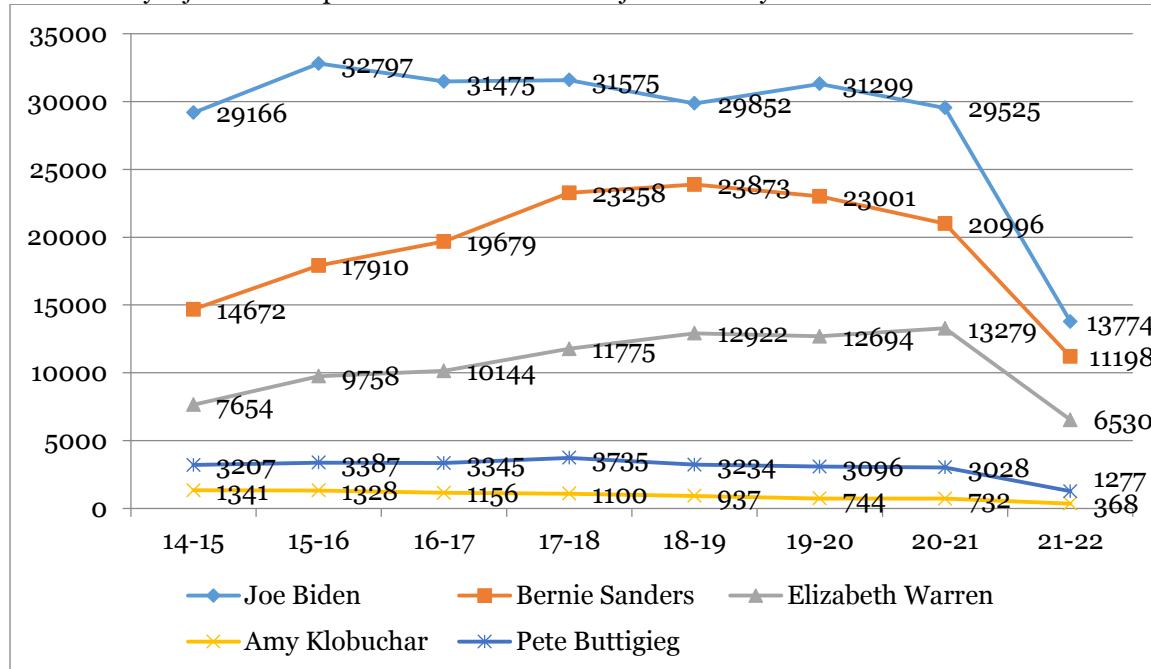
Zdroj: vlastní zpracování

Hodnota celkového sentimentu (suma skóre sentimentu u jednotlivých Tweetů zmiňujících daného kandidáta) u všech kandidátů v podstatě klesala. V časovém okně

mezi 17-18 hodinou byl zaznamenán skokový růst v případě Berniego Sanderse a Elizabeth Warren.

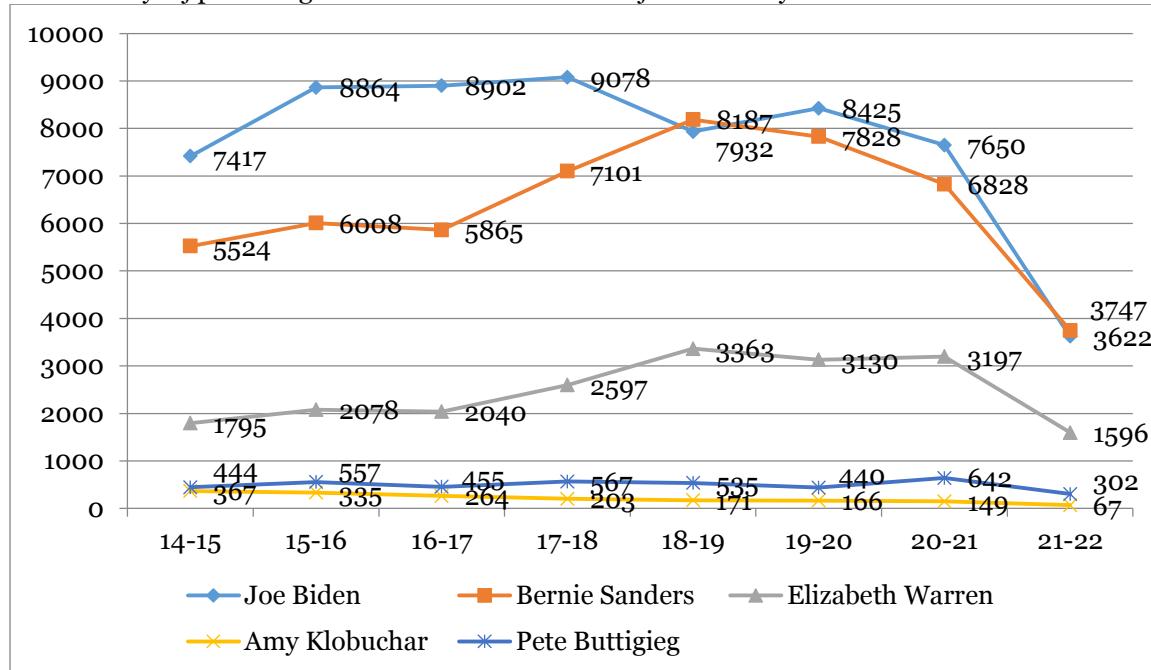
Analogicky došlo k růstu u Joe Bidena v okně mezi 18-19 hodinou. Podobné úrovně sentimentu dosahovali kandidáti Bernie Sanders a Elizabeth Warren, dále pak Amy Klobuchar a Pete Buttigieg. Nejvyšší hodnota celkového sentimentu byla po celou dobu příjmu dat zjištěna u Joe Bidena, který také ve volbách zvítězil.

Obr. 5-14 Vývoj celkového počtu Tweetů v intervalu jedné hodiny



Zdroj: vlastní zpracování

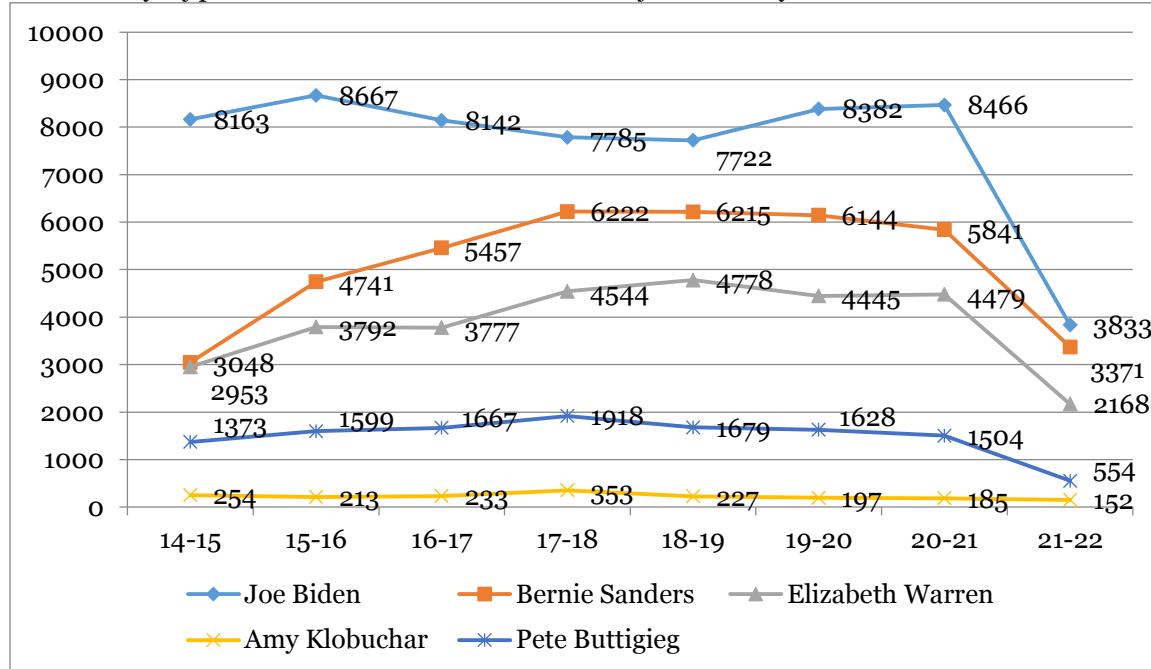
Obr. 5-15 Vývoj počtu negativních Tweetů v intervalu jedné hodiny



Zdroj: vlastní zpracování

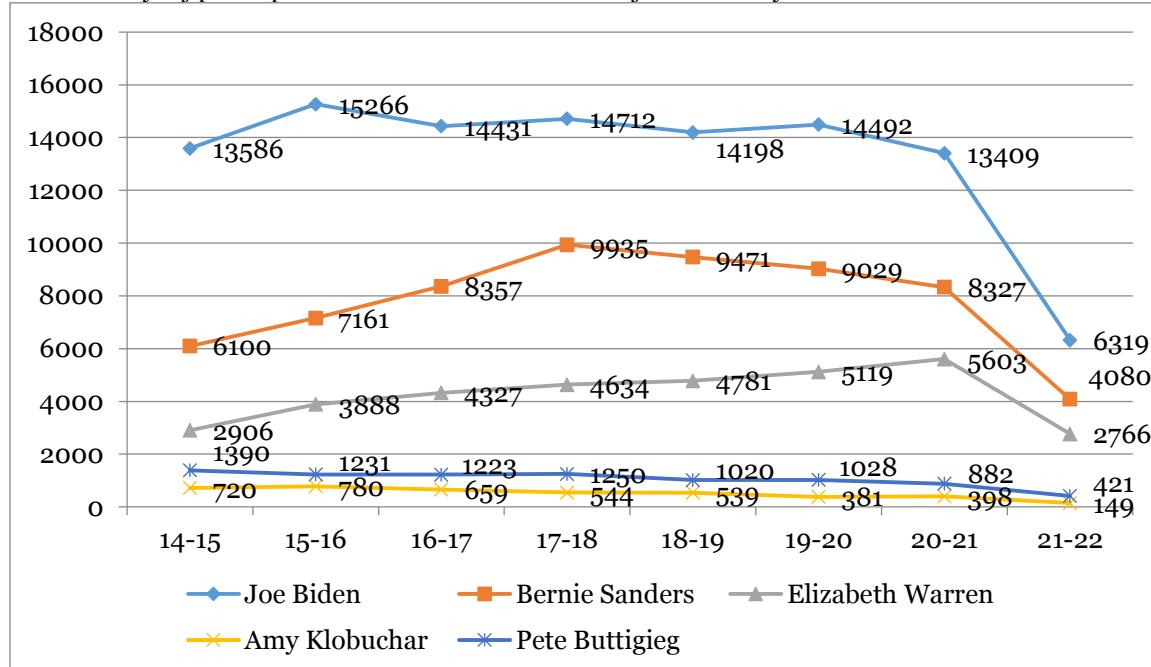
Z hlediska celkového počtu Tweetů je zřejmé, že jednotliví kandidáti po celou dobu příjmu dat dosahovali úrovně počtu Tweetů odpovídající výslednému pořadí kandidátů. Nejvyšší počet Tweetů vykazoval Joe Biden, následován Bernie Sandersem a dále Elizabeth Warren, Pete Buttigieg, Amy Klobuchar.

Obr. 5-16 Vývoj počtu neutrálních Tweetů v intervalu jedné hodiny



Zdroj: vlastní zpracování

Obr. 5-17 Vývoj počtu pozitivních Tweetů v intervalu jedné hodiny



Zdroj: vlastní zpracování

V případě metrik počtu negativních, neutrálních a pozitivních Tweetů je patrné, že jejich vývoj kopíruje trend celkového počtu Tweetů, a to především pokud jde o „pořadí“ jednotlivých kandidátů. Jedinou výjimku tvoří počet negativních příspěvků v časovém

okně 18:00 – 19:00 hodin, kdy byl u Bernieho Sanderse zaznamenán vyšší počet negativních Tweetů, než dosahoval Joe Biden.

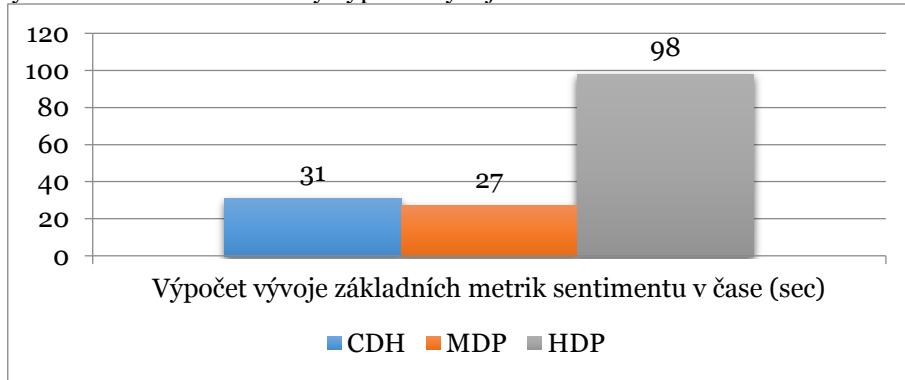
Na všech grafech je patrný výrazný pokles v posledním časovém okně. Tento nesouvisí s frekvencí publikování Tweetů ze strany uživatelů, ale spíše je projevem dosažení, respektive blížícího se překročení kvóty Tweetů, které lze v rámci volně dostupné varianty Twitter Filtered Stream API získat.

Výpis 5-5 spark_sentiment_static_quries.py – výpočet vývoje základních metrik sentimentu ze získaných Tweetů v čase (vlastní zpracování dle (ASF, 2019w))

```
...
91
92 windowed_sentiment_metrics_df = tweets_df\
93 .groupBy(tweets_df.tag, window(tweets_df.created_at, "60 minutes"))\
94 .agg(
95     sum(tweets_df.score).alias("Celkový sentiment"),
96     sum(tweets_df.negative).alias("Počet negativních Tweetů"),
97     sum(tweets_df.neutral).alias("Počet neutrálních Tweetů"),
98     sum(tweets_df.positive).alias("Počet pozitivních Tweetů"),
99     count(tweets_df.tag).alias("Celkový počet Tweetů")
100 )\
101 .orderBy("window")
102
103 windowed_sentiment_metrics_results = windowed_sentiment_metrics_df\
104 .writeStream\
105 .outputMode("complete")\
106 .format("console")\
107 .option("truncate", "false")\
108 .option("numRows", 40)\
109 .queryName("windowed_sentiment_metrics_results")\
110 .start()
```

Kód, jehož prostřednictvím byly získány údaje jednotlivých metrik, obsahuje Výpis 5-5. Zde je evidentní, že oproti základnímu programovacímu modelu Spark Structured Streaming jsou příspěvky seskupovány a ještě navíc seřazeny právě podle časového okna.

Obr. 5-18 Výkonnostní charakteristiky výpočtu vývoje metrik sentimentu v čase



Zdroj: vlastní zpracování

Způsob výpočtu a použití agregace nad proudem přicházejících dat zůstává prakticky identický. Aplikaci funkcionality operací nad časovými okny lze označit za velice intuitivní. Výkonnostní metriky doby zpracování Spark úlohy zahrnující výpočet vývoje pěti sledovaných metrik sentimentu v časových intervalech jedné hodiny znázorňuje Kód, jehož prostřednictvím byly získány údaje jednotlivých metrik, obsahuje Výpis 5-5. Zde je evidentní, že oproti základnímu programovacímu modelu Spark Structured Streaming jsou příspěvky seskupovány a ještě navíc seřazeny právě podle časového okna.

Obr. 5-18.

Oproti kalkulaci sumárních hodnot se doba trvání výpočtu hodnot za jednotlivé intervaly více než zdvojnásobila. Také v tomto případě byla úloha realizována nejrychleji v případě distribuce MDP, následované CDH a HDP. V rámci distribuce HDP zpracování dané úlohy Spark trvalo více než třikrát déle než v případě distribucí CDH a MDP. V případě dané úlohy Spark byl zaznamenán značný výkonnostní rozdíl v případě distribuce HDP oproti MDP a CDH.

5.5 Shrnutí

V rámci této kapitoly byla realizována praktická úloha zpracování big data. Jednotlivé distribuce frameworku Hadoop, které byly předmětem komparace, byly nasazovány postupně, pokud možno v identické konfiguraci tak, aby byla zajištěna srovnatelnost výkonu jednotlivých distribucí. Každá ze zkoumaných distribucí poskytuje vlastní službu pro nasazení, správu a monitoring (případně s využitím externích open source komponent) clusteru Hadoop. Určitá specifika jsou charakteristická především pro distribuci MDP, vyplývají z architektury distribuce a současně determinují vysší nároky na první výkonnostně-kapacitní parametry použité infrastruktury. Všechny distribuce lze při důsledném dodržení pokynů instalace a především přípravy prostředí bez větších problémů nasadit a využít. Pokud jde o rychlosť a přívětivost celého procesu nasazení, jeví se jako nejlepší z distribucí HDP, následována CDH a MDP. Tento názor je nicméně subjektivní a může být ovlivněn mimo jiné tím, že autor práce disponuje větším rozsahem zkušeností právě s distribucí HDP.

Jako zdroj big data bylo využito Twitter Filtered Stream API v rozsahu, jehož prostřednictvím bylo přijato půl milionu Tweetů týkajících se vybraných kandidátů v primárních volbách, jež se konaly ve Spojených státech amerických v rámci tzv. volebního superúterý. Uvedené API umožnilo příjem dat skutečně v reálném čase, jelikož se nejedná o „tradiční“ API, kterému odesíláme požadavek, a jež vrací odpověď, ale naopak je založeno na tom, že je navázáno trvalé spojení, v jehož rámci API odesílá Tweety odpovídající definovaným pravidlům (v případě praktické úlohy se jednalo o název kandidáta a přirozený jazyk, v němž byl příspěvek vytvořen a publikován). Současně byly v rámci příjmu dat s využitím knihovny afinn vypočítány hodnoty, tj. skóre a polarita sentimentu pro každý jednotlivý Tweet.

Data byla přijata do dedikované platformy pro příjem a distribuci proudů data Kafka a následně postupně replikována do Kafka v rámci jednotlivých distribucí Hadoop, aby bylo

zajištěno, že výpočty a analýza dat budou provedeny nad identickou množinou dat. Pro replikaci byl využit nástroj Kafka Mirror Maker. Celý proces příjmu dat v rámci jednotlivých distribucí byl měřen a výsledky porovnány, přičemž bylo zjištěno, že nejrychleji se podařilo přijmout data v rámci distribuce MDP, která byla ale oproti ostatním distribucím náročnější na využití zdrojů. Výsledky byly nicméně v podstatě srovnatelné u všech distribucí. Současně je nezbytné uvést, že v žádném případě neilustrují možnosti příjmu dat na straně Kafka, protože byla měřena replikace, nikoli příjem jako takový. Půl milionu Tweetů je v praxi možné do Kafka uložit v řádu vteřin, nicméně zdroj big data příspěvky vracel podstatně nižší rychlosti, a to především v závislosti na tom, jak rychle uživatelé příspěvky na sociální síti Twitter publikovali.

Pro analýzu získaných dat byl použit projekt Spark, konkrétně novější API Spark Structured Streaming, které se více blíží zpracování big data v téměř reálném čase (o zpracování dat v reálném čase v pravém smyslu slova nelze hovořit, protože Spark je založen na konceptu kontinuálního zpracování mikrodávek). Přijatá data byla v rámci Spark načtena z Kafka, transformována a následně analyzována. Realizovány byly dvě Spark úlohy. První Spark job zahrnoval výpočet klíčových metrik sentimentu, tj. celkovou hodnotu sentimentu, počet příspěvků a počet příspěvků v jednotlivých kategoriích sentimentu, agregované u všech kandidátů za celou množinu získaných příspěvků ze sociální sítě Twitter. Druhý Spark job zajistil výpočet uvedených hodnot pro jednotlivé kandidáty v rámci časových oken jedné hodiny po celou dobu, kdy byl realizován příjem big data ze sociální sítě Twitter.

Obě úlohy byly realizovány napříč pracovními uzly clusteru, takže nebylo možné měřit výkon na stroji, kde se nacházel Spark klient. Proto byly pro výkonnostní srovnání využity hodnoty doby trvání jednotlivých úloh dostupné v grafickém rozhraní Spark serveru. Bylo zjištěno, že v obou případech byly výsledky získány nejrychleji v případě distribuce MDP, dále CDH a HDP. V tomto případě však výsledky za srovnatelné označit nelze, jelikož v případě HDP byla doba realizace signifikantně delší, zejména u druhé Spark úlohy a distribuce HDP za MDP a CDH řádově zaostávala.

V rámci realizace praktické úlohy zpracování big data, která byla předmětem této kapitoly se podařilo nasadit jednotlivé zkoumané distribuce frameworku Hadoop, v reálném čase přijmout data ze sociální sítě Twitter do Kafka, postupně distribuovat, respektive replikovat data do instance Kafka v rámci jednotlivých distribucí a následně téměř v reálném čase získaná data analyzovat. Ze zkoumaných distribucí lze za nejvýkonnější označit MDP, jež byla následována CDH a HDP. Z hlediska uživatelské přívětivosti, pokud se jedná o nasazení a provoz distribuce, lze subjektivně za nejlepší označit CDH a dále HDP a MDP.

6 Komparace vybraných distribucí frameworku Apache Hadoop

Tato kapitola obsahuje finální komparaci zkoumaných distribucí frameworku Apache Hadoop, v jejímž rámci budou využity prezentované teoretické poznatky a především závěry a informace, jež byly zjištěny v průběhu realizace praktické úlohy. V první části kapitoly budou nejprve vymezena metodologická východiska a specifikována zvolená kritéria komparace distribucí Hadoop. V další části budou při striktní aplikaci zvolené metody komparace aplikována zvolená kritéria. Závěrečná část prezentuje vyhodnocení provedené komparace distribucí frameworku Apache Hadoop.

6.1 Vymezení metody a kritérií komparace

Komparace distribucí frameworku Apache Hadoop je v rámci této práce realizována primárně s cílem doporučit potenciálním uživatelům optimální distribuci, kterou by bylo možné snadno a relativně rychle nasadit a využít pro ad-hoc nebo kontinuální zajištění procesu příjmu big data a/nebo některé z jeho částí. Jelikož je na trhu dostupných více distribucí frameworku Apache, musí potenciální uživatel uvažující o využití některé z nich pro řešení zpracování big data, nutně projít rozhodovacím procesem zahrnujícím (Richter, a další, 2016) řešení rozhodovacích problémů s více variantami, posuzování jednotlivých variant a výběr optimální varianty. Řešený rozhodovací problém představuje v kontextu této práce právě výběr optimální distribuce frameworku Hadoop pro zajištění procesu zpracování big data, přičemž jednotlivé distribuce reprezentují dostupné varianty, které je nutné posoudit a zvolit optimální variantu.

Pro realizaci komparace distribucí frameworku Apache Hadoop byly zvoleny metody (Švecová, a další, 2016) vícekriteriálního rozhodování aplikované s cílem stanovení preferenčního uspořádání komparovaných variant, a to konkrétně metoda stanovení vah kritérií metodou postupného rozvrhu vah a metoda vícekriteriálního hodnocení variant stanovením hodnoty (užitku) variant s využitím přímého stanovení dílčích hodnocení jednotlivých kritérií.

Při výběru kritérií komparace distribuce frameworku Hadoop byla konfrontována množina kritérií aplikovaných v rámci vybraných odborných prací z dané problémové oblasti, jež byly předmětem Rešerše literatury s (Richter, a další, 2016) obecnými zásadami pro tvorbu souboru kritérií a specifickými požadavky na soubor kritérií, mezi které patří úplnost, minimální rozsah, operacionalita, neredundance a nezávislost. Původní záměr vytvoření rozsáhlé množiny kritérií byl i s ohledem na výsledky zjištěné v rámci předchozích kapitol přehodnocen a bylo upřednostněno vymezení užší množiny klíčových koherentních kritérií komparace. Řada kritérií nabývajících napříč jednotlivými distribucemi prakticky identických hodnot proto byla s ohledem na celkový význam nakonec vypuštěna. Jednalo se například o dostupnost a funkcionality instalátoru

distribuce a poskytovanou funkcionalitu obecně, jelikož všechny zkoumané distribuce Hadoop pokrývají v podstatě téměř shodnou množinou funkcionality z oblasti zpracování big data, pouze prostřednictvím odlišné množiny projektů. Náklady na pořízení distribuce nebyly mezi kritéria zahrnuty, protože ceny za příslušnou licenci a/nebo podporu nejsou veřejně dostupné.³² Zvolená kritéria jsou uvedena v Tab. 6.1.

Tab. 6.1 Zvolená kritéria komparace, stanovené váhy kritérií

Skupina kritérií	Váha kritérií	Kritérium	Váhy Kritérií	Výsledné Váhy
S ₁ Licence	0,3	K ₁ Licence	0,3	0,09
		K ₂ Volně dostupné repositáře	0,3	0,09
		K ₃ Možnost využití pro komerční účely bez licence	0,2	0,06
		K ₄ Podíl open source projektů v distribuci	0,2	0,06
S ₂ Nasazení	0,2	K ₅ Možnost nasazení on-premise	0,2	0,04
		K ₆ Možnost nasazení v cloudu	0,3	0,06
		K ₇ Uživatelská přívětivost nasazení	0,1	0,02
		K ₈ Rychlosť nasazení	0,1	0,02
		K ₉ Hardwarová náročnosť nasazení	0,3	0,06
S ₃ Provoz	0,2	K ₁₀ Dokumentace	0,2	0,04
		K ₁₁ Křivka učení	0,3	0,06
		K ₁₂ Přenositelnost postupů do jiné distribuce	0,3	0,06
		K ₁₃ Uživatelská přívětivost grafického rozhraní	0,2	0,04
S ₄ Výkon	0,3	K ₁₄ Vzorová úloha zpracování big data #1	0,4	0,12
		K ₁₅ Vzorová úloha zpracování big data #2	0,6	0,18

Zdroj: vlastní zpracování

Celkem bylo zvoleno 15 kritérií komparace, která byla zařazena do čtyř kategorií, mezi které patřily oblasti licencování, nasazení, provozu a výkonu distribuce Hadoop. Pro stanovení vah kritérií byla zvolena metoda (Švecová, a další, 2016) postupného rozvrhu vah. Nejprve byly stanoveny vahy pro jednotlivé kategorie kritérií, při dodržení požadavku na normování (součet vah napříč kategoriemi je roven jedné). Následně byly stanoveny rovněž normované vahy jednotlivých specifikovaných kritérií (součet vah kritérií v rámci skupiny je roven jedné). Výsledné vahy jednotlivých kritérií byly získány pronásobením vahy skupiny kritérií a vahy každého jednotlivého kritéria. Výsledné vahy jsou opět normované, jelikož součet výsledných vah kritérií je roven jedné. Zvolená kritéria a vahy přiřazené kritériím jsou subjektivní.

³²⁾ Společnost Cloudera uvádí v rámci veřejného ceníku (Cloudera, 2020b) pro enterprise verzi CDH i HDP cenu 10000\$ za jeden uzel za rok. Společnost MapR nicméně cenu za jednotlivé placené edice distribuce MDP veřejně neprezentuje.

Kategorie kritérií S₁ Licence byla zaměřena na zhodnocení míry otevřenosti/uzavřenosti distribuce Hadoop, tedy typu licence, pod kterou jsou balíky distribuce vydávány, dále zda jsou repositáře distribuce volně dostupné pro vyzkoušení a/nebo využití, možnost využití distribuce pro komerční účely bez zakoupené licence a konečně podíl open source projektů v distribuci. Při využití distribuce zdarma a především v případě investice do pořízení a provozu distribuce je relevantní zvážit, zda se pořízením/využitím určitého řešení uživatel/investor nedostává do situace označované jako tzv. proprietární uzamčení, kdy není možné flow a/nebo data bez větších problémů či investic migrovat k jinému poskytovateli analogického řešení. Vyšší váhy byly přiřazeny kritériím licence a možnosti volného stažení/získání repositářů distribuce, zatímco o jednu desetinu nižší váhy byly přiřazeny kritériím možnosti využití distribuce pro komerční účely bez licence a podílu open source projektů v distribuci.

Druhá kategorie kritérií byla zaměřena na hodnocení procesu nasazení distribucí Hadoop. Možnost relativně snadného a rychlého nasazení distribuce frameworku Hadoop se pro jeho využití v praxi jeví jako velice významná, a to zejména s ohledem na frekventovanou potřebu nasazení více než jednoho clusteru Hadoop (pro různé účely či cílové skupiny uživatelů nebo jako standby zálohu, pro testování, upgrady, atd.). Proto byla do dané skupiny zařazena kritéria možnosti nasazení on-premise, jakožto standardu, dále možnosti nasazení v cloudu (cloud dnes využívají/začínají využívat prakticky všechny kategorie firem bez ohledu na velikost, větší společnosti mohou dokonce provozovat vlastní cloudová prostředí), uživatelské přívětivosti procesu nasazení, rychlosti nasazení a HW nároků na prostředí, do kterého je distribuce nasazována. Nejvyšší váhy byly přiřazeny kritériím možnosti nasazení v cloudu, která se v současné době stává/stala standardem nahrazujícím nasazení v on-premise prostředí a dále hardwarové náročnosti nasazení, která do značné míry determinuje HW nároky pro každý další uzel clusteru po celou dobu životního cyklu clusteru Hadoop. Nižší váha byla přiřazena možnosti nasazení on-premise. Nejnižší váha byla zvolena pro uživatelskou přívětivost procesu nasazení a rychlosť nasazení. V obou případech nepřikládá autor této práce vyšší význam těmto kritériím, přičemž důvodem je skutečnost, že plnohodnotný cluster distribuce Hadoop zatím nepředstavuje prostředí, které by bylo možné nasadit v řádu minut, ale spíše hodin až jednotek dnů. Proto zpravidla takový cluster není, a to ani v případě, že by se jednalo o ad hoc nasazení, které bude po realizaci požadované úlohy odstraněno, nasazován na dobu v řádu hodin, ale spíše dnů až týdnů či měsíců. Určitá tolerance nižší uživatelské přívětivosti či delší doby nasazení, na rozdíl od ostatních kritérií ze skupiny nasazení, je tudíž v daném kontextu relevantní.

Dále byla vymezena kategorie provozu orientovaná primárně na hodnocení smysluplnosti investic lidských zdrojů do provozu jednotlivých distribucí Hadoop. Jak již bylo uvedeno, kritéria funkcionality byla z výsledné množiny kritérií komparace vypuštěna, protože jejich hodnocení by se napříč jednotlivými zkoumanými distribucemi frameworku Hadoop příliš nelišilo. Proto byla zařazena kritéria zaměřená na nutné aktivity související s provozem clusteru Hadoop. V rámci kategorie provoz byla zvolena kritéria dokumentace hodnotící dostupnost a úroveň dokumentace distribuce z hlediska podrobnosti a srozumitelnosti, dále kritérium křivky učení hodnotící, jak rychle je možné do procesu správy clusteru nastoupit a ovládnout potřebné znalosti a kompetence, kritérium přenositelnosti postupů do jiné distribuce vyjadřující smysl investice času a lidských zdrojů do získání znalostí a

kompetencí správy a administrace clusteru distribuce Hadoop a rovněž kritérium uživatelské přívětivosti systému pro nasazení, správu, provoz a monitoring clusteru dané distribuce Hadoop. Pro kritéria křivky učení a přenositelnosti postupů správy do jiné distribuce Hadoop byly zvoleny váhy vyšší, zatímco pro kritéria dokumentace a uživatelské přívětivosti grafického rozhraní byly zvoleny váhy o jednu desetinu nižší.

Poslední kategorie kritérií komparace zahrnovala poznatky zjištěné v rámci realizace praktické úlohy zpracování big data a tedy kritéria výkonu při realizaci modelových Spark úloh. Oblast výkonu představovala jednu z klíčových kategorií komparace.

6.2 Aplikace kritérií komparace

Kritéria komparace byla na distribuce frameworku Hadoop aplikována v souladu s (Švecová, a další, 2016) jednoduchou metodou stanovení hodnoty (užitku) variant, kdy celkové hodnocení variant je stanoveno jako vážený součet dílčích ohodnocení variant vzhledem k jednotlivým kritériím podle následujícího vzorce (Švecová, a další, 2016):

$$H^j = \sum_{i=1}^n v_i \times h_i^j \text{ pro } j = 1, 2, \dots, m,$$

kde H^j celkové ohodnocení (hodnota) j -té varianty,
 v_i váha i -tého kritéria,
 h_i^j dílčí ohodnocení j -té varianty vzhledem k vzhledem k i -tému kritériu,
 n počet kritérií hodnocení,
 m počet variant.

Dílčí hodnocení jednotlivých kritérií bylo provedeno metodou (Švecová, a další, 2016) založenou na přímém stanovení dílčích hodnocení, která byla stanovena přímo přiřazením bodů ze stupnice 1 až 10, na níž 1 odpovídá nejhoršímu hodnocení a 10 hodnocení nejlepšímu.

Kritéria komparace byla aplikována na základě uvedených metod vícekriteriálního hodnocení variant. Výsledky aplikace kritérií komparace v souladu s uvedenými metodami jsou uvedeny v Tab. 6.2. Pro každé kritérium byly jednotlivým variantám (distribucím) přímo přiděleny body ze stanovené stupnice (sloupec b u jednotlivých distribucí v Tab. 6.2). Dílčí hodnocení jednotlivých variant z hlediska zvolených kritérií byla získána pronásobením přiřazených bodů s váhami stanovenými pro jednotlivá kritéria (sloupec h u jednotlivých distribucí v Tab. 6.2). Celkové hodnocení jednotlivých variant bylo získáno váženým součtem dílčích hodnocení.

První kritérium bylo ohodnoceno 10 body v případě distribucí CDH a HDP, protože byly poskytovány na základě licence Apache 2.0, na rozdíl od distribuce MDP, která byla dostupná v souladu s proprietární licencí (MapR EULA) a proto jí bylo v případě tohoto kritéria přiřazeno pouze minimální hodnocení, tedy 1 bod. Druhým kritériem hodnocení bylo, zda byly repositáře distribuce veřejně a tedy volně dostupné. Zde bylo maximální hodnocení přiřazeno distribuci MDP, v jejímž případě byly repositáře veřejně dostupné.

Obtížnější bylo stanovení bodů pro distribuce CDH a HDP z hlediska kritéria K₂. V případě verzí, které byly nasazeny v rámci realizace praktické úlohy zpracování big data, se jednalo o veřejně volně dostupné repositáře. Jak již ale bylo uvedeno, v případě nových verzí již tato skutečnost neplatí. Přestože tedy byly komparovány jiné verze, nakonec byl vývoj v rámci nových verzí daných distribucí Hadoop zohledněn a nebyl přiřazen maximální ani minimální počet bodů, ale střední hodnota 5 bodů. V případě kritéria možnosti využití distribuce pro komerční účely bez zakoupené licence byl analogicky reflektován vývoj u distribucí CDH a HDP a opět jim bylo přiřazeno 5 bodů, zatímco distribuci MDP bylo z hlediska daného kritéria přiřazeno minimální hodnocení 1 bodu, jelikož EULA, kterou uživatel při nasazení clusteru distribuce MDP musí odsouhlasit, takové využití striktně vylučuje.

Tab. 6.2 Výsledky aplikace zvolených kritérií komparace

Kritérium	Váha	CDH		MDP		HDP	
		b	h	b	h	b	h
K ₁	0,09	10	0,90	1	0,09	10	0,90
K ₂	0,09	5	0,45	10	0,90	5	0,45
K ₃	0,06	5	0,30	1	0,06	5	0,30
K ₄	0,06	6	0,36	3	0,18	9	0,54
K ₅	0,04	10	0,40	10	0,40	10	0,40
K ₆	0,06	10	0,60	10	0,60	10	0,60
K ₇	0,02	7	0,14	6	0,12	8	0,16
K ₈	0,02	7	0,14	6	0,10	8	0,16
K ₉	0,06	8	0,48	5	0,30	8	0,48
K ₁₀	0,04	8	0,32	6	0,24	10	0,40
K ₁₁	0,06	8	0,48	5	0,30	8	0,48
K ₁₂	0,06	8	0,48	4	0,24	8	0,48
K ₁₃	0,04	8	0,32	7	0,28	9	0,36
K ₁₄	0,12	8	0,96	10	1,20	6	0,72
K ₁₅	0,18	8	1,44	10	1,80	5	0,90
Suma	1	116	7,77	94	6,81	119	7,33
Pořadí			1		3		2

Pozn: b – body, h – hodnocení; hodnocení provedeno za komparované verze distribucí, pokud v textu není uvedeno jinak

Zdroj: vlastní zpracování

V případě posledního kritéria ze skupiny S₁ byly přiřazeny body rovnoměrně odstupňovány pro jednotlivé distribuce Hadoop adekvátně k reálnému podílu open source projektů v množině tvořící danou distribuci. Nejvyšší dílčí hodnocení bylo přiřazeno téměř 100% open source distribuci HDP, nižší hodnocení distribuci CDH a nejnižší hodnocení

distribuci MDP vykazující ryze proprietární charakter v rovině jádra distribuce doplněného o určitou množinu open source projektů.

Možnost nasazení distribuce on-premise a v prostředí cloutu byla u všech distribucí hodnocena shodně, a to maximálním počtem bodů. V případě uživatelské přívětivosti byly jednotlivé varianty odstupňovány po jednom bodu, aby byl snížen vliv subjektivních preferencí, přičemž nejvíše byla z daného hlediska hodnocena distribuce HDP, dále CDH a konečně MDP. Analogicky bylo provedeno dílčí hodnocení z hlediska rychlosti nasazení distribuce, přičemž motiv v tomto případě představovalo hodnocení kritéria s ohledem na skutečnost, že distribuce je zpravidla (i v případě ad hoc zaměření) nasazována na delší dobu. Poslední kritérium ze skupiny zaměřené na nasazení distribuce bylo hodnoceno shodně pozitivně v případě distribucí CDH a HDP a méně pozitivně v případě distribuce MDP, kdy v rámci realizace praktické úlohy zpracování big data vyplynulo, že pro nasazení distribuce je nezbytné dedikovat vyšší objem HW zdrojů.

Z hlediska dokumentace byla maximálním počtem bodů ohodnocena distribuce HDP, dále pak vždy o dva body méně získaly distribuce CDH a MPD. Dílčí hodnocení kritéria křivky učení bylo v případě distribucí CDH a HDP nastaveno na 8, pro distribuci MDP na 5 bodů, a to především s ohledem na podíl proprietárních „nestandardních“ technologií. Podobně bylo hodnoceno kritérium přenositelnosti postupů správy získaných v rámci jedné distribuce do prostředí distribuce jiné. Úroveň přidělených bodů byla u distribucí CDH a HDP identická, v případě MDP poloviční, jelikož množství postupů symptomatických pro jednotlivé proprietární reimplementation klíčových projektů jádra distribuce MDP bylo možné v rámci jiných distribucí Hadoop využít pouze částečně nebo vůbec. Uživatelská přívětivost grafického rozhraní systému pro nasazení, správu, provoz a monitoring clusteru distribuce Hadoop byla hodnocena s odstupňováním po jednom bodu, nejlépe v případě HDP, dále CDH a nakonec MDP.

V rámci skupiny kritérií S₄ byly body přiřazovány u dvou výkonnostních kritérií odpovídajících typovým Spark úlohám provedeným v rámci realizace praktické úlohy zpracování big data. V případě obou kritérií bylo dílčí hodnocení provedeno tak, aby maximálně korespondovalo s výkonnostními charakteristikami zjištěnými právě v průběhu realizace praktické úlohy zpracování big data.

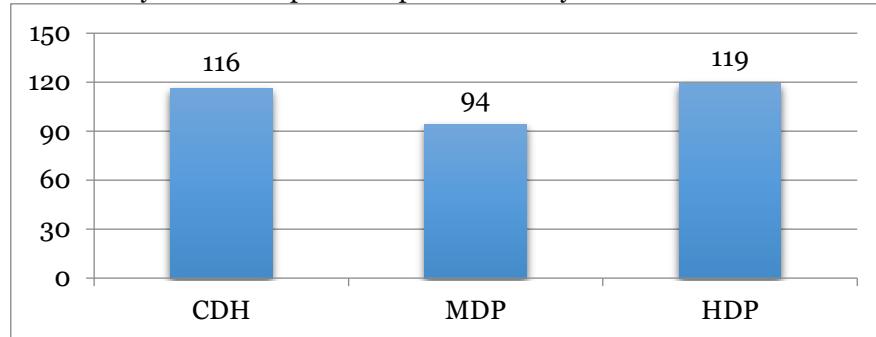
6.3 Vyhodnocení výsledků komparace

Aplikací stanovených kritérií komparace při striktním uplatnění uvedených metod vícekriteriálního hodnocení variant bylo zjištěno preferované pořadí jednotlivých hodnocených variant, tj. komparovaných distribucí frameworku Hadoop.

Výsledné pořadí se v důsledku stanovených vah pro skupiny kritérií a jednotlivá kritéria lišilo od pořadí před započtením vah. Z Obr. 6-1 je patrné, že nejvíce bodů bylo přiděleno distribuci HDP a dále pak CDH a MDP. Bodový rozestup mezi nejlépe a nejhůře hodnocenou distribucí nebyl značný a činil pouze 25 bodů, což odpovídalo 21%. Rozdíl mezi body přidělenými první a druhé distribuci činil pouze 3 body, tj. 2,5%. Signifikantně

méně bodů bylo přiděleno distribuci MDP, a to i přes výrazně lepší výsledky v rovině výkonnostních charakteristik.

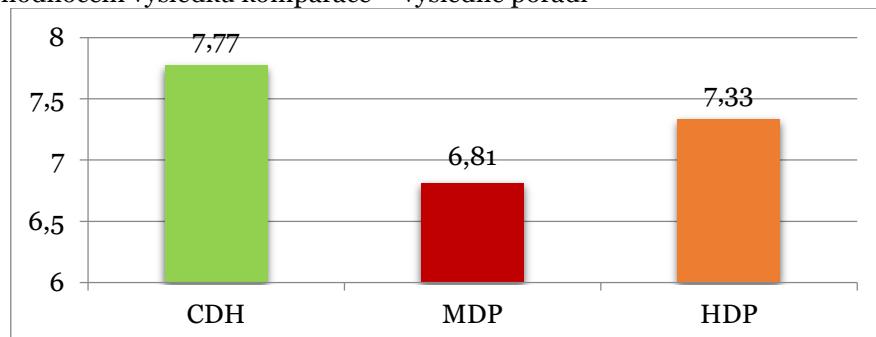
Obr. 6-1 Vyhodnocení výsledků komparace – přidělené body



Zdroj: vlastní zpracování

Uvedené pořadí se však změnilo po započtení vah jednotlivých kritérií, jak je to znázorněno na Obr. 6-2. Jako ideální varianta byla hodnocena distribuce CDH, následovaná distribucí HDP a konečně distribucí MDP. Ani v případě finálního pořadí jednotlivých komparovaných distribucí nebyl odstup mezi jednotlivými variantami příliš signifikantní, jelikož distribuce HDP skončila s hodnocením pouze o 5,7% nižším a distribuce MDP o 12,4%.

Obr. 6-2 Vyhodnocení výsledků komparace – výsledné pořadí



Zdroj: vlastní zpracování

V rámci komparace distribucí frameworku Apache Hadoop bylo zjištěno, že nejlepší variantu dle zvolených kritérií komparace představovala distribuce CDH. Na druhém místě pouze s minimálním odstupem skončila distribuce HDP a jako nejméně příznivá varianta byla identifikována distribuce MDP.

6.4 Shrnutí

V rámci této kapitoly byla s využitím metod vícekriteriálního rozhodování provedena komparace distribucí frameworku Hadoop. Na základě stanovených kritérií při aplikaci zvolených metod bylo zjištěno, že nejvýhodnější variantu ze zkoumaných distribucí frameworku Hadoop reprezentovala distribuce CDH, následovaná HDP a MDP. Výsledky komparace byly nicméně napříč jednotlivými distribucemi natolik blízké, že pro účely dané typové úlohy zpracování big data lze v podstatě zvolit kteroukoli z komparovaných

distribucí frameworku Hadoop, jelikož žádné z komparovaných řešení ostatní signifikantně kvalitativně ani kvantitativně z celkového pohledu nepřevyšuje a bez ohledu na zvolenou distribuci bude v případě zvolené typové úlohy zpracování big data vždy dosaženo požadovaných výsledků.

Distribuce CDH, která byla v rámci této kapitoly vyhodnocena jako nejvýhodnější varianta, představuje velice solidní řešení, které je možné bez problémů použít pro takřka jakýkoli use case z oblasti zpracování big data, a to i v případě, že uživatel zvolí pouze komunitní free licenci. Skutečná síla distribuce spočívá v pokročilé funkcionalitě a integraci dalších doplňkových řešení z produktového portfolia společnosti Cloudera. V případě využití enterprise edice společně s dalšími doplňkovými řešeními CDH představuje nejlepší a nejvíce komplexní řešení, jaké je v současnosti na trhu distribucí frameworku Hadoop k dispozici.

Řešení HDP, které bylo hodnoceno jako druhá nejlepší varianta a jež v současné době zastřešuje rovněž společnost Cloudera, představuje také etablovanou distribuci, kterou je vhodné zvolit zejména v případě, že uživatel/zákazník hledá řešení s nejvyšším podílem open source projektů s cílem eliminovat v maximální možné míře proprietární technologie. Současně se jedná o distribuci, která v případě využití volně dostupné edice determinuje zřejmě nejnižší náklady na nasazení a provoz. V případě volby distribuce HDP je nicméně nutné respektovat, že z hlediska výkonnostních charakteristik nebude dosaženo takových výsledků jako v případě distribuce CDH a především MDP. I přesto je možné distribuci HDP v řadě případů bez problémů použít.

Jako nejméně výhodná varianta byla v rámci této kapitoly hodnocena distribuce frameworku Hadoop společnosti MapR Technologies, kterou v současné době vlastní společnost HPE. Tato distribuce reprezentuje řešení zformované nad jádrem tvořeným množinou proprietárních reimplementací klíčových služeb distribuce frameworku Hadoop (distribuovaný souborový systém, NoSQL databáze pro Hadoop a platforma pro distribuovaný příjem a distribuci dat). Distribuce MDP je oproti ostatním komparovaným řešením vhodná primárně pro případy, kdy je nutné bez ohledu na podíl open source projektů tvořících distribuci dosahovat řádově nejlepších výsledků z hlediska výkonu a rychlosti zpracování big data. Jak bylo uvedeno, není cena za licence distribuce MDP veřejně dostupná, takže nelze plně zhodnotit, zda tato distribuce determinuje skutečně nejvyšší náklady ze všech komparovaných řešení.

Z provedené komparace distribucí frameworku Hadoop vyplývá, že z hlediska zvolené typové úlohy zpracování big data lze pro dosažení požadovaných výsledků zvolit kterékoli z posuzovaných řešení. Jako nejlepší varianta byla vyhodnocena distribuce CDH, dále pak HDP a MDP. Potenciální uživatel/zákazník využívající distribuci frameworku Hadoop by proto měl v procesu výběru distribuce Hadoop reflektovat především skutečné celkové náklady na distribuci frameworku Apache Hadoop.

Závěr

Předmětem této diplomové práce byla komparace distribucí frameworku Apache Hadoop. V úvodu práce byly stanoveny dva hlavní cíle. První hlavní cíl, spočívající v analýze dostupných distribucí frameworku Apache Hadoop, byl úspěšně splněn v rámci teoretické části práce, tj. druhé až čtvrté kapitoly, které byly zaměřeny na oblast big data, framework Apache Hadoop a distribuce frameworku Apache Hadoop. Druhý hlavní cíl, zaměřený na provedení komparace vybraných distribucí frameworku Apache Hadoop, byl úspěšně splněn realizací praktické části práce, tj. páté a šesté kapitoly, v jejichž rámci byla provedena typová úloha příjmu a zpracování big data v (témař) reálném čase a komparace distribucí CDH, MDP a HDP.

Každý ze zmíněných hlavních cílů byl dekomponován na tři dílkové cíle. První dílkový cíl, představení oblasti big data, byl naplněn v rámci druhé kapitoly. V této části textu byla podrobně charakterizována a představena oblast big data, v jejímž rámci nachází framework Hadoop v praxi uplatnění. Nejprve byly vymezeny základní pojmy a charakteristiky big data. Dále byl popsán proces analýzy big data a nastíněny koncepty a technologie pro analýzu big data. V závěru kapitoly byly uvedeny příklady analýzy big data v praxi.

Další dílkový cíl zahrnující detailní charakteristiku frameworku a ekosystému Apache Hadoop byl splněn v rámci třetí kapitoly. Tato kapitola byla věnována frameworku Apache Hadoop. Nejprve byly nastíněny okolnosti vzniku a praktického řešení potřeb efektivního uložení a zpracování big data a následně detailně popsán vznik a rozvoj frameworku Apache Hadoop včetně aktuálního stavu projektu ve verzi Hadoop 3. Součástí byla také podrobná charakteristika architektury a funkcionality frameworku Apache Hadoop a klíčových projektů ekosystému Hadoop. V poslední části dané kapitoly byly představeny příklady reálného využití frameworku Hadoop ze strany společností Twitter, Spotify a LinkedIn.

V rámci čtvrté kapitoly byl naplněn dílkový cíl spočívající v analýze dostupných distribucí frameworku Apache Hadoop. V tomto textu byla detailně vymezena množina řešení, která v minulosti formovala trh s distribucemi frameworku Hadoop. Bylo zjištěno, že trh distribucí Hadoop se v podstatě konsolidoval na tři hlavní poskytovatele, mezi které patřily společnosti Cloudera, MapR Technologies a Hortonworks. Řešení těchto producentů, tedy distribuce CDH, MPD a HDP, pak byly věnovány následující podkapitoly, které obsahovaly detailní popis vzniku a vývoje, podrobnou charakteristiku architektury a funkcionality a model poskytování každé jednotlivé distribuce. V rámci těchto podkapitol bylo zjištěno, že v průběhu let 2018 a 2019 došlo k drastickému propadu trhu s distribucemi Hadoop, především vlivem rostoucí adopce cloudových služeb právě na úkor distribucí Hadoop, v důsledku čehož došlo ke sloučení společností Cloudera a Hortonworks a akvizici společnosti MapR Technologies ze strany HPE. Současně bylo zjištěno, že repositáře nových verzí distribucí CDH a MDP již v budoucnu nebudou volně dostupné.

Dílčí cíle zahrnující zajištění zdroje produkujícího big data v reálném čase a prezentace použití vybraných distribucí frameworku Apache Hadoop pro příjem a zpracování big data z daného zdroje v reálném čase byly splněny v rámci páté kapitoly. V této části byla provedena typová úloha zpracování big data zahrnující příjem příspěvků ze sociální sítě Twitter v reálném čase a následné zpracování těchto dat v téměř reálném čase. Součástí praktické úlohy bylo také nasazení clusteru distribucí CDH, MDP a HDP. Tweety byly získány prostřednictvím Twitter Filtered Stream API skutečně v reálném čase. Příjem dat byl realizován v průběhu volebního superúterý a byl zaměřen na získání Tweetů obsahujících jména vybraných kandidátů v rámci primárních voleb ve Spojených státech amerických. Uvedeným způsobem bylo získáno během necelých osmi hodin 500 tisíc Tweetů. Současně byla s využitím knihovny *afinn* provedena analýza sentimentu. Získaná data byla uložena do nástroje Kafka. Následně došlo k v rámci jednotlivých komparovaných distribucí k transformaci a analýze dat prostřednictvím nástroje Spark, podporujícím zpracování v téměř reálném čase. Provedeny byly dvě Spark úlohy zahrnující výpočet sledovaných metrik sentimentu za celou množinu získaných dat a pro jednotlivá stanovená časová okna.

V rámci poslední kapitoly byl úspěšně naplněn dílčí cíl provedení komparace vybraných distribucí frameworku Apache Hadoop z hlediska možností zpracování big data v reálném čase. Pro tyto účely byly využity metody vícekriteriálního hodnocení variant. Aplikací metod stanovení vah kritérií metodou postupného rozvrhu vah a stanovení hodnoty (užitku) variant s využitím přímého stanovení dílčích hodnocení jednotlivých kritérií bylo zjištěno následující preferenční pořadí komparovaných distribucí frameworku Apache Hadoop: CDH, HDP, MDP. Výsledky u jednotlivých variant nicméně nevykazovaly výrazné odchylky a bylo tedy konstatováno, že pro danou typovou úlohu zpracování big data je možné využít prakticky kteroukoli ze zkoumaných distribucí. V případě reálné volby konkrétní distribuce pro specifický případ užití proto bylo doporučeno zaměření na kritéria nákladů na nasazení a provoz distribuce.

Všechny stanovené hlavní i dílčí cíle práce byly úspěšně splněny. Na tomto základě lze hlavní doporučení pro potenciální uživatele distribucí frameworku Hadoop zformulovat v tom smyslu, že je vždy skutečně relevantní zvážit, zda uživatel/společnost skutečně řeší big data, a zda potřebují plnohodnotnou distribuci frameworku Hadoop. Využití distribuce frameworku Hadoop determinuje signifikantní personální i finanční náklady, přičemž distribuce Hadoop, ale i dílčí nástroje pro zpracování big data jsou skutečně efektivní pouze v případě zpracování big data. V opačném případě může být jejich použití značně kontraproduktivní a nákladné. Trh s distribucemi Hadoop se v podstatě uzavírá, nové verze distribucí CDH a HDP již nebudou volně dostupné. Nesprávné rozhodnutí tak může potenciálně determinovat značné ztráty.

Mezi hlavní výstupy realizované práce spadá podrobná charakteristika vývoje trhu distribucí frameworku Apache Hadoop, detailní analýza klíčových distribucí Hadoop, tj. CDH, MPD a HDP, včetně podrobného popisu architektury a funkcionality. Další výstup představuje typová úloha přijetí dat ze sociální sítě Twitter v reálném čase s využitím nástroje Kafka a zpracování získaných big data pomocí projektu Spark, respektive API Spark Structured Streaming. Výsledky práce je možné použít pro získání rychlého vhledu do oblastí big data a distribucí frameworku Apache Hadoop, dále mohou sloužit jako

podklad pro výběr konkrétní distribuce frameworku Hadoop a v neposlední řadě jako koncepce řešení (proof of concept) pro příjem a zpracování big data ze sociální sítě Twitter.

Přestože práce detailně pojednala řadu aspektů distribucí frameworku Apache Hadoop, stále existuje prostor pro další rozvoj daného tématu. Práci by bylo možné dále rozvíjet například směrem k otázkám využití v prostředí cloudu, případně komparace nástrojů distribuce s konkurenčními řešeními, která nabízejí zejména poskytovatelé velkých cloudových platforem, např. AWS, GCP, Azure. Tato práce byla zaměřena na využití distribuce Hadoop pro zpracování big data ve stavu, v jakém jsou oficiálně k dispozici. Proto bylo zpracování big data provedeno pouze v téměř reálném čase. Navazující práce by se tudíž mohly zaměřit i na možnost kombinace distribuce s některým z projektů umožňujícím zpracování big data skutečně v reálném čase.

Použitá literatura

ACHARI, S. 2015. *Hadoop Essentials : Delve into the key concepts of Hadoop and get thorough Understanding of the Hadoop ecosystem.* 1. vyd. Birmingham: Packt Publishing. 172 s. ISBN: 978-1-78439-668-8.

ADRIAN, M. 2017. IBM Ends Hadoop Distribution, Hortonworks Expands Hybrid Open Source. *Gartner Blog* [online]. Gartner. [Cit. 18. 5. 2019]. Dostupné z: <https://blogs.gartner.com/merv-adrian/2017/06/21/ibm-ends-hadoop-distribution-hortonworks-expands-hybrid-open-source/>.

ALAPATI, S. R. 2017. *Expert Hadoop Administration : Managing, Tuning, and Securing Spark, YARN, and HDFS.* 1. vyd. Boston: Addison-Wesley. 848 s. ISBN: 978-0-13-456719-5.

ALAPATI, S. R. 2018. *Expert Apache Cassandra Administration : Install, configure, optimize, and secure Apache Cassandra databases.* 1. vyd. New York: Apress. 496 s. ISBN: 978-1-4842-3125-8.

ALIBABA. 2019. Versioning. *Alibaba Cloud E-MapReduce Documentation* [online]. Alibaba. [Cit. 21. 5. 2019]. Dostupné z: <https://www.alibabacloud.com/help/doc-detail/28073.html>.

AMAZON. 2019a. Amazon EMR 5.x Release Versions. *AWS EMR Documentation* [online]. Amazon. [Cit. 21. 5. 2019]. Dostupné z:

<https://docs.aws.amazon.com/emr/latest/ReleaseGuide/emr-release-5x.html>.

AMAZON. 2019b. Amazon EMR Management Guide. *AWS Documentation* [online]. Amazon. [Cit. 19. 5. 2019]. Dostupné z:

<https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-mgmt.pdf>.

AMAZON. 2019c. Amazon EMR with the MapR Distribution for Hadoop Pricing. *AWS EMR Documentation* [online]. Amazon. [Cit. 24. 11. 2019]. Dostupné z:

<https://aws.amazon.com/emr/mapr/pricing/>.

ANKAM, V. 2016. *Big Data Analytics.* 1 vyd. Birmingham: Packt Publishing. 326 s. ISBN: 978-1-78588-469-6.

ANTONY, B., BOUDNIK, K., ADAMS, CH., SHAO, B., LEE, C., SASAKI, K. 2016. *Professional Hadoop.* 1 vyd. Indianapois: John Wiley & Sons. 216 s. ISBN: 978-1-119-26717-1.

ARENADATA. 2019. Arenadata Hadoop. [online]. Arenadata. [Cit. 21. 5. 2019]. Dostupné z: <https://arenadata.tech/en/products/hadoop/>.

- ASF. 2012. What is Bigtop, and Why Should You Care? *ASF Blog* [online]. ASF. [Cit. 19. 5. 2019]. Dostupné z: https://blogs.apache.org/bigtop/entry/bigtop_and_why_should_you.
- ASF. 2014. Apache Hive. [online]. Apache Software Foundation. [Cit. 8. 5. 2019]. Dostupné z: <https://hive.apache.org/>.
- ASF. 2016a. Apache Cassandra. *Apache Cassandra* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://cassandra.apache.org/>.
- ASF. 2016b. Architecture. *Apache Chukwa 0.8.0 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://chukwa.apache.org/docs/r0.8.0/design.html>.
- ASF. 2016c. Overview. *Apache Chukwa 0.8.0 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <http://chukwa.apache.org/index.html>.
- ASF. 2016d. SerDe. *Hive wiki* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://cwiki.apache.org/confluence/display/Hive/SerDe>.
- ASF. 2017a. Apache Mahout. *Apache Mahout 0.13.0 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://mahout.apache.org/>.
- ASF. 2017b. Getting Started. *Apache Pig 0.17.0 Documentation* [online]. Apache Software Foundation. [Cit. 7. 5. 2019]. Dostupné z: <http://pig.apache.org/docs/r0.17.0/start.html>.
- ASF. 2017c. Mahout Overview. *Apache Mahout 0.13.0 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://mahout.apache.org/docs/latest/index.html>.
- ASF. 2018a. Apache Avro 1.8.2 Getting Started (Java). *Apache Avro 1.8.2 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://avro.apache.org/docs/current/gettingstartedjava.html>.
- ASF. 2018b. Apache Bigtop. *Apache Bigtop* [online]. Apache Software Foundation. [Cit. 19. 5. 2019]. Dostupné z: <https://bigtop.apache.org/>.
- ASF. 2018c. *Apache Pig* [online]. Apache Software Foundation. [Cit. 7. 5. 2019]. Dostupné z: <https://pig.apache.org/>.
- ASF. 2018d. Bigtop 1.3.0 Release. *Apache Bigtop Wiki* [online]. Apache Software Foundation. [Cit. 19. 5. 2019]. Dostupné z: <https://cwiki.apache.org/confluence/display/BIGTOP/Bigtop+1.3.0+Release>.
- ASF. 2018e. Overview. *Apache Avro 1.8.2 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z: <https://avro.apache.org/docs/current/>.
- ASF. 2019a. Ambari Design. *Apache Ambari Documentation* [online]. Apache Software Foundation. [Cit. 4. 5. 2019]. Dostupné z: https://issues.apache.org/jira/secure/attachment/12559939/Ambari_Architecture.pdf.

- ASF. 2019b. *Apache Hadoop* [online]. Apache Software Foundation. [Cit. 15. 4. 2019]. Dostupné z: <https://hadoop.apache.org/>.
- ASF. 2019c. Apache Hadoop YARN. *Apache Hadoop 3.1.2 Documentation* [online]. Apache Software Foundation. [Cit. 3. 5. 2019]. Dostupné z: <https://hadoop.apache.org/docs/r3.1.2/hadoop-yarn/hadoop-yarn-site/YARN.html>.
- ASF. 2019d. Apache HBase. *Apache HBase* [online]. Apache Software Foundation. [Cit. 7. 5. 2019]. Dostupné z: <https://hbase.apache.org/index.html>.
- ASF. 2019e. Apache HBase Reference Guide. *Apache HBase* [online]. Apache Software Foundation. [Cit. 7. 5. 2019]. Dostupné z: <http://hbase.apache.org/book.html>.
- ASF. 2019f. *Apache Spark* [online]. Apache Software Foundation. [Cit. 8. 5. 2019]. Dostupné z: <https://spark.apache.org/>.
- ASF. 2019g. Apache Tez. *Apache Tez Documentation* [online]. Apache Software Foundation. [Cit. 8. 5. 2019]. Dostupné z: <https://tez.apache.org/index.html>.
- ASF. 2019h. Architecture. *Ozone* [online]. Apache Software Foundation. [Cit. 28. 4. 2019]. Dostupné z: <https://hadoop.apache.org/ozone/docs/0.3.0-alpha/concepts.html>.
- ASF. 2019i. Cluster Mode Overview. *Apache Spark 2.3.4 Documentation* [online]. Apache Software Foundation. [Cit. 8. 5. 2019]. Dostupné z: <https://spark.apache.org/docs/latest/cluster-overview.html>.
- ASF. 2019j. Download. *Ozone* [online]. Apache Software Foundation. [Cit. 28. 4. 2019]. Dostupné z: <https://hadoop.apache.org/ozone/downloads/>.
- ASF. 2019k. Frequently Asked Questions. *Ozone* [online]. Apache Software Foundation. [Cit. 27. 4. 2019]. Dostupné z: <https://hadoop.apache.org/ozone/faq/>.
- ASF. 2019l. Hadoop Commands Guide. *Apache Hadoop 3.1.2 Documentation* [online]. Apache Software Foundation. [Cit. 28. 4. 2019]. Dostupné z: <https://hadoop.apache.org/docs/r3.1.2/hadoop-project-dist/hadoop-common/CommandsManual.html>.
- ASF. 2019m. *Hadoop Releases* [online]. Apache Software Foundation. [Cit. 26. 4. 2019]. Dostupné z: <https://archive.apache.org/dist/hadoop/common/>.
- ASF. 2019n. *Hadoop Roadmap* [online]. Apache Software Foundation. [Cit. 26. 4. 2019]. Dostupné z: <https://wiki.apache.org/hadoop/Roadmap>.
- ASF. 2019o. HDFS Architecture. *Apache Hadoop 3.1.2 Documentation* [online]. Apache Software Foundation. [Cit. 28. 4. 2019]. Dostupné z: <https://hadoop.apache.org/docs/r3.1.2/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>.
- ASF. 2019p. MapReduce Tutorial. *Apache Hadoop 3.1.2 Documentation* [online]. Apache Software Foundation. [Cit. 3. 5. 2019]. Dostupné z:

<https://hadoop.apache.org/docs/r3.1.2/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html>.

ASF. 2019q. Overview. *Apache Ambari Documentation* [online]. Apache Software Foundation. [Cit. 4. 5. 2019]. Dostupné z:
<https://ambari.apache.org/>.

ASF. 2019r. Overview. *Apache ZooKeeper 3.4 Documentation* [online]. Apache Software Foundation. [Cit. 5. 5. 2019]. Dostupné z:
<https://zookeeper.apache.org/doc/r3.4.14/zookeeperOver.html>.

ASF. 2019s. Ozone [online]. Apache Software Foundation. [Cit. 27. 4. 2019]. Dostupné z:
<https://hadoop.apache.org/ozone/>.

ASF. 2019t. Powered by Apache *Hadoop* [online]. Apache Software Foundation. [Cit. 26. 4. 2019]. Dostupné z: <https://wiki.apache.org/hadoop/PoweredBy>.

ASF. 2019u. Quick Start Guide. *Apache Hadoop 3.2.0 Documentation* [online]. Apache Software Foundation. [Cit. 3. 5. 2019]. Dostupné z:
<https://hadoop.apache.org/docs/r3.2.0/hadoop-yarn/hadoop-yarn-applications/hadoop-yarn-submarine/QuickStart.html>.

ASF. 2019v. Spark Overview. *Apache Spark 2.4.3 Documentation* [online]. Apache Software Foundation. [Cit. 8. 5. 2019]. Dostupné z:
<https://spark.apache.org/docs/latest/>.

ASF. 2019w. Spark Structured Streaming Programming Guide. *Apache Spark 2.3.4 Documentation* [online]. Apache Software Foundation. [Cit. 9. 5. 2019]. Dostupné z:
<http://spark.apache.org/docs/latest/structured-streaming-programming-guide.html>.

ASF. 2019x. Submarine [online]. Apache Software Foundation. [Cit. 3. 5. 2019]. Dostupné z: <https://hadoop.apache.org/submarine/>.

ASF. 2020a. *Apache Kafka* [online]. Apache Software Foundation. [Cit. 4. 1. 2020]. Dostupné z: <https://kafka.apache.org/>.

ASF. 2020b. Download. *Ozone* [online]. Apache Software Foundation. [Cit. 11. 4. 2020]. Dostupné z: <https://hadoop.apache.org/ozone/downloads/>.

ASF. 2020c. Kafka 2.1 Documentation. *Apache Kafka Documentation* [online]. Apache Software Foundation. [Cit. 4. 1. 2020]. Dostupné z:
<https://kafka.apache.org/21/documentation.html>.

ASF. 2020d. Submarine [online]. Apache Software Foundation. [Cit. 11. 4. 2020]. Dostupné z: <https://submarine.apache.org/>.

AURSWAMY, V. 2015. *How LinkedIn uses Automic for Big Data Processes* [online]. LinkedIn. [Cit. 10. 5. 2019]. Dostupné z:
<https://www.slideshare.net/automic/how-linkedin-uses-automic-for-big-data-processes>.

- BEKKER, A. 2018. *Cassandra vs. HBase : twins or just strangers with simillar looks?* [online]. ScienceSoft. [Cit. 7. 5. 2019]. Dostupné z: <https://www.scnsoft.com/blog/cassandra-vs-hbase>.
- BELL, L. 2014. *Intel dropping its own distribution of Hadoop big data analytics in favour of Cloudera* [online]. The Inquirer. [Cit. 18. 5. 2019]. Dostupné z: <https://www.theinquirer.net/inquirer/news/2336750/intel-dropping-its-own-distribution-of-hadoop-big-data-analytics-in-favour-of-cloudera>.
- BELL, J. 2015. *Machine Learning : Hands-On for Developers and Technical Professionals*. 1. vyd. Indianapolis: John Wiley & Sons. 408 s. ISBN: 978-1-118-88906-0.
- BERKA, P. 2003. *Dobývání znalostí z databází*. 1. vyd. Praha: Academia. 366 s. ISBN 80-200-1062-9.
- BHATIA, R. 2018. The Hortonworks-Cloudera Merger Is Actually Hadoop's Obituary. *Analytics India Magazine* [online]. Analytics India Magazine. [Cit. 7. 12. 2019]. Dostupné z: <https://analyticsindiamag.com/the-hortonworks-cloudera-merger-is-actually-hadoops-obituary/>.
- BHUSHAN, M. 2018. *Big Data & Hadoop : Learn by Example*. 1. vyd. New Delhi: BPB Publications. 333 s. ISBN: 978-93-8655-199-3.
- BINANI, S., GUTTI, A., UPADHYAY, S. SQL vs. NoSQL vs. NewSQL – A Comparative Study. *Communications on Applied Electronics* [online]. 2016, 6(1), 43-46. ISSN: 2394-4714. Dostupné z: <https://pdfs.semanticscholar.org/9d92/d600ff1a4bec346b6ca3fe6d8bf9677294b2.pdf>.
- BONACI, M. 2015. *The History of Hadoop* [online]. A Medium Corporation. [Cit. 27. 2. 2019]. Dostupné z: <https://medium.com/@markobonaci/the-history-of-hadoop-68984a11704>.
- BROTÁNEK, J. *Apache Hadoop jako analytická platforma*. Praha, 2017. Diplomová práce. Vysoká škola ekonomická v Praze. Fakulta informatiky a statistiky. Vedoucí práce doc. Ing. Ota Novotný, Ph.D.
- BUYYA, R., CALHERIOS, R., DASTJERDI, A. V. 2016. *Big Data : Principles and Paradigms*. 1. vyd. Cambridge: Elsevier. 468 s. ISBN: 978-0-12-805394-2.
- CISION PR NEWSWIRE. 2015. Hortonworks SmartSense Brings Preventative Maintenance for Big Data to Customers [online]. Cision PR Newswire. [Cit. 7. 12. 2019]. Dostupné z: <https://www.prnewswire.com/news-releases/hortonworks-smartsense-brings-preventative-maintenance-for-big-data-to-customers-300164192.html>.
- CLOUDERA. 2009. Cloudera Announces New Distribution for Hadoop to Bring Data Processing Power to Enterprises. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2009-05-16-cloudera-announces-new-distribution-for-hadoop-to-bring-data-processing-power-to-enterprises.html>.

CLOUDERA. 2010. Cloudera Unveils Cloudera Enterprise. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z:
<https://www.cloudera.com/about/news-and-blogs/press-releases/2010-06-29-cloudera-unveils-cloudera-enterprise.html>.

CLOUDERA. 2011a. Cloudera Expands Enterprise Management Software For Apache Hadoop With End-to-End Visibility For Big Data Analytics. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z:
<https://www.cloudera.com/about/news-and-blogs/press-releases/2011-12-08-cloudera-expands-enterprise-management-software-for-apache-hadoop-with-end-to-end-visibility-for-big-data-analytics.html>.

CLOUDERA. 2011b. Cloudera Unveils Industry's First Full Lifecycle Management and Automated Security Solution for Apache Hadoop Deployments. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z:
<https://www.cloudera.com/about/news-and-blogs/press-releases/2011-06-29-cloudera-unveils-industry-s-first-full-lifecycle-management-and-automated-security-solution-for-apache-hadoop-deployments.html>.

CLOUDERA. 2011c. Cloudera Unveils the First Tested, Integrated, 100% Apache-Licensed Hadoop Distribution For On-Premise and Cloud Computing. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z:
<https://www.cloudera.com/about/news-and-blogs/press-releases/2011-04-12-cloudera-unveils-the-first-tested-integrated-100-apache-licensed-hadoop-distribution-for-on-premise-and-cloud-computing.html>.

CLOUDERA. 2012a. Cloudera Announces Game-Changing, Real-Time Query on Hadoop and Leads a New Era of Data Management. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2012-10-24-cloudera-announces-game-changing-real-time-query-on-hadoop-and-leads-a-new-era-of-data-management.html>.

CLOUDERA. 2012b. Cloudera Delivers CDH4 Beta. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2012-04-24-cloudera-delivers-cdh4-beta.html>.

CLOUDERA. 2012c. Cloudera Introduces Fourth Generation of Its Big Data Platform to Drive Ease of Use, Integration and Adoption of Apache Hadoop for the Enterprise. *Cloudera Press Release* [online]. Cloudera. [Cit. 27. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2012-06-05-cloudera-introduces-fourth-generation-of-its-big-data-platform-to-drive-ease-of-use-integration-and-adoption-of-apache-hadoop-for-the-enterprise.html>.

CLOUDERA. 2013a. Cloudera Accelerates Platform for Big Data With New Enterprise-Required Advancements. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-02-26-cloudera-accelerates-platform-for-big-data-with-new-enterprise-required-advancements.html>.

CLOUDERA. 2013b. Cloudera Announces Cloudera Developer Kit, Enabling Developers to Build Hadoop Apps Faster. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-05-08-cloudera-announces-cloudera-developer-kit-enabling-developers-to-build-hadoop-apps-faster.html>.

CLOUDERA. 2013c. Cloudera Democratizes Apache Hadoop for Enterprise End Users with Open Source, Interactive Search. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-06-04-cloudera-democratizes-apache-hadoop-for-enterprise-end-users-with-open-source-interactive-search.html>.

CLOUDERA. 2013d. Cloudera Enterprise 5 Sets New Standard for Data Management : Lays Foundation for The Enterprise Data Hub. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-10-29-cloudera-enterprise-5-sets-new-standard-for-data-management-lays-foundation-for-the-enterprise-data-hub.html>.

CLOUDERA. 2013e. Cloudera Search Now Generally Available for Open Source Users and Enterprise Subscribers. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-09-05-cloudera-search-now-generally-available-for-open-source-users-and-enterprise-subscribers.html>.

CLOUDERA. 2013f. Cloudera Ships Impala 1.0 : Industry's First Production-Ready SQL-on-Hadoop Solution. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-04-30-cloudera-ships-impala-1-0-industry-s-first-production-ready-sql-on-hadoop-solution.html>.

CLOUDERA. 2013g. Cloudera Takes Hadoop Security to the Next Level With Sentry : Fine-Grained Authorization for Impala and Apache Hive. *Cloudera Press Release* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2013-07-24-cloudera-takes-hadoop-security-to-the-next-level-with-sentry-fine-grained-authorization-for-impala-and-apache-hive.html>.

CLOUDERA. 2014a. Cloudera Delivers Industry's First Complete Enterprise Data Hub. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2014-02-03-cloudera-delivers-industry-s-first-complete-enterprise-data-hub.html>.

CLOUDERA. 2014b. Cloudera Enterprise 5 Now Generally Available : Delivers Next Generation Platform to Enable the Enterprise Data Hub. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2014-04-02-cloudera-enterprise-5-now-generally-available-delivers-next-generation-platform-to-enable-the-enterprise-data-hub.html>.

CLOUDERA. 2014b. Cloudera Enterprise 5 Now Generally Available : Delivers Next Generation Platform to Enable the Enterprise Data Hub. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2014-04-02-cloudera-enterprise-5-now-generally-available-delivers-next-generation-platform-to-enable-the-enterprise-data-hub.html>.

CLOUDERA. 2014c. Cloudera Unveils Cloudera Enterprise 5.2. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2014-10-14-cloudera-unveils-cloudera-enterprise-5-2.html>.

CLOUDERA. 2015a. Cloudera Adds Google Cloud Platform Support with Release of Cloudera Director 1.5. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2015-08-12-cloudera-adds-google-cloud-support-with-cloudera-director.html>.

CLOUDERA. 2015b. Cloudera Enterprise Data Hub Edition Provides Enterprise-Ready Hadoop for the Microsoft Azure Marketplace. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2015-09-24-cloudera-enterprise-data-hub-edition-provides-enterprise-ready-hadoop-for-microsoft-azure.html>.

CLOUDERA. 2015c. Cloudera Launches Kudu, New Hadoop Storage for Fast Analytics on Fast Data. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2015-09-28-cloudera-launches-kudu-new-hadoop-storage-for-fast-analytics-on-fast-data.html>.

CLOUDERA. 2015d. Cloudera Navigator Optimizer Provides Active Data Optimization for Hadoop Workloads. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2015-11-19-cloudera-navigator-optimizer-provides-active-data-optimization.html>.

CLOUDERA. 2015e. Real-Time Messaging System Apache Kafka Now Fully Integrated with Cloudera. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2015-02-18-real-time-messaging-system-apache-kafka.html>.

CLOUDERA. 2016a. Cloudera Announces Beta Release of Cloudera Desktop. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2016-01-10-cloudera-announces-beta-release-cloudera-desktop15.html>.

CLOUDERA. 2016b. Cloudera Announces Cloudera Director 2.0 for the Easiest Way to Manage Enterprise Hadoop Workloads Across Cloud Environments. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2016-01-21-cloudera-director-2-0-easiest-way-to-manage-enterprise-hadoop-workloads-across-cloud-environments.html>.

CLOUDERA. 2016c. Cloudera Delivers Enterprise-Grade Real-Time Streaming and Machine Learning with Apache Spark 2.0 and Drives Community Innovation with Apache Kudu 1.0. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z:

<https://www.cloudera.com/about/news-and-blogs/press-releases/2016-09-29-cloudera-delivers-enterprise-grade-real-time-streaming-and-machine-learning-apache-spark-2-0-apache-kudu-1-0.html>.

CLOUDERA. 2016d. Cloudera Eases the Path for Analytics on Apache Hadoop with the General Availability of Cloudera Navigator Optimizer and Cloudera Enterprise 5.8. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z:

<https://www.cloudera.com/about/news-and-blogs/press-releases/2016-07-21-cloudera-eases-the-path-for-analytics-on-apache-hadoop-with-cloudera-enterprise-5-8.html>.

CLOUDERA. 2016e. Latest Release of Cloudera Enterprise Brings Better Performance and Operations Efficiency Across Workloads and Users. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2016-04-07-Latest-Release-Cloudera-Enterprise-Brings-Better-Performance-Operations-Efficiency-Across-Workloads-Users.html>.

CLOUDERA. 2017a. Cloudera Announces General Availability of Apache Kudu with Release of Cloudera Enterprise 5.10. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z:

<https://www.cloudera.com/about/news-and-blogs/press-releases/2017-01-31-cloudera-announces-general-availability-of-apache-kudu-with-release-of-cloudera-enterprise-5-10.html>.

CLOUDERA. 2017b. Cloudera Announces General Availability of Data Science Workbench to Accelerate Data Science and Machine Learning in the Enterprise. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z:

<https://www.cloudera.com/about/news-and-blogs/press-releases/2017-05-01-cloudera-announces-general-availability-of-data-science-workbench-to-accelerate-data-science-and-machine-learning-in-the-enterprise.html>.

CLOUDERA. 2017c. Cloudera changes the game in cloud-based data warehouses with Cloudera Altus Analytic DB (beta). *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2017-11-28-cloudera-changes-the-game-in-cloud-based-data-warehouses-with-altus.html>.

CLOUDERA. 2017d. Cloudera Introduces Altus Data Engineering for Microsoft Azure Beta Enabling New Hybrid and Multi-cloud Data Analytic Workflows. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2017-09-27-cloudera-introduces-altus-data-engineering-microsoft-azure-hybrid-multi-cloud-data-analytic-workflows.html>.

CLOUDERA. 2017e. Cloudera Launches Altus to Simplify Big Data Workloads in the Cloud. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z:

<https://www.cloudera.com/about/news-and-blogs/press-releases/2017-05-24-cloudera-launches-altus-to-simplify-big-data-workloads-in-the-cloud.html>.

CLOUDERA. 2017f. Cloudera Launches SDX, a Shared Data Experience for the Enterprise. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2017-09-26-cloudera-simplifies-big-data-initiatives-with-shared-data-experience-sdx.html>.

CLOUDERA. 2017g. Cloudera to Accelerate Data Science and Machine Learning for the Enterprise with New Data Science Workbench. *Cloudera Press Release* [online]. Cloudera. [Cit. 3. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2017-03-14-cloudera-to-accelerate-data-science-and-machine-learning-for-the-enterprise-with-new-data-science-workbench.html>.

CLOUDERA. 2018a. Cloudera Accelerates Enterprise Machine Learning from Research to Production. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2018-05-21-cloudera-accelerates-enterprise-machine-learning-from-research-to-production.html>.

CLOUDERA. 2018b. Cloudera and Hortonworks Announce Merger to Create World's Leading Next Generation Data Platform and Deliver Industry's First Enterprise Data Cloud. *Cloudera Press Release* [online]. Cloudera. [Cit. 19. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2018-10-03-cloudera-and-hortonworks-announce-merger-to-create-worlds-leading-next-generation-data-platform-and-deliver-industrys-first-enterprise-data-cloud.html>.

CLOUDERA. 2018c. Cloudera Announces Preview of Cloud-Native Machine Learning Platform to Accelerate the Industrialization of AI. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2018-12-05-cloudera-announces-preview-of-cloud-native-machine-learning-platform-to-accelerate-the-industrialization-of-ai.html>.

CLOUDERA. 2018d. Cloudera Delivers Its Most Powerful Machine Learning and Analytics Platform - Cloudera Enterprise 6. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2018-09-10-cloudera-delivers-its-most-powerful-machine-learning-and-analytics-platform-cloudera-enterprise-6.html>.

CLOUDERA. 2018e. Cloudera Enterprise : The Modern Platform for Machine Learning and Analytics Optimized for the Cloud. *Cloudera Resources Library* [online]. Cloudera. [Cit. 23. 5. 2019]. Dostupné z: <https://www.cloudera.com/content/dam/www/marketing/resources/datasheets/cloudera-enterprise-datasheet.pdf.landing.html>.

CLOUDERA. 2018f. Cloudera Introduces the Industry's First Machine Learning and Analytics Platform-as-a-Service Built with a Shared Data Experience (SDX). *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z:

<https://www.cloudera.com/about/news-and-blogs/press-releases/2018-03-06-cloudera-introduces-the-industrys-first-machine-learning-and-analytics-platform-as-a-service-built-with-a-shared-data-experience-sdx.html>.

CLOUDERA. 2018g. Cloudera Leads the Way in Data Warehousing for Hybrid Cloud. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2018-08-28-cloudera-leads-the-way-in-data-warehousing-for-hybrid-cloud.html>.

CLOUDERA. 2019a. CDH 6.2.x Packaging. *Cloudera Enterprise 6.2.x Documentation* [online]. Cloudera. [Cit. 12. 6. 2019]. Dostupné z: https://www.cloudera.com/documentation/enterprise/6/release-notes/topics/rg_cdh_62_packaging.html.

CLOUDERA. 2019b. CDH Overview. *Cloudera Enterprise 6.2.x Documentation* [online]. Cloudera. [Cit. 19. 6. 2019]. Dostupné z: https://www.cloudera.com/documentation/enterprise/6/6.2/topics/cdh_intro.html.

CLOUDERA. 2019c. Cloudera and Hortonworks Complete Planned Merger. *Cloudera Press Release* [online]. Cloudera. [Cit. 19. 5. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2019-01-03-cloudera-and-hortonworks-complete-planned-merger.html>.

CLOUDERA. 2019d. Cloudera Downloads. *Cloudera* [online]. Cloudera. [Cit. 24. 5. 2019]. Dostupné z: <https://www.cloudera.com/downloads.html>.

CLOUDERA. 2019e. Cloudera Enterprise 6.2.0 Released. *Cloudera Community News & Announcements* [online]. Cloudera. [Cit. 12. 6. 2019]. Dostupné z: <https://community.cloudera.com/t5/Community-News-Release/ANNOUNCE-Cloudera-Enterprise-6-2-0-Released/td-p/88501>.

CLOUDERA. 2019f. Cloudera Installation Guide. *Cloudera Enterprise 6.2.x Documentation* [online]. Cloudera. [Cit. 7. 12. 2019]. Dostupné z: <https://www.cloudera.com/documentation/enterprise/6/6.2/topics/installation.html>.

CLOUDERA. 2019g. Cloudera Manager. *Cloudera Enterprise 6.2.x Documentation* [online]. Cloudera. [Cit. 19. 6. 2019]. Dostupné z: https://www.cloudera.com/documentation/enterprise/latest/topics/cloudera_manager.html.

CLOUDERA. 2019h. Cloudera Manager Frequently Asked Questions. *Cloudera Enterprise 6.0.x Documentation* [online]. Cloudera. [Cit. 23. 5. 2019]. Dostupné z: https://www.cloudera.com/documentation/enterprise/6/6.0/topics/cm_faqs.html.

CLOUDERA. 2019i. Cloudera Unlocks Opportunity at the Edge Accelerating Enterprise Data Cloud Vision. *Cloudera Press Release* [online]. Cloudera. [Cit. 10. 6. 2019]. Dostupné z: <https://www.cloudera.com/about/news-and-blogs/press-releases/2019-03-27-cloudera-unlocks-opportunity-at-the-edge-accelerating-enterprise-data-cloud-vision.html>.

CLOUDERA. 2019j. Find a Cloudera Partner. *Cloudera* [online]. Cloudera. [Cit. 19. 6. 2019]. Dostupné z: <https://www.cloudera.com/partners/partners-listing.html>.

CLOUDERA. 2019k. Hortonworks Support Services Policy. *Cloudera* [online]. Cloudera. [Cit. 8. 12. 2019]. Dostupné z: <https://www.cloudera.com/legal/hwx/support-services-policy.html>.

CLOUDERA. 2019l. Index of cdh-releases-rcs/org/apache/spark/spark-sql-kafka-0-10_2.11/2.3.0.cloudera2. *Cloudera* [online]. Cloudera. [Cit. 14. 3. 2020]. Dostupné z: https://repository.cloudera.com/cloudera/cdh-releases-rcs/org/apache/spark/spark-sql-kafka-0-10_2.11/2.3.0.cloudera2/.

CLOUDERA. 2020a. [ANNOUNCE] Cloudera Enterprise 6.3.3 Released. *Cloudera Community* [online]. Cloudera. [Cit. 14. 3. 2020]. Dostupné z: <https://community.cloudera.com/t5/Product-Announcements/ANNOUNCE-Cloudera-Enterprise-6-3-3-Released/td-p/289016>.

CLOUDERA. 2020b. Cloudera Pricing. [online]. Cloudera. [Cit. 5. 4. 2020]. Dostupné z: <https://www.cloudera.com/products/pricing.html>.

CORONEL, C., MORRIS, S. 2019. *Database Systems : Design, Implementation & Management*. 13. vyd. Boston: Cengage. 805 s. ISBN: 978-1-337-62790-0.

DATAFLAIR, 2018. *Apache Hive Architecture & Components* [online]. DataFlair. [Cit. 8. 5. 2019]. Dostupné z: <https://data-flair.training/blogs/apache-hive-architecture/>.

DATAFLAIR, 2019. *Hadoop 2 vs Hadoop 3 – Why You Should Work on Hadoop Latest Version* [online]. DataFlair. [Cit. 27. 4. 2019]. Dostupné z: <https://data-flair.training/blogs/hadoop-2-vs-hadoop-3/>.

DATASTAX. 2013. DataStax Brisk Distribution. [online]. DataStax. [Cit. 21. 5. 2019]. Dostupné z: <https://github.com/riptano/brisk>.

DATASTAX. 2019. About DSE Hadoop. *DataStax Enterprise 5.0 Documentation* [online]. DataStax. [Cit. 21. 5. 2019]. Dostupné z: https://docs.datastax.com/en/datastax_enterprise/5.0/datastax_enterprise/ana/dsehadoop/dseHadoop.html.

DAVENPORT, T. H., HARRIS, J. G., 2017. *Competing on Analytics : The New Science of Winning*. 2. vyd. Boston: Harvard Business Review Press. eISBN: 978-1-63369-373-9.

DEAN, J., GHEMAWAT, S. MapReduce : Simplified Data Processing on Large Clusters. In: *OSDI'04: Sixth Symposium on Operating System Design and Implementation* [online]. San Francisco: CA, 2004, s. 137-150. Dostupné z: <https://static.googleusercontent.com/media/research.google.com/cs//archive/mapreduce-osdi04.pdf>.

DEMCHENKO, Y., DE LAAT, C., MEMBREY, P. *Defining Architecture Components of the Big Data Ecosystem*. In: *International Conference on Collaboration Technologies and Systems* [online]. Minneapolis: CTS, 2014. s. 104-112. Dostupné z:

https://www.researchgate.net/publication/269272409_Defining_architecture_components_of_the_Big_Data_Ecosystem.

DEUTSCHER, M. 2013. WANdisco Releases Free Hadoop Distro. *SiliconANGLE* [online]. *SiliconANGLE*. [Cit. 19. 5. 2019]. Dostupné z: <https://siliconangle.com/2013/02/11/wandisco-releases-free-hadoop-distro/>.

DEZYRE. 2016. How LinkedIn uses Hadoop to leverage Big Data Analytics? [online]. DeZyre. [Cit. 9. 5. 2019]. Dostupné z: <https://www.dezyre.com/article/how-linkedin-uses-hadoop-to-leverage-big-data-analytics/229>.

DOWNARD, I. 2017. Kafka vs. MapR Event Store: Why MapR?. The MapR Blog [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/blog/kafka-vs-mapr-streams-why-mapr/>.

EBBERS, M., DE SOUZA, R. G., LIMA, M. C., MCCULLAGH, P., NOBLES, M., VANSTEE, D., WATERS, B. 2013. *Implementing IBM InfoSphere BigInsightson IBM System x* [online]. 2. vyd. New York: IBM. 224 s., ISBN: 0738438286. Dostupné z: <http://www.redbooks.ibm.com/redbooks/pdfs/sg248077.pdf>.

EMC. 2013. EMC Introduces World's Most Powerful Hadoop Distribution : Pivotal HD. *EMC Press Release* [online]. EMC. [Cit. 18. 5. 2019]. Dostupné z: <https://www.emc.com/about/news/press/2013/20130225-04.htm>.

EMC. 2015. *Data Science & Big Data Analytics : Discovering, Analyzing, Visualizing and Presenting Data*. 1. vyd. Indianapolis: John Wiley & Sons. 410 s. ISBN: 978-1-118-87613-8.

ERRAISI, A., BELANGOUR, A., TRAGHA, A. Digging into Hadoop-based Big Data Architectures. *International Journal of Computer Science Issues* [online]. 2017, **14**(6), 52-59. ISSN: 1694-0784. Dostupné z: <https://ijcsi.org/papers/IJCSI-14-6-52-59.pdf>.

FASALE, A., KUMAR, N. 2015. *YARN Essentials : A comprehensive, hands-on guide to install, administer, and configure Settings in YARN*. 1. vyd. Birmingham: Packt Publishing. 174 s. ISBN: 978-1-78439-173-7.

FAYYAD, U., PIATESKY-SHAPIRO, G., SMYTH, P. From Data Mining to Knowledge Discovery in Databases. *AI Magazine* [online]. 1996, **17**(3), 37-54. ISSN: 0738-4602. Dostupné z: <https://www.aaai.org/ojs/index.php/aimagazine/article/viewFile/1230/1131>.

FOLEY, M. J. 2013. Microsoft makes available its Azure-based Hadoop service. *ZDNet* [online]. ZDNet. [Cit. 19. 5. 2019]. Dostupné z: <https://www.zdnet.com/article/microsoft-makes-available-its-azure-based-hadoop-service/>.

FONTAINE, R. 2018. *How Apache Hadoop 3 Adds Value Over Apache Hadoop 2* [online]. Hortonworks. [Cit. 27. 4. 2019] Dostupné z: <https://hortonworks.com/blog/hadoop-3-adds-value-hadoop-2/>.

FRAMPTON, M. 2015. *Mastering Apache Spark : Gain expertise in processing and storing data by using advanced techniques with Apache Spark*. 1. vyd. Birmingham: Packt Publishing. 318 s. ISBN: 978-1-78398-714-6.

FU, M. 2018. Leaving the Nest : Heron donated to Apache Software Foundation. *Twitter Blog* [online]. Twitter. [Cit. 9. 5. 2019]. Dostupné z: https://blog.twitter.com/engineering/en_us/topics/open-source/2018/heron-donated-to-apache-software-foundation.html.

GARTNER. *Advanced Analytics*. In: GARTNER. 2019. *Gartner IT Glossary* [online]. Stamford: Gartner. [Cit. 15. 4. 2019]. Dostupné z: <https://www.gartner.com/it-glossary/advanced-analytics/>.

GARTNER. *Big Data*. In: GARTNER. 2019. *Gartner IT Glossary* [online]. Stamford: Gartner. [Cit. 15. 4. 2019]. Dostupné z: <https://www.gartner.com/it-glossary/big-data/>.

GÁLA, L., POUR, J., ŠEDIVÁ, Z. 2015. *Podniková informatika : Počítačové aplikace v podnikové a mezipodnikové praxi*. 3., aktualizované vydání. Praha: Grada Publishing. 240 s. ISBN: 978-80-247-5457-4.

GHEMAWAT, S., GOBIOFF, H., LEUNG, S. The Google File System. In: *Proceedings of the 19th ACM Symposium on Operating Systems Principles* [online]. Bolton Landing: ACM, 2003, s. 20-43. ISBN: 1-58113-757-5. Dostupné z: <http://static.googleusercontent.com/media/research.google.com/en//archive/gfs-sosp2003.pdf>.

GILLIN, P. 2019. Big-data bombshell: MapR may shut down as investor pulls out after ‘extremely poor results’. SiliconANGLE [online]. SiliconANGLE. [Cit. 24. 11. 2019]. Dostupné z:

<https://siliconangle.com/2019/05/30/mapr-may-shut-investor-pulls-following-extremely-poor-results/>.

GOOGLE. 2019a. 1.4.x release versions. *Google Cloud Dataproc documentation* [online]. Google. [Cit. 21. 5. 2019]. Dostupné z:

<https://cloud.google.com/dataproc/docs/concepts/versioning/dataproc-release-1.4>.

GOOGLE. 2019b. Cloud Dataproc Image version list. *Google Cloud Dataproc documentation* [online]. Google. [Cit. 21. 5. 2019]. Dostupné z: <https://cloud.google.com/dataproc/docs/concepts/versioning/dataproc-versions>.

GOYAL, S., PARASHAR, A., SHROTRIYA, A. *Application of Big Data Analytics in Cloud Computing via Machine Learning*. In: MITTAL, M., BALAS, V. E., HEMANTH, D. J., KUMAR, R. 2018. *Data Intensive Computing Applications for Big Data*. 1. vyd. Amsterdam: IOS Press. s. 236-266. ISBN: 978-1-61499-813-6.

GRADY, N., CHANG, W. 2015. *NIST Big Data Interoperability Framework : Volume 1, Definitions* [online]. 1. finální verze. Gaithersburg: National Institute of Standards and Technology. 19 s. NIST Special Publication 1500-1. Dostupné z: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-1.pdf>.

GUALTIERI, M., YUHANNA, N. 2014. *The Forrester Wave : Big Data Hadoop Solutions, Q1 2014* [online]. Forrester Research. [Cit. 11. 5. 2019]. Dostupné z: http://hortonworks.com/wp-content/uploads/2014/02/The_Forrester_Wave_Big_Data_Solutions_Q1_2014.pdf.

GUALTIERI, M., YUHANNA, N. 2016a. *The Forrester Wave : Big Data Hadoop Cloud Solutions, Q2 2016* [online]. Forrester Research. [Cit. 12. 5. 2019]. Dostupné z: <https://www.forrester.com/report/The+Forrester+Wave+Big+Data+Hadoop+Cloud+Solutions+Q2+2016/-/E-RES126541>.

GUALTIERI, M., YUHANNA, N. 2016b. *The Forrester Wave : Big Data Hadoop Distributions, Q1 2016* [online]. Forrester Research. [Cit. 12. 5. 2019]. Dostupné z: <http://www.datascienceassn.org/sites/default/files/Hadoop%20Vendor%20Evaluations%202016.pdf>.

HAMMOUD, M., SAKR, M. F. *Distributed Programming for the Cloud : Models, Challenges and Analytical Engines*. In: SAKR, S., GABER, M. M. 2014. *Large Scale and Big Data : Processing and Management*. 1. vyd. Boca Raton: CRC Press. s. 1-38. ISBN: 978-1-4665-8151-7.

HAN, Z., HONG, M., WANG, D. 2017. *Signal Processing and Networking for Big Data Applications*. 1. vyd. Cambridge: Cambridge University Press. 474 s. ISBN: 978-1-107-12438-7.

HASHEMI, M. 2017. The Infrastructure Behind Twitter : Scale. *Twitter Blog* [online]. Twitter. [Cit. 9. 5. 2019]. Dostupné z: https://blog.twitter.com/engineering/en_us/topics/infrastructure/2017/the-infrastructure-behind-twitter-scale.html.

HE, J., RINALDI, B., SAHA, G. 2018. *First-Class Support for Long Running Services on Apache Hadoop YARN* [online]. Hortonworks. [Cit. 27. 4. 2019]. Dostupné z: <https://hortonworks.com/blog/first-class-support-long-running-services-apache-hadoop-yarn/>.

HOLUBOVÁ, I., KOSEK, J., MINAŘÍK, K., NOVÁK, D. 2015. *Big Data a NoSQL databáze*. 1. vyd. Praha: Grada Publishing. 288 s. ISBN: 978-80-247-5466-6.

HORTONWORKS. 2011. Support [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z: <http://web.archive.org/web/2011105202011/http://hortonworks.com/support/>.

HORTONWORKS. 2012a. Hortonworks Data Platform v1.0 Powered by Apache Hadoop : Installing and Configuring HDP using Hortonworks Management Center. *Hortonworks* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: http://hortonworks.com/wp-content/uploads/2012/06/HMC_User_Guide.pdf.

HORTONWORKS. 2012b. Support [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z: <http://web.archive.org/web/20120307021947/http://hortonworks.com/support/>.

HORTONWORKS. 2013a. Hortonworks Data Platform : Installing HDP Using Apache Ambari. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: https://docs.cloudera.com/HDPDocuments/HDP1/HDP-1.2.0/bk_using_Ambari_book/bk_using_Ambari_book-20130114.pdf.

HORTONWORKS. 2013b. Hortonworks Data Platform 2.0 Now Generally Available. *Hortonworks* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: <http://investors.hortonworks.com/static-files/d212769b-5f64-4b72-877e-4ad90dc98f77>.

HORTONWORKS. 2014a. Hadoop Support Subscription [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z: <http://web.archive.org/web/20141117230143/http://hortonworks.com/support/>.

HORTONWORKS. 2014b. Hortonworks Data Platform : HDP-2.2.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.2.0/bk_HDP_RelNotes/bk_HDP_RelNotes-20141202.pdf.

HORTONWORKS. 2014c. Hortonworks Data Platform : Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.1.1/bk_releasenotes_hdp_2.1/bk_releasenotes_hdp_2.1-20140422.pdf.

HORTONWORKS. 2015a. HDF Support Subscription [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z: <http://web.archive.org/web/20151006065058/http://hortonworks.com/hdf-support/>.

HORTONWORKS. 2015b. HDP Support Subscription [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z: <http://web.archive.org/web/20151128135932/http://hortonworks.com/support/>.

HORTONWORKS. 2015c. Hortonworks Data Platform : HDP-2.3.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.3.0/bk_HDP_RelNotes/bk_HDP_RelNotes.pdf.

HORTONWORKS. 2015d. Hortonworks DataFlow : Release Notes. *HDF Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: https://docs.cloudera.com/HDPDocuments/HDF1/HDF-1.1.0/bk_HDF_RelNotes/bk_HDF_RelNotes.pdf.

HORTONWORKS. 2016a. HDP Support Subscription [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z: <http://web.archive.org/web/20160322020522/http://hortonworks.com/support/>.

HORTONWORKS. 2016b. Hortonworks Data Platform : HDP-2.4.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.4.0/bk_HDP_RelNotes/bk_HDP_RelNotes.pdf.

HORTONWORKS. 2016c. Hortonworks Data Platform : HDP-2.5.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.5.0/bk_release-notes/bk_release-notes.pdf.

HORTONWORKS. 2016d. Hortonworks DataFlow : Release Notes. *HDF Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/HDF2/HDF-2.0.0/bk_release-notes/bk_release-notes.pdf.

HORTONWORKS. 2016d. *How to install Tez UI Standalone and use it to debug Hive Queries* [online]. Hortonworks. [Cit. 8. 5. 2019]. Dostupné z:
<https://community.hortonworks.com/articles/56145/how-to-install-tez-ui-standalone-and-use-it-to-deb-1.html>.

HORTONWORKS. 2017a. Hortonworks Data Platform : HDP-2.6.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.6.0/bk_release-notes/bk_release-notes.pdf.

HORTONWORKS. 2017b. Hortonworks DataFlow : Release Notes. *HDF Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/HDF3/HDF-3.0.0/bk_release-notes/bk_release-notes.pdf.

HORTONWORKS. 2017c. Hortonworks DataPlane Service : Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/DP/DP-1.0.0/bk_dps-release-notes/bk_dps-release-notes.pdf.

HORTONWORKS. 2018a. DAS 1.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/DAS/DAS-1.0.0/release-notes/das-release-notes.pdf>.

HORTONWORKS. 2018b. Hortonworks Data Platform : HDP 3.0.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.0.0/release-notes/hdp-release-notes.pdf>.

HORTONWORKS. 2018c. Hortonworks Data Platform : HDP 3.1.0 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.0/index.html>.

HORTONWORKS. 2018d. SmartSense Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/SS1/SmartSense-1.5.1/release-notes/SmartSense_Release_Notes.pdf.

HORTONWORKS. 2019a. Apache Ambari Installation. *Ambari Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/Ambari-2.7.3.0/bk_ambari-installation/content/ch_Getting_Ready.html.

HORTONWORKS. 2019b. Cloudbreak Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/Cloudbreak/Cloudbreak-2.9.1/release-notes/cb_release-notes.pdf.

HORTONWORKS. 2019c. DataPlane Platform 1.2.3 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/DP/DP-1.2.3/release-notes/dp-release-notes.pdf>.

HORTONWORKS. 2019d. Hortonworks Data Platform : Apache Solr Search Installation. *Hortonworks Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
https://docs.cloudera.com/HDPDocuments/HDPS/HDPS-5.0.0/bk_solr-search-installation/bk_solr-search-installation.pdf.

HORTONWORKS. 2019e. Hortonworks Data Platform : HDP 3.1. *Hortonworks Documentation* [online]. Hortonworks. [Cit. 10. 6. 2019]. Dostupné z:
<https://hortonworks.com/products/data-platforms/hdp/>.

HORTONWORKS. 2019f. Hortonworks Data Platform : HDP 3.1 Release Notes. *Hortonworks Documentation* [online]. Hortonworks. [Cit. 8. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.0/release-notes/content/relnotes.html>.

HORTONWORKS. 2019g. Hortonworks Data Platform : HDP 3.1.4 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.4/release-notes/hdp-release-notes.pdf>.

HORTONWORKS. 2019h. Hortonworks Data Platform : HDP 3.1.5 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 11. 1. 2020]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/HDP3/HDP-3.1.5/release-notes/hdp-release-notes.pdf>.

HORTONWORKS. 2019i. Hortonworks DataFlow. *Hortonworks* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z: <https://hortonworks.com/products/data-platforms/hdf/>.

HORTONWORKS. 2019j. Hortonworks DataFlow : Release Notes. *HDF Documentation* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:
<https://docs.cloudera.com/HDPDocuments/HDF3/HDF-3.4.1.1/release-notes/hdf-release-notes.pdf>.

HORTONWORKS. 2019k. Quick Facts About the Business, Technology and Business Teams. *Hortonworks* [online]. Hortonworks. [Cit. 7. 12. 2019]. Dostupné z:

[https://hortonworks.com/about-us/quick-facts/.](https://hortonworks.com/about-us/quick-facts/)

HORTONWORKS. 2019l. Understanding Ambari Metrics System. *Ambari Documentation* [online]. Hortonworks. [Cit. 4. 5. 2019]. Dostupné z:

https://docs.hortonworks.com/HDPDocuments/Ambari-2.7.3.0/using-ambari-core-services/content/amb_understanding_ambari_metrics_service.html.

HORTONWORKS. 2020. DAS 1.4.4 Release Notes. *HDP Documentation* [online]. Hortonworks. [Cit. 15. 2. 2020]. Dostupné z:

<https://docs.cloudera.com/HDPDocuments/DAS/DAS-1.4.4/release-notes/das-release-notes.pdf>.

HPE. 2019. HPE advances its intelligent data platform with acquisition of MapR business assets. HPE Press Releases [online]. HPE. [Cit. 24. 11. 2019]. Dostupné z:

<https://www.hpe.com/us/en/newsroom/press-release/2019/08/hpe-advances-its-intelligent-data-platform-with-acquisition-of-mapr-business-assets.html>.

HUAWEI. 2019. Big Data. [online]. Huawei. [Cit. 20. 5. 2019]. Dostupné z:
<https://e.huawei.com/en/solutions/cloud-computing/big-data>.

HUESKE, F., KALAVRI, V. 2019. *Stream Processing with Apache Flink : Fundamentals, Implementation, and Operation of Streaming Applications*. 1. vyd. Sebastopol: O'Reilly. 310 s. ISBN: 978-1-491-97429-2.

HWANG, K., CHEN, M. 2017. *Big-Data Analytics for Cloud, IoT and Cognitive Computing*. 1. vyd. Hoboken: John Wiley & Sons. 432 s. ISBN: 978-1119247029.

CHEN, M., MAO, S., ZHANG, Y., LEUNG, V. C. M. 2014. *Big Data : Related Technologies, Challenges, and Future Prospects*. 1. vyd. Heidelberg: Springer. 102 s. ISBN: 978-3-319-06245-7.

CHIHOUB, H., IBRAHIM, S., ANTONIU, G., PEREZ, M. S. *Consistency Management in Cloud Storage Systems*. In: SAKR, S., GABER, M. M. 2014. *Large Scale and Big Data : Processing and Management*. 1. vyd. Boca Raton: CRC Press. s. 325-356. ISBN: 978-1-4665-8151-7.

INTEL. 2013. Intel Aims to Enrich Lives by Unlocking the Power of Big Data. *News Release* [online]. Intel. [Cit. 18. 5. 2019]. Dostupné z: <https://newsroom.intel.com/news-releases/intel-aims-to-enrich-lives-by-unlocking-the-power-of-big-data/>.

JAIN, V. K. 2017. *Big Data and Hadoop*. 1. vyd. New Delhi: Khanna Book Publishing. 622 s. ISBN: 978-93-82609-13-1.

KIŠKA, V. *Integrace Big Data a datového skladu*. Praha, 2017. Diplomová práce. Vysoká škola ekonomická v Praze. Fakulta informatiky a statistiky. Vedoucí práce doc. Ing. Ota Novotný, Ph.D.

KOBIELUS, J. G., POWERS, S., HOPKINS, B., EVELSON, B., COYNE, S. 2012. *The Forrester Wave : Enterprise Hadoop Solutions, Q1 2012* [online]. Forrester Research. [Cit. 20. 5. 2019]. Dostupné z:

ftp://ftp.software.ibm.com/software/in/Forrester_Enterprise_Hadoop_Solutions_Q1_2012_2.pdf.

KOMMENDA, N. 2020. Democratic primary delegate count – latest : Joe Biden and Bernie Sanders are competing to become the Democrats' nominee for president. *The Guardian* [online]. Guardian New & Media. [Cit. 14. 3. 2020]. Dostupné z: <https://www.theguardian.com/us-news/ng-interactive/2020/mar/18/democratic-primary-delegate-count-latest>.

KRISHNAN, S. 2016. Discovery and Consumption of Analytics Data at Twitter. *Twitter Blog* [online]. Twitter. [Cit. 9. 5. 2019]. Dostupné z: https://blog.twitter.com/engineering/en_us/topics/insights/2016/discovery-and-consumption-of-analytics-data-at-twitter.html.

KUMAR, A. 2018. *Architecting Data-Intensive Applications : Develop scalable, data-intensive, and robust applications the smart way*. 1. vyd. Birmingham: Packt Publishing. 340 s. ISBN: 978-1-78646-509-2.

KVASNICA, M. *Nová generace datového skladu reklamního systému Sklik.cz*. Zlín, 2017. Diplomová práce. Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky. Vedoucí práce prof. Ing. František Schauer, DrSc.

LANEY, D. 2001. *3D Data Management : Controlling Data Volume, Velocity, and Variety* [online]. Stamford: META Group. 3 s. Dostupné z: <https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf>.

LI, N. 2017. Big Data Processing at Spotify : The Road to Scio (Part 1). *Spotify Labs* [online]. Spotify. [Cit. 9. 5. 2019]. Dostupné z: <https://labs.spotify.com/2017/10/16/big-data-processing-at-spotify-the-road-to-scio-part-1/>.

LIU, H. 2018. Hortonworks DataFlow (HDF) 3.1 blog series part 5: Introducing Apache NiFi-Atlas integration [online]. Cloudera Blog. [Cit. 7. 12. 2019]. Dostupné z: <https://blog.cloudera.com/hdf-3-1-blog-series-part-6-introducing-nifi-atlas-integration/>.

LUBLINSKY, B., SMITH, K. T., YAKUBOVICH, A. 2013. *Professional Hadoop Solutions*. 1. vyd. Indianapolis: John Wiley & Sons. 474 s. ISBN: 978-1-118-61193-7.

LUU, H. 2018. *Beginning Apache Spark 2 : With Resilient Distributed Datasets, SparkSQL, Structured Streaming and Spark Machine Learning Library*. 1. vyd. New York: Apress. 408 s. ISBN: 978-1-4842-3578-2.

MACÁK, M. *Tools for big data analysis*. Brno, 2018. Diplomová práce. Masarykova univerzita. Fakulta informatiky. Vedoucí práce doc. Ing. RNDr. Barbora Bühnová, Ph.D.

MANE, S. B., SAWANT, Y., KAZI, S., SHINDE ,V. Real Time Sentiment Analysis of Twitter Data Using Hadoop. *International Journal of Computer Science and Information*

Technologies [online]. 2014, 5(3), 3098-3100. ISSN: 0975-9646. Dostupné z: <http://ijcsit.com/docs/Volume%205/vol5issue03/ijcsit2014050393.pdf>.

MAPR. 2011. MapR 1.0 Documentation. MapR Documentation archive [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: <https://mapr.com/docs/archive/relnotes/attachments/13828494/14712866.pdf>.

MAPR. 2012. MapR 2.0 Documentation. MapR Documentation archive [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: <https://mapr.com/docs/archive/relnotes/attachments/13828478/14712857.pdf>.

MAPR. 2013. MapR 3.0 Documentation. MapR Documentation archive [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: <https://mapr.com/docs/archive/relnotes/attachments/13828470/14712861.zip>.

MAPR. 2014a. MapR Technologies Surpasses 700 Customers. MapR Press Releases [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: <https://mapr.com/company/press-releases/mapr-technologies-surpasses-700-customers/>.

MAPR. 2014b. Release Notes : Version 4.0.0 FCS Release Notes. MapR Documentation archive [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/archive/relnotes/Version-4.0.0-FCS-Release-Notes_26280852.html.

MAPR. 2015. MapR Announces MapR Streams, Creating the Industry's First and Only Converged Data Platform. MapR Press Releases [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/company/press-releases/mapr-announces-mapr-streams-creating-industrys-first-and-only-converged-data/>.

MAPR. 2016a. About MapR 5.1. MapR 5.1 Documentation[online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/51/c_51_home.html.

MAPR. 2016b. About MapR 5.2. MapR 5.2 Documentation [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/52/c_52_home.html.

MAPR. 2016c. MapR Achieves Another Record Quarter with 99% Software Subscription License Growth. MapR Press Releases [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/company/press-releases/mapr-achieves-another-record-quarter-99-software-subscription-license-growth/>.

MAPR. 2016d. Release Notes : Release Notes. MapR Documentation archive [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/archive/relnotes/Release-Notes_13370000.html.

MAPR. 2016d. Release Notes : Version 5.0 Release Notes. MapR Documentation archive [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/archive/relnotes/Version-5.0-Release-Notes_30377219.html.

MAPR. 2017a. MapR Announces 81% Revenue Growth for First Fiscal Quarter. MapR Press Releases [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z:
<https://mapr.com/company/press-releases/mapr-announces-first-quarter-growth/>.

MAPR. 2017b. MapR Extends Convergence to the IoT Edge. MapR Press Releases [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z:
<https://mapr.com/company/press-releases/mapr-extends-convergence-iot-edge/>.

MAPR. 2018a. 6.1 Administration. MapR 6.1 Documentation [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z:
<https://mapr.com/docs/61/AdministratorGuide/ClstrAdminOverview.html>.

MAPR. 2018b. 6.1 Development. MapR 6.1 Documentation [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/docs/61/ApplicationDev.html>.

MAPR. 2018c. 6.1 Platform. MapR 6.1 Documentation [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: https://mapr.com/docs/61/MapROverview/c_overview_intro.html.

MAPR. 2018d. About MapR 6.0. MapR 6.0 Documentation [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/60/c_60_home.html.

MAPR. 2018e. About MapR 6.1. MapR 6.1 Documentation [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/61/c_60_home.html.

MAPR. 2019a. Component Versions for Released MEPs. MapR 6.1 Documentation [online]. MapR. [Cit. 8. 12. 2019]. Dostupné z:
https://mapr.com/docs/home/InteropMatrix/Component_versions_all_MEPs.html.

MAPR. 2019b. Contributor: John Schroeder. The MapR Blog [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: <https://mapr.com/blog/author/john-schroeder/>.

MAPR. 2019c. Contributor: M. C. Srivas. The MapR Blog [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: <https://mapr.com/blog/author/mc-srivas/>.

MAPR. 2019d. MapR Announces Clarity Program. MapR [online]. MapR. [Cit. 10. 6. 2019]. Dostupné z: <https://mapr.com/products/differentiation/cloudera/clarity/>.

MAPR. 2019e. MapR Control System. MapR [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/products/mapr-control-system/>.

MAPR. 2019f. MapR Editions. MapR Products [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/products/mapr-distribution-editions/>.

MAPR. 2019g. MapR End User License Agreement. MapR Resources [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/resources/eula/>.

MAPR. 2019h. MapR Installer. MapR 6.1 Documentation [online]. MapR. [Cit. 7. 12. 2019]. Dostupné z: <https://mapr.com/docs/home/MapRInstaller.html>.

MAPR. 2019i. Release History for Major Releases. MapR 6.1 Documentation [online]. MapR. [Cit. 23. 11. 2019]. Dostupné z: https://mapr.com/docs/61/InteropMatrix/r_release_dates.html.

MARAVIĆ, I. 2016. Spotify's Event Delivery – The Road to the Cloud (Part II). *Spotify Labs* [online]. Spotify. [Cit. 9. 5. 2019]. Dostupné z: <https://labs.spotify.com/2016/03/03/spotifys-event-delivery-the-road-to-the-cloud-part-ii/>.

MAYO, M. 2016. *Top Big Data Processing Frameworks* [online]. KDnuggets. [Cit. 21. 4. 2019]. Dostupné z: <https://www.kdnuggets.com/2016/03/top-big-data-processing-frameworks.html>.

MCDONALD, C. 2017. *Getting Started with Apache Spark : From Inception to Production* [online]. 2. vyd. Santa Clara: MapR. 193 s. Dostupné z: <https://go.mapr.com/rs/846-BMC-777/images/Spark2018eBook.pdf>.

MCDONALD, C. 2017. Database Comparison: An In-Depth Look at How MapR Database Does What Cassandra, HBase, and Others Can't. The MapR Blog [online]. MapR. [Cit. 24. 11. 2019]. Dostupné z: <https://mapr.com/blog/database-comparison-an-in-depth-look-at-mapr-db/>.

MCDONALD, C., DOWNARD, I. 2018. *Getting Started with Apache Spark : From Inception to Production* [online]. 2. vyd. Santa Clara: MapR. 193 s. Dostupné z: <https://go.mapr.com/rs/846-BMC-777/images/Spark2018eBook.pdf>.

MELL, P., GRANCE, T. 2011. *The NIST Definition of Cloud Computing* [online]. 1. verze. Gaithersburg: National Institute of Standards and Technology. 7 s. NIST Special Publication 800-145. Dostupné z: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>.

MICROSOFT. 2013. Windows Azure HDInsight is now Generally Available! *Azure Blog* [online]. Microsoft. [Cit. 19. 5. 2019]. Dostupné z: <https://azure.microsoft.com/cs-cz/blog/windows-azure-hdinsight-is-now-generally-available/>.

MICROSOFT. 2014. Hortonworks Makes HDP 2.0 for Windows Server Generally Available. *SQL Server Blog* [online]. Microsoft. [Cit. 7. 12. 2019]. Dostupné z: <https://cloudblogs.microsoft.com/sqlserver/2014/01/21/hortonworks-makes-hdp-2-0-for-windows-server-generally-available/>.

MICROSOFT. 2019a. Set up clusters in HDInsight with Apache Hadoop, Apache Spark, Apache Kafka, and more. *Azure HDInsight Documentation* [online]. Microsoft. [Cit. 19. 5. 2019]. Dostupné z: <https://docs.microsoft.com/en-us/azure/hdinsight/hdinsight-hadoop-provision-linux-clusters>.

MICROSOFT. 2019b. What are the Apache Hadoop components and versions available with HDInsight. *Azure HDInsight Documentation* [online]. Microsoft. [Cit. 2. 5. 2019]. Dostupné z: <https://docs.microsoft.com/en-us/azure/hdinsight/hdinsight-component-versioning>.

MILOŠ, M. *Nástroje pro Big Data Analytics*. Praha, 2013. Diplomová práce. Vysoká škola ekonomická v Praze. Fakulta informatiky a statistiky. Vedoucí práce doc. Ing. Jan Pour, CSc.

MUJUMDAR, P. 2013. *How HiveServer2 Brings Security and Concurrency to Apache Hive* [online]. Cloudera. [Cit. 8. 5. 2019]. Dostupné z:

<https://blog.cloudera.com/blog/2013/07/how-hiveserver2-brings-security-and-concurrency-to-apache-hive/>.

MURTHY, A. 2012. *Apache Hadoop YARN – Background and an Overview* [online]. Hortonworks. [Cit. 25. 4. 2019] Dostupné z: <https://hortonworks.com/blog/apache-hadoop-yarn-background-and-an-overview/>.

MURTHY, A. 2013. *Apache Hadoop 2 is now GA!* [online]. Hortonworks. [Cit. 26. 4. 2019] Dostupné z: <https://hortonworks.com/blog/apache-hadoop-2-is-ga/>.

NEMSCHOFF, M. 2016. *Gartner 2016 Magic Quadrant for Data Warehouse and Database Management Solutions for Analytics* [online]. MapR. [Cit. 12. 5. 2019]. Dostupné z:
<https://mapr.com/blog/gartner-2016-magic-quadrant-data-warehouse-and-database-management-solutions-analytics/>.

NGO, A. T. *Analýza nástrojů pro práci s Big Daty*. Praha, 2018. Diplomová práce. Vysoká škola ekonomická v Praze. Fakulta informatiky a statistiky. Vedoucí práce prof. Ing. Zdeněk Molnár, CSc.

NIELSEN, F. A. 2011. A New ANEW : Evaluation of a word list for sentiment analysis in microblogs. Dostupné z:

http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/6006/pdf/imm6006.pdf.

NOVOTNÝ, O., POUR, J., SLÁNSKÝ, D. 2005. *Business Intelligence : Jak využít bohatství ve vašich datech*. 1. vyd. Praha: Grada Publishing. 256 s. ISBN: 80-247-1094-3.

ODPI. 2019a. About. *ODPi* [online]. ODPI. [Cit. 12. 5. 2019]. Dostupné z: <https://www.odpi.org/about>.

ODPI. 2019b. Compliance Directory. *ODPi* [online]. ODPI. [Cit. 12. 5. 2019]. Dostupné z: <https://www.odpi.org/projects>.

ODPI. 2019c. Specifications. *ODPi* [online]. ODPI. [Cit. 12. 5. 2019]. Dostupné z: <https://www.odpi.org/projects/specifications>.

ONDATAENGINEERING. 2018. Hortonworks Data Platform Search. *OnDataEngineering* [online]. OnDataEngineering. [Cit. 7. 12. 2019]. Dostupné z:
<https://ondataengineering.net/technologies/hortonworks-data-platform-search/>.

ONDATAENGINEERING. 2019. Cloudbreak. *OnDataEngineering* [online]. OnDataEngineering. [Cit. 7. 12. 2019]. Dostupné z:
<https://ondataengineering.net/technologies/cloudbreak/>.

- PANETTIERI, J. 2016. *Pivotal Kills Hadoop Distribution, Embraces Hortonworks* [online]. ChannelE2E. [Cit. 18. 5. 2019]. Dostupné z: <https://www.channele2e.com/software/big-data/pivotal-kills-hadoop-distribution-embraces-hortonworks/>.
- PATTAMSETTI, R. M. R. 2017. *Distributed Computing in Java 9 : Make the best of Java for distributing applications*. 1. vyd. Birmingham: Packt Publishing. 304 s. ISBN: 978-1-78712-699-2.
- PIVOTAL. 2015. Pivotal Big Data Suite Accelerates Digital Transformation with Upgraded Apache Hadoop Distribution, Next Generation Analytical Performance. *Pivotal Press Release* [online]. Pivotal. [Cit. 18. 5. 2019]. Dostupné z: <https://pivotal.io/big-data/press-release/pivotal-big-data-suite-accelerates-digital-transformation>.
- PRAHBU, C. S. R. 2011. *Grid and Cluster Computing*. 4. vyd. Delhi: PHI Learning. ISBN: 978-81-203-3428-1.
- PYNE, S., RAO, P., B. L. S., RAO, S. B. *Big Data Analytics : Views from Statistical and Computational Perspectives*. In: PYNE, S., RAO, P., B. L. S., RAO, S. B. 2016. *Big Data Analytics : Methods and Applications*. 1. vyd. New Delhi: Springer. s. 1-10. ISBN: 978-81-322-3626-9.
- RICHTER, J., FOTR, J., ŠVECOVÁ, L. 2016. *Výběr kritérií, tvorba variant a stanovení jejich důsledků*. In: FOTR, J., ŠVECOVÁ, L., HRŮZOVÁ, H., RICHTER, J. 2016. *Manažerské rozhodování : postupy, metody, nástroje*. 3., přepracované vyd. Praha: Ekopress. s. 119-152. ISBN: ISBN 978-80-87865-33-0.
- RONTHAL, A. M., EDJLALI, R., GREENWALD, R. 2018. *Magic Quadrant for Data Management Solutions for Analytics* [online]. Gartner. [Cit. 12. 5. 2019]. Dostupné z: <https://b2bsalescafe.files.wordpress.com/2018/03/magic-quadrant-for-data-management-solutions-for-analytics.pdf>.
- ROTTINGHUIS, J. 2019. Partly Cloudy : The start of a journey into the cloud. *Twitter Blog* [online]. Twitter. [Cit. 9. 5. 2019]. Dostupné z: https://blog.twitter.com/engineering/en_us/topics/infrastructure/2019/the-start-of-a-journey-into-the-cloud.html.
- RUNKLER, T. A. 2016. *Data Analytics : Models and Algorithms for Intelligent Data Analysis*. 2. vyd. Wiesbaden: Springer Vieweg. 150 s. ISBN: 078-3-658-14074-8.
- SAMMER, E., WHITE, T. 2013. Cloudera Development Kit (CDK) : Hadoop Application Development Made Easier. *Cloudera Engineering Blog* [online]. Cloudera. [Cit. 28. 5. 2019]. Dostupné z: <https://blog.cloudera.com/blog/2013/05/cloudera-development-kit-cdk/>.
- SANGHEETA, S., SREEJA, A. K. No Science No Humans, No New Technologies No Changes : „Big Data a Great Revolution“. *International Journal of Computer Science and Information Technologies* [online]. 2015, 6(4), 3269-3274. ISSN: 0975-9646. Dostupné z: <http://ijcsit.com/docs/Volume%206/vol6issue04/ijcsit2015060405.pdf>.

SARKAR, D. 2018. Emotion and Sentiment Analysis : A Practitioner's Guide to NLP [online]. KDnuggets. [Cit. 4. 1. 2020]. Dostupné z:
<https://www.kdnuggets.com/2018/08/emotion-sentiment-analysis-practitioners-guide-nlp-5.html>.

SCOTT, J. A. 2015. *Getting Started with Apache Spark : From Inception to Production* [online]. 1. vyd. San Jose: MapR. 84 s. Dostupné z: https://www.bigdata-toronto.com/2016/assets/getting_started_with_apache_spark.pdf.

SHAW, S., VERMEULEN, A. F., GUPTA, A., KJERRUMGAARD, D. 2016. *Practical Hive : A Guide to Hadoop's Data Warehouse System*. 1. vyd. New York: Apress. 256 s. ISBN: 978-1-4842-0272-2.

SHVACHKO, K. V., ZHANG, Z., KROGEN, E. 2018. *Scaling Hadoop at LinkedIn* [online]. LinkedIn. [Cit. 9. 5. 2019]. Dostupné z:
https://www.slideshare.net/Hadoop_Summit/scaling-hadoop-at-linkedin-107176757.

SINGH, CH., KUMAR, M. 2019. *Mastering Hadoop 3 : Big data processing at scale to unlock unique business Insights*. 1. vyd. Birmingham: Packt Publishing. 512 s. ISBN: 978-1-78862-044-4.

SMITH, D., LEONG, L., BALA, R. 2018. *Magic Quadrant for Cloud Infrastructure as a Service, Worldwide* [online]. Stamford: Gartner. 25 s. Dostupné z:
<https://www.docdroid.net/wROPdh5/gartner-2018-magic-quadrant-for-cloud-infrastructure-as-a-service-worldwide.pdf>.

STEINBACH, C. 2016. *The Past, Present, and Future of Hadoop @ LinkedIn* [online]. LinkedIn. [Cit. 9. 5. 2019]. Dostupné z:
<https://www.slideshare.net/cwsteinbach/the-past-present-and-future-of-hadoop-at-linkedin>.

SYNCFUSION. 2019. Big Data – No Big Deal. [online]. Syncfusion. [Cit. 21. 5. 2019]. Dostupné z: <https://www.syncfusion.com/products/big-data>.

ŠVECOVÁ, L., FOTR, J. *Hodnocení variant rozhodování*. In: FOTR, J., ŠVECOVÁ, L., HRŮZOVÁ, H., RICHTER, J. 2016. *Manažerské rozhodování : postupy, metody, nástroje*. 3., přepracované vyd. Praha: Ekopress. s. 153-216. ISBN: ISBN 978-80-87865-33-0.

TAN, W., VAVILAPALLI, V. K. 2018a. *Apache Hadoop 3.1.0 released. And a look back!* [online]. Hortonworks. [Cit. 27. 4. 2019] Dostupné z:
<https://hortonworks.com/blog/apache-hadoop-3-1-0-released-and-a-look-back/>.

TAN, W., VAVILAPALLI, V. K. 2018b. *First Class GPUs support in Apache Hadoop 3.1, YARN & HDP 3.0* [online]. Hortonworks. [Cit. 27. 4. 2019] Dostupné z:
<https://hortonworks.com/blog/gpus-support-in-apache-hadoop-3-1-yarn-hdp-3/>.

TURKINGTON, G., MODENA, G. 2015. *Learning Hadoop 2 : Design and implement data processing, lifecycle management, and analytic workflows with the cutting-edge toolbox of Hadoop 2*. 1. vyd. Birmingham: Packt Publishing. 382 s. ISBN: 978-1-78328-551-8.

TWITTER. 2019a. Apply for access. *Twitter Developer* [online]. Twitter. [Cit. 24. 9. 2019]. Dostupné z: <https://developer.twitter.com/en/apply-for-access>.

TWITTER. 2019b. Create an app. *Twitter Developer* [online]. Twitter. [Cit. 24. 9. 2019]. Dostupné z: <https://developer.twitter.com/en/apps/create>.

TWITTER. 2019c. Filter realtime Tweets. *Twitter Developer* [online]. Twitter. [Cit. 15. 10. 2019]. Dostupné z:

<https://developer.twitter.com/en/docs/tweets/filter-realtime/overview>.

TWITTER. 2019d. Filtered stream – Overview. *Twitter Developer* [online]. Twitter. [Cit. 7. 12. 2019]. Dostupné z:

<https://developer.twitter.com/en/docs/labs/filtered-stream/overview>.

TWITTER. 2019e. Get access to the Twitter API. *Twitter Developer* [online]. Twitter. [Cit. 24. 9. 2019]. Dostupné z: <https://developer.twitter.com/en/application>.

TWITTER. 2019f. Getting started. *Twitter Developer* [online]. Twitter. [Cit. 14. 10. 2019]. Dostupné z: <https://developer.twitter.com/en/docs/basics/getting-started>.

TWITTER. 2019g. Pricing. *Twitter Developer* [online]. Twitter. [Cit. 14. 10. 2019]. Dostupné z: <https://developer.twitter.com/en/pricing>.

TWITTER. 2019h. Sample realtime Tweets. *Twitter Developer* [online]. Twitter. [Cit. 15. 10. 2019]. Dostupné z: <https://developer.twitter.com/en/docs/tweets/sample-realtime/overview>.

TWITTER. 2019i. Search Tweets. *Twitter Developer* [online]. Twitter. [Cit. 15. 10. 2019]. Dostupné z: <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets>.

TWITTER. 2019j. Spark Sentiment Analysis App. *Twitter Developer* [online]. Twitter. [Cit. 24. 9. 2019]. Dostupné z: <https://developer.twitter.com/en/apps/16853186>.

TWITTER. 2019k. Twitter API Status. *Twitter API Status* [online]. Twitter. [Cit. 14. 10. 2019]. Dostupné z: <https://api.twitterstat.us/>.

TWITTER. 2020a. Brand Resources. *Twitter Company* [online]. Twitter. [Cit. 4. 1. 2020]. Dostupné z: https://about.twitter.com/en_us/company/brand-resources.html.

TWITTER. 2020b. Filtered stream – Quickstart. *Twitter Developer* [online]. Twitter. [Cit. 4. 1. 2020]. Dostupné z:
<https://developer.twitter.com/en/docs/labs/filtered-stream/quick-start>.

VAUGHAN, J. 2019. *Cloudera and Hortonworks combo to push CDP, machine learning* [online]. TechTarget. [Cit. 10. 6. 2019] Dostupné z:

<https://searchdatamanagement.techtarget.com/news/252455907/Cloudera-and-Hortonworks-combo-to-push-CDP-machine-learning>.

WANDISCO. 2013. WANdisco releases new version of Hadoop Distro; WDD 3.1.1 simplifies Hadoop deployments. WANdisco Press Release [online]. WANdisco. [Cit. 19. 5. 2019]. Dostupné z: <https://www.wandisco.com/news-events/press-releases/wandisco-releases-new-version-of-hadoop-distro-wdd-3-1-1-simplifies-hadoop-deployments>.

WEBER INFORMATICS. 2018. Download kafka-clients JAR 2.0.0 with all dependencies. *JAR Download* [online]. Weber Informatics. [Cit. 14. 3. 2020]. Dostupné z: <https://jar-download.com/artifacts/org.apache.kafka/kafka-clients/2.0.0/source-code>.

WOODIE, A. 2018. New Cloudera Plots a Course Toward a Unified Future [online]. Datanami. [Cit. 11. 1. 2020]. Dostupné z: <https://www.datanami.com/2018/10/24/new-cloudera-plots-a-course-toward-a-unified-future/>.

WU, D., SAKR, S., ZHU, L. *Big Data Programming Models*. In: ZOMAYA, A. Y., SAKR, S. 2017a. *Handbook of Big Data Technologies*. 1. vyd. Cham: Springer. s. 31-64. ISBN: 978-3-319-49339-8.

WU, D., SAKR, S., ZHU, L. *Big Data Storage and Data Models*. In: ZOMAYA, A. Y., SAKR, S. 2017b. *Handbook of Big Data Technologies*. 1. vyd. Cham: Springer. s. 3-30. ISBN: 978-3-319-49339-8.

Přílohy

Příloha A: Milníky vývoje distribuce CDH

Rok	Měsíc	Událost
2008	Březen	<ul style="list-style-type: none">• Založena společnost Cloudera
2009	Březen	<ul style="list-style-type: none">• Produkční CDH 1.x (Hadoop, Hive, Pig)
	Srpna	<ul style="list-style-type: none">• Doug Cutting se připojuje k týmu Cloudera
2010	Červen	<ul style="list-style-type: none">• Neprodukční CDH 3.x v edicích Express a Enterprise
	Duben	<ul style="list-style-type: none">• Produkční CDH 3.x (Hadoop, HBase, Hive, Pig, Sqoop, Flume, Hue, Oozie, ZooKeeper), podpora 64bit
2011	Červen	<ul style="list-style-type: none">• CDH 3.5 zahrnující Service Configuration Manager
	Prosinec	<ul style="list-style-type: none">• CDH 3.7 – SCM → Cloudera Manager
2012	Duben	<ul style="list-style-type: none">• CDH 4.x beta – HDFS HA, YARN + MapReduce 2
	Červen	<ul style="list-style-type: none">• Produkční CDH 4.x
	Říjen	<ul style="list-style-type: none">• Beta Impala – BI dotazování nad big data v rámci CDH
2013	Únor	<ul style="list-style-type: none">• CDH 4.5 – BDR, Cloudera Navigator 1.0
	Duben	<ul style="list-style-type: none">• Produkční Impala 1.0 – MPP pro BI SQL nad Hadoop
	Červen	<ul style="list-style-type: none">• Cloudera Search beta – vyhledávací engine pro CDH
	Červenec	<ul style="list-style-type: none">• Sentry – RBAC autorizace pro Hive, Impala
	Září	<ul style="list-style-type: none">• Produkční Cloudera Search
	Říjen	<ul style="list-style-type: none">• Neprodukční CDH 5.x (Hadoop, HBase, Hue, Flume, Hive, Cloudera Impala, Mahout, Oozie, Cloudera Search, Sentry, Sqoop)
2014	Duben	<ul style="list-style-type: none">• Produkční CDH 5.x
	Říjen	<ul style="list-style-type: none">• CDH 5.2 – bezpečnostní a funkční vylepšení, Impala 2.0, Cloudera Director
2015	Únor	<ul style="list-style-type: none">• Dokončena integrace Kafka v rámci CDH
	Srpna	<ul style="list-style-type: none">• Cloudera Director 1.5 (podpora AWS a GCP)
	Září	<ul style="list-style-type: none">• Distribuce CDH dostupná v Microsoft Azure Marketplace
	Listopad	<ul style="list-style-type: none">• Beta verze Kudu
		<ul style="list-style-type: none">• CDH 5.5 (+ beta Cloudera Navigator Optimizer)
2016	Leden	<ul style="list-style-type: none">• Cloudera Director 2.0
	Duben	<ul style="list-style-type: none">• CDH 5.7 – Hive-on-Spark
	Červenec	<ul style="list-style-type: none">• CDH 5.8 – produkční verze Cloudera Navigator Optimizer
	Září	<ul style="list-style-type: none">• Kudu 1.0
2017	Leden	<ul style="list-style-type: none">• CDH 5.10 (+ Kudu)
	Březen	<ul style="list-style-type: none">• Beta verze Cloudera Data Science Workbench
	Květen	<ul style="list-style-type: none">• Produkční verze Cloudera Data Science Workbench

	Září	<ul style="list-style-type: none"> • Cloudera Altus v rámci AWS • Cloudera Shared Data Experience • Beta verze Cloudera Altus v rámci Microsoft Azure • Beta verze Cloudera Altus Analytic DB
	Listopad	<ul style="list-style-type: none"> • Beta verze Cloudera Altus Analytic DB
2018	Květen	<ul style="list-style-type: none"> • Beta verze Cloudera Data Science Workbench 1.4 • Produkční verze Cloudera Altus v rámci Microsoft Azure • Beta verze CDH 6.0
	Srpen	<ul style="list-style-type: none"> • Produkční verze Cloudera Altus Data Warehouse (dříve Cloudera Altus Analytic DB)
	Září	<ul style="list-style-type: none"> • Produkční verze CDH 6.0, Cloudera Workload Experience Manager
	Říjen	<ul style="list-style-type: none"> • Akvizice Hortonworks a zahájení sloučení
2019	Leden	<ul style="list-style-type: none"> • Dokončeno sloučení Cloudera a Hortonworks – Cloudera poskytuje CDH i HDP, CDF i HDF + vlastní portfolio
	Březen	<ul style="list-style-type: none"> • Produkční verze CDH 6.2.0
2020	Březen	<ul style="list-style-type: none"> • Produkční verze CDH 6.3.3 – první verze CDH, v jejímž případě nejsou repositáře veřejně dostupné

Zdroj: data (Bonachi, 2015; Cloudera, 2009; Cloudera, 2010; Cloudera, 2011c; Cloudera, 2011b; Cloudera, 2011a; Cloudera, 2012b; Cloudera, 2012c; Cloudera, 2012a; Cloudera, 2013f; Cloudera, 2013a; Cloudera, 2013c; Cloudera, 2013g; Cloudera, 2013e; Cloudera, 2013d; Cloudera, 2014b; Cloudera, 2014c; Cloudera, 2015e; Cloudera, 2015a; Cloudera, 2015b; Cloudera, 2015c; Cloudera, 2015d; Cloudera, 2016b; Cloudera, 2016e; Cloudera, 2016d; Cloudera, 2016c; Cloudera, 2017a; Cloudera, 2017g; Cloudera, 2017b; Cloudera, 2017e; Cloudera, 2017f; Cloudera, 2017d; Cloudera, 2017c; Cloudera, 2018a; Cloudera, 2018g; Cloudera, 2018d; Cloudera, 2019c; Cloudera, 2019d; Cloudera, 2019e; Cloudera, 2020a), vlastní zpracování

Příloha B: Milníky vývoje distribuce MDP

Rok	Měsíc	Událost
2011	Červen	<ul style="list-style-type: none"> Založena společnost MapR Technologies Produkční MDP 1.x (MapR-FS, MapReduce, HBase, ZooKeeper, MCS) – edice M3, M5
2012	Srpna	<ul style="list-style-type: none"> Produkční MDP 2.x (MapR-FS, MapReduce, HBase, ZooKeeper, MCS, Cascading, Flume, Hive, Mahout, Oozie, Pig, Sqoop, Whirr)
2013	Květen	<ul style="list-style-type: none"> MDP 3.x – (MapR-FS, MapReduce, HBase, MapR-DB, ZooKeeper, MCS, Cascading, Flume, Hive, Mahout, Oozie, Pig, Sqoop, Whirr) – nová edice M7 (optimalizovaná proprietární komponenta HBase – MapR-DB) MDP dostupná v rámci Amazon EMR
2014	Červen	<ul style="list-style-type: none"> MDP 4.x (MapR-FS, MapReduce, YARN, HBase, MapR-DB, ZooKeeper, MCS, Cascading, Flume, Hive, Mahout, Oozie, Pig, Sqoop, Whirr, Spark, Hue, Impala) – přechod na Hadoop 2.x + YARN
2015	Červenec	<ul style="list-style-type: none"> MDP 5.x (MapR-FS, MapReduce, YARN, HBase, MapR-DB, ZooKeeper, MCS, Cascading, Flume, Hive, Mahout, Oozie, Pig, Sqoop, Spark, Hue, Impala, Sqoop2, Sentry, Drill)
2016	Únor	<ul style="list-style-type: none"> MDP 5.1 (MapR-FS, MapReduce, YARN, HBase, MapR-DB, MapR-Streams, MCS, Cascading, Flume, Hive, Mahout, Oozie, Pig, Sqoop, Spark, Hue, Impala, Sqoop2, Sentry, Druid, Myriad) – změna názvu na MapR Converged Data Platform, změna názvů edic
	Srpna	<ul style="list-style-type: none"> MDP 5.2 – navíc externí nástroje pro monitoring – Grafana, Kibana, dále zavedeny častější aktualizace prostřednictvím MEP, MapR-FS mění název na MapR-XD
2017	Listopad	<ul style="list-style-type: none"> MDP 6.0 – redesign grafického rozhraní MCS, podpora integrace S3, OpenStack Manilla, MDP kontejner pro vývojáře
2018	Září	<ul style="list-style-type: none"> MDP 6.1 – vylepšení zabezpečení, možnost šifrování dat v clusteru, MapR-Streams mění název na MapR-ES, vyšší verze Kafka, ZooKeeper
2019	Květen	<ul style="list-style-type: none"> Po selhání jednání s investory zakladatel John Schroeder informuje zaměstnance, že pokud se nepodaří získat finančního partnera, bude nucen společnost MapR Technologies uzavřít
	Srpna	<ul style="list-style-type: none"> HPE realizuje akvizici MapR Technologies, včetně MDP

Zdroj: data (Gillin, 2019; MapR, 2011; MapR, 2012; MapR, 2013; MapR, 2014b; MapR, 2016a; MapR, 2016b; MapR, 2016d; MapR, 2016d; MapR, 2018d; MapR, 2018e; MapR, 2019i), vlastní zpracování

Příloha C: Milníky vývoje distribuce HDP

Rok	Měsíc	Událost
2011	Červenec	<ul style="list-style-type: none"> • 24 bývalých zaměstnanců ze společnosti Yahoo! zakládá společnost Hortonworks
2012	Červenec	<ul style="list-style-type: none"> • HDP 1.0.0.12 (Hortonworks Management Center, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Ganglia, Nagios) - první oficiálně dostupná verze HDP
2013	Leden	<ul style="list-style-type: none"> • HDP 1.2.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Ganglia, Nagios) – Hortonworks Management Center mění název na Ambari
	Říjen	<ul style="list-style-type: none"> • HDP 2.0.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Ganglia, Nagios) – přechod distribuce HDP na Hadoop 2
2014	Leden	<ul style="list-style-type: none"> • HDP 2.0.0 pro Windows Server
	Duben	<ul style="list-style-type: none"> • HDP 2.1.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Tez, Hue, Storm, Falcon, Knox, Accumulo, Phoenix, Solr, Ganglia, Nagios)
	Prosinec	<ul style="list-style-type: none"> • Hortonworks Data Platform Search • HDP 2.2.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Tez, Hue, Storm, Falcon, Knox, Accumulo, Phoenix, Avro, Ranger, DataFu, Kafka, Slider, Solr, Ganglia, Nagios), Ganglia a Nagios označeny jako deprecated, monitoring clusteru HDP nadále zajišťuje Ambari
	Červenec	<ul style="list-style-type: none"> • HDP 2.3.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Tez, Hue, Storm, Falcon, Knox, Accumulo, Phoenix, Avro, Ranger, DataFu, Kafka, Slider, Solr, Calcite, Cascading, Spark), Nástroje Ganglia a Nagios z distribuce odstraněny, nahrazeny funkcionalitou Ambari
2015	Červenec	<ul style="list-style-type: none"> • Cloudbreak 1.0 – nástroj pro automatizované nasazování HDP nad cloudovou infrastrukturou AWS, GCP, Azure, OpenStack
	Říjen	<ul style="list-style-type: none"> • SmartSense 1.0.0 – produkt pro proaktivní monitoring clusteru HDP
	Prosinec	<ul style="list-style-type: none"> • HDF 1.1.0 (Ambari, NiFi, MiNiFi, Schema Registry, Streaming Analytics Manager, Kafka, Storm, Ranger) – platforma pro zpracování proudů dat v reálném čase
2016	Březen	<ul style="list-style-type: none"> • HDP 2.4.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Tez, Hue, Storm, Falcon, Knox, Accumulo, Phoenix, Avro, Ranger, DataFu, Kafka, Slider, Calcite, Solr, Cascading, Spark, Atlas)

	Srpen	<ul style="list-style-type: none"> • HDP 2.5.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Tez, Hue, Storm, Falcon, Knox, Accumulo, Phoenix, Avro, Ranger, DataFu, Kafka, Slider, Calcite, Solr, Cascading, Spark, Zeppelin, Atlas)
	Září	<ul style="list-style-type: none"> • HDF 2.0.0
	Duben	<ul style="list-style-type: none"> • HDP 2.6.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Flume, Mahout, Tez, Hue, Storm, Falcon, Knox, Accumulo, Phoenix, Avro, Ranger, DataFu, Kafka, Slider, Calcite, Solr, Cascading, Spark, Zeppelin, Atlas), Falcon, Flume, Mahout, Slider, Cascading, Hue označeny jako deprecated
2017	Červen	<ul style="list-style-type: none"> • HDF 3.0.0, Technical preview Hortonworks Schema Registry, Hortonworks Streaming Analytics Manager
	Říjen	<ul style="list-style-type: none"> • Hortonworks DataPlane Service (později Hortonworks DataPlane Platform) 1.0.0 pro aplikace Data Steward Studio, Data Lifecycle Manager
	Červenec	<ul style="list-style-type: none"> • HDP 3.0.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Tez, Storm, Knox, Accumulo, Phoenix, Ranger, DataFu, Kafka, Calcite, Solr, Spark, Atlas, Druid, Livy, Spark, Zeppelin, Superset), přechod distribuce na Hadoop 3, přepracované grafické rozhraní Ambari
2018	Srpen	<ul style="list-style-type: none"> • Data Analytics Studio 1.0.0
	Říjen	<ul style="list-style-type: none"> • Oznámeno sloučení Hortonworks a Cloudera
	Listopad	<ul style="list-style-type: none"> • SmartSense 1.5.1
	Prosinec	<ul style="list-style-type: none"> • HDP 3.1.0 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Tez, Storm, Knox, Accumulo, Phoenix, Ranger, DataFu, Kafka, Calcite, Solr, Spark, Atlas, Druid, Livy, Spark, Zeppelin, Superset), poslední verze HDP před dokončením sloučení Hortonworks se společností Cloudera
	Leden	<ul style="list-style-type: none"> • Dokončeno sloučení Hortonworks s Cloudera
	Březen	<ul style="list-style-type: none"> • Hortonworks DataPlane Platform 1.2.3
	Květen	<ul style="list-style-type: none"> • Cloudbreak 2.9.1
	Srpen	<ul style="list-style-type: none"> • HDF 3.4.1.1
2019		<ul style="list-style-type: none"> • HDP 3.1.4 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Tez, Storm, Knox, Accumulo, Phoenix, Ranger, DataFu, Kafka, Calcite, Solr, Spark, Atlas, Druid, Livy, Spark, Zeppelin, Superset)
	Prosinec	<ul style="list-style-type: none"> • Hortonworks Data Platform Search 5.0.0 • HDP 3.1.4 (Ambari, Hadoop, Pig, Hive, HBase, ZooKeeper, Oozie, Sqoop, Tez, Storm, Knox, Accumulo, Phoenix, Ranger, DataFu, Kafka, Calcite, Solr, Spark, Atlas, Druid, Livy, Spark, Zeppelin, Superset), repositáře distribuce přestávají být veřejně dostupné
2020	Leden	<ul style="list-style-type: none"> • Data Analytics Studio 1.4.4

Zdroj: data (Hortonworks, 2019k; Hortonworks, 2012a; Hortonworks, 2013a; Hortonworks, 2013b; Microsoft, 2014; Hortonworks, 2014c; Hortonworks, 2014b; OnDataEngineering, 2018; Hortonworks, 2015d; Cision PR Newswire, 2015; Hortonworks, 2015c; Hortonworks, 2016b; Hortonworks, 2016c; Hortonworks, 2016d; Hortonworks, 2017a; Hortonworks, 2017b; Hortonworks, 2017c; Hortonworks, 2018a; Hortonworks, 2018b; Hortonworks, 2018c; Cloudera, 2018c; Hortonworks, 2019c; Hortonworks, 2019d; Hortonworks, 2018d; Cloudera, 2019c; Hortonworks, 2019g; Hortonworks, 2019h; Hortonworks, 2019j; Hortonworks, 2020), vlastní zpracování

Příloha D: Verze a komponenty nasazených distribucí frameworku Apache Hadoop

Služba/komponenta	CDH 6.2.0	HDP 3.1.0	MDP 6.1.0³³
Accumulo	-	1.7.0	-
Atlas	-	1.1.0	-
Avro	1.8.2	-	-
Calcite	-	1.16.0	-
DataFu	-	1.3.0	-
Drill	-	-	1.15.0.0
Druid	-	0.12.1	-
Flume	1.9.0	-	1.8
Hadoop	3.0.0	3.1.1	-
HBase	2.1.2	2.0.2	1.1.13
Hive	2.1.1	3.1.0	2.3.4
Hue	4.3.0	-	4.3.0
Impala	3.2.0	-	2.10**
Kafka	2.1.0	2.1.0	-
Kite SDK	1.0.0	-	-
Knox	-	1.0.0	-
Kudu	1.9.0	-	-
Livy	-	0.5.0	0.5.0
Mahout	-	-	-
Myriad	-	-	0.2
Oozie	5.1.0	4.3.1	5.1.0
Parquet	1.9.0	-	-
Phoenix	-	5.0.0	-
Pig	0.17.0	0.16.0	0.16

³³) Distribuce MDP je typická tím, že core komponenty Hadoop nahrazuje vlastní reimplementací, což je důvod, proč verze Hadoop v tabulce není uvedena. Distribuce MDP ve verzi 6.1.0 nepodporovala nasazení služby Impala pro OS Ubuntu, a proto v rámci realizace praktické úlohy zpracování big data tato komponenta nasazena nebyla. Přesto je v tabulce verze dané komponenty uvedena pro zajištění možnosti srovnání. Distribuce MDP substituuje komponentu Kafka vlastní alternativou v podobě Mapr Event Store, která byla ve verzi MDP 6.1.0 kompatibilní s Kafka Streams 1.1.

Ranger	-	1.2.0	-
Sentry	2.1.0	-	1.7.0
Solr	7.4.0	-	-
Spark	2.4.0	2.3.2	2.4.4.0
Sqoop	1.4.7	1.4.7	1.4.7
Sqoop2	-	-	1.99.7
Storm	-	1.2.1	-
Tez	-	0.9.1	0.9.1
Zeppelin	-	0.8.0	-
ZooKeeper	3.4.5	3.4.6	-

Pozn: V případě distribuce CDH není služba Flume v rámci komunitní edice k dispozici.
 Zdroj: vlastní zpracování, dle (Cloudera, 2019a; Hortonworks, 2019f; MapR, 2019a)

Příloha E: Srovnání kroků průvodce instalací distribuce frameworku Apache Hadoop

Krok	CDH	HDP	MDP
Úvodní obrazovka	✓ (1)	✗	✓ (1)
Akceptace licence distribuce	✓ (2)	✗	✗
Výběr edice licence	✓ (3)	✗	✓ (2)
Specifikace názvu clusteru	✓ (4)	✓ (1)	✓ (4)
Specifikace množiny uzlů v clusteru	✓ (5)	✓ (3)	✓ (5,6)
Výběr instalačních repositářů	✓ (6)	✓ (2)	✓ (2)
Akceptace licence JDK	✓ (7)	✗	✗
Specifikace přístupových údajů k uzlům clusteru	✓ (8)	✓ (3)	✓ (3, 4, 5)
Nastavení konfigurace disků	✗	✗	✓ (5)
Instalace agentů distribuce na uzly clusteru	✓ (9)	✓ (4)	✓ (8)
Instalace balíků distribuce na uzly clusteru	✓ (10)	✗	✓ (8)
Validace instalovaného clusteru	✓ (11)	✗	✗
Výběr služeb (Komponent) clusteru Hadoop	✓ (12)	✓ (5)	✓ (2)
Rozmístění služeb napříč uzly clusteru	✓ (13)	✓ (6, 7)	✓ (2,7)
Nastavení a validace přístupu k DBMS	✓ (14)	✓ (8)	✓ (3)
Schválení konfigurace	✓ (15)	✓ (9)	✓ (7)
Instalace vybraných služeb clusteru Hadoop	✓ (16)	✓ (10)	✓ (8)
Aplikace zvolené licence v clusteru	✗	✗	✓ (9)
Shrnutí	✓ (17)	✓ (11)	✓ (10)

Pozn: V závorce je indikována posloupnost v rámci průvodce nasazením distribuce Hadoop.
Zdroj: vlastní zpracování