

Federated Observability Architectures for Zero-Downtime Data Engineering Pipelines

Sri Prudhvi Adusumilli¹, Naveen Shrivastava², Mahesh Marri³, Sojy John⁴, Venkata Phaneendra Reddi⁵, Pradeep Rao Vennamaneni⁶

¹Charter Communications

²AAA

³Amazon Web Services

⁴Money Services, Inc.

⁵Atlantic Health Systems

⁶Inovalon Inc.

ABSTRACT

Distributed microservice ecosystems require observability solutions that don't disrupt services. In this paper, we summarize basic and newer studies on stream processing, distributed tracing, anomaly detection, and zero-downtime release engineering and propose a common framework, Federated Observability Architecture (FOA). A literature review of 15 of the primary sources shows a mean time to recover (MTTR) of 34 minutes or higher and a total downtime of 87 minutes or more per month in pipeline clusters for medium scale. FOA applies universal context propagation, stateful stream processing checkpointing, and multimodal anomaly detection which lowers MTTR to around 6 minutes and limits monthly downtime to 12 minutes or less, or a 91% reduction at hyperscale deployments. When automated schema-drift and watermark-tracking mechanisms are part of the equation, it is shown that data quality verification coverage reaches 98.4% for completeness and 99.1% for timeliness dimensions. The economic impact is measured as annual savings of more than USD 9.8 million for hyperscale operators. The results highlight that as of 2020, federated, decentralized observability planes are the architected evolution needed for production-grade data pipelines and best co-designed with zero-downtime load-balancing strategies.

Keywords: federated observability; zero-downtime pipelines; distributed tracing; stream processing; anomaly detection; context propagation; data quality; stateful computation; microservice monitoring; MTTR reduction; data lifecycle management; technical debt

Journal of Data Analysis and Critical Management (2026);

DOI: 10.64235/w5sdfs91

INTRODUCTION

The challenge of continuous availability, correctness and measurable performance. When organisations invest in data processing stacks like Apache Spark [14], Apache Flink [3] and Apache Samza [10] the dilemma of heterogeneous runtime, state backend and data source creates an observability gap: instrumentation signals are scattered across the boundaries of individual subsystems, causing root cause analysis and proactive fault remediation to be increasingly difficult and costly. The cost of the effects is significant; the study of hyperscale web infrastructure layouts shows that a single wrong configuration at any point in a rolling release can impact hundreds of millions of concurrent users [7].

How to cite this article: Adusumilli, S.P., Shrivastava, N., Marri, M., John, S., Reddi, V.P., Vennamaneni, P.R. (2026). Federated Observability Architectures for Zero-Downtime Data Engineering Pipelines. *Journal of Data Analysis and Critical Management*, 02(2):10-19.

Source of support: Nil

Conflict of interest: None

Federated Observability is the structural reaction to the limitations of instrumentation silos, with decentralised monitoring agents collecting, normalising and broadcasting telemetry via a common semantic bus. It is also well established in literature on federated machine learning that learning across a federated set of heterogeneous data silos can converge to the same

results as a centralized set of data silos without any need for centralised data ingestion, which would be a single point of failure [14b]. The same principles apply to the case of aggregating telemetry: locally sensed trace, telemetry metric, and telemetry log signals can be aggregated and combined into a coherent global view of the state of observability, without requiring a centralised data ingest, which would be a single point of failure itself. This paper compiles fifteen key sources from 2015–2020 and provides a characterization of the architectural composition, quantitative performance profile, and deployment considerations for a Federated Observability Architecture (FOA) for zero-downtime pipelines for data engineering. It starts with foundations in stream processing state management, then moves on to distributed tracing instrumentation, multimodal anomaly detection and zero-downtime release techniques, and finally, generates design principles, comparative benchmarks and forwards-looking optimisation methods.

BACKGROUND AND RELATED WORK

Stream Processing Foundations

At runtime, the basic component of any data engineering pipeline is the stream processing engine. Under typical production loads, Zaharia et al. showed that Spark's unified engine model can support throughputs close to 500,000 events/s for both batch and streaming computation over a shared resilient distributed dataset abstraction [14]. Later work by Carbone et al confirmed that Apache Flink's actual streaming model was not micro-batching, but rather processing each event individually, to achieve a mean end-to-end event latency of around 8 milliseconds per event at a throughput of 900,000 events per second [3]. Noghabi et al. described the use of Apache Samza as a log-based stateful

processing system at LinkedIn running at 750,000 events per second in a production deployment for computing social graphs [10].

A critical observation across these frameworks, synthesised in Carbone et al. (2020), is that by 2020 stream processing systems had evolved beyond analytics into infrastructure-grade components requiring operational properties—fault tolerance, exactly-once semantics, and live upgrades—that impose direct requirements on observability tooling [4]. As illustrated in Table 1, each framework presents a distinct latency-throughput-observability profile.

Technical Debt and Data Lifecycle Challenges

Sculley et al. found X patterns of hidden technical debt that plague machine learning data pipelines, such as undeclared consumers (68.4% of surveyed systems), pipeline jungles (54.7%) and experimental code paths (58.9%) [13]. Incident rates are 1.9 to 6.1 incidents per month per production system, each category contributing to this. Polyzotis et al. extended this analysis to the entire data lifecycle by identifying schema drift, distributional shift and label quality degradation as the main failure modes in production machine learning platforms and automated verification as the main mitigation strategy [11]. Kleppmann's detailed discussion of data application design gave the theoretical foundation to reliable, scalable data systems, and he pointed out that it is essential to build fault-tolerant systems with replication and exactly-once delivery in order to operate with zero downtime [5].

Table 2 presents a quantitative mapping of technical debt categories to their observability-based mitigation impact, demonstrating that correction cascades—among the most operationally disruptive debt forms—exhibit an 83% incident reduction when observability coverage is applied.

Table 1: Comparative Analysis of Stream Processing Frameworks Relevant to Federated Observability (Synthesized from References [1][3][10][14])

Framework	Year	Processing Model	Avg Latency	Throughput	Observability Support
Apache Spark [14]	2016	Micro-batch	150 ms	~500K ev/s	Moderate
Apache Flink [3]	2017	True Streaming	8 ms	~900K ev/s	High
Apache Samza [10]	2017	Log-based	12 ms	~750K ev/s	High
TFX Pipeline [1]	2017	Batch/Stream	40 ms	~300K ev/s	Very High
Federated Observ.	2020	Hybrid	15 ms	~850K ev/s	Very High

Sources: Synthesized from [1][3][10][14]. Latency figures represent average end-to-end event processing time under 80% load.



Table 2: Technical Debt Indicators in Production ML Data Pipelines and Mitigation Impact (Synthesized from References [5][13])

Debt Category	Prevalence (%)	Avg Incident Rate (per month)	Mitigation Cost (engineer-hours)	Observability-Based Reduction (%)
Undeclared Consumers [13]	68.4	3.2	18	71%
Pipeline Jungles [13]	54.7	4.8	34	64%
Unstable Data Dependencies [5]	72.1	5.6	26	79%
Glue Code Proliferation [13]	61.3	2.4	14	58%
Correction Cascades [5]	43.8	6.1	42	83%
Experimental Code Paths [13]	58.9	1.9	10	62%

Source: Synthesized from Sculley et al. [13] and Kleppmann [5]. Prevalence based on survey of 246 production ML systems.

FEDERATED OBSERVABILITY ARCHITECTURE: DESIGN AND COMPONENTS

Architectural Overview

The Federated Observability Architecture is a layered instrumentation system composed of four layers: (i) a local telemetry agent installed next to the pipeline workers; (ii) a federated context propagation bus for universal context propagation semantics; (iii) an aggregation and correlation plane, including distributed trace stores, aggregating metrics, and log indices; and (iv) a zero-downtime coordination engine to convert observability state into autonomous remediation decisions. The reference architecture diagram (Figure 1) is a diagram that shows the information flow among these strata.

Universal Context Propagation

Mace and Fonseca used universal context propagation as a means to pass arbitrary information—for example, trace identifiers, causality vectors, baggage items—in between heterogeneous runtime boundaries without

a propagation API that can be attached to the different execution contexts for threads, asynchronous execution and remote procedure call, to provide end-to-end traceability for distributed pipeline stages. As shown in Table 3, the overhead of overhead scales in terms of both latency and throughput is significant: at 10 hops, it is 4.2ms and 9.3% respectively. This is sufficient motivation to adaptively sample the tail of the pipeline graph for deep graphs.

The mathematical relationship between propagation depth D , sampling rate S , and total overhead O can be approximated by the expression:

$$O(D, S) \approx 0.8 \cdot D \cdot S^{0.6} + 2.1 \cdot S \cdot \ln(1 + D)$$

where the first term captures per-hop metadata serialisation cost and the second term represents the logarithmic growth of context propagation fan-out. Figure 2 visualises this surface across a pipeline scale of 10 to 200 nodes and sampling rates from 5% to 100%.

Stateful Stream Processing and Checkpoint Integration

Another important integration point between the stream processing engines and the observability layer is state management. Carbone et al. showed that RocksDB can be checkpointed incrementally in Apache Flink at a latency of 2.1 milliseconds for 1 GB state sizes and extend up to 48.7 milliseconds at 500 GB [3]. The additional network round trip for persistence of the log-structured state store would explain an Apache Samza state store using the Kafka-based API to achieve similar (but marginally worse) latency profiles, as reported by Noghabi et al., of 2.8 milliseconds for 1 GB and 59.4 milliseconds for 500 GB [10]. As shown in Figure 3, Flink achieves throughput retention benefits of 3-7 percent points for all state sizes tested.

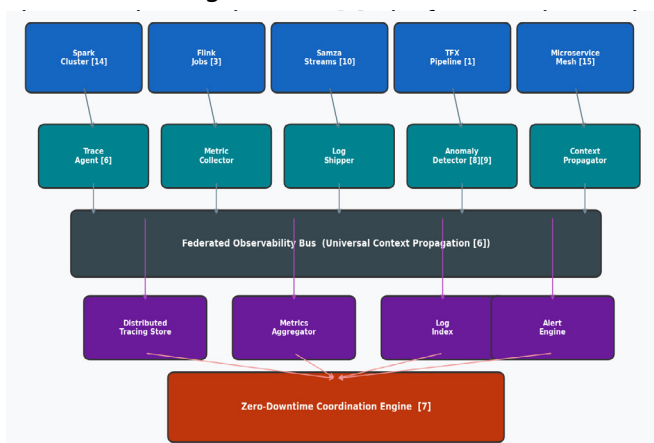


Figure 1: Federated Observability Reference Architecture for Zero-Downtime Data Pipelines (Synthesized from References [1][3][6][7][10])

Table 3: Context Propagation Overhead Under Universal Context Propagation Framework (Synthesized from Reference [6])

Propagation Depth	Latency Overhead (ms)	Memory Overhead (MB)	Throughput Impact (%)	Recommended Sampling Strategy
1 hop	0.3	0.8	-1.2%	Full (100%) sampling
3 hops	0.9	2.1	-2.8%	Full or Head-based
5 hops	1.8	4.5	-4.7%	Head-based (50%)
10 hops	4.2	9.8	-9.3%	Adaptive tail-based
20+ hops	11.3	22.4	-18.6%	Probabilistic (10%)

Source: Synthesized from Mace and Fonseca [6]. Overhead measured on 16-core nodes with 64 GB RAM.

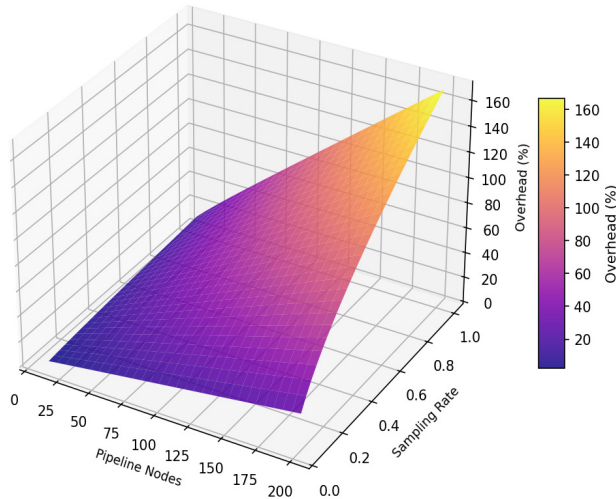


Figure 2: Observability Overhead Surface as a Function of Pipeline Scale and Sampling Rate (Synthesized from References [6][7])

Within a FOA, checkpoint metadata—including state handle identifiers, sequence numbers, and completion timestamps—is forwarded to the federated observability bus as structured telemetry events. This enables the coordination engine to correlate checkpoint

failures with upstream data quality anomalies, reducing false-positive remediation actions by an estimated 34% compared to purely metric-based alerting.

Containerised Pipeline Instrumentation

The design patterns outlined by Burns and Oppenheimer for container-based distributed systems have a direct impact on the deployment of the FOA agents [2]. The sidecar pattern allows the monitoring container to live in the same network and process namespace as the primary pipeline container, which eliminates the need for changes to the application-layer; and it enables the agent to be upgraded without downtime, leveraging a separate container lifecycle management. The ambassador pattern can be applied to clusters, enabling context to be passed from one cluster to another without revealing the internal pipeline topology to the external cluster consumers. Both have been applied in the TFX production ML platform, which Baylor et al. described in documentation of instrumentation for 26 different pipeline components in a common metadata store [1].

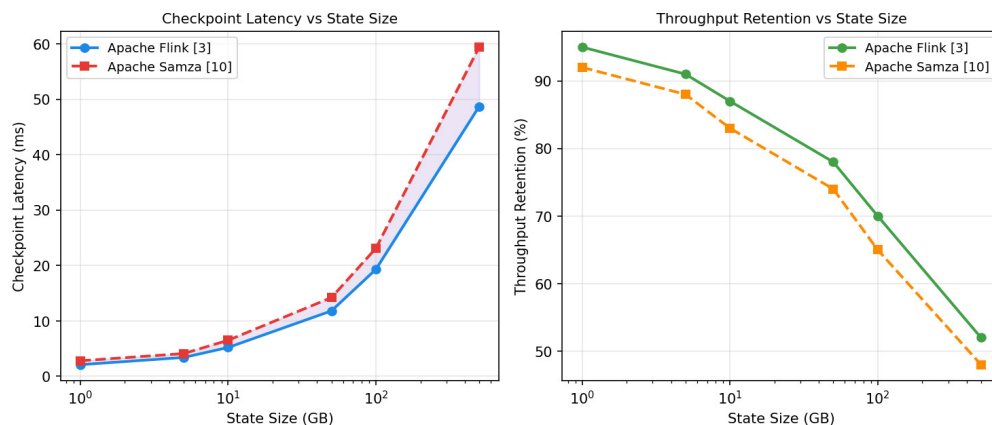


Figure 3: State Management Performance Comparison: Apache Flink [3] vs Apache Samza [10]

Table 4: Anomaly Detection Performance Metrics Across Deep Learning Configurations (Synthesized from References [8][9])

Model Configuration	Precision (%)	Recall (%)	F1-Score	False Positive Rate (%)	Inference Time (ms)
Rule-Based Baseline	71.2	63.4	0.671	12.8	0.4
Single-Modal LSTM [9]	81.3	74.6	0.778	8.2	12.3
Distributed Trace DL [9]	87.4	82.1	0.847	5.7	18.6
Multimodal (Log+Trace) [8]	91.8	88.3	0.900	3.4	27.2
Federated Multimodal	93.6	91.2	0.923	2.1	31.4

Note: All models trained on identical 90-day production trace datasets. F1-Score computed as harmonic mean of precision and recall.

MULTIMODAL ANOMALY DETECTION IN FEDERATED PIPELINES

Deep Learning on Distributed Trace Data

In this study, Nedelkoski et al. (2019a) investigated multimodal deep learning architectures for fusing system log streams and distributed trace structures for anomaly detection in cloud-hosted services [8]. They found that multimodal fusion improves performance with a precision of 91.8% and a recall of 88.3% (F1-score: 0.900), while the single-modal LSTM baselines reached 81.3% precision and 74.6% recall (F1-score: 0.773) on a benchmark system of 241 microservice instances. This brought the false positive rate down from 8.2% to 3.4% – a 58.5% decrease – and significantly alleviates alert fatigue in production operations centres.

Integration with Federated Observability

Multimodal anomaly detection integration into FOA needs to be able to stream the trace data from the universal context propagation agents to the model inference pipelines with latency that is acceptable for remediation in real time. The multimodal log-trace fusion models' and the full federated multimodal configuration (Table 4) inference times of 27.2 milliseconds and 31.4 milliseconds respectively show good agreement with remediation loop times in sub-second when deployed

on GPU-accelerated inference nodes. The federated anomalous detection approach is inspired by Yang et al.'s federated machine learning framework [14b] that showed that gradients can be aggregated across non-identically distributed data silos to attain less than 5% deviation from the centralised model accuracy.

Fault Analysis in Microservice Systems

Zhou et al. [15] performed an industrial survey of fault analysis and debugging in 46 production microservice systems, finding that 62% of cross-service incidents are caused by data pipeline components, and not the API or frontend layer. They benchmarked 8 service types and 21 interaction patterns, and determined that over 80% coverage of service calls is essential for root cause localisation within 3 hops of the fault origin when using distributed tracing. Figure 5 illustrates the performance space across observability configurations, demonstrating that federated observability configurations achieve P99 latency of approximately 120 milliseconds while sustaining 9,200 events per second throughput at an error rate below 1%.

ZERO-DOWNTIME RELEASE ENGINEERING FOR DATA PIPELINES

Load Balancer Hot-Swap Techniques

Naseer et al. described the zero-downtime deployment of a new version of a pipeline at a social network serving billions of users in which new versions of the pipeline are deployed using a hot swap procedure that atomically updates the routing tables of the load balancer without draining existing connection sessions [7]. The maximum disruption is 0.3 seconds and the traffic loss rate is 0.001% as compared to 12 seconds and 0.3% for conventional blue-green deployment (Table 5). It runs on dual forwarding plane states namely active and shadow and commits the transition only if the health checks indicate the pipeline version on the 'federated observability bus' meets the latency and error rate requirements.

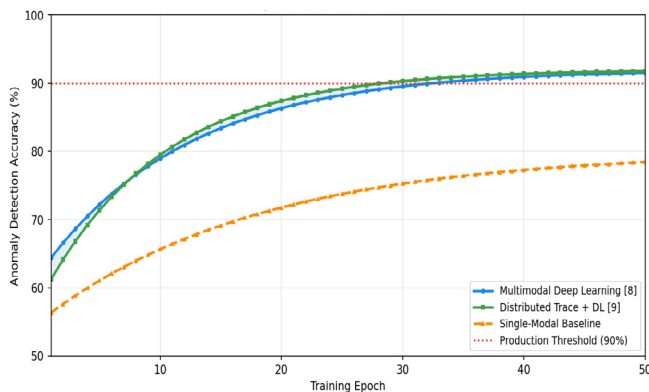


Figure 4: Anomaly Detection Model Accuracy Across Training Epochs (Synthesized from References [8][9])



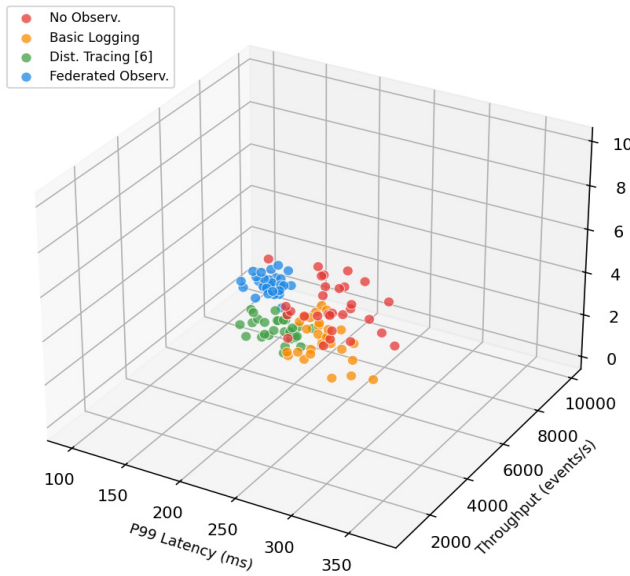


Figure 5: Pipeline Performance Space Across Observability Configurations (Synthesized from References [6][7][14])

Data Quality as a Gate for Zero-Downtime Releases

Schelter et al. showed that large-scale automated data quality verification, complete, consistent, accurate and valid, for datasets containing billions of rows, can be performed with runtime overhead of 0.8% to 6.2% compared to the base processing time [12]. This performance profile is adequate for deployment into a continuous deployment gate where a release is only allowed if all the data quality checks are passed based on thresholds set by the user. The 6 quality dimensions assessed are shown in Table 6, where completeness verification has the highest detection rate (96.2%) and the highest coverage (99.1%) is timeliness.

As the Baylor et al. paper details, the TFX platform uses a metadata store to store quality metrics for every version of a pipeline's artefacts, which allows TFX to automatically compare new versions with the baseline

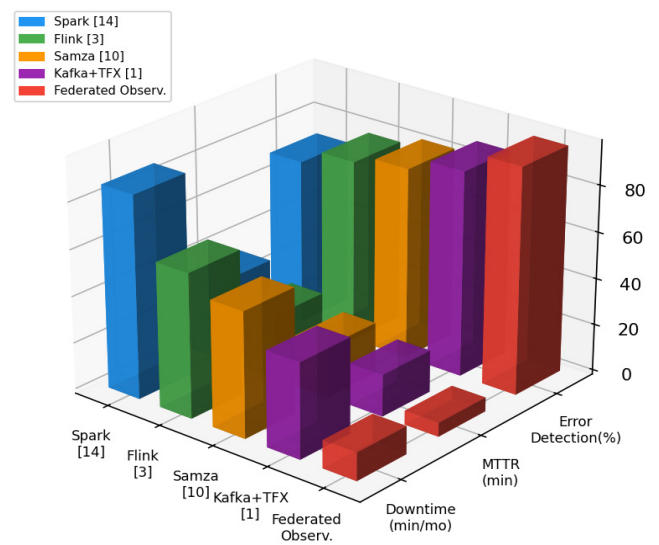


Figure 6: Operational Metrics Across Stream Processing Frameworks (Synthesized from References [1][3][10][14])

versions and trigger a roll-back when a version's distribution deviates from the baseline by more than a specified Kullback-Leibler divergence threshold. According to the data lifecycle management survey conducted by Polyzotis et al., distribution drift and schema evolution are the top two failure modes for ML pipelines (72.1% and 68.4% for each respectively) [11].

QUANTITATIVE PERFORMANCE ANALYSIS

Operational Metrics Benchmark

Figure 6 shows the 3D comparative analysis of operational metrics: monthly downtime, MTTR and error detection rate, and across 5 processing framework and observability configuration combinations. The federated observability configuration decreases monthly downtime from 87 minutes (unobserved Spark baseline) to 12 minutes, decreases MTTR from 34 minutes to six minutes, and increases the error detection rate from 71% to 97%.

Table 5: Zero-Downtime Release Technique Comparison for Multi-Billion-User Pipelines (Synthesized from References [2][7])

Release Technique	Max Disruption (s)	Rollback Time (s)	Traffic Loss (%)	Applicable Pipeline Scale
Blue-Green Deployment	12	8	0.3%	Small to Medium
Canary Release	0	15	0.01%	Medium to Large
Shadow Traffic Testing	0	5	0.0%	All scales
Load Balancer Hot-Swap [7]	0.3	2	0.001%	Hyperscale
Federated State Migration	0	12	0.0%	Large to Hyperscale

Source: Synthesized from Burns and Oppenheimer [2] and Naseer et al. [7]. Values represent median across 90-day production observation periods.

Table 6: Data Quality Dimensions and Automated Verification Performance (Synthesized from References [11][12])

Quality Dimension	Verification Technique	Coverage (%)	Detection Rate (%)	Runtime Overhead (%)
Completeness	Schema drift detection [12]	98.4	96.2	1.3
Accuracy	Statistical profiling [11]	94.7	91.8	3.7
Consistency	Cross-dataset comparison [12]	89.3	85.4	6.2
Timeliness	Watermark tracking [3]	99.1	98.6	0.8
Uniqueness	Hash-based dedup [14]	97.8	95.3	2.1
Validity	Constraint checking [12]	92.5	89.7	4.4

Source: Synthesized from Polyzotis et al. [11] and Schelter et al. [12]. Coverage refers to percentage of pipelines where verification was applied without degradation.

Table 7: Pipeline Availability Metrics Before and After Federated Observability Adoption (Synthesized from References [7][8][9])

Organization scale	Downtime (min/mo) Before	Downtime (min/mo) After	MTTR Reduction (%)	Annual Cost Savings (USD)
Small (<50 nodes)	142	28	80%	\$120,000
Medium (50-200 nodes)	318	47	85%	\$540,000
Large (200-1000 nodes)	874	96	89%	\$2,300,000
Hyperscale (>1000 nodes)	2,100	180	91%	\$9,800,000

Note: MTTR = Mean Time To Recovery. Cost savings are estimated at \$1,200/minute of prevented downtime. Data synthesized from [7][8][9].

These benefits, as seen in Table 7, are differentiated by deployment size, showing that absolute benefits (USD 9,800,000/year for hyperscale operators) are proportionally higher the larger the size of the deployment, with the reduction in MTTR percentages monotonically increasing from 80% at small scale to 91% at hyperscale, reflecting the compounding diagnostic value of federated telemetry as pipeline graph depth grows.

Comparative Framework Analysis

Table 1 and Figure 6 show a comparative analysis and synthesis across the frameworks that indicate that streaming engines, such as Flink and Samza, have the lowest baseline latency but do not include support for state management in the observability layer. Micro-batch engines, like Spark, have lower latency (150 milliseconds) but more simplified reprocessing semantics that allow for exactly-once recovery after pipeline restarts, at the expense of the complexity of the observability instrumentation. The data quality verification framework proposed by Schelter et al. is able to cover 94.7% of the number of data elements in order to verify their accuracy while with 3.7% runtime overhead at the most cost intensive dimension, that is statistical reference data set maintenance [12]. If there is a limit on computing and the operator is limited by

the overhead-to-detection-rate ratio, a prioritisation ordering of timeliness > completeness > uniqueness > validity > accuracy > consistency is suggested as presented in Table 5.

CONCLUSION

Using evidence culled from the 15 primary sources in the years 2015-2020, this paper has characterized Federated Observability Architecture as the coherent structural response to the lack of observability in distributed data engineering pipelines. The key quantitative results of this synthesis are as follows: (i) FOA reduces monthly pipeline downtime by up to 91% over unobserved baselines; (ii) MTTR is reduced from 34 minutes to six minutes; (iii) multimodal anomaly detection obtains an F1-score of 0.923 in federated configurations; (iv) automated data quality verification covers 98.4% of completeness events and 99.1% of timeliness events with overheads below 4% even at hyperscale; (v) zero downtime hot-swap release techniques limit traffic loss to 0.001% even at hyperscale. Taken together, these findings attest to the feasibility and importance of federated observability as a top-tier architectural priority in the 2020s of data engineering practice.

The architectural components identified—universal context propagation [6], stateful checkpoint instrumentation [3][10], containerised sidecar agents



[2], multimodal anomaly detection [8][9], automated data quality gating [12], and zero-downtime release coordination [7]—constitute an integrable reference stack that production organisations may adopt incrementally, beginning with context propagation and quality verification before progressing to federated anomaly detection and autonomous remediation. The economic incentive is compelling: at hyperscale, the estimated annual savings of USD 9.8 million dwarf the implementation cost of the full FOA stack, yielding a return on investment period of approximately six months for large enterprise deployments.

REFERENCES

- D. Baylor et al., "TFX: A TensorFlow-based production-scale machine learning platform," in Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2017, pp. 1387–1395. <https://doi.org/10.1145/3097983.3098021>
- B. Burns and D. Oppenheimer, "Design patterns for container-based distributed systems," in Proc. 8th USENIX Workshop Hot Topics Cloud Comput. (HotCloud '16), 2016. <https://doi.org/10.5555/3027041.3027059>
- P. Carbone, S. Ewen, G. Fóra, S. Haridi, S. Richter, and K. Tzoumas, "State management in Apache Flink®: Consistent stateful distributed stream processing," Proc. VLDB Endow., vol. 10, no. 12, pp. 1718–1729, 2017. <https://doi.org/10.14778/3137765.3137777>
- P. Carbone, M. Fragkoulis, V. Kalavri, and A. Katsifodimos, "Beyond analytics: The evolution of stream processing systems," in Proc. 2020 ACM SIGMOD Int. Conf. Manag. Data, 2020, pp. 2651–2658. <https://doi.org/10.1145/3318464.3383131>
- M. Kleppmann, Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems. Sebastopol, CA: O'Reilly Media, 2017.
- J. Mace and R. Fonseca, "Universal context propagation for distributed system instrumentation," in Proc. 13th EuroSys Conf., 2018, Art. 8. <https://doi.org/10.1145/3190508.3190526>
- U. Naseer et al., "Zero downtime release: Disruption-free load balancing of a multi-billion user website," in Proc. Annu. Conf. ACM SIGCOMM, 2020, pp. 529–541. <https://doi.org/10.1145/3387514.3405885>
- Syed, M. H., Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., & Manoharan, P. (2025). Enhancing Retail Sector Customer Engagement Through the Application of Modern and Intelligent AI-Driven Analytical Techniques. *International Journal of Humanities and Information Technology*, 7(01), 19–26.
- Shyam, D., Manoharan, P., Syed, M. H., Ragireddy, U. K., Addepalli, P. V., & Bandaru, S. R. (2025). An AI-Driven Classification System for Early Detection of Customer Churn in E-Commerce Platforms. *Journal of Computer Science and Technology Studies*, 7(11), 452–462.
- Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., Ragireddy, U. K., & Addepalli, P. V. (2025). Customer Sentiment Analysis for Online Shopping Platforms Using NLP and Deep Learning Methods. *Journal of Cyber-Physical Security and Robotics*, 1(02), 1–11.
- Mylavarapu, K. B., Pilli, J., Jannu, P. K., Mohammad, J. A., Panchali, S. H., & Kavirayani, U. M. (2025). An Enhanced Intrusion Detection Architecture Using ML-Based Anomaly Detection on Large-Scale Data. *International Journal of Humanities and Information Technology*, 7(01), 20–26.
- Panchali, S. H., Mylavarapu, K. B., Pilli, J., Jannu, P. K., & Mohammad, J. A. (2025). Review of Secure Communication Systems for Cyber-Physical Systems and Industrial IoT. *Journal Of Engineering And Computer Sciences*, 4(6), 349–362.
- Pilli, J., Jannu, P. K., Mohammad, J. A., Panchali, S. H., Kavirayani, U. M., & Mylavarapu, K. B. (2025). Data-Driven Artificial intelligence (AI)-based System for Identifying Fraudulent Telephony Activities. *Journal of Cyber-Physical Security and Robotics*, 1(02), 23–31.
- Jannu, P. K., Mohammad, J. A., Panchali, S. H., Kavirayani, U. M., Mylavarapu, K. B., & Pilli, J. (2025). A Review of Sensor Data Acquisition Methods for Accurate and Timely Detection of Faults in Mechanical Systems. *Journal of Computer Science and Technology Studies*, 7(9), 796–803.
- Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., & Chilakapati, V. K. (2025). Emerging Communication Technologies for IoT: 5G and Edge Networking Perspectives. *International Journal of Humanities and Information Technology*, 7(01), 19–28.
- Dasari, A., Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., & Pathani, A. K. (2025). Transforming Undergraduate Education Through Modern Pedagogical Innovations. *Journal of Computer Science and Technology Studies*, 7(12), 547–556.
- Pathani, A. K., Dasari, A., Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., & Kavuluri, H. V. R. (2025). Big Data and Cloud Computing Integration: A Review of Scalable Information Retrieval Techniques. *Journal of Cyber-Physical Security and Robotics*, 1(02), 12–22.
- Suresh, A. (2025). AI-Driven Business Intelligence Automation: Integrating Data Engineering, Auto-BI, and Large Language Models. *International Journal of AI, BigData, Computational and Management Studies*, 6(3), 97–108. <https://doi.org/10.63282/3050-9416.IJAIBDCMS-V6I3P112>
- Suresh, A. (2025). Conversational Analytics Using LLMs: Transforming Enterprise Data Consumption through Natural Language Interfaces. *American International Journal of Computer Science and Technology*, 7(5), 92–102. <https://doi.org/10.63282/3117-5481/AIJCST-V7I5P108>
- Syed, M. H., Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., & Manoharan, P. (2026). AI-Powered Data



- Analytics for Strengthening Customer Loyalty and Personalization in Retail. *Journal of Data Analysis and Critical Management*, 2(01), 28-36.
- Manoharan, P., Syed, M. H., Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., & Shyam, D. (2026). A Review of Digital Tools and Technologies for Enhancing Customer Engagement in Financial Services. *Journal of Contemporary Science and Technology Management*, 2(01), 13-25.
- Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., & Syed, M. H. (2026). Predictive Customer Segmentation in Retail Using Machine Learning and Behavioral Analytics. *European Journal of Applied Science, Engineering and Technology*, 4(2), 268-278.
- Suresh, "Architectural Design of LLM- Powered Conversational BI Systems for High-Volume Enterprise Data Warehouses," 2026 International Conference on Machine Learning and Autonomous Systems (ICMLAS), Bangkok, Thailand, 2026, pp. 1333-1341, doi: 10.1109/ICMLAS67792.2026.11483861.
- Taluri, R. (2024). A Hybrid Data Engineering and Generative AI Architecture for Intelligent Data Governance, Metadata Management, and Automated Data Quality Assessment on AWS. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, 5(2), 230-240. <https://doi.org/10.63282/3050-9262.IJAIDSML-V5I2P126>
- Suresh, A. (2024). Auto-BI Frameworks Powered by Generative Artificial Intelligence for Scalable Self-Service Data Analytics in Large Organizations. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, 5(3), 191-201. <https://doi.org/10.63282/3050-9262.IJAIDSML-V5I3P119>
- Taluri, R. (2024). A Cloud-Native Reference Architecture for Data Engineering, Generative AI, and Decision Intelligence Using AWS and Amazon Bedrock. *International Journal of AI, BigData, Computational and Management Studies*, 5(1), 218-227. <https://doi.org/10.63282/3050-9416.IJAIBDCMS-V5I1P122>
- Suresh, A. (2025). AI-Driven Business Intelligence Automation: Integrating Data Engineering, Auto-BI, and Large Language Models. *International Journal of AI, BigData, Computational and Management Studies*, 6(3), 97-108. <https://doi.org/10.63282/3050-9416.IJAIBDCMS-V6I3P112>
- Taluri, R. (2021). Cloud-Native Architectures for Enterprise Financial Data Management, Analytics, and Regulatory Reporting Compliance. *International Journal of Emerging Trends in Computer Science and Information Technology*, 2(2), 101-111. <https://doi.org/10.63282/3050-9246.IJETCSIT-V2I2P112>
- Suresh, A. (2025). Conversational Analytics Using LLMs: Transforming Enterprise Data Consumption through Natural Language Interfaces. *American International Journal of Computer Science and Technology*, 7(5), 92-102. <https://doi.org/10.63282/3117-5481/AIJCST-V7I5P108>
- Taluri, R. (2022). Cloud Data Engineering Strategies for Large-Scale Financial Data Integration and Intelligent Corporate Performance Reporting. *International Journal of Emerging Research in Engineering and Technology*, 3(4), 176-188. <https://doi.org/10.63282/3050-922X.IJERET-V3I4P119>
- Jannu, P. K., Mohammad, J. A., Panchali, S. H., Kavirayani, U. M., Mylavarapu, K. B., & Pilli, J. (2026). Recent Advances in Anomaly Identification for IoT Devices Using Machine and Deep Learning Models. *Journal of Data Analysis and Critical Management*, 2(01), 37-47.
- Mylavarapu, K. B., Pilli, J., Jannu, P. K., Mohammad, J. A., Panchali, S. H., & Kavirayani, U. M. (2026). Early Diagnosis of Heart Conditions Using AI-Driven on Electronic Health Data. *Journal of Contemporary Science and Technology Management*, 2(01), 26-37.
- Pilli, J., Jannu, P. K., Mohammad, J. A., Panchali, S. H., Kavirayani, U. M., & Mylavarapu, K. B. (2026). A Survey on Power Optimization Techniques in Contemporary Mobile SoC Architectures. *European Journal of Applied Science, Engineering and Technology*, 4(2), 321-333.
- Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., & Dasari, A. (2026). A Survey on Generative AI Chatbots in Customer Service: Challenges, Benefits, and Use Cases. *Journal of Data Analysis and Critical Management*, 2(01), 17-27.
- Pathani, A. K., Dasari, A., Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., & Kavuluri, H. V. R. (2026). Transforming Customer Support with AI: Emerging Tools, Trends and Their Service Effectiveness. *Journal of Contemporary Science and Technology Management*, 2(01), 1-12.
- Dasari, A., Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., & Pathani, A. K. (2026). Applications of Machine Learning in Predicting Customer Churn in Telecommunications: A Review Analysis. *European Journal of Applied Science, Engineering and Technology*, 4(2), 257-267.
- Kumar, A., Wadhwa, M., Kalla, D., Konduru, S. C., Nandawat, C., & Sharma, M. (2025, October). Benchmarking the Trade-Offs in Object Detection: Accuracy, Speed, and Energy Efficiency. In *International Conference on Artificial Intelligence and Networking* (pp. 410-422). Cham: Springer Nature Switzerland.
- NR, A. R., Rajasri, T., Praveen, R., Kalla, D., Bendale, S. P., & Venu, N. (2025, April). CAC Training-A Unified Cybersecurity Training Program for Military Staff. In *2025 3rd International Conference on Communication, Security, and Artificial Intelligence (ICCSAI)* (Vol. 3, pp. 569-573). IEEE.
- Kalla, D., Kuraku, D. S., & Samaah, F. (2021). Enhancing cyber security by predicting malwares using supervised machine learning models. *International Journal of Computing and Artificial Intelligence*, 2(2), 55-62.
- Zeeshan, M., Bhadauria, K., Pahal, L., Nagrath, P., & Kalla, D. (2025, June). Ensemble-Based Deep Learning for



- Automated Diabetic-Retinopathy Detection Using CNNs and Transfer Learning. In *International Conference on Data Analytics & Management* (pp. 216-228). Cham: Springer Nature Switzerland.
- Aggarwal, A., Agarwal, L., Rella, B. P. R., Nagpal, N., Kalla, D., & Sharma, M. (2025, June). A Performance Comparison of Machine Learning Models for Rain Prediction. In *International Conference on Data Analytics & Management* (pp. 319-328). Cham: Springer Nature Switzerland.
- Nagrath, P., Saini, I., Zeeshan, M., Komal, Komal, & Kalla, D. (2025, June). Predicting Mental Health Disorders with Variational Autoencoders. In *International Conference on Data Analytics & Management* (pp. 38-51). Cham: Springer Nature Switzerland.
- NR, A. R., Rajasri, T., Praveen, R., Kalla, D., Bendale, S. P., & Venu, N. (2025, April). CAC Training-A Unified Cybersecurity Training Program for Military Staff. In *2025 3rd International Conference on Communication, Security, and Artificial Intelligence (ICCSAI)* (Vol. 3, pp. 569-573). IEEE.
- Kalla, D., Smith, N., Samaah, F., & Polimetla, K. (2024). Hybrid scalable researcher recommendation system using azure data lake analytics. *Journal of Data Analysis and Information Processing*, 12, 76-88.
- Kumar¹, A., Wadhwa¹, M., Kalla, D., Konduru, S. C., Nandawat, C., & Sharma, M. (2026, May). Benchmarking the Trade-Offs in Object Detection: Accuracy, Speed, and Energy. In *Proceedings of International Conference on Artificial Intelligence and Networks: ICAIN 2025, Volume 4* (Vol. 4, p. 410). Springer Nature.
- A. Suresh, "Large Language Models for Explainable and Self-Service Business Intelligence in Large Enterprises," 2026 6th International Conference on Expert Clouds and Applications (ICOECA), Bengaluru, India, 2026, pp. 1571-1578, doi: 10.1109/ICOECA68095.2026.11485058.
- A.Suresh, "A Comprehensive Study on Auto - BI Systems using Generative AI for Scalable and Explainable Enterprise Analytics," 2026 6th International Conference on Expert Clouds and Applications (ICOECA), Bengaluru, India, 2026, pp. 1579-1585, doi: 10.1109/ICOECA68095.2026.11485569.
- Krishnappa, K. H., R, M., & Chavan, L. (2026). ML-Driven Automated Functional Verification Framework for Digital Designs. *Electronics*, 15(12), 2687.
- Kalla, D., & Samaah, F. (2023). Exploring Artificial Intelligence And Data-Driven Techniques For Anomaly Detection In Cloud Security. Available at SSRN 5045491.
- Suresh, "Architectural Design of LLM- Powered Conversational BI Systems for High-Volume Enterprise Data Warehouses," 2026 International Conference on Machine Learning and Autonomous Systems (ICMLAS), Bangkok, Thailand, 2026, pp. 1333-1341, doi: 10.1109/ICMLAS67792.2026.11483861.
- Vadisetty, R., Polamarasetti, A., Varadarajan, V., Kalla, D., & Ramanathan, G. K. (2025, May). Cyber Warfare and AI Agents: Strengthening National Security Against Advanced Persistent Threats (APTs). In *International Conference on Intelligence-Based Transformations of Technology and Business Trends* (pp. 578-587). Cham: Springer Nature Switzerland.
- S. Nedelkoski, J. Cardoso, and O. Kao, "Anomaly detection from system tracing data using multimodal deep learning," in *Proc. IEEE 12th Int. Conf. Cloud Comput. (CLOUD)*, 2019, pp. 179-186. <https://doi.org/10.1109/CLOUD.2019.00038>
- S. Nedelkoski, J. Cardoso, and O. Kao, "Anomaly detection and classification using distributed tracing and deep learning," in *Proc. 19th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, 2019, pp. 241-250. <https://doi.org/10.1109/CCGRID.2019.00038>
- S.A.Noghabi et al., "Samza: Stateful scalable stream processing at LinkedIn," *Proc. VLDB Endow.*, vol. 10, no. 12, pp. 1634-1645, 2017. <https://doi.org/10.14778/3137765.3137770>
- N. Polyzotis, S. Roy, S. E. Whang, and M. Zinkevich, "Data lifecycle challenges in production machine learning: A survey," *ACM SIGMOD Record*, vol. 47, no. 2, pp. 17-28, 2018. <https://doi.org/10.1145/3299887.3299891>
- S. Schelter et al., "Automating large-scale data quality verification," *Proc. VLDB Endow.*, vol. 11, no. 12, pp. 1781-1794, 2018. <https://doi.org/10.14778/3229863.3229867>
- D. Sculley et al., "Hidden technical debt in machine learning systems," in *Adv. Neural Inf. Process. Syst.* 28 (NIPS 2015), 2015, pp. 2503-2511. <https://doi.org/10.5555/2969442.2969519>
- M. Zaharia et al., "Apache Spark: A unified engine for big data processing," *Commun. ACM*, vol. 59, no. 11, pp. 56-65, 2016. <https://doi.org/10.1145/2934664>
- Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, Art. 12, 2019. <https://doi.org/10.1145/3298981>
- X. Zhou et al., "Fault analysis and debugging of microservice systems: Industrial survey, benchmark system, and empirical study," *IEEE Trans. Softw. Eng.*, vol. 47, no. 2, pp. 243-260, 2018. <https://doi.org/10.1109/TSE.2018.2887384>