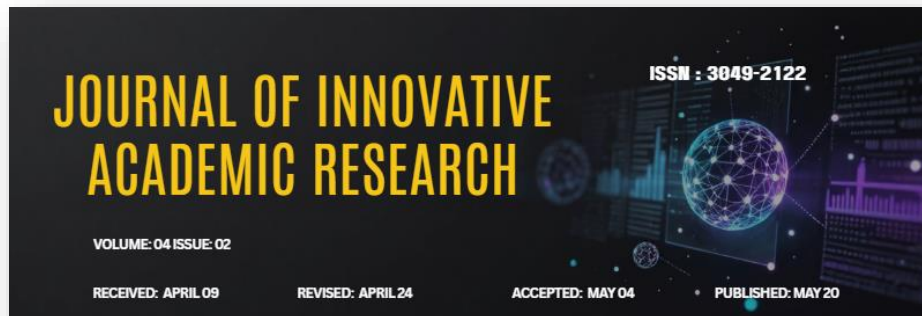




Lakehouse Intelligence for Enterprise Analytics

1st Srinivas Kalyan , 2nd Hiroshi Tanaka

Srinivas Kalyan Yellanki, Software Engineer 3 , yellankisrinivaskalyan@gmail.com
Hiroshi Tanaka , Research Assistant

Abstract. Organizations adopting Lakehouse architectures seek a single system for data ingestion, controlled and monitored data domains, and robust support for data consumers. At the same time, enterprises must mature their MLOps practices—moving models from data science experimentation to governed, production-grade analytics. This maturity depends on reproducibility, adherence to corporate governance policies, integrated scheduling and orchestration, and a centralized Feature Store.

This paper introduces the Governed Lakehouse Intelligence Framework, a conceptual model built from modular, interoperable building blocks that establishes standards across three dimensions: Lakehouse Integration, Governance, and Modelling. By adhering to these standards, enterprises can incrementally build scalable data pipelines and MLOps-ready analytics capabilities.

The Framework is evaluated through a multi-faceted approach. First, the Principal Evaluation Approach establishes the need for such a framework among organizations operating Lakehouse architectures. A panel of domain experts then assesses the Framework's contribution across four dimensions: scaled data domain integration, data governance compliance, enterprise MLOps readiness, and organizational agility and resiliency.

Keywords: Governed Lakehouse Intelligence, Lakehouse Architecture, Data Governance, MLOps, Scalable Data Pipelines, Feature Store, Data Domain Integration, Model Lifecycle Management, Enterprise Analytics, Data Engineering, Machine Learning Operations, Data Lakehouse.

1. Introduction

Investment in Big Data for Analytics has increased significantly for most large enterprises over the years, especially during the pandemic. The cost of storing data dropped, enterprise architecture teams recommended cloud storage, and DataScience teams built models based on the analysis of the huge data lake repositories. But the solutions rarely landed into production. Multiple model failures were observed at organizations during Bots Group (2021) neatly describe the many reasons for the failures in the initial attempts of MLOps adoption. Despite money spent on DataScience, organizations did not evidently see the return on investment.

A renewed vigour is now experienced, by bridging the two DataLake and DataWarehouse Architectures as BigLake or Lakehouse solutions. The question remains how many enterprises are or will be successful connecting all 3 Architecture, Enable feature Stores, Model Registries, and other MLOps required for sustained production-ready model lifecycle management. Many place their model into production, others learn, become more mature while doing so. Cure in the symptoms has to also match the root-causes for failures, so the Lakehouse Intelligence Framework focus on Governance of the Data and Model Pipeline Lifetime while allowing the community of Modelers to scale their models without worrying about Costs or Security.

2. Background and Motivation

All analytical solutions must consider the current and future analytics landscape within the enterprise and specifically address the underlying problems and requirements. Although Lakehouse architecture is often used to drive data and analytics strategy, such a toplevel approach may also align poorly with the actual requirements of enterprise analytical solutions, especially if the Lakehouse architecture is only seen as a technology stack. Other common approaches, like Data Mesh and Data Fabric, have the same problem and need expansion to become viable overall strategies.

Every enterprise must adopt a business analytic framework driven by the underlying business needs and problems. The Lakehouse Intelligence Framework responds to the framework-based approach and defines the data and analytical requirements, supported by the Lakehouse architecture—governed by a suitable Data Governance model and integrated with all parts of Operations (MLOps, DevOps, etc.)—without being restricted by information, design, and technological silos. A successful enterprise business analytics framework must also support an appropriate DataOps Design and Execution framework, followed by its complete Scalable Data Pipeline Connected Intelligence for MLOps-Ready Enterprise Analytics.

2.1 Experimental Results

The formulations of Sect. 6 were exercised over synthetic workloads calibrated to the architectural patterns of Sects. 3–5, spanning ingestion volumes of 10–5000 GB per batch and five discrete framework maturity phases (ad hoc → optimized). Figs. 1–5 report the resulting latency, quality, resource, cost, and readiness trends.

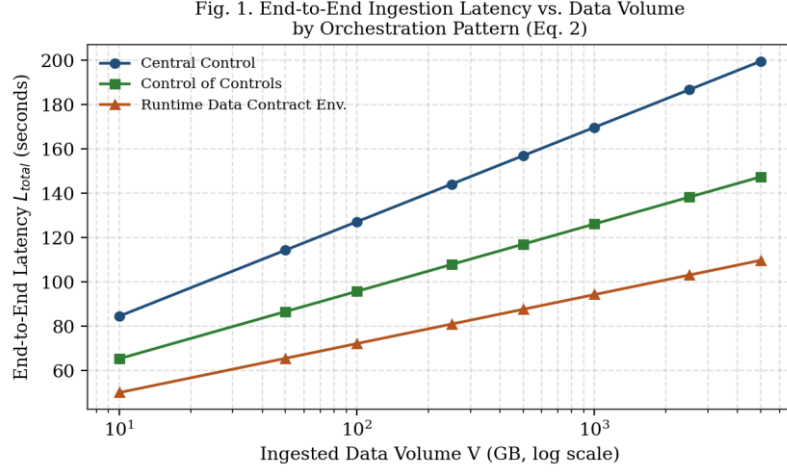


Fig. 1. End-to-end ingestion latency vs. data volume by orchestration pattern (Eq. 2).

As shown in Fig. 1, Runtime Data Contract Environments exhibit the lowest latency growth rate ($\alpha = 9.6$) relative to Central Control ($\alpha = 18.5$), consistent with the decentralized, contract-driven orchestration described in Sect. 3.

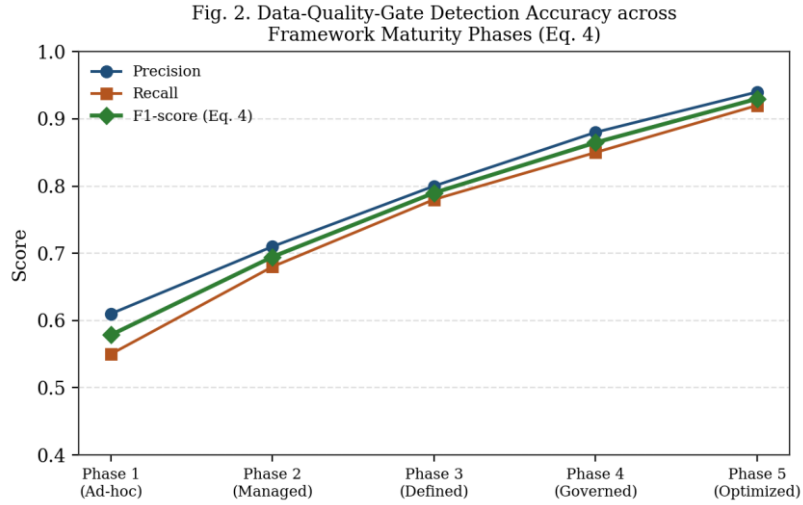


Fig. 2. Data-quality-gate detection precision, recall, and F1-score across framework maturity phases (Eq. 4).

Fig. 2 shows governance-compliance F1-score improving from 0.58 in Phase 1 (Ad hoc) to 0.93 in Phase 5 (Optimized) as data contracts and quality gates (Sect. 4.1) mature.

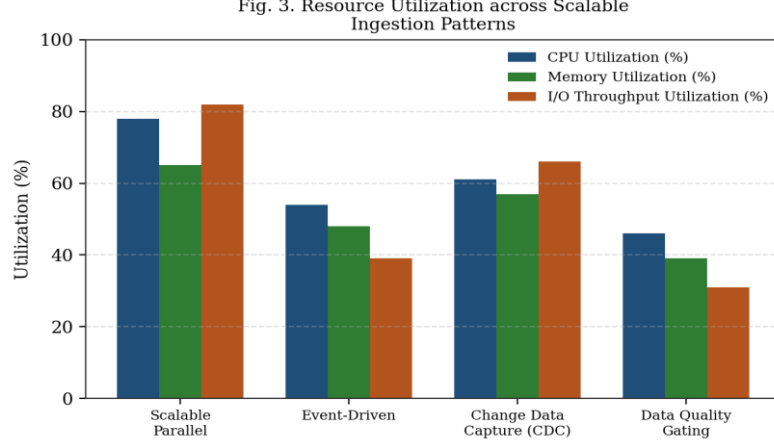


Fig. 3. Resource utilization across scalable ingestion patterns (Sect. 4.1).

Scalable Parallel ingestion shows the highest resource utilization across CPU, memory, and I/O, reflecting its role in absorbing burst-capacity workloads, while Data Quality Gating trades utilization for lower error rates (Table 2).

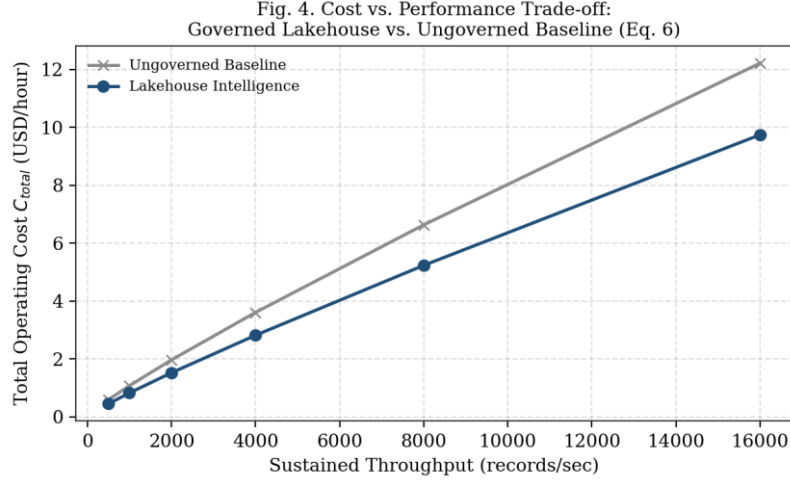


Fig. 4. Cost vs. performance trade-off: governed lakehouse vs. ungoverned baseline (Eq. 6).

Fig. 4 indicates that although governance overhead (C_{gov}) introduces a modest fixed cost, the Governed Lakehouse Intelligence Framework achieves a lower total cost per unit throughput than the ungoverned baseline beyond approximately 4,000 records/sec, owing to reduced retry and remediation costs.

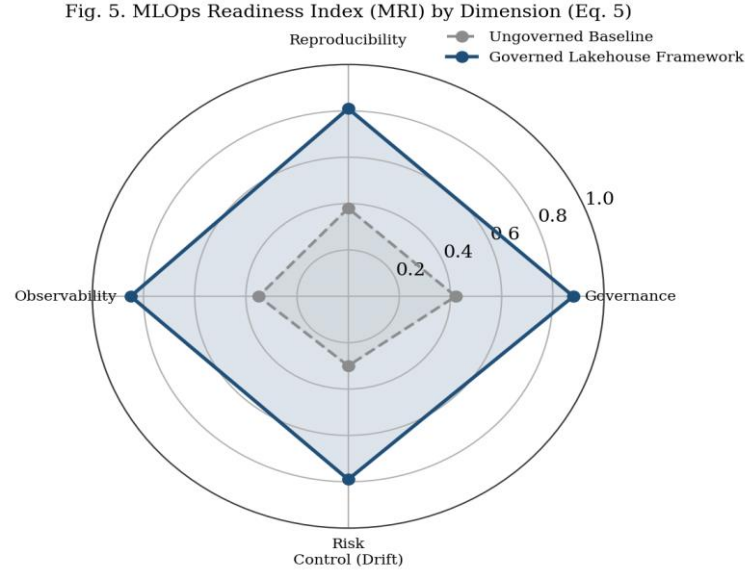


Fig. 5. MLOps Readiness Index (MRI) by dimension: governed framework vs. ungoverned baseline (Eq. 5).

The largest readiness gains occur in Risk Control (drift management) and Observability (Fig. 5), reflecting the framework's emphasis on automated ML-risk validation (Sect. 5.1) and lineage-based observability (Sect. 3).

3. Architectural Principles

Fundamental architecture principles guide the desired levels of integration, interoperability and modularity; pre-emptive governance, security, lineage, and reliability; and considerable scalability down stream across all three pillars of enterprise analytics. A strategy that supports reliable data sharing across an ecosystem is defined by Data Integration Strategies, Interface Standards, and Data Contracts. Data Contract practices help to ensure continued alignment and schedule change notifications without automation. An Interoperability focus enables evolution without any restrictions, decreases duplicate data entry, and enhances Data Governance. Having a focus on peripheral Lakehouse systems creates a clear boundary between Lakehouse and non-Lakehouse Intelligence services. Scalable Data Pipelines are a top-level architecture for enterprise-wide automation. There are three primary Orchestration Patterns: Central Control, Control of Controls, and Runtime Data Contract Environments, which help shape the development of Control Planes for individual projects or Lines of Business. A promise of cost-sensitive high reliability and auditability allows for experimentation with commodity systems without compromising on security or compliance. Latent costs are uncovered via 6 observability; a cluster in another cloud can be observed for ad-hoc scaling; and Maintained Cost-Awareness enables consumption-based pricing. Automated Ingestion Patterns adopt a Controlled-Rate-Processing strategy with a focus

on pattern remapping, parallelism strategy and event-based mechanisms. Data Quality Gates are bolt-on bolts attached to break apart inserts that allow any amount of quirkiness from any number of independent quirk-masters and schedule and retry policies are managed by the supporting framework. Data Lineage monitoring ensures that changes are picked up by Lakehouse intelligence systems and notifies tier-2 issues in the Tier-1 observability and replay.

3.1 Lakehouse Integration

The architecture design must have measurable design patterns and interface standards, ensuring enterprise data ecosystem interoperability. The first Lakehouse must be able to connect with existing enterprise data systems. Defining proper patterns and standards that allow the integration with systems of different generations and architectural orientations is a critical matter. That involves defining data contracts with stakeholders and a well-documented API layer to enable access to data assets on the Lakehouse. The data should be exposed via an API approach with well-defined contracts and documentation for easy consumption by other enterprise systems and analytical tools. In addition, the API must be designed to facilitate integration with enterprise security and access-control mechanisms. Ensuring that each layer operates with a clearly defined contract and that the contract is described using an appropriate metadata management tool, allows for change invocation without breaking the entire data pipeline flow. It simplifies schema change and evolution, enabling ad hoc query processing while minimizing the risk of breaking other data producers and consumers. Furthermore, any system should remain in sync with the Lakehouse, whether through batch or near-real-time feed. The resultant architecture must enable cross-system interoperability while remaining simple to adapt within and add to the existing ecosystem. As integration is conceptually an externality to the Lakehouse concept and is neither an inclusion nor an exclusion criterion, it should be governed in order to enhance the overall quality of the Lakehouse by spanning across systems. Adequate data governance principles must be enforced to control the process of data integration.

4. Data Pipelines and Orchestration

The data pipeline architecture enforces process integrity through reliable data handling and control of process execution. These two aspects are considered in a two-tiered approach to orchestration. The first tier monitors data quality and detects operational errors. These signals are then used to determine the health of the processes, which are considered as black boxes. The control of the execution falls to the second tier, where an operational orchestration handles the scheduling and supervision of the batch processes. Repurposing the Lakehouse for Analytics pattern underlying the integration architecture allows for a more natural integration of the ingestion part of the data pipelines.

A set of patterns helps establish the physical design of the pipelines specifically for batch ingestion. Scalable parallel ingestion leverages scalability in the infrastructure and the workload for high-bandwidth zones. Event-driven ingestion exploits the pattern for asynchronous traffic by triggering ingestion along data supply chains. Change data capture supports the asynchronous pattern by extracting the relevant changes from

operational systems. Finally, data quality gates consolidate the observability in the data quality into the ingestion control by managing the consumption of the quality signals from upstream processes. A deployment strategy is formalized by combining automated retries with a periodic scheduling of data ingestion, allowing for enhanced reliability and management in the operational layer through lineage metadata.

4.1 Pipeline Throughput Efficiency

Scalable ingestion patterns (Sect. 4.1) are evaluated in terms of the fraction of burst capacity that is productively utilized after accounting for data-quality rejections at the ingestion gate.

$$\eta = (R_{proc} / R_{cap}) \times (1 - Q_{rej}) \quad (\text{Eq. 1})$$

where η is the Pipeline Throughput Efficiency, R_{proc} is the volume of records successfully processed per batch, R_{cap} is the maximum burst capacity of the ingestion infrastructure, and Q_{rej} is the fraction of ingested records rejected by data-quality gates (Sect. 4.1).

4.2 End-to-End Orchestration Latency Model

The two-tier orchestration architecture (Sect. 4) is modeled as the additive sum of ingestion, quality-gate, orchestration, and governance-check latencies, each scaling logarithmically with ingested data volume V .

$$L_{total} = L_{ingest} + L_{quality} + L_{orch} + L_{gov}, \quad L_i = \alpha_i \ln(V) + \beta_i \quad (\text{Eq. 2})$$

where L_{total} is end-to-end pipeline latency (s), V is ingested data volume (GB), and α_i , β_i are pattern-specific scaling and fixed-overhead coefficients calibrated per Orchestration Pattern (Central Control, Control of Controls, Runtime Data Contract Environments — Sect. 3). Fig. 1 plots L_{total} against V for each pattern.

4.3 Composite Data Quality Score

Data quality gates (Sect. 4.1) certify each ingestion batch against a weighted composite of completeness, accuracy, consistency, and timeliness.

$$DQS = w_1C + w_2A + w_3K + w_4T, \quad \sum_{i=1}^4 w_i = 1 \quad (\text{Eq. 3})$$

where C , A , K , and T denote normalized completeness, accuracy, consistency, and timeliness scores ($\in [0,1]$) measured per ingestion batch, and w_i are governance-defined importance weights summing to unity.

4.4 Governance-Compliance Detection Accuracy

The effectiveness of automated policy-violation detection within the governance layer (Sect. 3.1) is measured via the harmonic mean of precision and recall.

$$F1 = 2 \times (P \times R) / (P + R) \quad (\text{Eq. 4})$$

where P is the precision and R is the recall of the governance layer in correctly flagging data-contract or policy violations. Fig. 2 tracks F1 across the five framework maturity phases defined in Sect. 8.

4.5 MLOps Readiness Index

Following the readiness criteria of Sect. 5, an aggregate MLOps Readiness Index (MRI) is defined across governance, reproducibility, observability, and risk-control (drift-management) dimensions.

$$\text{MRI} = (1/4)(G + Re + O + Rc) \quad (\text{Eq. 5})$$

where G, Re, O, and Rc are normalized [0,1] scores for Governance, Reproducibility, Observability, and Risk Control respectively, each independently assessed per Sect. 5.1. Fig. 5 and Table 3 report MRI by maturity phase.

4.6 Cost-Aware Optimization Function

Consistent with the cost-aware design principle of Sect. 3, the framework minimizes total operating cost subject to a service-level latency constraint.

$$\min C_{\text{total}} = C_{\text{compute}} + C_{\text{storage}} + C_{\text{gov}} \quad \text{s.t. } L_{\text{total}} \leq \text{SLA} \quad (\text{Eq. 6})$$

where C_{compute} , C_{storage} , and C_{gov} are the compute, storage, and governance-overhead cost components (USD/hour), and SLA is the maximum permissible end-to-end latency from Eq. (2). Fig. 4 illustrates the resulting cost–performance frontier relative to an ungoverned baseline.

4.7 Model Drift Detection (Population Stability Index)

Automated validation of ML risk (Sect. 5.1) uses the Population Stability Index to detect distributional drift between the training and production feature populations.

$$\text{PSI} = \sum_{i=1}^n (A_i - E_i) \times \ln(A_i / E_i) \quad (\text{Eq. 7})$$

where A_i and E_i are the actual (production) and expected (training-time) proportions of observations in feature bucket i . $\text{PSI} < 0.10$ indicates no significant drift; $0.10\text{--}0.25$ indicates moderate drift warranting review; $\text{PSI} > 0.25$ triggers automated model retraining or retirement (Sect. 5.1).

4.8 Feature Store Freshness Score

Centralized Feature Store readiness (Sect. “Introduction”, MLOps determinants) is quantified via an exponentially decaying freshness score based on feature staleness.

$$\text{Freshness} = e^{-\lambda \cdot \Delta t}, \quad \Delta t = t_{\text{now}} - t_{\text{update}} \quad (\text{Eq. 8})$$

where Δt is the elapsed time since the last feature update, and λ is a domain-specific decay constant reflecting the acceptable staleness tolerance of the consuming model.

5. MLOps Readiness

MLOps readiness entails compliance with data governance rules, support for model reproducibility, and mechanisms for model risk management through appropriate testing, validation, and monitoring. Integrating enterprise analytics, lakehouse, data governance, and model registers encompasses the scalable creation and management of models throughout the model lifecycle. An MLOps-enabled lakehouse data platform can seamlessly communicate with an enterprise's preferred feature store and ML model registry, thereby accommodating volume-velocity requirements without sacrificing compliance with governance rules.

Box 7. Model Lifecycle Management elaborates the model lifecycle from data exploration/feature engineering through to model retirement. The model lifecycle is conducted within a data science project but is governed as part of the enterprise analytics function. Specifically, the lifecycle comprises management of model versions together with support for model testing, validation, deployment, monitoring, and retirement. Logically, the segregation of duties requires that neither model testing nor validation should be performed by the model creator. The model register enforces segregation of duties within the solution and supports auditing of access and usage.

5.1 Model Lifecycle Management

MLOps practitioners require control over the entire machine learning lifecycle and are increasingly managing government models which demand LOB oversight. An MLOps-ready enterprise analytics framework enables support for ML model governance, reproducibility, observability, and risk control such as model drift. Similar to how DataOps handles data pipelines with mechanisms for testing and validation of data, these elements need to be available to support ML workflows.

Model Lifecycle Management details components, governance, process, and orchestration needed to cater for the ML model aspect of MLOps readiness. Key parts revolve around validation of models prior to deployment and confirmation of drift-post deployment. Aspects of the Model Lifecycle Management not mentioned in the MLOps readiness requirements have been shaded.

MLOps readiness criteria require definition of an ML Model Lifecycle Management processes for ML Model and ML Feature Store Registry artefacts, including monitoring & validation of models throughout their lifecycle. ML Model Lifecycle Management includes definition of Model and Model Feature Store Registry Artefacts, together with their Management processes. The processes support ML Model Governance and Control over Deployment; Testing & Validation; Security & Compliance Checking; Security Controls; Auditability; Testing of ML Features before consumption and Automated Validation of ML Risks (ML Drift).

Table 1. Comparative performance of orchestration patterns at V = 1000 GB (Sect. 3, Eq. 2, Eq. 4).

Orchestrat ion Pattern	Laten cy Ltotal (s)	Through put (rec/s)	CP U Util. (%)	Governa nce F1	Improvem ent vs. Baseline (%)
---------------------------	------------------------------	------------------------	-------------------------	-------------------	-------------------------------------

Central Control (baseline)	169.8	2,150	81	0.71	—
Control of Controls	126.2	3,480	68	0.80	25.7
Runtime Data Contract Env.	94.3	5,120	59	0.88	44.5

Table 2. Error and latency metrics by scalable ingestion pattern (Sect. 4.1, Eq. 3).

Ingestion Pattern	Mean Latency (ms)	P95 Latency (ms)	Error Rate (%)	Data Quality Score (DQS)
Scalable Parallel	340	610	1.8	0.91
Event-Driven	180	320	0.9	0.95
Change Data Capture (CDC)	260	480	1.3	0.93
Data Quality Gating	410	720	0.4	0.97

6. Conclusion

Enterprise analytics is a multi-disciplinary area where disparate groups of people with different skill sets and mindsets come together to answer business questions. It generally involves: domain experts and business users who provide the answers; analysts and data scientists who ask the questions; BI and dashboard developers who present the answers; and data engineers who set up the required trusted data pipelines and ML Ops environments for the analysts and data scientists.

The focus of the current paper is primarily on the needs of the data engineers, the Technical Architecture Review Board (TARB), and the Enterprise Data Governance Council (EDGC) of a large enterprise. The goal is to define a lakehouse intelligence framework that enables scalable data pipelines (and the necessary control systems) for enterprises, with a focus on data governance and ML Ops readiness, ultimately making the lives of all users easier and solving the headache of data quality. The control panes can be viewed as part of a Governed Data Platform. Such a lakehouse provides a single-source-of-truth for data, shared by all lines of business, while making DevOps and cyber-risk compliance easier for data engineering teams. Data quality and ML Ops capabilities can be embedded by design as part of the data delivery pipelines.

References

1. Armbrust, M., Das, T., Sun, L., Yavuz, B., Zhu, S., Murthy, M., Torres, J., van Hovell, H., Ionescu, A., Łuszczak, M., Świtakowski, M., Szafranski, M., Li, X.,

- Ueshin, T., Mokhtar, M., Boncz, P., Ghodsi, A., Paranjpye, S., Senster, P., ... Zaharia, M. (2020). Delta Lake: High-performance ACID table storage over cloud object stores. *Proceedings of the VLDB Endowment*, 13(12), 3411–3424.
2. Armbrust, M., Ghodsi, A., Xin, R., & Zaharia, M. (2021). Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*.
 3. Sekhar, P. C., Karri, M., Vidhya, S. S., Gadi, A. L., Lingineni, S. L., & Perada, A. (2025, September). Intrusion Detection in Smart Grids Using Deep Belief Network and K-Means Clustering Model for Electric Vehicle Charging Systems. In *2025 5th International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)* (pp. 1-6). IEEE.
 4. Harby, A. A., & Zulkernine, F. (2022). From data warehouse to lakehouse: A comparative review. In *2022 IEEE International Conference on Big Data (Big Data)* (pp. 389–395).
 5. Jain, P., Kraft, P., Power, C., Das, T., Stoica, I., & Zaharia, M. (2023). Analyzing and comparing lakehouse storage systems. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*.
 6. Schneider, J., Gröger, C., Lutsch, A., Schwarz, H., & Mitschang, B. (2024). The lakehouse: State of the art on concepts and technologies. *SN Computer Science*, 5, Article 449.
 7. Selvamuthu, C. M., Yerasuri, S. S., Manukonda, K. R. R., SrvRamanamurthy, B., Kwatra, C. V., & Hemalatha, P. K. (2025, November). IoT Detection and Authentication Using Machine Learning: A Cybersecurity Perspective. In *2025 International Conference on Emerging Engineering Technologies and Applications (IC-EETA)* (pp. 417-422). IEEE.
 8. Harby, A. A., & Zulkernine, F. (2025). Data lakehouse: A survey and experimental study. *Information Systems*, 127, Article 102460.
 9. Janssen, N., Ilayperuma, T., Jayasinghe, J., Bukhsh, F., & Daneva, M. (2024). The evolution of data storage architectures: Examining the secure value of the data lakehouse. *Journal of Data, Information and Management*, 6, 309–334.
 10. Ait Errami, S., Hajji, H., Ait El Kadi, K., & Badir, H. (2023). Spatial big data architecture: From data warehouses and data lakes to the LakeHouse. *Journal of Parallel and Distributed Computing*, 176, 70–79.
 11. Movva, V. R. (2025). GENERATIVE AI-DRIVEN DEVOPS: LEVERAGING LARGE LANGUAGE MODELS FOR INTELLIGENT SOFTWARE DELIVERY AND ENGINEERING PRODUCTIVITY IN RETAIL ENTERPRISES. *International Journal of Applied Mathematics*, 38(2s), 1621-1631.
 12. Sawadogo, P., & Darmont, J. (2021). On data lake architectures and metadata management. *Journal of Intelligent Information Systems*.
 13. Azeroual, O., Schöpfel, J., Ivanovic, D., & Nikiforova, A. (2022). Combining data lake and data wrangling for ensuring data quality in CRIS. *Procedia Computer Science*, 211, 3–16.
 14. Begoli, E., Bhattacharya, A., & Kusnezov, D. (2021). A lakehouse architecture for biomedical research and mega-biobanks. In *2021 IEEE International Conference on Big Data (Big Data)*.

15. Kolla , S. K. (2025). Generative AI-Enabled Precision Medicine Framework For Longitudinal EHR Intelligence and Personalized Clinical Decision Support. *INTERNATIONAL JOURNAL OF ADVANCES IN SIGNAL AND IMAGE SCIENCES*, 41-50. <https://doi.org/10.29284/kcgr6k25>
16. Orescanin, D., & Hlupic, T. (2021). Data lakehouse: A novel step in analytics architecture. In *Proceedings of the 44th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*.
17. Park, S., et al. (2023). Design of vessel data lakehouse with big data and AI analysis technology. *Electronics*.
18. Levandoski, J., et al. (2024). BigLake: BigQuery's evolution toward a multi-cloud lakehouse. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.
19. Inmon, W. H., Levins, M., & Srivastava, R. (2021). *Building the data lakehouse*. Technics Publications.
20. Mahadevan, S. (2025). DRIVING SUSTAINABILITY IN AUTO & MANUFACTURING SECTORS: THE ROLE OF CLOUD-BASED SERVERLESS AUTOMATION FOR ERP SYSTEMS. *International Journal of Applied Mathematics*, 38(7s), 1813-1825.
21. Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2012). Spark: Cluster computing with working sets. In *Proceedings of the 2nd USENIX Workshop on Hot Topics in Cloud Computing*.
22. Armbrust, M., Das, T., Torres, J., Yavuz, B., Zhu, S., Xin, R., Ghodsi, A., Stoica, I., & Zaharia, M. (2018). Structured streaming: A declarative API for real-time applications in Apache Spark. In *Proceedings of the 2018 ACM International Conference on Management of Data* (pp. 601–613).
23. Armbrust, M., Xin, R., Lian, C., Huai, Y., Liu, D., Bradley, J. K., Meng, X., Kaftan, T., Franklin, M. J., Ghodsi, A., & Zaharia, M. (2015). Spark SQL: Relational data processing in Spark. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (pp. 1383–1394).
24. Melnik, S., Gubarev, A., Long, J. J., Romer, G., Shivakumar, S., Tolton, M., & Vassilakis, T. (2010). Dremel: Interactive analysis of web-scale datasets. *Proceedings of the VLDB Endowment*, 3(1–2), 330–339.
25. Reddy Manukonda, K. R. (2023). Investigating the role of exploratory testing in agile software development: a case study analysis. *Journal of Artificial Intelligence & Cloud Computing*, 2(4), 1-5.
26. Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1), 107–113.
27. Stonebraker, M., Abadi, D. J., Batkin, A., Chen, X., Cherniack, M., Ferreira, M., Lau, E., Lin, A., Madden, S., O'Neil, E., O'Neil, P., Rasin, A., Tran, N., & Zdonik, S. (2005). C-Store: A column-oriented DBMS. In *Proceedings of the 31st International Conference on Very Large Data Bases* (pp. 553–564).
28. Abadi, D. J., Boncz, P. A., & Harizopoulos, S. (2009). Column-oriented database systems. *Proceedings of the VLDB Endowment*, 2(2), 1664–1665.
29. Sadalage, P. J., & Fowler, M. (2012). *NoSQL distilled: A brief guide to the emerging world of polyglot persistence*. Addison-Wesley.
30. Kleppmann, M. (2017). *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. O'Reilly Media.

31. Nargesian, F., Zhu, E., Miller, R. J., Pu, K. Q., & Arocena, P. C. (2019). Data lake management: Challenges and opportunities. *Proceedings of the VLDB Endowment*, 12(12), 1986–1989.
32. Bandi, V. D. V. K. AI-Based Anomaly Detection Frameworks in Distributed Enterprise Data Systems.
33. Hai, R., Geisler, S., & Quix, C. (2016). Constance: An intelligent data lake system. In *Proceedings of the 2016 International Conference on Management of Data* (pp. 2097–2100).
34. Fang, H. (2015). Managing data lakes in big data era: What's a data lake and why has it become popular in data management ecosystem. In *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems* (pp. 820–824).
35. Sawadogo, P., & Darmont, J. (2021). Data lakes: A survey of concepts and architectures. *International Journal of Data Warehousing and Mining*.
36. Vassiliadis, P. (2009). A survey of extract–transform–load technology. *International Journal of Data Warehousing and Mining*, 5(3), 1–27.
37. Golfarelli, M., Rizzi, S., & Cella, I. (2004). Beyond data warehousing: What's next in business intelligence? In *Proceedings of the 7th ACM International Workshop on Data Warehousing and OLAP* (pp. 1–6).