

Mircea-Radu-Serban Dobrau

Emerging Web Technologies

Report Sem 2

by Mircea-Radu-Serban Dobrau

FILE	2787_MIRCEA-RADU-SERBAN_DOBRAU_MIRCEA-RADU-SERBAN_DOBRAU_EMERGING_WEB_TECHNOLOGIES_REPORT_SEM_2_93287_1773097060.DOCX		
TIME SUBMITTED	28-MAR-2018 08:54 PM (UTC+0100)	WORD COUNT	3476
SUBMISSION ID	85753422	CHARACTER COUNT	20410

Mircea-Radu-Şerban Dobrău

P147987

Web Development (BSc)

**Emerging Web Technologies
(M6X01096)**

Michael Dacey

Abstract

It is estimated that in the following years data will be measured in terms of zettabytes in terms of illustrating the rapid growth of data traffic, where one zettabyte equals one sextillion (10^{21}) bytes. [1] As a result, businesses and enterprises have been slowly moving from traditional models of managing, processing and storing data with relational database management systems (RDBMS). "Big Data" is a term used for describing the storage, management and analysis of large volumes of data using new types of database management systems such as Apache Cassandra. This report starts off with a description of 4 Big Data principles: Volume, Velocity, Variety and Veracity. Next, Netflix is given as an example of a service using Big Data to power its functionality alongside how the four principles fit, as well as research on how Netflix attempts to solve big data challenges and how the 4 Big Data principles apply. Furthermore, the distributed Apache Cassandra NoSQL database is introduced along with the presentation of 5 of its properties: its' Data Model, peer-to-peer (P2P) architecture and resulting fault-tolerance, horizontal scalability, and CQL (Cassandra Query Language). The report ends with the benefits a web application receives directly or indirectly by using Cassandra as its database management system (DBMS).

Table of Contents

Big Data	4
Four V's	4
Case Study: Netflix.....	7
Hadoop Technology: Cassandra	10
Introduction	10
Data-Model.....	11
Peer-to-peer. Fault tolerance.....	11
Horizontal scalability.....	12
CQL.....	13
Cassandra Benefits	14
References	15

Big Data

Four V's

Volume

The "Volume" principle refers to the fact of data growing beyond terabytes into petabytes or even exabytes (1 million terabytes). [2, p. 1] As data has been traditionally stored in tables and files this presents a challenge for distributed environments in managing and processing the data. Attacks on databases which involve generating ambiguous data is also an issue with traditional data. [2, p. 84]

A different research paper [3] points out that data often comes from location-dependent and diverse data streams that contain different kinds of data generated at high speeds (or velocity) from multiple types of sensors; These data sources include wearable technologies, cloud-based services, enterprise data warehouses and NoSQL databases.

This problem is addressed with the help of commodity hardware and distributed storage solutions such as the Hadoop Distributed File System (HDFS) [3].

Velocity

The "Velocity" Principle on the other hand refers to the increasing rate of data that is being collected, store and analyzed/processed. Additionally, it represents the ability for building predictive models and using more data than can be used using traditional data sources [4, pp. 151–152]

Another definition of the velocity principle is that data content is constantly evolving through the addition of extra data collections as well as data coming from legacy or archived data collections, as well as data coming through streams from multiple sources. [5, p. xx]

This challenge can be solved with software doing real-time processing, as well as processing of incoming data streams and in-memory computing. [3] Another approach includes querying of “event” histories, which suppose that performing dynamic and continuous/recurrent querying and processing is possible on logs which are continuously fed with incoming events.

On the other hand, complex event processing (CEP) supposes that streams cannot be stored, and on-line processing and delivery is performed at a rate which eventually combines streams with stored data. [6, p. 4]

Variety

The novelty of this issue has been brought about by new types of technologies such as portable devices and sensors. [2, p. 95] In terms of issues and vulnerabilities, the “variety of data” says that from a structural point of view data can be of multiple types [2, p. 84]:

- Structured (Such as financial transactions [7, p. 31])
- Unstructured (Such as images [7, p. 31] or data that doesn't reside within a data model or schema [8, p. 34])
- Single Factor
- Multi Factor
- Probabilistic
- Linked
- Dynamic

In other words, data can come under many forms such as documents, web files, XML files, multimedia, IOT data, open data etc. [3]

All these forms of data must be integrated together [8, p. 133] so that they can be processed and analyzed. This challenge is being addressed by software solutions [3] such as NoSQL which is schema-less [3].

Veracity

Veracity refers to the amount of data inaccuracies being present. This principle deals with challenges of incompleteness, reliability, uncertainty and quality of the data. Veracity should in a good scenario be *consistent* enough for software solutions to automate the processing of inaccurate data. [3] Data may inevitably be prone to errors in formatting, spelling or syntax as it comes from a wide variety of sources [9, p. 334].

In relation to veracity, data can either have a high degree of veracity if it contains a large amount of noise (data that cannot be turned into useful information) in comparison to useful data or a low amount of veracity of data if it comes in a controlled manner. [7, p. 31]

A different approach on the problem of veracity is that data can change its form occasionally. Therefore, this type of event must be handled every time the data changes its form, otherwise the data is of no use and it wastes resources. The veracity issue also involves other issues such as availability of resources, authenticity, trustworthiness, origin, accountability and reputation. [2, p. 84]

It is worth observing that the Veracity attribute has been added by IBM on top of the first Big Data notation by Douglas Laney [10] consisting of Volume, Velocity and Variety. [11, p. 6]

Case Study: Netflix

Netflix is a company specializing in streaming of digital media and video-on-demand such as TV shows and movies. It was founded in 1997 [12] by Reed Hastings and Marc Randolph. [13, p. 167] According to statistics site Statista.com Netflix sports a number of 117 million subscribers as of Q4 2017 [14]. Additionally, a study from 2013 states that Netflix accounts for 32.25% of fixed-network downstream traffic from 9 PM to 12 AM. [15] Its' proprietary recommendation engine (previously called) Cinematch is a set of algorithms which suggests customers movies they might like watching [13, p. 172].

Volume

Netflix collects very large amounts of data pertaining to the customers' movie viewing preferences, as well as other information. Data is captured in two types, active and passive. Active data capture involves data acquired directly from the customers, such as [13, pp. 170–171]:

- Name, address, credit card billing information
- Preferences in movies, movie ratings and movie recommendations
- [16, p. 18] Various tags for movies, effectively establishing many new media micro-genres

On the other hand, passive data involves:

- Customer searches, family viewing habits, day of week rental behaviour
- IP addresses

As streaming of content creates a large amount of operational data, Netflix further by collecting this data with the scope of a deeper understanding of the customer and his/her TV watching patterns. The operational data that is collected includes [13, p. 179]:

- Time of day movie was watched
- Whether the movie was watched towards the end
- Whether the movie was paused
- The device that was used for streaming
- [16, p. 20] Delays caused by buffering, bitrate and how often playback is stopped either by the user or due to network limitations

To solve this problem, Netflix moved from traditional relational database management system (RDBMS) and oracle databases to NoSQL and Cassandra [16, p. 21]

Velocity

According to statistics site Statista.com, the number of household subscribing to Netflix worldwide is expected to reach around 114 million by 2020, with the past number from 2016 being 81 million. [17] In other words, Netflix must deal with an increase of data coming from on average 8 million new subscribers each year, illustrating the Velocity principle of an ever-increasing rate of data being stored, processed, and analyzed.

As a solution to this issue Netflix uses Hadoop technologies whose scope are to allow distributed processing of Big Data across computer clusters [18, p. 55], in this case Hive [18, p. 358] for processing data. [16, p. 21]

Variety

Netflix's largest data issue is unstructured data: video and audio.

As has been mentioned previously, Netflix has given an outline to messy unstructured data such as video and audio by paying thousands of viewers to tag movies with themes, issues and motifs. As a result, Netflix has "nearly 80,000" micro-genres to choose from.

Furthermore, Netflix has begun automating the process by taking snapshots at particular user actions or set intervals and storing them in JPEG format, after which technologies such as facial recognition and colour analysis is used to determine the content of the movie [16, pp. 21–22].

Towards solving the issue of merging multiple datasets Netflix uses Pig [18, p. 438] for combining datasets together. [16, p. 21]

Veracity

In a blog post from 2015, Netflix states [19] (July 2015) that server administrators use a technique called cluster analysis for finding unhealthy servers. The clustering algorithm used for visualization of clusters is DBSCAN [20].

Next, in regards of authenticity of data that is sent to Netflix (e.g. movie ratings) Netflix states in its Terms of Use [21] that the account owner is solely responsible for updating and maintaining the accuracy of the information that is being sent to Netflix. Netflix may terminate or put an account on hold with the scope of avoiding fraudulent activity.

Finally, content providers/owners and others may inflate their digital ranking on Netflix through what is called a “shilling attack” [22], giving fake ratings to media to boost its visibility/ranking on Netflix. Unfortunately, no research has been found on how Netflix deals with these kinds of attacks.

It is possible that Netflix switched from its old 5-star ranking system to a thumbs-up/thumbs-down approach in order to minimize the damage caused by shilling attacks, which in regard to recommendations also takes into account combined ratings of a user's other Netflix users with similar taste in movies. [23]

Hadoop Technology: Cassandra

Introduction

Cassandra was originally developed at Facebook for the purpose of searching inboxes. [24, p. 4] Eventually the project was released as an open source project at on Google Code in July of 2008. The project's original authors are Avinash Lakshman and Prashant Malik [25, p. 6]

The abstract of the 2010 original paper by Avinash Lakshman and Prashant Malik [26] defines Cassandra as a distributed storage solution for working with large amounts of structured data which in turn is spread across multiple nodes which may or may not be spread across multiple data centers.

Additionally, the way in which it manages persistence makes services relying on it highly scalable and reliable. Although it shares design and implementation strategies like traditional RDBMSs, it does not support a full relational data-model, it provides clients instead with a data model whose data layout and format can be dynamically controlled.

Finally, it was designed to run on cheap commodity hardware and to have high write throughput with minimal sacrifice in read efficiency.

Below is a brief description of some of the main properties of the Cassandra database.

Data-Model

Cassandra is a wide-column data store and/or a non-relational database, usually called a NoSQL database. It is based on a flexible schema which consists of a relational database schema equivalent called a "keyspace" inside of which are column families. The keyspace acts as a logical namespace for holding column families together. [27, p. 24] [28, p. x]

Next, the column-family store is the equivalent of a multi-dimensional map. Cassandra stores multiple columns of data together as a row which in turn is associated with a row key. Column families do not need to store null columns and are not bound by a rigid schema or require the defining of a schema upfront. The main component of a column is usually called the primary key or the row key, and all data which shares a common row key is stored together. [29, p. 13]

In other words, what makes Cassandra different from traditional RDBMSs consists of the flexibility of the table structure: namely that different rows may have different columns. Alternatively, a row may now contain particular columns or any varied number of columns. Finally, a row may contain columns which were not defined in the column's specification. [28, p. x]

Peer-to-peer. Fault tolerance

The storage architecture of Cassandra is Peer-to-peer (P2P), meaning that each node in a cluster has the same role. In other words, Cassandra is a decentralized database. This architecture enables Cassandra to have features such as Replication and partitioning.

Replication for example represents a process whereby Cassandra maintains a predefined number of replicas of data on various data sites which in Cassandra are called Nodes. Partitioning on the other hand is a scheme which defines where data may be distributed across multiple nodes for achieving high data availability and high performance on data read/write requests. [30, p. 7]

Additionally, all peers in a Cassandra database are equal as there is no concept of master and slave instances. This leads to a simpler architecture. [24, p. 9]

Therefore, Cassandra allows one to replicate data across multiple nodes in contrast with traditional relational databases which require complex architectural setup for replicating data across multiple nodes for fault tolerance. In Cassandra each update can be made either synchronously or asynchronously. As a result, Cassandra enables a very high fault tolerance and/or availability. [24, p. 4]

Horizontal scalability [31, p. 36]

Horizontal scalability is defined as the ability to expand storage and processing capacity of a database by adding more servers and/or hardware to work for the database cluster. In traditional relational RDBMSs, once the storage/processing capacity has been exceeded or nearing its limits, the data set had to be sharded among multiple independent database instances which in turn have no knowledge of each other. In other words, the application is held responsible for knowing which instances contain particular pieces of data.

A Cassandra database on the other hand is from the beginning deployed as a cluster of interdependent instances. As a result, the cluster is a single entity from the client application's viewpoint. Cassandra doesn't need to know where each piece of data belongs to, whenever data is read or written on any instance in the cluster the respective node forwards the request to the cluster instance which truly contains the data.

As a result, more instances can be easily added to the cluster as Cassandra takes care of rebalancing the data so that each node has an equal share and database work continues smoothly. Furthermore, scaling down [27, p. 16] is also possible.

CQL

Cassandra uses the Cassandra Query Language (CQL) for accessing Cassandra databases. It has a syntax familiar to the traditional SQL for relational database and can be either used from the shell with `cqlsh` or client APIs. Many SQL statements are equally valid using CQL [28, p. 55]

Some of the main properties of the CQL syntax [28, p. 55] can be outlined as follows:

- Identifiers (unless enclosed in double brackets) and keywords are case-insensitive
- A CQL identifier is a letter followed by any sequences of letters, digits and the underscore character
- String literals are specified with a single quote. Quotation marks inside a CQL query must be delimited or escaped with an additional single quote (e.g. 'Dan''s House')

On the other hand, however, CQL lacks constructs for JOIN statements and some constructs such as retrieving update time of a column do not have an SQL equivalent. [31, p. 58]
Instead, Cassandra encourages denormalization of data by using collections [32, p. 318]

Cassandra Benefits

Notable benefits of using Cassandra for web applications are availability (against small-scale failures [33]), low latency and flexibility.

As data can be stored and replicated into nodes spread across multiple geographical areas, users spanning from one geographical region to another have lower latency when accessing data, in comparison to traditional relational databases in which data is stored only in one place, thus offering varying degrees of data access latency depending on the users' proximity to the data site.

On the other hand, the other benefit, high availability of data, is also due to the decentralized manner of storing data. If a small amount of the available clusters is non-functional due to errors or power outages the database will refer queries to functional nodes. Therefore, the database is resistant against small-scale attacks. [34, p. 66] [33] Research on the validity of Cassandra data availability further confirms this benefit. [35]

Finally, due to the flexibility of Cassandra's family-column data model, new types of attributes can be easily added as a Cassandra table is not constrained by its columns or lack thereof. Blogs or agile websites can benefit from this in comparison to relational databases which would require temporary unavailability whenever updating the database with new types of data. [36, p. 3]



References

- [1] T. Butts, "Welcome to the Zettabyte Era," *TV Technol.*, vol. 34, no. 6, p. 4, Jun. 2016.
- [2] S. Mazumder, R. S. Bhadoria, and G. C. Deka, *Distributed Computing in Big Data Analytics: Concepts, Technologies and Applications*. Springer, 2017.
- [3] V. C. Storey and I.-Y. Song, "Big data technologies and Management: What conceptual modeling can do," *Data Knowl. Eng.*, vol. 108, pp. 50–67, Mar. 2017.
- [4] N. Siddiqi, *Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards*. John Wiley & Sons, 2017.
- [5] J. J. Berman, *Principles of Big Data: Preparing, Sharing, and Analyzing Complex Information*. Newnes, 2013.
- [6] M. A. Resources Information, *Big Data: Concepts, Methodologies, Tools, and Applications: Concepts, Methodologies, Tools, and Applications*. IGI Global, 2016.
- [7] T. Erl, W. Khattak, and P. Buhler, *Big Data Fundamentals: Concepts, Drivers & Techniques*. Prentice Hall, 2016.
- [8] Z. Mahmood, *Data Science and Big Data Computing: Frameworks and Methodologies*. Springer, 2016.
- [9] S. Yu and S. Guo, *Big Data Concepts, Theories, and Applications*. Springer, 2016.
- [10] D. Laney, "3D data management: Controlling data volume, velocity and variety," *META Group Res. Note*, vol. 6, no. 70, 2001.
- [11] R. Buyya, R. N. Calheiros, and A. V. Dastjerdi, *Big Data: Principles and Paradigms*. Morgan Kaufmann, 2016.
- [12] "Business Search - Business Entities - Business Programs | California Secretary of State." [Online]. Available: <https://businesssearch.sos.ca.gov/CBS/SearchResults?SearchType=CORP&SearchCriteria=Netflix&SearchSubType=Keyword>. [Accessed: 10-Feb-2018].
- [13] R. Walker, *From Big Data to Big Profits: Success with Data and Analytics*. Oxford University Press, 2015.
- [14] * All products require an annual contract Prices do not include sales tax, "Netflix subscribers count 2017 | Statistic," *Statista*. [Online]. Available: <https://www.statista.com/statistics/250934/quarterly-number-of-netflix-streaming-subscribers-worldwide/>. [Accessed: 10-Feb-2018].
- [15] "Netflix Still King of the Downstream: Study | Multichannel." [Online]. Available: <http://www.multichannel.com/news/content/netflix-still-king-downstream-study/358600>. [Accessed: 10-Feb-2018].
- [16] B. Marr, *Big Data in Practice: How 45 Successful Companies Used Big Data Analytics to Deliver Extraordinary Results*. John Wiley & Sons, 2016.
- [17] * All products require an annual contract Prices do not include sales tax, "Number of Netflix households worldwide 2015 | Statistic," *Statista*. [Online]. Available: <https://www.statista.com/statistics/273885/quarterly-subscriber-numbers-of-netflix/>. [Accessed: 10-Feb-2018].
- [18] V. K. Jain, *Big Data and Hadoop*. Khanna Publishing, 2017.
- [19] N. T. Blog, "Tracking down the Villains," *Netflix TechBlog*, 14-Jul-2015. .
- [20] E. Schubert, J. Sander, M. Ester, H. Kriegel, and X. Xu, "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN," *ACM Trans. Database Syst. TODS*, vol. 42, no. 3, pp. 1–21, Aug. 2017.

- [21] "Netflix Terms of Use." [Online]. Available: <https://help.netflix.com/legal/termsfuse>. [Accessed: 13-Feb-2018].
- [22] I. Gunes, C. Kaleli, A. Bilge, and H. Polat, "Shilling attacks against recommender systems: a comprehensive survey," *Artif. Intell. Rev.*, vol. 42, no. 4, pp. 767–799, 2014.
- [23] "Netflix Ratings & Recommendations," *Help Center*. [Online]. Available: <https://help.netflix.com/en/node/9898>. [Accessed: 13-Feb-2018].
- [24] S. R. Alapati, *Expert Apache Cassandra Administration*. Apress, 2017.
- [25] R. Bradberry and E. Lubow, *Practical Cassandra: A Developer's Approach*. Addison-Wesley, 2013.
- [26] A. Lakshman and P. Malik, "Cassandra: A Decentralized Structured Storage System," *SIGOPS Oper Syst Rev*, vol. 44, no. 2, pp. 35–40, Apr. 2010.
- [27] J. Carpenter and E. Hewitt, *Cassandra: The Definitive Guide: Distributed Data at Web Scale*. O'Reilly Media, Inc., 2016.
- [28] D. Vohra, *NoSQL Web Development with Apache Cassandra*. Cengage Learning, 2015.
- [29] C. Y. Kan, *Cassandra Data Modeling and Analysis*. Packt Publishing Ltd, 2014.
- [30] V. Mishra, *Beginning Apache Cassandra Development*. Apress, 2014.
- [31] M. Brown, *Learning Apache Cassandra*. Packt Publishing Ltd, 2015.
- [32] K. Chandrasekaran, *Essentials of Cloud Computing*. CRC Press, 2014.
- [33] F. Yang, A. A. Chien, and H. S. Gunawi, "Resilient Cloud in Dynamic Resource Environments," in *Proceedings of the 2017 Symposium on Cloud Computing*, New York, NY, USA, 2017, pp. 627–627.
- [34] G. Lucio, *Pervasive Cloud Computing Technologies: Future Outlooks and Interdisciplinary Perspectives: Future Outlooks and Interdisciplinary Perspectives*. IGI Global, 2013.
- [35] C. Pérez-Miguel, A. Mendiburu, and J. Miguel-Alonso, "Modeling the availability of Cassandra," *J. Parallel Distrib. Comput.*, vol. 86, pp. 29–44, 2015.
- [36] A. B. M. Moniruzzaman and S. A. Hossain, "NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison," *ArXiv13070191 Cs*, Jun. 2013.

Mircea-Radu-Serban Dobrau Emerging Web Technologies Report Sem 2

GRADEMARK REPORT

FINAL GRADE

85/100

GENERAL COMMENTS

Instructor

Big data application. (20) 20

Discussion of Hadoop technology (20) 19

Quality of website (Typescript) (20) 18

Single page application (20) 16

Dynamic content using AJAX (15) 12

Software testing (5) 0

Very good analysis of the four Vs of big data, and the provided case study was very good.

Typescript used with the Marionette framework.
REST webservice provided with database handling.
Marionette handling was a bit repetitive.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14



Comment 1

Very nice consise analysis of Cassandra.

PAGE 15

PAGE 16
