Application of Parallel and Distributed Systems
Institute for Parallel and Distributed Systems
University of Stuttgart

Architectures and Implementations of Data Lakehouses: Case Studies from Industrial Practice

Jan Schneider, Christoph Gröger, Arnold Lutsch,
Holger Schwarz and Bernhard Mitschang

In: Proceedings of the 25th International Conference on Business Information Systems (BIS 2025), Poznań, Poland, Lecture Notes in Business Information Processing (LNBIP), Volume X, pp. [X], Springer Nature, 2025.

DOI: https://doi.org/X

BIBTEX:

```
@inproceedings{Schneider2025,
    author={Jan Schneider and Christoph Gr{\"{o}}ger and Arnold Lutsch and Holger Schwarz and Bernhard
    Mitschang},
    title={Architectures and Implementations of Data Lakehouses: Case Studies from Industrial Practice},
    booktitle={Proceedings of the 25th International Conference on Business Information Systems (BIS 2025),
    Poznań, Poland},
    series= {Lecture Notes in Business Information Processing},
    volume={X},
    pages={XX-XX},
    publisher={Springer Nature},
    year={2025},
    doi={https://doi.org/X}
}
```

Architectures and Implementations of Data Lakehouses: Case Studies from Industrial Practice

Jan Schneider[1], Christoph Gröger[2], Arnold Lutsch[2], Holger Schwarz[1] and Bernhard Mitschang[1]

[1] University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany
{firstname}.{lastname}@ipvs.uni-stuttgart.de
[2] Robert Bosch GmbH, Borsigstraße 4, 70469 Stuttgart, Germany
{firstname}.{lastname}@de.bosch.com

Abstract. Data analytics and AI constitute key drivers for business innovation and are hence indispensable for ensuring the competitiveness of enterprises in the digital age. Yet, they can only be leveraged when sufficient data management is carried out, which requires the design and development of suitable data platforms. In recent years, so-called data lakehouses found their way into practice, which promise to combine the benefits of data warehouses and data lakes. However, only little is known about the data and technology architectures that are utilized for these data platforms in practice, as well as the experiences that enterprises have made with them. To address this gap, we conducted four case studies on large-scale, real-world data lakehouse implementations from the industrial sector that are used for various kinds of analytics and AI applications. This paper presents our within-case and cross-case results, which provide insights on common architectural decisions, practical experiences and observed challenges. They outline directions for future research and can support enterprises in the design of suitable data platforms.

Keywords: Data Lakehouse, Data Platforms, AI Applications, Case Studies, Industry Experience.

1 Introduction

Data analytics and Artificial Intelligence (AI) constitute key ingredients for the digital transformation of enterprises, as they allow to derive insights from data and thus enable data-driven business decisions. Recent advances in the field of AI are particularly promising for industrial enterprises, as they generate large amounts of heterogeneous data across the industrial value chain that provide huge potentials for business process optimization and business model innovation. Examples include predictive maintenance [6], AI-based optical inspection on the factory shop floor and data-driven engineering [28]. In order to be able to leverage these analytics- and AI-driven business opportunities, data management is crucial. In industrial practice, data management accounts for up to 80% of the implementation efforts for AI use cases, with data platforms constituting

the core IT components for it [10]. *Data warehouses* [12] represent the most conventional type of data platform and are primarily utilized for reporting and Online Analytical Processing (OLAP) [3]. However, their rigid data models and limitations in the support of advanced analytics [5] led to the increased utilization of *data lakes* [11], which store all incoming data in its raw format on scalable and directly accessible storage systems. In general, data warehouses and data lakes possess rather diverging properties and support different kinds of analytics [20].

Since around 2020, so-called *data lakehouses* [2] have found their way into practice. They pursue to combine benefits of data warehouses and data lakes and are typically implemented with open-source frameworks such as Delta Lake, Apache Hudi and Apache Iceberg [1, 14, 21]. As illustrated in Fig. 1, these frameworks control the read and write access of processing engines for data that resides on highly-scalable storage systems. By managing technical metadata such as log files, they provide additional features for data processing, e.g. relational semantics, ACID guarantees and an increased performance for batch and stream processing [1, 14]. Since these frameworks allow to store data similarly to data lakes, but still provide typical capabilities of data warehouses, the resulting data platforms can cover analytical workloads from both worlds.

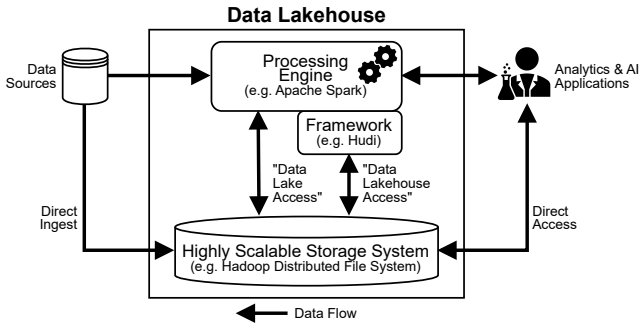

Fig. 1: High-level view on the components of a typical data lakehouse.

However, little is known about data and technology architectures of data lakehouse implementations in industrial practice, e.g. in terms of data modeling and technologies. Also, there is a lack of insights on practical experiences and indications for future research. This leads to the following research questions:

RQ1: What architectures, i.e. data and technology architectures, are used for data lakehouse implementations in practice?
RQ2: What practical experiences have enterprises made during the development and operation of data lakehouse implementations?
RQ3: What directions for future research result from the practical experiences?

To address these questions, we studied four real-world cases at a globally operating manufacturing group, in which data lakehouse implementations have been developed, tested and operated for between one and three years in different

productive settings with huge data volumes and different analytical use cases, including AI applications. For each case, we examined its context, systematically analyzed the data and technology architecture of the corresponding implementation and collected practical experiences from the responsible analytics teams. This paper presents our within-case and cross-case results of the case studies. It builds on our previous preliminary work [22], in which we briefly described one exemplary implementation of a data lakehouse in practice. The paper at hands takes this as a starting point to make the following contributions: It a) *reviews related work* from literature (cf. Section 2), b) *introduces the method* that we applied, as well as the four cases and their contexts (cf. Section 3), c) systematically *analyzes and compares the data and technology architectures* of the investigated implementations (cf. Section 4.1), d) derives a *generalized data lakehouse architecture* from the four cases and highlights *interesting findings* (cf. Section 4.2), e) discusses the *practical experiences* gathered from the analytics teams (cf. Section 5) and f) points to *future research directions* (cf. Section 6).

2 Related Work

In literature, several implementations of data lakehouses are presented, spanning different domains like healthcare (cf. [29, 24]), cyber security (cf. [27, 1]) and telecommunication (cf. [26]). However, these works are not suitable for answering the research questions RQ1-RQ3 due to several reasons:
Lack of cross-case comparisons: To the best of our knowledge, there are no works available yet that compare multiple independent data lakehouse implementations. Instead, the existing works focus only on implementations from individual settings, which limits the generalizability of the insights (cf. [26, 27]).
Heterogeneous architecture assessments: The architecture assessments in literature differ strongly in their purpose, scope, granularity and form of representation: Some of them are detailed and cover far-reaching properties like data processes and data formats (cf. [24, 4]), while others stay high-level and only outline superficial aspects, such as the involved technologies (cf. [27, 15]). This heterogeneity impedes a structured analysis and comparison of the architectures due to the lack of an universally applicable architecture framework.
Unclear level of maturity: Several works do not explicitly state whether the presented data lakehouses constitute conceptual designs, prototypes or data platforms in productive use (cf. [18, 27]), While for some it can be assumed that they are used in production (cf. [4, 30]), it remains unclear for how long these data platforms have been operated and hence how stable the insights are.
Missing practical experiences: Prolonged experiences and challenges that occurred during the development and operation of a data lakehouse implementation are barely discussed in literature. Consequently, no cross-case comparisons are conducted either, preventing the derivation of future research directions.

To address these shortcomings in our work, we a) designed the four case studies with the goal of performing cross-case comparisons in mind (cf. Section 3.1), b) selected and applied an architecture framework as guidance for analyzing and

comparing the architectures of the data lakehouse implementations in a systematic manner (cf. Section 4), c) only investigated data platforms that have been in productive use for between one and three years and d) explicitly gathered practical experiences and challenges in the course of interviews (cf. Section 5).

3 Method and Case Overview

This section first explains the method and scope that we applied in the course of our case studies. Afterwards, the investigated cases are introduced.

3.1 Method and Scope of Study

We decided to conduct case studies [7], as this method is suitable for answering questions of "what" and "how" [19]. Furthermore, case studies are recognized as suitable method for complex topics where context needs to be taken into consideration [7]. These properties apply to the domain of our research questions, as architectures of data platforms are subject to a wide range of organizational and technical requirements, while the practical experiences are likely to depend on contextual factors such as data volumes and the types of analytical use cases.

In the scope of this research, we studied four real-world cases at a globally operating manufacturing group with a world-wide network of factories and suppliers. In each case, one analytics team was in charge of designing, developing and operating a data platform that corresponds to a data lakehouse as outlined in Section 1 and has been in productive use for between one and three years. The four analytics teams belong to completely different business units with different employees, processes and IT system landscapes and thus developed their solutions independently of each other for their requirements, without centrally defined standards. Consequently, our approach can be considered a multiple case study [19] that assesses several cases in different settings and hence may lead to more generalizable results in comparison to a single case study.

For each case, we conducted interviews (45-60 minutes) with members of the responsible analytics team, who described themselves as either data engineers, data architects or solution architects and already had prior experience in the development of data warehouses and data lakes. In total, six interviews were conducted, which followed a semi-structured approach with questions prepared in advance[3]. The interviewers also responded dynamically to answers of the participants with suitable follow-up questions. To enable a systematic comparison of the data and technology architectures across the cases (cf. RQ1), we utilized the *Data Lake Architecture Framework* by Giebler et al. [8] as reference for the interview questions and guidance for the within- and cross-case analyses. The framework is defined on a generalized level, facilitating its application to various kinds of data-lake-like data platforms, including data lakehouses.

[3] For details details on the interview structure and prepared questions, please see: https://gist.github.com/schneijan/41cb39804a7eaa5dac84d14432390b90

Table 1: Overview of business and IT aspects of the four investigated cases.

		C1: Production Analytics	C2: MES Analytics	C3: Product Lifecycle Analytics	C4: Telemetry Analytics
Business Asp.	**Product Lifecycle Phases**	Production Execution	Production Execution	All	Operations and Service
	Users	Data Analysts, Data Scientists	Data Analysts, Data Scientists	Data Analysts, Data Scientists	Data Analysts, Data Scientists, Service Engin.
IT Aspects	**Source Systems**	MES, Image Repository	MES	IoT Devices, CRM, ERP, Social Media	IoT Devices
	Data Types	Structured, Unstructured	Structured	Structured	Structured
	Data Volume	Terabytes	Petabytes	Terabytes	Terabytes
	Analytical Workloads	Reporting, OLAP, DM/ML, NRT Reporting	Reporting, OLAP, DM/ML, Str. Analytics	Reporting, DM/ML	Reporting, OLAP, DM/ML, Exploration
	Types of Analytics	Descriptive, Diagnostic, Predictive			

Engin.: Engineers; MES: Manufacturing Execution System; CRM: Customer Relationship Management System; ERP: Enterprise Resource Planning System; DM/ML: Data Mining/Machine Learning; NRT: Near-realtime; Str.: Streaming

3.2 Overview of the Cases

In the following, we provide an overview of the investigated cases at the manufacturing group. For reasons of confidentiality, we generalize certain aspects. Table 1 contains business and IT aspects of the four cases C1-C4. The first business aspect refers to the phases of the industrial product lifecycle [25], for which the data lakehouse is utilized, while the second one lists the roles of the involved users. The IT aspects cover details about the source systems, the types and volumes of the managed data, as well as the envisaged analytical workloads.

Production Analytics (C1): In this case, different electronic components for vehicles are produced along a production line. The involved machines and sensors generate data about the quality of produced components and the condition of the machines. The goal of the analytics team was the construction of a data platform that is capable of collecting, managing and preparing this manufacturing data for various analyses. This also includes near-time reporting, since some analysis results are supposed to be displayed on dashboards along the shop floor. Furthermore, machine learning for the detection of faulty components needs to be carried out. In our previous work [22], we described this case in detail.

MES Analytics (C2): This case deals with the mass production of engine parts. On the shop floor, data from machines and sensors is generated and collected in a Manufacturing Execution System (MES) and then ingested into a data platform. Its goal is to enable the processing of the generated data for

flexible ad-hoc analytics, including the generation of Key Performance Indicators (KPIs) and visualizations, data mining and machine learning activities for optimizing the manufacturing processes, as well as streaming analytics.
Product Lifecycle Analytics (C3): In the business unit of this case, different types of appliances for consumers are developed, produced and distributed. Across all phases of the industrial product lifecycle, primarily structured data about the different business processes is collected, including data about products, suppliers and resellers. This data is stored in operational IT systems, such as Customer Relationship Management (CRM) and Enterprise Resource Planning (ERP) systems. Additionally, field data from IoT-enabled appliances is collected and KPIs about social media activities are gathered. All this data needs to be ingested into a data platform to be processed and analyzed. The overall goal is to reduce the development time for new analytics applications (reporting, data mining and machine learning) by creating a single source of truth that provides quick and easy access to all the available data.
Telemetry Analytics (C4): The data platform implementation of this case supports the analyses of data from network-enabled IoT devices that are used by customers. The goal is to enable self-service analytics for reporting, OLAP, data mining and machine learning workloads on structured data. Parts of the data also represent technical logs of the IoT devices, which should be made available to service engineers who need it for identifying problems at the customer site.

4 Architectures of Data Lakehouses in Practice

This section addresses RQ1 by analyzing the data and technology architectures of the four data lakehouse implementations within and across cases. For this purpose, the Data Lake Architecture Framework by Giebler et al. [8] is utilized. This framework breaks down the architecture of a data platform into several aspects, of which the aspects *Data Organization* (A1), *Data Modeling* (A2) and *Data Flow* (A3) belong to the data architecture, while the aspects *Data Storage* (A4) and *Infrastructure* (A5) describe the technology architecture.

In the following, Section 4.1 assesses and compares the four data lakehouse implementations regarding these aspects. Subsequently, Section 4.2 derives a generalized data lakehouse architecture from the investigated implementations that combines their characteristics and points to interesting findings.

4.1 Analysis of the Data and Technology Architectures

Table 2 characterizes the investigated data lakehouse implementations in terms of the aspects A1-A5 from the architecture framework. The following subsections describe the individual aspects in detail and compare them across the cases.
Data Organization (A1): All data lakehouse implementations leverage the concept of data zones [23] for organizing data of different granularity, quality and application-specificity. However, the numbers and names of the zones differ between the cases. The implementations of C1, C3 and C4 all possess a "Raw"

Table 2: Data & technology architectures of the data lakehouse implementations.

	Aspects			C1: Production Analytics	C2: MES Analytics	C3: Product Lifecycle Analytics	C4: Telemetry Analytics
Data Architecture	**A1**	**Data Organization (Data Zones)**		Raw, Harmonization, Delivery, ML	Bronze, Silver, Gold	Raw, Curated, Aggregated, Mach. Learning	Raw, Metadata, Harmonization, Delivery, Tech
	A2	**Data Modeling**		Use-case-drv., Denormalized	Use-case-drv., Normalized	Use-case-drv., Multi-dim.	Use-case-drv., Normalized
	A3	**Data Flow**	Ingestion	Stream, Batch Ingest	Stream	Stream, Batch Ingest	Stream
			Processing	Stream, Batch	Stream, Batch	Batch	Stream
			Consumption	Batch Queries, Stream Sink, DM/ML Tools	Batch Queries, Stream Sink, DM/ML Tools	Batch Queries, DM/ML Tools	Batch Queries, Manual Access
Tech. Arch.	**A4**	**Data Storage**		Object Store			
	A5	**Infrastr.**	Deployment	Azure Cloud	Azure Cloud	Azure Cloud	AWS Cloud
			Storage	ADLS2	ADLS2	ADLS2	S3
			Processing	Databricks	Databricks	Databricks	Amazon EMR
			Framework	Delta Lake	Delta Lake	Delta Lake	Apache Hudi

Use-case-drv.: Use-case-driven; DM/ML: Data Mining/Machine Learning

zone. In addition, most of them have a zone that manages consolidated datasets ("Harmonization" in C1 and C4, "Curated" in C3). For reporting and OLAP, C1 and C4 provide a "Delivery" zone, while their "Machine Learning" zones are dedicated to data mining and machine learning activities. Despite its name, the "Metadata" zone of C4 is not related to metadata management, but rather corresponds to another "Delivery" zone containing master data about IoT devices. The zones "Aggregated" of C3, which contains pre-processed use-case-specific data, and "Tech" of C4, which holds log files for the service engineers, have no equivalent zones in the other cases. C2 constitutes an exception with regards to this architectural aspect, as its implementation uses the more generic medallion model [16], in which raw data is stored in the *Bronze* zone, processed and cleansed data in the *Silver* zone and use-case-specific data in the *Gold* zone.

Data Modeling (A2): In all four implementations, a schema-on-read approach with use-case-driven data modeling is carried out, which provides a high degree of flexibility. According to the interview participants, they have no central guidelines that would enforce the usage of certain modeling techniques, such as Data Vault [17]. However, in the cases C2 and C4, the data is already normalized, as it is extracted from the source systems in this shape. In addition, multi-dimensional data models [13] are found in C3. Interestingly, the data of C1 is intentionally de-normalized, which introduces additional redundancy, but increases the query performance, as expensive join operations can be avoided. According to the analytics team, this approach is economically reasonable, as computing power in the cloud is often more expensive than costs for additional storage space.

Data Flow (A3): For data ingestion, an event hub like Apache Kafka is utilized in all four cases, which temporarily stores and buffers stream data from

the source systems before it is moved to the data lakehouse. In parallel, C1 and C2 also employ batch processing for data ingestion, such as for image data that needs to be prepared for machine learning. The raw data that is stored on the data platforms is subsequently cleansed and prepared for the different types of analytics applications by using either batch or stream processing. Depending on the types of analytical workloads, the stored data is consumed in different ways: For reporting and OLAP, query engines are utilized to query the available data with SQL statements on behalf of reporting and visualization tools. Data that must be analyzed in near-realtime (cf. C1 and C2) is consumed by appropriate processing engines, such as Apache Spark or Apache Flink, and then made available in dedicated sinks for dashboards or similar applications. For data mining and machine learning, various data science tools like the MLlib module of Apache Spark can be used to work on the stored data. In C4, the service engineers can also access and explore log data directly via the underlying object store, for example in order to search for potential causes of device failures.

Data Storage and Infrastructure (A4, A5): All four investigated implementations are entirely deployed on public clouds. The implementations of C1-C3 leverage similar technology stacks, including Microsoft Azure as cloud provider, Azure Data Lake Storage Gen2 as storage system and the Databricks runtime as processing engine in combination with the Delta Lake framework. Only C4 uses a different stack, which runs on Amazon Web Services and comprises the Amazon Simple Storage Service (S3) as storage system, together with Apache Spark on Amazon EMR and the framework Apache Hudi. Consequently, all four implementations employ highly-scalable object stores for persisting the data.

4.2 Generalized Data Lakehouse Architecture and Findings

To provide a more concise answer to RQ1, we consolidated the architectures of the investigated data lakehouse implementations into a generalized architecture that is illustrated in Fig. 2. It combines the architectural characteristics of the four implementations and was derived by iteratively abstracting and merging their architectures. Hence, the generalized architecture can be seen as a template, of which each of the four data lakehouse implementations is an instance.

Two types of source data can be distinguished in the generalized architecture: *Structured Data* corresponds to data where the structure is already known in advance, such as measurement data from machines that complies to a pre-defined schema. In contrast, the structure of *Unstructured Data* is not known in advance, which includes image data (cf. C1). Semi-structured data does not occur, as all data in the four cases can be classified as either structured or unstructured data. While the unstructured data is only extracted as batches (cf. C1), the structured data can either be ingested as batches or as unbounded data streams. In the latter case, an event hub like Apache Kafka serves as a buffer.

The data lakehouse itself is based on a *cloud object store* and can be divided into three parts, where each part consists of one or multiple data zones. The first part embodies the *Raw Zones*, which hold raw and only slightly cleansed data. In the architectures of the four examined implementations, we observed three

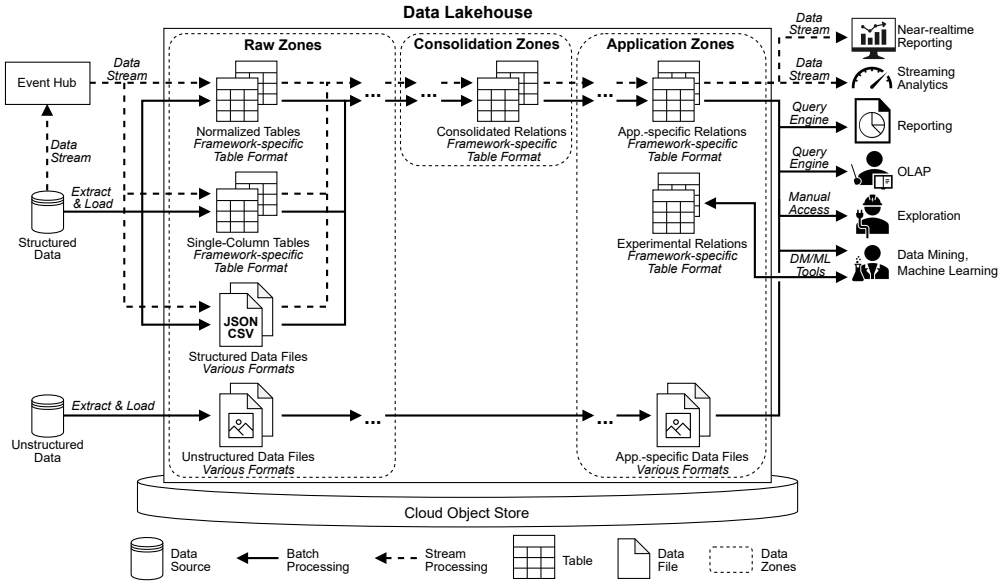

Fig. 2: Generalized data lakehouse architecture as derived from the four cases.

different approaches of how the structured raw data is stored: With *Normalized Tables*, the records of the new data are normalized during the ingestion and added to a table with columns that mostly match the structure of the data. In contrast, *Single-Column Tables* provide only one column, e.g. of type "text", in which the entire data record is stored. Finally, in the *Structured Data Files* approach, the new data is stored as files of its raw format, e.g. as CSV or JSON files. If the data is managed in tables, this is done in the framework-specific table format that is utilized, such as Delta or Hudi tables.

In contrast to the structured data, the unstructured data is always stored in its raw format. Depending on the storage approach that is chosen for the raw data, different processing steps follow, which are either carried out via batch or stream processing. In any case, the structured data eventually arrives in the *Consolidation Zones* part of the data platform, which includes zones for the management of harmonized and integrated data that provide a standardized, holistic view (cf. [9]). In these zones, the data is solely stored as tables of the framework-specific format. In the subsequent processing steps, the data is prepared for specific analytics applications. The resulting application-specific data then resides in *Application Zones*, where it can be accessed from users or processes for various analytical purposes. In addition, data scientists can derive new experimental data and store it as new tables in these zones. In the examined data lakehouse implementations, unstructured data is solely processed as batches and only rudimentary prepared before it is made available in an application zone.

In summary, several architectural observations can be made to answer RQ1:
Data zones for data lakehouses: The concept of data zones [11], which has originally been developed for data lakes, is applied in all four cases, which indicates that it has also proven useful in the context of data lakehouses.
Several storage approaches for structured raw data: We identified three different approaches for the storage of structured raw data, which is either managed as framework-specific tables or plain data files.
Combination of batch and stream processing: In industrial practice, both batch and stream processing seem to play an important role for the preparation of data. While data lakehouses support both modes in a flexible manner, still additional components for stream processing are required, such as an event hub.
Non-uniform data architectures: The four investigated data lakehouse implementations are all deployed in the cloud and leverage similar technology architectures, consisting of an object store, batch and stream processing engines, an event hub and a framework. However, in terms of the data architectures, they diverge noticeably. Based on our observations during the case studies, it can be concluded that apparently no best practices have yet emerged that would recommend modeling patterns for various data types and zones.

5 Practical Experiences

This section addresses research question RQ2 by presenting our cross-case analysis results regarding practical experiences with the data lakehouse approach.

In general, all interview participants had a positive opinion on the data lakehouse concept. Furthermore, they were finally satisfied with their implementation in terms of functional demands, performance, data freshness, interoperability and costs. They also indicated that they would use the same approach again for their present setting, as they saw benefits in comparison to their previously used implementations. Nevertheless, they also described several technical challenges that arose during the development and operation of the data lakehouse implementations, which are summarized in Table 3 and described below.

Table 3: Technical challenges that were encountered during the development and operation of the investigated data lakehouse implementations.

Encountered Challenges		**C1: Production Analytics**	**C2: MES Analytics**	**C3: Product Lifecycle Analytics**	**C4: Telemetry Analytics**
CH1	Configuration & Optimization	✗	✗	✗	✗
CH2	Processing Latency	✗	✗	✗	
CH3	Data Management	✗	✗		✗
CH4	Technology Selection		✗		✗

Configuration & Optimization (CH1): In the examined implementations, the frameworks Delta Lake and Apache Hudi constitute key components (cf. Sec-

tion 4). Both frameworks are highly configurable, providing hundreds of configuration parameters that allow to control different aspects, such as the reading and writing behavior, partitioning strategies and background processes. For example, Fig. 3 visualizes a series of measurements that we collected from the implementation of case C2 over a period of 30 days. This metric describes the total volume of storage that has been occupied on the underlying cloud object storage at the end of each day. Accordingly, the storage addressed around 2.69 petabytes of data storage on average, with a negative trend that is caused by public holidays on which the production was reduced. A pattern can be discovered, according to which the data volume continuously increases over the course of each week and then abruptly decreases each Saturday. While the increases can be attributed to new data arriving at the data platform, the decreases can be explained by optimization procedures that are carried out in the background. In particular, they likely reflect the impact of the `VACUUM` command[4] from the Delta Lake framework, which deletes data files that are no longer referenced and have exceeded a retention time, shrinking the data volume. A suitable configuration of this command is crucial to optimize the data platform in terms of performance and costs and has an influence on its operational behavior.

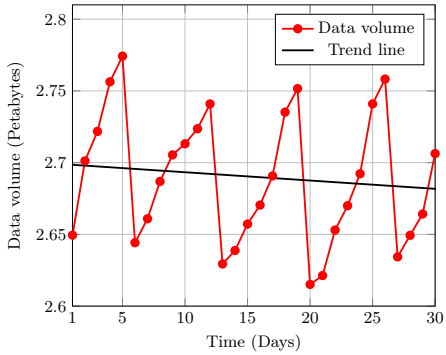

Fig. 3: Total volume of occupied storage space on the cloud object store of case C2 at the end of each day over a period of 30 days.

However, interview participants from all cases reported that finding a suitable configuration constituted a major challenge, as the impact, side-effects and interactions of the parameters were sometimes hard to predict. As a result, the developers were forced to test various configurations through empirical testing in a trial-and-error manner, which was time-consuming and required difficult trade-offs to be made, e.g. with respect to the read and write performance of tables. In addition, they stated that the optimization strongly differed from the tuning of traditional databases and required more technical knowledge.

[4] https://docs.databricks.com/en/delta/vacuum.html

Processing Latency (CH2): In three cases, latency issues were reported, which affected either SQL queries for reporting and OLAP workloads or stream processing jobs. The participants from C1 and C2 both stated that due to high latencies, data processing was only possible to a limited extent in near-realtime and suspected that the micro-batching approach of Apache Spark may constitute a bottleneck. However, as there are no strict timing requirements, these limitations could be tolerated. In C1, C2 and C3, latency problems were also noticeable when querying the data; however, these problems could be overcome with tunings at the frameworks and adjustments at the utilized compute cluster.
Data Platform Management (CH3): During the interviews for the cases C1, C2 and C4, challenges related to the management of the data lakehouse implementations and the handling of the stored data were mentioned. This pertains various activities at an operational level, such as the discovery and exploration of relevant datasets in different zones, the development, maintenance and documentation of data pipelines, the description of data semantics, the monitoring of data quality, the enforcement of access control, as well as activities related to the collaboration between data producers and consumers. From the view of the interview participants, the data lakehouse approach and the associated technologies constitute primarily a technical solution, but do not offer much support at the operational level. Although external data catalogs can be used for this purpose, they are considered to not sufficiently integrate with the data platform.
Technology Selection (CH4): During the development, it was not clear to the analytics teams which processing engine, e.g. Apache Spark or Apache Flink, and which framework, e.g. Delta Lake or Apache Iceberg, were most suitable for the envisaged use cases and according to which criteria these technologies should be selected. As a result, in both C2 and C4, two frameworks were tried out in parallel, and finally the decision was made in favor of Delta Lake for C2 and Apache Hudi for C4. To cope with these uncertainties, a technology-independent approach was employed in case C4, in which incoming data is stored as structured data files (cf. Section 4.2) before it is transformed to a framework-specific table. This allows them to switch to another framework at a later point.

It can be observed that challenges arose both during development and the operation of the implementations. While they generated high satisfaction among the analytics teams, the missing support for data platform management, the selection of suitable technologies and the configuration remain as open challenges.

6 Conclusion and Future Research Directions

Based on our cross-case study of four real-world implementations, we conclude that data lakehouses have proven as data platforms in large-scale industrial settings with petabytes of data, broad user bases and various analytical workloads. Hence, they can be considered state of the art in industrial practice. We found that the architectures of real-world implementations commonly apply data zones, leverage different approaches for the management of raw data, interleave batch

and stream processing, use an event hub as buffer for streaming data, comprise similar technology stacks and are typically deployed to public clouds.

Based on our architectural findings and the collected practical experiences and challenges, we derive directions for future research (RQ3): To address the uncertainties regarding the data architecture (cf. Section 4.1), *design patterns for data architectures* are required. In addition, the selection of appropriate technologies (cf. challenge CH4), e.g. in terms of processing engines like Apache Spark and frameworks like Apache Hudi, raises the need for *decision guidance*, i.e. criteria that allow to evaluate technologies for different settings. On a similar note, a systematic *method for the configuration and optimization of data lakehouse implementations* (cf. challenges CH1 and CH2) in terms of the utilized frameworks is necessary to ease their adaption in practice. Finally, we currently see the greatest need for research in the area of *data platform management* (cf. challenge CH3). Here, many different activities for various user groups need to be supported and tightly integrated with the data platform to improve its usability and to close the gap between technological solutions and operational processes.

Acknowledgments. We sincerely thank Thomas Kohler for his valuable inputs and profound support during the conduction of the case studies.

References

1. Armbrust, M., Das, T., Sun, L., et al.: Delta Lake: High-performance ACID Table Storage over Cloud Object Stores. Proceedings of the VLDB Endowment **13**(12), 3411–3424 (2020)
2. Armbrust, M., Ghodsi, A., Xin, R., et al.: Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics. In: 11th Conference on Innovative Data Systems Research (CIDR), Online Proceedings (2021)
3. Baars, H., Kemper, H.G.: Business Intelligence & Analytics. Springer Fachmedien Wiesbaden, Wiesbaden (2021)
4. Begoli, E., Goethert, I., Knight, K.: A Lakehouse Architecture for the Management and Analysis of Heterogeneous Data for Biomedical Research and Mega-biobanks. In: 2021 IEEE International Conference on Big Data. pp. 4643–4651. IEEE (2021)
5. Bose, R.: Advanced Analytics: Opportunities and Challenges. Industrial Management & Data Systems **109**(2), 155–172 (2009)
6. Dogan, A., Birant, D.: Machine Learning and Data Mining in Manufacturing. Expert Systems with Applications **166**, 114060 (2021)
7. Dul, J., Hak, T.: Case Study Methodology in Business Research. Routledge, London and New York (2008)
8. Giebler, C., Gröger, C., Hoos, E., Eichler, R., Schwarz, H., Mitschang, B.: 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 (2021)
9. Giebler, C., Gröger, C., Hoos, E., et al.: 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 (2020)
10. Gröger, C.: There is no AI without data. Communications of the ACM **64**(11), 98–108 (2021)

11. Hai, R., Koutras, C., Quix, C., et al.: Data Lakes: A Survey of Functions and Systems. IEEE Transactions on Knowledge and Data Engineering **35**(12), 12571–12590 (2023)
12. Inmon, W.H.: Building the Data Warehouse. Wiley technology publishing Timely, practical, reliable, Wiley, Indianapolis, Ind., 4th edition edn. (2005)
13. Iqbal, M., Mustafa, G., Sarwar, N., et al.: A Review of Star Schema and Snowflakes Schema. In: Intelligent Technologies and Applications, Communications in Computer and Information Science, vol. 1198, pp. 129–140. Springer Singapore (2020)
14. Jain, P., Kraft, P., Power, C., et al.: Analyzing and Comparing Lakehouse Storage Systems. In: Proceedings of the 13th Annual Conference on Innovative Data Systems Research (2023)
15. Kumar, D., Li, S.: Separating Storage and Compute with the Databricks Lakehouse Platform. In: 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA). pp. 1–2. IEEE (2022)
16. L'Esteve, R.C.: The Cloud Leader's Handbook. Apress, Berkeley, CA (2023)
17. Linstedt, D., Olschimke, M.: Building a Scalable Data Warehouse with Data Vault 2.0. Morgan Kaufmann (2015)
18. Liu, G., Pang, Z., Zeng, J., et al.: IoT Lakehouse: A New Data Management Paradigm for AIoT. In: Big Data – BigData 2023, Lecture Notes in Computer Science, vol. 14203, pp. 34–47. Springer Nature Switzerland (2023)
19. Meredith, J.: Building Operations Management Theory through Case and Field Research. Journal of Operations Management **16**(4), 441–454 (1998)
20. Nambiar, A., Mundra, D.: An Overview of Data Warehouse and Data Lake in Modern Enterprise Data Management. Big Data and Cognitive Computing **6**(4), 132 (2022)
21. Schneider, J., Gröger, C., Lutsch, A., et al.: The Lakehouse: State of the Art on Concepts and Technologies. SN Computer Science **5**(5) (2024)
22. Schneider, J., Lutsch, A., Gröger, C., et al.: First Experiences on the Application of Lakehouses in Industrial Practice. In: Proceedings of the 35th GI-Workshop Grundlagen von Datenbanken. vol. 3710, pp. 3–8 (2024)
23. Sharma, B.: Architecting data lakes: Data management architectures for advanced business use cases. O'Reilly Media, Sebastopol, CA, second edition edn. (2018)
24. Siddiqi, M.H., Idris, M., Alruwaili, M.: FAIR Health Informatics: A Health Informatics Framework for Verifiable and Explainable Data Analysis. Healthcare (Basel, Switzerland) **11**(12) (2023)
25. Stark, J.: Product Lifecycle Management (Vol. 1): 21st Century Paradigm for Product Realisation. Decision Engineering, Springer International, 4th edn. (2020)
26. Tang, X., Chai, C., Zhao, D., et al.: Separation Is for Better Reunion: Data Lake Storage at Huawei. In: 2024 IEEE 40th International Conference on Data Engineering (ICDE). pp. 5142–5155. IEEE (2024)
27. Tovarnak, D., Racek, M., Velan, P.: Cloud Native Data Platform for Network Telemetry and Analytics. In: 2021 17th International Conference on Network and Service Management (CNSM). pp. 394–396. IEEE (2021)
28. Wang, J., Xu, C., Zhang, J., Zhong, R.: Big Data Analytics for Intelligent Manufacturing Systems: A Review. Journal of Manufacturing Systems **62**, 738–752 (2022)
29. Xiao, Q., Zheng, W., Mao, C., Hou, W., Lan, H., et al.: MHDML: Construction of a Medical Lakehouse for Multi-source Heterogeneous Data. In: Health Information Science, Lecture Notes in Computer Science, vol. 13705, pp. 127–135 (2022)
30. Zhang, Y., Peng, B., Du, Y., Su, J.: GeoLake: Bringing Geospatial Support to Lakehouses. IEEE Access **11**, 143037–143049 (2023)