Modern Data Engineering

Shreyas Patil
VU Amsterdam
2759591
s.v.patil@student.vu.nl

Abstract

Data generated from different sources needs to be available for analysis as data science algorithms have evolved and gained much attention in modern times. However, data engineering aspect which is necessary to enable the analytics is easily overlooked and with many technologies available it is even more ambiguous for implementers and architects to decide on how to go about it. A qualitative analysis and view about the technologies and software designs that are prevalent in data engineering community is necessary. Since marketing can influence popularity and decision making on usage of technology the aim is also to provide clarity on the different terms such as data lakes, datamesh and lakehouse and review literature to identify the technical aspects involved in these concepts. Lakehouse represents the most modern design, incorporating technical aspects that make it suitable for bringing disparate data from an organization and providing support for a wide variety of analytical use cases, ranging from traditional BI and reporting to advanced machine learning.

1 Introduction

Data generation in different industries has increased and is producing data at exponential rate and in petabytescale as in seen in Figure 1. CERN, the European Organization for Nuclear Research, generated approximately 40 zettabytes of data using the Large Hadron Collider during their run in 2018, surpassing the storage capacity of Amazon cloud storage services, as reported in (19). Having the right knowledge about the different data management tools, strategy and concepts is an essential prerequisite in determining their suitability for a project's purpose. With the concept of Data lakes gaining popularity these days, the term seems to have ambiguous interpretations when it comes to its implementation feasibility as explained by (14) .The objective is to investigate technically how feasible is it to implement the concept of data lakes and is it always a suitable replacement for traditional data warehouse implementation that has been existing since 1990s.

RQ1 : How are different data management strategies evolving to suit variety of use-cases ?
RQ2 : Data lakes, DataMesh and Lakehouses are widely used terms in modern data management, to what extent are they technical or are they just marketing term?

With **RQ1** the objective is to first provide a background on why and how the technologies used in data engineering have developed, with **RQ2** the goal is to clarify the common misunderstandings created by marketing and offerings of different technologies that claim themselves to belong to those categories. It is organised in a manner of increasing popularity of concepts and implementations overall in data engineering across industries over the last 2 decades. section 2 describes the methodology behind selection of literature of study, section 3 describes the process of ETL(extract, transform and load) process, the term has synonymously been used for the general concept of data processing for analytical purpose both in academia and industry. In section 4 on data lakes and lakehouses describes

the modern way of managing and processing both structured and unstructured data, section 5 discusses the concept of data handling in a decentralized manner, contrary to the way of managing data at a single location as in data lakes and datawarehouses. section 6 summarizes the overall progress and consolidates the progress of all the concepts and designs along with their comparative relationship to each other. Finally, a general discussion on data engineering is presented in the conclusion on research questions is presented in section 7 and possible shortcomings of survey in section 8.

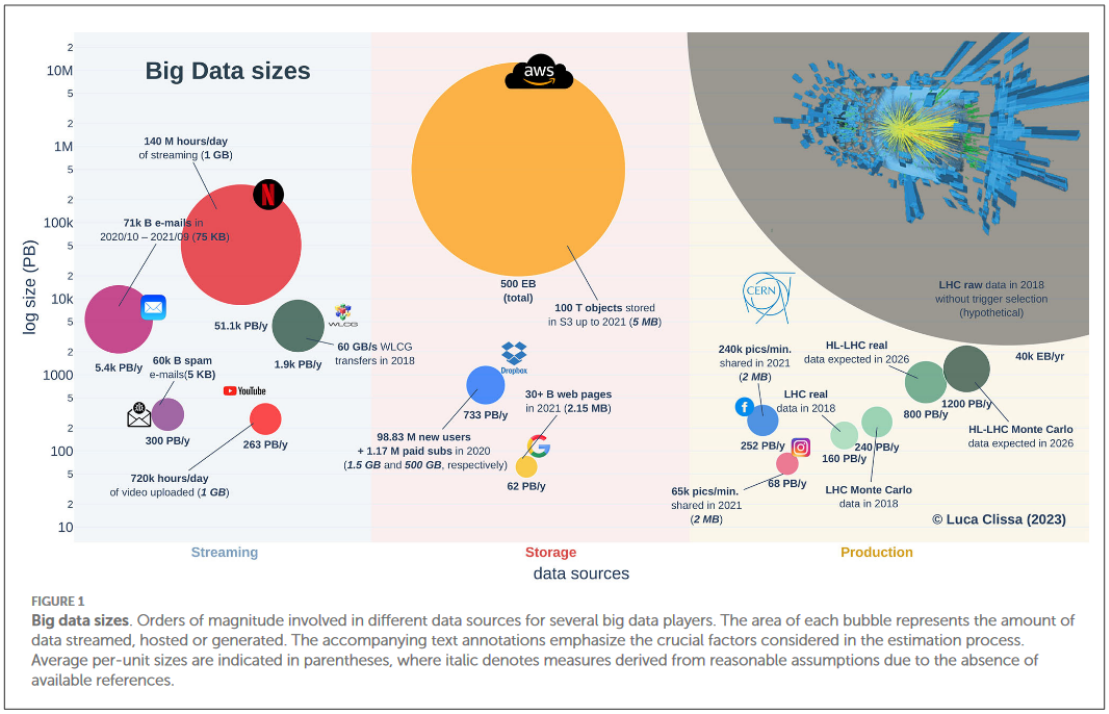

FIGURE 1
Big data sizes. Orders of magnitude involved in different data sources for several big data players. The area of each bubble represents the amount of data streamed, hosted or generated. The accompanying text annotations emphasize the crucial factors considered in the estimation process. Average per-unit sizes are indicated in parentheses, where italic denotes measures derived from reasonable assumptions due to the absence of available references.

Figure 1: Data sizes in modern times on Petabyte scale (19)

2 Study Design

Given the enormous volume of data produced in the present day, coupled with the data that has been accumulating since the 1990s, it is crucial to explore the historical methods of data management to gain a better understanding of their evolution. In this study, we commence with a review of foundational techniques, such as traditional data warehouses (1), and trace their progression towards the contemporary concepts of data lakes and lakehouses (16).

For the current survey, the selection of literature was conducted using the Google Scholar search engine. Keywords including data lakes, data warehouses, ETL, datamesh, and lakehouse were employed in the literature search. A filter based on the publication date was applied to retrieve only literature published after 2015, excluding earlier dates. While previous surveys have captured knowledge about existing data engineering strategies, this survey primarily focuses on staying up-to-date and thoroughly reviewing modern trends in data engineering.

3 Extract, Transform and Load (ETL)

The term ETL refers to the process of extracting, transforming, and loading data from source systems to target systems that collectively form the data warehouse. Typically, a data warehouse, used for analytical purposes and also known as OLAP (Online Analytical Processing), consists of data stored in fact and dimension tables. The fact table contains quantitative data, while the dimensions provide descriptive data that contextualizes the information in the fact table. Together, they form a star schema, with the fact table at the center and dimension tables pointed to by it (1).

ETL tools have been in existence since the 1990s and were traditionally well-suited for batch data processing, where data was loaded in batches during offline periods, often at night when business operations were halted. Popular technologies during this period included Informatica, Ab Initio, SSIS (SQL Server Integration Services), and Oracle Data Integrator (ODI), developed by various vendors. These tools relied on vendor-specific implementations for their features (1). Some of these tools supported parallel processing and real-time ETL through a change data capture mechanism (2). This mechanism allowed only the changes in the transactional database to be propagated to the data warehouse, eliminating the need for a full data load and enabling real-time updates in the warehouse.

However, with the growing adoption of cloud-based solutions for data warehousing, traditional approaches appear to be becoming obsolete. While the underlying concepts and principles remain the same, there has been a shift in the way data is handled. The mechanism of ELT (Extract Load Transform) is becoming increasingly prevalent, signifying that data is loaded first and processed later (1). As discussed in section 4, data lakes are gaining popularity and are built on the ELT principle.

3.1 Lambda and Kappa architecture

Given the imperative need for real-time analytics, having a data pipeline capable of delivering data as soon as it is generated has become essential. To address this requirement, lambda and kappa architectures have gained popularity (3). The fundamental objective of these architectures is to integrate real-time and batch data processing into a unified workflow. Organizations aim for real-time analytics while retaining the ability to perform batch analytics.

The lambda architecture, depicted in Figure 2, comprises a batch processing layer and a transient real-time processing layer (3). As of 2012, Twitter implemented the lambda architecture using Hadoop MapReduce for batch processing (20), and Storm for real-time processing (21). Storm provides the capability to express the processing of a stream of data as graphs, where vertices represent computations and edges depict data flow. While lambda architecture combines both modes of data processing, it introduces complexity in implementation. Essentially, it requires replicating processing logic for both batch and real-time pipelines. In many cases, this complexity raises the risk of different teams implementing the logic differently. Moreover, there is a potential for the semantics of computations to yield different results in real-time and batch analyses. For instance, temporary data loss during real-time processing may lead to different analyses, and if batch analysis is performed after data recovery, it will produce accurate results.

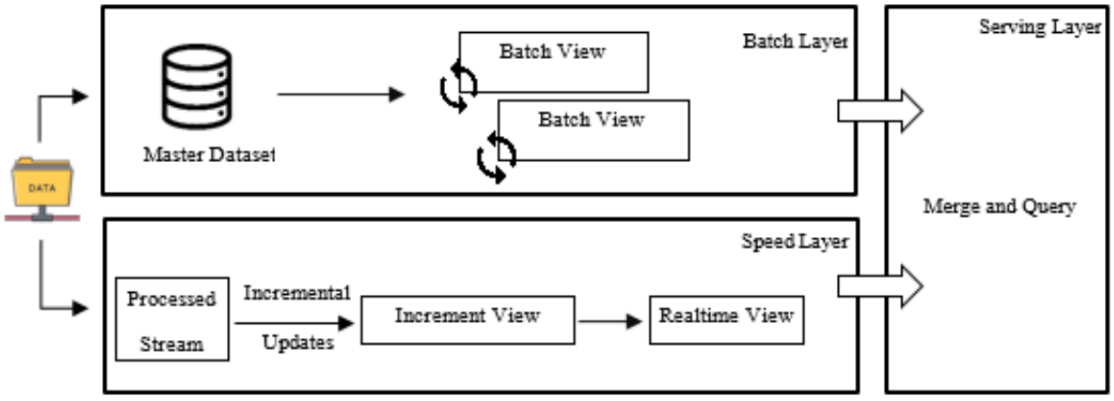

Figure 2: Typical Lambda Architecture (4)

Kappa architectures, illustrated in Figure 3, offer a more effective solution to the challenges mentioned earlier and are gaining widespread use. The key principle is treating everything as a stream of data, with a stream processing engine capable of handling all data, including batch data considered as a bound stream. Popular technologies employed in this architecture include Kafka Streams (22) and Apache Beam (23), enabling data to be processed in this manner. Spark (24) has also implemented capabilities for stream processing.

The architecture involves the concept of event time (the time at which data is produced) and processing time (the time at which data reaches the system for processing). Watermark boundaries represent the moving time windows within which all produced data is assumed to have reached the system. For example, a watermark of 5 minutes implies that the system considers data within the last 5 minutes for computing analytics, providing a solution to the negative impact of out-of-order or late-arriving data in real-time analysis (10).

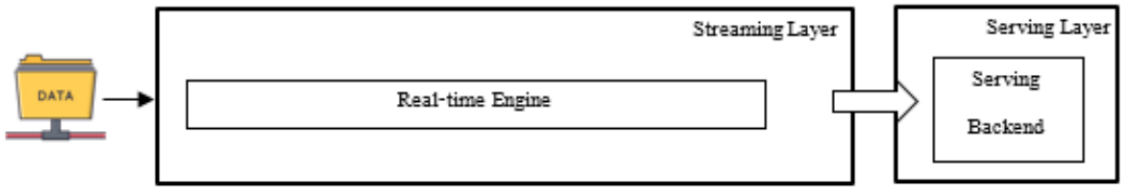

Figure 3: Typical Kappa Architecture (4)

3.2 Lambda vs Kappa in different experiments

Although the Kappa architecture appears to be a more versatile solution applicable across various scenarios, it is important not to dismiss the relevance of the Lambda architecture entirely. The choice between them is not necessarily about the inherent speed or efficiency of a stream processing engine compared to a batch processing engine on batch tasks. Instead, the Kappa architecture reduces the complexity associated with maintaining two versions of the same pipeline, as is the case in Lambda (3).

In a study conducted by (4) to compare the performance and usage of the two architectures, MapReduce was used in the batch layer, and Spark Streaming was employed in the real-time layer. The study found that for a word count task on a dataset of size 300 MB in a 2-worker and 1-master node setup, the processing time of the Lambda architecture was 2.2 times more than that of Kappa. However, there was no significant difference in memory usage. This suggests that the Kappa architecture is better suited for tasks prioritizing speed over accuracy, while Lambda architectures, despite requiring a higher setup and maintenance cost, offer more accurate analysis (4).

In another benchmarking study conducted by (5) on social network data using Microsoft Azure cloud, the Yahoo Flickr Creative Commons 100 Million (YFCC100M) dataset, a Flickr multimedia collection dataset, was employed to address an influence analysis problem. The study compared the total processing time, cost, and scalability of deployment for both Lambda and Kappa architectures.

The Lambda architecture was implemented using Apache Spark, which supports in-memory processing and leverages a read-only immutable collection of data items known as Resilient Distributed Dataset (RDD). Apache Spark supports both streaming and batch processing (11). For Kappa architecture, Apache Storm (21) was utilized, a distributed platform for processing raw streams of data. Both architectures were implemented with 2 executor nodes. In terms of performance based on processing time, the Lambda architecture performed on average 12-18 percent better. However, this result could be attributed to the faster in-memory computation provided by Spark. For a fair comparison of architectures, it might have been more appropriate to utilize Spark Streaming (11) in the Kappa architecture.Regarding scalability, increasing the number of executors from 2 to 4 reduced run times by 18 percent in both architectures. However, Kappa architecture was observed to scale better when implemented with resource-heavy nodes, or in other words, when vertical scaling was applied.

4 Data lakes

A data lake represents the modern approach to managing large volumes of data extracted from multiple sources, capable of accommodating data in various formats. It serves as the primary storage location in modern data engineering pipelines, accommodating structured, unstructured, and semi-structured data, in contrast to data warehouses that primarily handle structured data (9).

The primary objective of data lakes is to address the issue of data residing in different systems in isolated silos and being processed independently. Instead, they aim to centralize all data in a single location, regardless of its format, allowing analysts to access and analyze the entirety of the organization's produced data (8).

Data lakes adhere to the principle of schema on read, which means that data is stored without imposing any schema restrictions. The schema is inferred at the time of querying and usage. This is in contrast to data warehouses, which operate on the principle of schema on write, where data is checked against a schema before being written (8).

The comprehensive definition of what constitutes a data lake has been presented by (6), providing a framework that delineates nine aspects crucial for the design and implementation of a data lake. This framework offers a step-by-step process for designing a novel data lake architecture and assessing existing data lakes to identify their shortcomings. The decisions made in their design are segregated into aspects according to the framework.

Figure 4 illustrates the nine aspects, where aspects A-F are conceptual, each associated with an implementation decision, and each aspect serving as an abstraction above the one below it. Aspects G-I are applicable to the others collectively. The decisions made for the design aspect of the data lake influence each other, and the interactions between them are depicted in Figure 5.

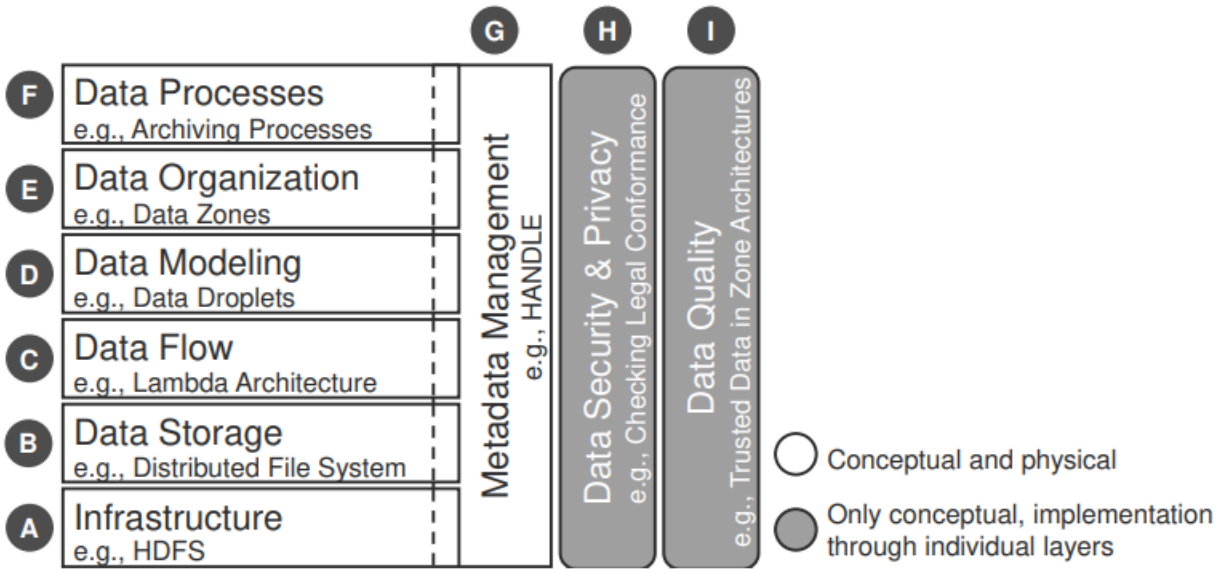

Figure 4: Essential Aspects of data lake as per Data lake Architecture framework (6)

The implementation of data lakes is thoroughly surveyed by (13), introducing a zone reference model. The paper explores existing models of data lake implementation and highlights the inconsistency and lack of a systematic approach in literature for implementing these models.

The paper delves into the design of a so-called zone architecture, which represents the stages through which data progresses. Each zone contains data with varying granularity, schema, and intended use case. The identified zones are Landing, Raw, Harmonized, Distilled, and Delivery zones (15). In the Landing and Raw zones, data is stored without modifications, including secure storage of personal data. The Harmonized and Distilled zones contain cleaned data tailored for specific use cases, while the Delivery zone is equivalent to data marts in traditional data warehousing solutions. The paper emphasizes that adhering to this data management and flow structure ensures the preservation of data history and is practical for usability, allowing different users of the data lake to view their desired versions of data. It also enables security measures across different zones by controlling access based on user groups (13).

The incorporation of data marts as part of the architecture positions the data lake as an extension of traditional data warehousing architecture. The paper suggests that different zones can be implemented using big data technologies, such as Kafka for landing zones and HDFS for the storage of other zones.

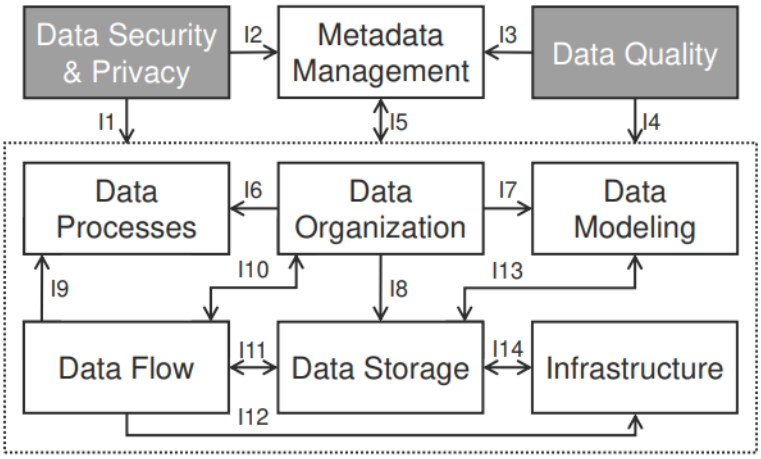

Figure 5: Interaction between the aspects of the Datalake where the source of arrow is the influencer and the destination is the influenced (6)

4.1 Challenges of Data lakes

Despite the goal of data lakes to centralize all data in a single location, several challenges are associated with handling them, as described by (7). Firstly, the extraction and discovery of datasets from the entire data lake require a well-set-up and maintained metadata management system (15). Secondly, since there are no restrictions on what is written to data lakes, there is a risk of incorrect or duplicate data, along with uncertainty about the semantics of the data and the possibility of corruption, especially without information about the data source. Data lakes are at a high risk of lacking governance, performance, and access control if not managed effectively. The lack of guarantees about performance and semantic interpretation stems from the primary focus of data lakes on storage, with less effort spent on enhancing usability for analytics (7). However, if the aspects of data lakes, as elaborated by (4), are well-utilized during the design and implementation phases, they can provide significant analytical value.

Most literature on data management and warehousing presents the concept of data lakes ambiguously, with no set technology or framework for their development. There is limited literature that combines conceptual knowledge of data lakes with the enabling technology. (13) comprehensively covers the technical aspects along with the conceptual aspects mentioned previously. It also notes that with advancements in the use cases of data for machine learning, which often rely on data being input in the form of embeddings (real number vector representations of data), challenges arise in designing data lakes for such use cases.

4.2 Lakehouse

In modern data management strategies, data lakes are often viewed and used primarily for storage. However, this approach introduces the overhead of maintaining a separate database or warehouse for crucial characteristics such as transaction management, indexing, caching, and metadata management to enable advanced analytics. This has led to a demand for data lakes to exhibit characteristics akin to data warehouses (13). The question arises: How can cloud storage be designed and leveraged for low-latency transactions and ACID guarantees? The answer to this query has given rise to the concept of the "lakehouse," aiming to fulfill the requirements of both worlds (16). However, due to the novelty of the lakehouse concept, there is considerable misconception about the technologies or architectures that constitute a lakehouse. Various vendors have marketed their tools as lakehouses without a clear understanding of the term.

A concrete and technical definition of a lakehouse is presented by (18). A lakehouse is essentially a fusion of a data warehouse and data lake, requiring careful consideration of tradeoffs when incorporating features from each. (17) specifies a set of requirements for what constitutes a lakehouse and evaluates popular technologies marketed as lakehouses, including Apache Iceberg (25), Delta Lake (26), Apache Hudi (27), Snowflake (28), Dremio (29), and Trino (30). The findings indicate that only Apache Iceberg, Delta Lake, and Apache Hudi, particularly in conjunction with the processing framework of Apache Spark, can be considered true lakehouses. In contrast, Snowflake, Dremio, and Trino offer some features of a lakehouse but need to be used in combination with other technologies to meet all the requirements.

Delta Lake (26) is a database management system layer that provides ACID guarantees and is built on top of cloud object storage, such as AWS S3 and Azure Blob. As organizations increasingly store all their data in the cloud due to its scalability for storage and independent computing, there emerged a challenge in performing efficient operations directly on the data stored in cloud storage, particularly update and delete operations. These operations are not straightforward and time-efficient, as modifying the entire object can slow down reads until all writes to the object are completed.

Delta Lake addresses these challenges by implementing the capability of updates and rollbacks on data objects in cloud stores using efficient log structures. These structures not only facilitate time travel in data but also support schema evolution. In the event of changes in table schema, old data files can still be read without rewriting them, as the history of schema updates is saved in transaction logs. Delta Lake provides a solution for use cases where multiple data warehouses had to be maintained for different purposes like data warehousing, business intelligence, and machine learning, offering a more convenient and unified approach with simple Delta Lake tables (18). A pictorial architecture of open-source data lakehouse Apache Iceberg (25) can be seen in Figure 6 which highlights how the operations on the actual data are managed via a hierarchy of metadata on those files and thus abstracting the details of storage and its management.

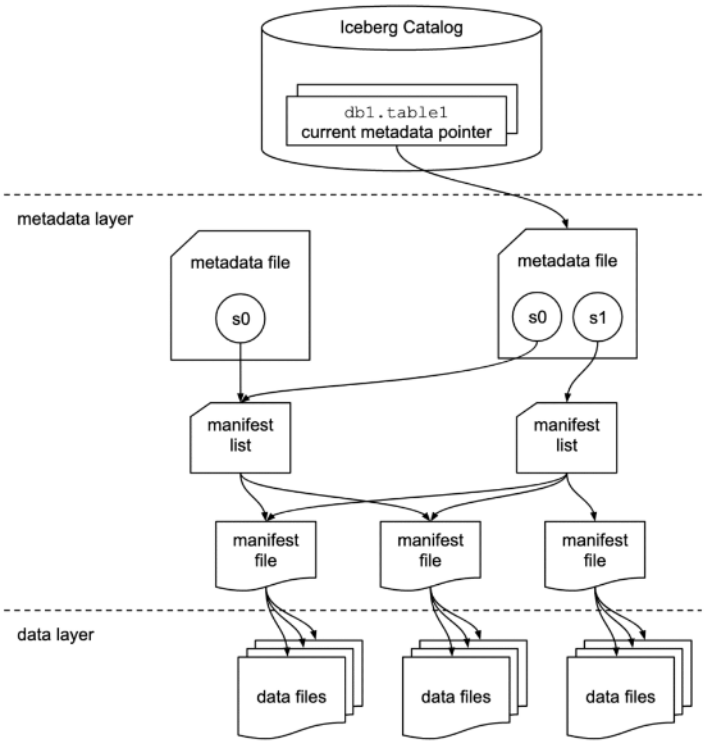

Figure 6: Open source Lakehouse - Apache Iceberg Architecture (25)

A overview of how the data platforms have evolved over time to lakehouse architecture can be seen in Figure 8. In Figure 7 the option 4 on integrated architecture is the lakehouse paradigm, option 2 requires maintaining multiple copies of data to cater to the needs of different usecases and option 3 is at risk of data in data lake being out of sync or different from the data warehouse version. Lakehouse

in Figure 8 also reveals the capability of lakehouses to handle structured as well as unstructured data while the older architectures of data warehouses were primarily built to handle structured data.

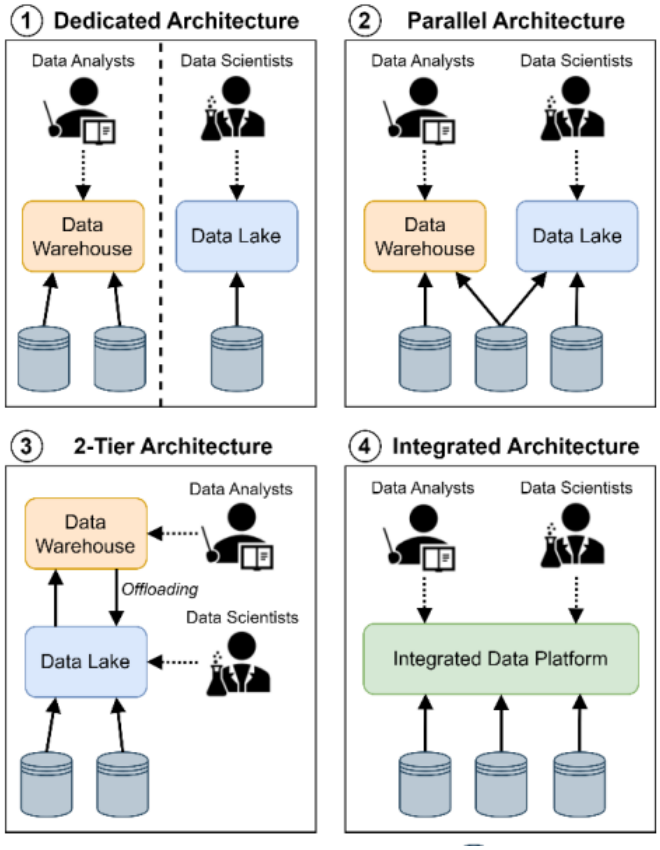

Figure 7: Variations of combining the data warehouse and data lake paradigms (17)

5 Data Mesh

The adoption of data lakes represents a relatively new approach to managing data, driven by the challenges associated with trying to manage data from various sources in a monolithic manner. Traditional approaches often result in a lack of ownership, as no single team can effectively take full responsibility for all the data in an organization. The burden of managing such data can be overwhelming.

One of the challenges is the difficulty of discovering quality datasets within the vast ocean of data. Organizing and making sense of this data is a significant challenge. Additionally, different business units may have different semantics about their data, making it challenging to link data items across various units.

The concept of Data Mesh introduces a decentralized approach to managing data. In this model, each business domain is individually responsible for managing and curating datasets for usage by others, establishing clear ownership. Data is treated as a direct product rather than a byproduct of business operations. A dedicated data product owner is assigned to manage the data product's lifecycle. This approach shifts the responsibility of processing data for analytical purposes closer to the source of its creation, as depicted in Figure 9. Individual business domains are accountable for the quality and security of the data (12).

This shift from a centralized to a decentralized way of handling data has been adopted by companies such as Zalando and Netflix (12).

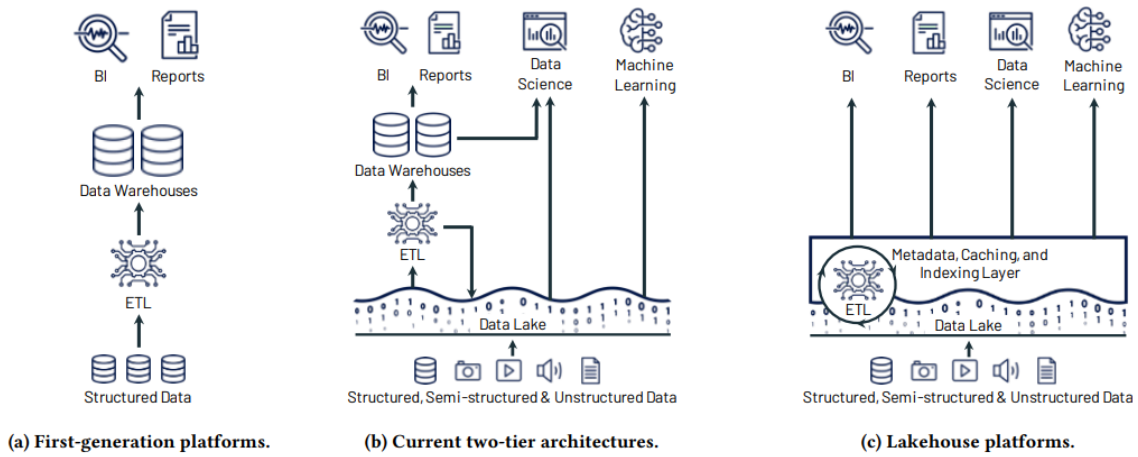

Figure 8: Evolution of data platforms towards lakehouse paradigm, to enable development of single platform which will enable analytics on structured and unstructured data of entire organisation (16)

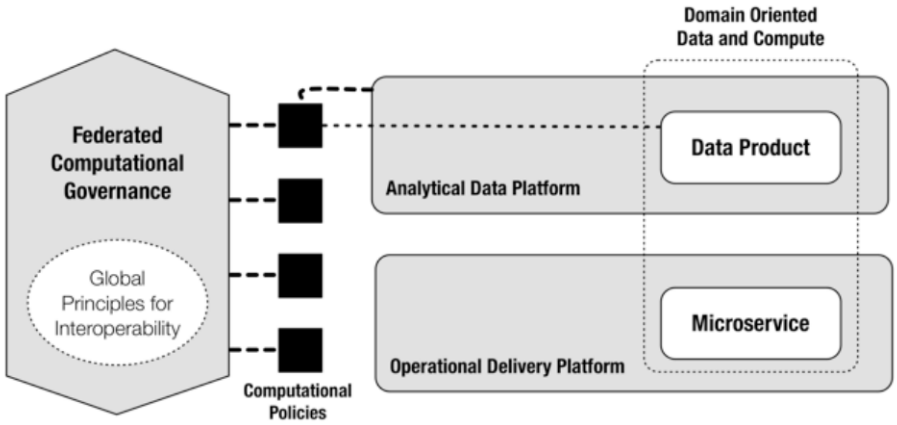

Figure 9: Conceptual implementation of Data mesh , with approach towards moving the analytical data processing and ownership close to its source service (12)

6 Discussion

Over the last 2 decades there have been rapidly changing use cases of data in all industries that produce data, for not just descriptive but also predictive analytics. Modern usecases influence and depend on the way data engineering is done prior to analysis. Since 90s analytical data preparation was done primarily on structured data of businesses and ETL was the term used for this activity and it was primarily batch based processed done on weekly or overnight basis. However, with data being generated in real time also created a requirement to analyse it in real time which made the concepts of Lambda and Kappa architectures popular, while Lambda is about processing the batch and real time data independently and later merging, Kappa tends to view all data in terms of streams. Lambda from the study has been found to provide accurate results compared to Kappa but has additional overhead of maintaining 2 separate codebases and environments. With rapid advancements in artificial intelligence requiring large amount of data, has created the requirement of unstructured data of organisations to being used, this has given popularity to the ideas of data lakes which is capable of storing all data as objects in cloud.

While data lakes are primarily developed and viewed from storage perspective, and it being separate from data warehouses that have been built previously to handle historical data of organisations, made it necessary to have a single platform to manage all the structured and

unstructured data along with necessary database management properties which has give rise to the concept and implementation of lakehouses. However, since metadata is handled by different businesses differently as each has it's own semantics about data, the concept of data mesh has been proposed which encourages the processing of data at the source of its creation and the domain or business responsible for creating it should also be responsible for building its own analytical product which would be interoperable with data from other business. Data mesh is more of a way of handling data and there is no technology that actually is an implementation of the concept whereas lakehouse and data lakes are more concrete implementations technically and can be used off the shelf in industry.

A comparison of datawarehouse , data lakes and lakehouses is as shown in Figure 10, with green cells representing a pro and red a con.

	Data warehouse	**Data lake**	**Data lakehouse**
Type of Data	Only Structured	Structured / Unstructured	Structured / Unstructured
Performance	High performance	Achieving high performance is challenge	High Performance
File and Table format	Proprietary and vendor lock in	Open formats like Parquet supported and no vendor lock in	Open formats like Parquet supported and no vendor lock in
Support for ACID	Supported	No Support	Supported
Configurations required	Very Minimal as everything is provided by vendor	Lot of configurations required to assemble data platform that is performant as warehouse	Relatively less as the "table format" component enables high performance on data lake
Support for advanced ML loads	Machine learning not supported	ML workloads are supported	ML workloads are supported
Scalability	Very expensive to scale as storage and compute provided by vendor hence tightly coupled	Scalable as storage and compute is decoupled, with usage of cloud services	Scalable as storage and compute is decoupled, with usage of cloud services

Figure 10: Comparison of Data warehouses, Data lakes and lakehouses

7 Conclusion

As far as research question **RQ1** is concerned data lakes and lakehouses have technical aspect to them and are implementable products while data mesh still being a business strategy and way of handling data without much emphasis on what technology would enable it. Lakehouses in particular seems to be the most in demand product, yet the very few technologies actually support variety of features while others need to be used in composition with other technologies to be considered as a lakehouse implementation.A more detailed overview of the progress of technology for data engineering has been discussed in section 6.
With respect to research question **RQ2** it can be observed that there have been a proliferation of technologies and the way of marketing them using certain terms has created ambiguity regarding what the technology or product is capable . This often misleads organisations in decision making for their usecase. Nevertheless the trend in data management strategy has been increasingly been affected by adoption of cloud services for storage, compute and security. From traditional ETLs which were designed for large batches on premise, in modern times the support of data management for machine learning and advanced analytics has promoted the usage of data lakes and lakehouse architectures. While Kappa and lambda architectures have been conceptual but yet they signify the need of organisations to simultaneously support streaming as well as batch analytical workloads.

8 Limitations of Survey

Firstly, an exhaustive analysis of all available technologies would provide a more clear picture of what are the other technologies or products that could possibly be better or provide novel feature and designs. Secondly, within the limited scope of the survey the current study could be biased towards technologies for instance deltalakes, that are popular which indirectly hints at it being affected by how marketing them has made them popular. However, as much as possible effort is towards focus on the technical concepts related to the terminologies rather than the offering of a product from a specific vendor

References

[1] Mukherjee, R. and Kar, P., 2017, January. A comparative review of data warehousing ETL tools with new trends and industry insight. In 2017 IEEE 7th International Advance Computing Conference (IACC) (pp. 943-948). IEEE.

[2] Muddasir, N.M. and Raghuveer, K., 2017, September. Study of methods to achieve near real time ETL. In 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC) (pp. 436-441). IEEE.

[3] Lin, J., 2017. The lambda and the kappa. IEEE Internet Computing, 21(05), pp.60-66.

[4] Sanla, A. and Numnonda, T., 2019, July. A comparative performance of real-time big data analytic architectures. In 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC) (pp. 1-5). IEEE.

[5] Persico, V., Pescapé, A., Picariello, A. and Sperlí, G., 2018. Benchmarking big data architectures for social networks data processing using public cloud platforms. Future Generation Computer Systems, 89, pp.98-109.

[6] Giebler, C., Gröger, C., Hoos, E., Eichler, R., Schwarz, H. and Mitschang, B., 2021, September. The data lake architecture framework: a foundation for building a comprehensive data lake architecture. In Conference for Database Systems for Business, Technology and Web (BTW) (Vol. 70469).

[7] Nargesian, F., Zhu, E., Miller, R.J., Pu, K.Q. and Arocena, P.C., 2019. Data lake management: challenges and opportunities. Proceedings of the VLDB Endowment, 12(12), pp.1986-1989.

[8] Khine, P.P. and Wang, Z.S., 2018. Data lake: a new ideology in big data era. In ITM web of conferences (Vol. 17, p. 03025). EDP Sciences.

[9] Mehmood, H., Gilman, E., Cortes, M., Kostakos, P., Byrne, A., Valta, K., Tekes, S. and Riekki, J., 2019, April. Implementing big data lake for heterogeneous data sources. In 2019 ieee 35th international conference on data engineering workshops (icdew) (pp. 37-44). IEEE.

[10] Feick, M., Kleer, N. and Kohn, M., 2018. Fundamentals of real-time data processing architectures lambda and kappa. SKILL 2018-Studierendenkonferenz Informatik.

[11] Zaharia, M., Xin, R.S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M.J. and Ghodsi, A., 2016. Apache spark: a unified engine for big data processing. Communications of the ACM, 59(11), pp.56-65.

[12] Machado, I.A., Costa, C. and Santos, M.Y., 2022. Data mesh: concepts and principles of a paradigm shift in data architectures. Procedia Computer Science, 196, pp.263-271.

[13] Giebler, C., Gröger, C., Hoos, E., Schwarz, H. and Mitschang, B., 2020, October. A zone reference model for enterprise-grade data lake management. In 2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC) (pp. 57-66). IEEE.

[14] Hai, Rihan, et al. "Data Lakes: A Survey of Functions and Systems." IEEE Transactions on Knowledge and Data Engineering (2023).

[15] Sawadogo, P. and Darmont, J., 2021. On data lake architectures and metadata management. Journal of Intelligent Information Systems, 56, pp.97-120.

[16] Armbrust, M., Ghodsi, A., Xin, R. and Zaharia, M., 2021, January. Lakehouse: a new generation of open platforms that unify data warehousing and advanced analytics. In Proceedings of CIDR (Vol. 8).

[17] Schneider, J., Schwarz, H. and Mitschang, B., 2023. Assessing the Lakehouse: Analysis, Requirements and Definition.

[18] Armbrust, M., Das, T., Sun, L., Yavuz, B., Zhu, S., Murthy, M., Torres, J., van Hovell, H., Ionescu, A., Łuszczak, A. and Świtakowski, M., 2020. Delta lake: high-performance ACID table storage over cloud object stores. Proceedings of the VLDB Endowment, 13(12), pp.3411-3424.

[19] Clissa, L., Lassnig, M. and Rinaldi, L., 2023. How big is Big Data? A comprehensive survey of data production, storage, and streaming in science and industry. Frontiers in big Data, 6.

[20] https://hadoop.apache.org/docs/r1.2.1/index.htmlMapReduce

[21] https://storm.apache.org/

[22] https://kafka.apache.org/

[23] https://beam.apache.org/

[24] https://spark.apache.org/

[25] https://iceberg.apache.org/

[26] https://docs.databricks.com/en/delta/index.html

[27] https://hudi.apache.org/

[28] https://www.snowflake.com/en/

[29] https://www.dremio.com/

[30] https://trino.io/