ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 6 Issue 3 July 2026
Paper Id: JETA-V6I3P107/ Page No: 52-59

Review Article

Autonomous Multi-Cloud Data Pipeline Orchestration Using AI-Driven Observability and Self-Healing ETL Frameworks

Sreenivasa Reddy Vemareddy
Belhaven University, 1500 Peachtree St, Jackson, MS 39202

Received Date: 24 July 2026 Revised Date: 29 July 2026 Accepted Date: 03 August 2026

Abstract: *As new and multiple cloud platforms emerge, each providing different features and capabilities, the data journey is becoming more complex and more important, resulting in an autonomous multi-cloud data pipeline orchestration problem such as container platforms, serverless services, streaming engines, data lakes, and disparate cloud providers host many analytical workloads. This review covers the impact of AI-powered observability and self-healing ETL solutions in adaptive control in these distributed environments. A survey of the existing literature reveals that there is no single established research stream that is sufficient for handling the autonomous multi-cloud ETL orchestration problem, but it is a cross-section of the following streams of research: cloud-native orchestration, distributed stream processing, data quality monitoring, root-cause analysis, anomaly detection and self-adaptive systems. Several reported studies have been conducted in the fields of resource scheduling, pipeline elasticity, container portability, log-based anomaly detection, and quality measurement, but much remains to be done to break all of the restrictions. The major limitations identified include insufficient cross-cloud empirical validation, weak causal reasoning, limited evidence of successful automated remediation, heavy reliance on labelled incident data, and inadequate integration between data observability and orchestration policies. The article concludes by calling for future research to explore the operability of ETL pipelines as adaptive socio-technical systems that are closed in an operational loop with observability, policy, lineage and remediation.*

Keywords: *AI Observability. Autonomous Orchestration, Data Pipelines, Multi-Cloud, Self-Healing ETL*

I. INTRODUCTION

Autonomous multi-cloud data pipeline orchestration is the process of coordinating the execution, monitoring, adaptation and recovery of the ETL workload across multiple cloud environments. It has gained academic relevance thanks to the shift of today's data pipelines from batch scripts to distributed, containerized, streaming, event-driven and service-mediated systems. The cloud-related challenges in application design are focused on heterogeneity, elasticity and provider diversity in the next generation cloud research [1]. The reliability issue is also found to be shifted from provisioning a static infrastructure to coordinating dynamically with orchestration of containers, decomposition of services, continuous delivery and modular deployment. Scheduling is necessary but insufficient for effective ETL orchestration in that environment except to perform compute, do data checks, evolve schema, handle dependencies and recover from incidents.

This problem has been exacerbated by the advent of the container platforms and serverless services, which reduce the scale of deployment units, makes them ephemeral, and makes them more mobile between cloud boundaries. The container research literature highlights portability and resource isolation as key benefits of containerization, while complexity in monitoring, networking, state management and scheduling under a heterogeneous workload is noted to be a challenge [3]. The other tension in serverless studies is that though function level elasticity helps with operational elasticity, it also increases the complexity of orchestration of pipelines, opacity, cold-start variability, lack of provider control, and limited state control [4]. As a result, autonomous multi-cloud ETL is an interdisciplinary research area that connects several areas such as cloud computing, data engineering, software reliability, machine learning operations and autonomic computing.

The research literature is somewhat fragmented, and although industry interest has been growing in “data observability” and “self-healing pipelines”, the research is still somewhat disjointed. Studies on scheduling are based either on cost, delay or the objectives of satisfying deadlines without considering semantic data quality. Research on data quality is typically around data profiling, rule checking, completeness and consistency checks, monitoring and detection; it fails to typically connect detected quality events with automated correction actions by multiple providers. Most AI techniques applied to logs, metrics, traces, and time-series signals rely on stable telemetry schemas, labelled incident data, or customized microservice environments. While the literature on self-adaptive systems gives a vocabulary for a control loop, there is not much empirical evidence of autonomous ETL repair.

The objective of this review is to critically evaluate the peer reviewed journal articles related to autonomous multi-cloud data pipeline orchestration using AI-powered frameworks with observability and self-healing ETL capabilities. This

review does not claim the existence of a mature and unified body of literature. Instead, it is an evaluation of the contribution of neighboring, but technically necessary research areas to and their restriction of the suggested research area. The review focuses on orchestration mechanisms, observability inputs, anomaly detection via machine-learning, data quality checking and remediation logic, portability and cross-cloud validation. The common theme is that future advances will increasingly depend on architectural approaches that integrate pipeline lineage and data quality evidence with pipeline telemetry into safe, explainable, and policy-aware recovery actions rather than on another isolated scheduling or anomaly-detection mechanism.

II. LITERATURE REVIEW

There are three interrelated groups of literature that are relevant to autonomous multi-cloud data pipeline orchestration: cloud-native orchestration, data pipeline reliability, and AI-driven operational intelligence. The earliest technical base is in research on resource scheduling. Singh and Chana demonstrate that cost, utilization, makespan, energy and deadline constraints are key factors that have been emphasized in the research of cloud scheduling [5]. These types of goals are still applicable for multi-cloud ETL pipelines, but pipeline workloads introduce additional dependencies that aren't fully represented by generic scheduling taxonomies. Even if compute allocation is efficient, due to an error in the transformation step or ingestion of the source data with a delayed or corrupted schema, the issue can result in downstream workloads being invalidated. Similarly, distributed stream-processing surveys indicate that frameworks like Spark Streaming, Flink-like engines, Storm-like architecture and Kafka-centred architectures can meet the requirements for latency, state, throughput and fault tolerance [6]. However, reliability is a common way of comparing stream-processing capabilities as a checkpointing/recovery rather than as an end-to-end pipeline between providers.

The other concepts that fall into this category are linked to data lakes, data quality and observability foundations. Giebler et al. describe data-lake architectures in which the interaction with metadata, governance and ingestion control is crucial [7]. Firmani et al. suggest that big-data quality cannot be assessed solely in terms of accuracy, but volume, heterogeneity, timeliness and fitness-for-use changes the data quality assessment [8]. Cai and Zhu also claim that the quality assessment is a multi-dimensional problem in the context of big data, covering the dimensions of completeness, consistency, timeliness, credibility and accessibility [9]. Ehrlinger and Wöß provide a review from a tool-centric point of view, arguing that the tools used to measure and monitor the quality of data can be used to profile and evaluate the rules and that autonomous data repair is poorly represented in the tools that are available [10]. All these studies indicate a need for more than pipeline task retry in order to achieve self-healing ETL. It demands semantic observability: the ability to determine if data continues to be reliable and up-to-date, and if they meet the requirements of the downstream analytical contract.

Table 1: Summary of Key Findings

Ref	Focus	Key Findings
[5]	Cloud resource scheduling	Scheduling objectives balance cost, time, energy, and QoS, but semantic data dependencies receive limited attention.
[6]	Distributed data stream processing	Stream engines support low-latency execution and fault tolerance, but multi-cloud portability and observability integration remain secondary.
[7]	Data lake architecture	Metadata and governance are central to pipeline reliability, especially when raw and curated data zones coexist.
[8]	Big data quality meaning	Quality must be evaluated through context-sensitive dimensions rather than narrow accuracy metrics.
[9]	Big data quality assessment	Completeness, consistency, timeliness, and credibility create measurable constraints for ETL monitoring.
[10]	Data quality tools	Profiling and rule-based monitoring are common, while automated remediation support remains limited.
[11]	Deep anomaly detection	Deep models capture complex deviations but face interpretability, drift, and label scarcity constraints.
[12]	Time-series anomaly detection	Algorithm selection depends on seasonality, sampling, drift, and ground-truth availability.
[13]	Cloud anomaly detection and RCA	Logs, metrics, traces, and dependency graphs are necessary for meaningful fault localization.

The third cluster is on observability and failure intelligence using AI. While deep anomaly detection surveys demonstrate the ability of neural techniques to detect more complex deviations of high-dimensional operational signals, it has been found that they are vulnerable to distribution shift, limited labels, unclear thresholds, and poor interpretability [11]. This restriction is supported by the literature on time-series anomaly detection, which has clearly shown that the

effectiveness of algorithms varies greatly based on seasonality, drift, noise, sampling rate and the quality of the ground truth data [12]. For cloud applications, Soldani and Brogi find that logs, metrics, traces, dependency graphs and topology information are the most important components for detecting anomalies and root causes in microservice-based applications [13]. This is true for ETL orchestration, because the health of a pipeline is related to other pipeline events such as an ingestion lag, a container crash, a schema mismatch, or an event of warehouse throttling that can happen on any of the pipeline's telemetry planes. Autonomous remediation requires correlating data-lineage and workflow-dependency data with operational symptoms.

Operational context is provided by deployment and self-adaption studies. Leitner et al. show how different pricing schemes and workloads of cloud providers affect cost management of microservice-based cloud applications deployed [14]. microservices but cross service architecture has been mainly directed towards deployment, scalability and maintainability of microservices, but monitoring and microservices runtime governance are still not well addressed, as shown in [15]. Weyns et al. review the scientific evidence regarding claims in self-adaptive systems and provide many examples of falsified adaptive mechanisms in realistic environments [16]. In the context of autonomous (multi-cloud) ETL, it's not only about being recovered correctly, but also about being recovered correctly and without cascading errors, adhering to policies and being able to roll back the remediation process.

Field is not monolithic in terms of its method of operation as seen in Table 1. Instead, autonomous multi-cloud ETL is a composite research area: scheduling optimizes ETL execution, stream-processing frameworks manage continuous work, correctness comes from quality studies, signal interpretation occurs with anomaly detection, and control-loop logic comes from self-adaptive systems. What is missing is a robust empirical study that together combines these three elements in closed loop and cross cloud ETL process and has proven repair actions.

III. METHODOLOGY

There , AI diagnosis, and policy-governed remediation. This is represented in the form of a closed operational pattern in Figure 1, not as a generic pipeline diagram.

Table 2: Method Comparison

Ref	Method	Strengths	Limitations
[5]	Cloud scheduling taxonomy	Clarifies optimization objectives for workload placement.	Limited treatment of data correctness and lineage.
[6]	Framework comparison for stream processing	Captures latency, state, throughput, and recovery semantics.	Multi-cloud control is not the central evaluation basis.
[8]	Conceptual analysis of big data quality	Broadens quality beyond accuracy.	Does not specify autonomous repair mechanisms.
[10]	Tool review for data quality monitoring	Identifies measurable observability functions.	Repair remains rule-bound or manual in many tools.
[11]	Deep anomaly detection review	Covers high-dimensional detection methods.	Interpretability and drift remain persistent constraints.
[12]	Time-series anomaly detection review	Connects algorithm choice to signal properties.	Benchmark transfer to cloud ETL remains limited.
[13]	Survey of anomaly detection and RCA	Links telemetry types to cloud fault localization.	Empirical grounding often remains service-specific.
[16]	Evidence review for self-adaptive systems	Supplies control-loop logic for self-healing.	Evidence strength varies across realistic deployments.

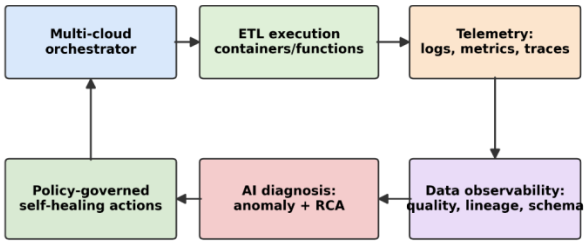

Figure 1: Closed-Loop Architecture for Autonomous Multi-Cloud Self-Healing ETL

As illustrated in Figure 1, distributed ETL execution is surrounded by orchestration, telemetry, semantic data observability, AI diagnosis, and policy-governed remediation within a provider-aware control loop. It's because it's not only retrying but the need to be data-lineage aware and causal to diagnose and act based on the policy. The concept of cloud orchestration, stream-processing, data quality, anomaly detection, root cause analysis and self-adaptive systems were adopted from literature [1]–[6, 8]–[13, 16].

IV. DISCUSSION

The most significant change, as highlighted in the reviewed studies, is from infrastructure-centred orchestration to runtime control, which is driven by telemetry. There has been previous work that focused on both choosing adequate resources for achieving specific goals within a time and cost boundary, and choosing the quality of service and utilization bound resource sets [5]. This is still a key requirement for multi-cloud ETL as ETL jobs may be CPU intensive, require more memory on the ETL machine, be stateful and/or have different costs for transporting the data. However, all scheduling taxonomies reported above do not take into account all kinds of data failures. While task completion is the only measure of success for ETL, additional factors to consider are schema compatibility, quality limits, lineage integrity, freshness and downstream reproducibility. Therefore, one of the most important research gaps exists in the literature between computational scheduling and the semantic data reliability.

To narrow the gap in stream-processing studies, the properties of execution of pipelines near real-time pipelines are taken advantage of. The differences between the different distributed stream-processing frameworks, in terms of processing model, fault tolerance, state management and latency control, are presented by Isah et al. [6]. The intent of such dimensions is that they are directly correlated with the autonomous ETL jobs – checkpointing, replay, event-time processing, and backpressure have an impact on whether or not a self-healing framework can recover without duplicating output or losing any data. However, the reported comparisons are not sufficient to solve the multi-cloud orchestration problem. A framework may provide a powerful checkpointing service for provider-controlled clusters, but there are still a lot of questions regarding how providers should be inter-connecting, how identities are to be managed, storage semantics, storage replication, failover management and so on. Autonomous multi-cloud self-healing extends beyond stream-engine fault tolerance.

The other trend that data quality studies bring is infrastructure telemetry should be included in the pipeline observability, alongside the data quality semantic signals. As shown by Firmani et al. and Cai and Zhu [8, 9] big data quality is multidimensional and dependent on the context. The finding will impact AI-based observability. Many reasons can trigger an operational incident such as an increase in CPU load, restart of a container, schema drift identified by the infrastructure monitoring system, missing required field, late arriving event or invalid transformation. Ehrlinger and Wöß also show that full automation of the data repair is not possible using data quality monitoring tools and that these can be used for profiling and measurement as well [10]. Literature thus substantiates the difference between the process of detection and healing. There are many systems that will be able to identify invalid, corrupted, or suspicious data, but few systems that will be able to come up with a remedy, defend it, implement it and review it.

The strengths of AI's anomaly detection capabilities contribute to predictive power yet come with methodological weaknesses. High dimensional and nonlinear patterns are reported beneficial in Deep anomaly detection analysis [11]. Studies in the field of anomaly detection with time series data show that the reliability of the detection depends on seasonality, a drift, noise, sampling and on the availability of the ground truth [12]. The results are important for multi-cloud ETL because there are differences between the distributions of the various telemetry providers, regions, workloads and data domains. After modifying the ingestion pattern and deploying a detector in the new provider, it is possible that it will generate excessive alerts and/or miss incidents based on a fixed metric on the specific provider. In this context, observability involves more than model selection; it requires robust feature engineering, lineage-aware context, threshold governance, drift monitoring, and interpretable drift analysis.

Some of the current limitations in the literature exist when it comes to research on Root Cause Analysis (RCA) as a key step between observability and self-healing. Multiple telemetry sources like logs, metrics, traces and dependency information are needed to effectively detect anomalies and analyse their root causes in cloud applications, as shown by the authors of [13] and suggested by Soldani and Brogi. With ETL pipelines this need is compounded as errors get passed through service and data dependencies. A remediation engine that doesn't have lineage and topology can re-start a corrupted load or an incorrect service or mask a data quality defect. The literature surveyed for this study, therefore, suggests that initiating healing from alerts is not the right approach, but graph-informed remediation.

Economic and architectural coupling are restricting the multi-cloud autonomy. This is supported by deployment cost studies, as well as architecture studies. Unfortunately cloud service providers do not have identical deployment costs, as Leitner et al. have shown, depending on the workload [14]. Unexpected costs of self-healing in ETL pipelines include cross-

cloud egress, duplicate computations, larger size of checkpoint storage and emergency overprovisioning. Some of the major research challenges identified by Di Francesco et al. for microservice architecture (MSA) are scalability and maintainability and runtime governance is still a challenge [15]. This is crucial because microservices are hard to coordinate in the event of failure and this complexity is generally expressed in an autonomous ETL. One aspect of the improvement may work against the others, such as by increasing queue pressure, impacting consistency or going over budget of the provider.

The literature on self-adaptive systems is a salutary reminder of the bold promises of automation. Weyns et al. show that self-adaptation evidence does not always have a high level of strength [16]. This is particularly relevant for self-healing ETL, as if there is success in the small testbed, but not in multi-cloud pipelines, then it's not a sign of success. Consider fault injection, data corruption situations, schema drift, delayed sources, provider-throttling, network-partitioning, incomplete lineage and rollback-correctness. While there is a wealth of concepts in the existing literature, there is little evidence of end-to-end, closed-loop self-healing ETL in realistic multi-cloud environments.

Table 3: Results Comparison

Ref	System / Context	Metric / Basis of Comparison	Outcome
[5]	Cloud workload scheduling	Cost, utilization, deadline, QoS objectives	Optimization criteria are mature, but data validity is external to most models.
[6]	Distributed stream processing	Latency, throughput, state, fault tolerance	Execution recovery is supported, but cross-cloud policy and lineage remain limited.
[7]	Data lake environments	Metadata, governance, ingestion control	Pipeline reliability depends on metadata quality and zone discipline.
[10]	Data quality monitoring tools	Profiling, rule checks, quality dimensions	Detection functions are stronger than autonomous remediation functions.
[11]	Deep anomaly detection	High-dimensional anomaly learning	Strong pattern detection is offset by interpretability and drift limitations.
[12]	Time-series anomaly detection	Seasonality, drift, labels, signal noise	Algorithm transfer is sensitive to telemetry context and ground-truth quality.
[13]	Cloud application RCA	Logs, metrics, traces, dependency graphs	Multi-source observability improves localization but remains hard to generalize.
[16]	Self-adaptive systems	Evidence for adaptive claims	Closed-loop claims require stronger empirical validation in realistic settings.

The results from the literature review are summarized in an evidence-distribution view in the areas of capabilities required for the autonomous multi-cloud ETL (see figure 2). It is important because there is a visible imbalance, that is, there is more evidence for orchestration, data quality and anomaly detection individually than when integrated into a single system that is self-healing across providers. As seen in Figure 2, the articles analysed have been grouped according to the primary capability area they make a contribution to autonomous multi-cloud ETL. As seen in the figure, there is less evidence of integration than there is of evidence at the component level, especially with closed-loop remediation. Journal articles that have been peer-reviewed and reviewed manually [1]–[16].

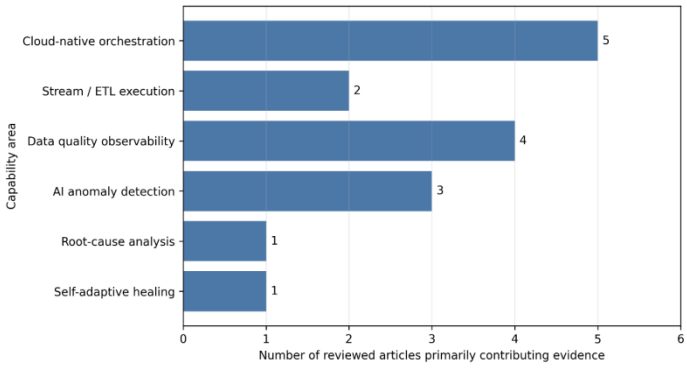

Figure 2: Evidence Distribution Across Autonomous Multi-Cloud ETL Capability Areas

In Figure 3, the ability of three architectures is compared, namely the traditional ETL automation, cloud-native orchestration and AI-driven self-healing ETL. It is important because it shows that the proposed topic needs to be a balanced level of maturity not only in terms of portability, observability, diagnosis, and safe remediation but also from a technical

standpoint as well. The relative capability maturity of these three technologies (rule-based ETL, cloud-native orchestration and AI-powered self-healing ETL) is compared along five dimensions in Figure 3. The graphic highlights the need for the orchestration control, semantic observability and adaptive remediation principles to work together to enable autonomous multi-cloud data pipeline orchestration. From the literature reviewed and interpreted, the scheduling, containers, serverless computing, data quality monitoring, anomaly detection, root-cause analysis, and self-adaptation are explored in this paper. The reviewed and interpreted literature includes scheduling [1]–[8], containers [9, 10], serverless computing [11, 12], data quality monitoring [13, 14], anomaly detection [15], root-cause analysis [16] and self-adaptation [1]–[16].

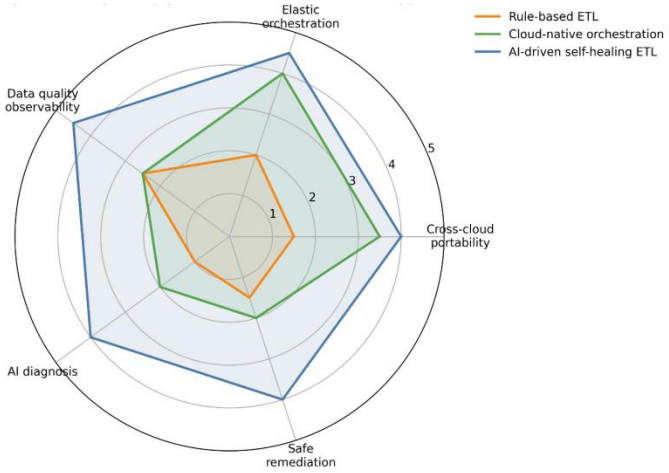

Figure 3. Comparative Capability Profile of ETL Orchestration Approaches

The results combined show that there are four outcomes to measure autonomous multi-cloud ETL. First, it's about execution, being efficient, and that's not enough. If it's not cheap, it's not possible, or possible but that will lead to stale or corrupted data. Secondly, observability needs to be built upon infrastructure telemetry and semantic data signals. Operational symptoms are found in logs, metrics and traces; Data quality checks, schema monitors, lineage graphs and freshness indicators discover data analytical correctness. Third, the self-healing should be policy controlled. Data contracts, budget rules, compliance limits and human escalation limits should limit the use of automated redo, reroute, replay, quarantine, rollback, compensation, or scaling. Fourth, cross cloud validation is required. While a method that manages to work in one Kubernetes cluster or one managed stream service does not necessarily extend to multi-cloud ETL, there are differences in identity systems, storage consistency, network latency, failure semantics and billing models across different providers.

The common issue in literature between the two is that autonomy is sometimes referred to as a technical control challenge, and ETL self-healing can also be considered a data governance challenge. Classifying anomalies and triggering actions on them can be done with machine learning and orchestrators but none of these guarantees that logs are still "analytically valid" if they have been fixed. This is a contradiction which is the reason why many promising techniques are still partial. The combination of these to repeatable multi-cloud experiments is rarely seen in peer-reviewed papers, while scheduling papers optimize placement, data quality papers specify dimensions, anomaly papers improve detection and self-adaptive systems papers propose control loops. The most significant missing research component is then the empirical integration – validated frameworks that tie together telemetry, lineage, quality rules, causal diagnosis and controlled remediation in realistic cross-cloud failures.

V. FUTURE DIRECTIONS

Current work on autonomous multi-cloud data pipeline orchestration research is directed towards integrated evaluation environments with the integration of multiple data quality faults, data pipeline lineage, data pipeline telemetry, and provider heterogeneity with the execution of the data pipeline. While there are some good points for reference provided by the current scheduling and stream processing approaches, the future will introduce new metrics such as schema drift, late-arriving data, duplicate data events, corrupted transformations, provider throttling, network partitioning, and cross-cloud recovery cost [5,6,60]. These benchmarks would enable operational correctness as well as semantic correctness to be measured against self-healing ETL.

The second major priority is causal observability. While existing anomaly detection can be applicable, it can only detect the deviation without knowing if the deviation is caused by an upstream source, transformation logic, orchestration failure, storage inconsistency or downstream load constraint [11]–[13]. New frameworks must include dependency graphs,

data-lineage graphs, service topology and temporal telemetry to allow for causal ranking for remediation. This is important because unsafe automated repair can cause data corruption that can be worsened by this direction.

The third priority is the policy-aware remediation. Self-healing ETL should not be synonymous with automated retry mechanisms. Remediation should be based on clear policies using data contracts and privacy boundaries, as well as the cloud data-egress and migration costs, SLAs, audit requirements, and the opportunity to roll back. The self-adaptive systems research [16] provides a theoretical underpinning for such control loops, but in real data-engineering contexts there is a need to provide further evidence.

Finally, but by no means least, there is the human oversight and governance that can be explained. Multi-cloud ETL pipelines can be used for various applications, such as financial reporting, clinical analytics, security monitoring, and operational decision-making. It is therefore crucial that automated processes can keep a record of why the repair is being undertaken, confirm the incident and offer escalation paths when deciding on repair options. Interdisciplinary efforts between the cloud systems community and data governance, reliability engineering, machine learning operations and others could lead to measurable, safe, and accountable autonomous systems, and not just self-sustaining systems.

VI. CONCLUSION

Orchestrating data pipelines using AI-driven autonomous multi-cloud is a new research area that merges a number of well-established, but poorly connected, literatures. The following technical foundations are all important: cloud-native orchestration, distributed stream processing, data quality monitoring, anomaly detection, root-cause analysis and self-adaptive systems. From the literature reviewed one of the best take-home messages is that pipeline autonomy is not solely dependent on execution efficiency. It has to be continually interpreted with respect to infrastructure telemetry, semantic data quality, lineage, dependency structure, and policy constraints.

Persistent gaps remain. Currently, there is limited evidence of end-to-end self-healing ETL across multiple cloud vendors, and most of the work that has been completed has been on isolated components like scheduling, monitoring or detection. Automated remediation is particularly immature as it needs to reason about the causes, to ensure that the data is correct, to be aware of costs, to consider compliance requirements, and to take reversible action if the data is to be recovered. This field's outcome is dependent on making safe adaptation decisions that are grounded in observability evidence by using integrated frameworks. To achieve scholarly progress, realistic benchmarks, fault-injection studies, clear evaluation measures, and improved correlations of cloud reliability and data governance are necessary. The importance is that reliable analytical infrastructure relies on pipelines which are needed to remain correct and available in the multi-cloud environment where the clouds are heterogeneous, dynamic and failure-prone.

- **Interest Conflicts:** The author declares that there is no conflict of interest concerning the publishing of this paper.

VII. REFERENCES

[1] Varghese, B., & Buyya, R. (2018). Next generation cloud computing: New trends and research directions. Future Generation Computer Systems, 79, 849–861.

[2] Kratzke, N., & Quint, P.-C. (2017). Understanding cloud-native applications after 10 years of cloud computing—A systematic mapping study. Journal of Systems and Software, 126, 1–16.

[3] Pahl, C., Brogi, A., Soldani, J., & Jamshidi, P. (2019). Cloud container technologies: A state-of-the-art review. IEEE Transactions on Cloud Computing, 7(3), 677–692.

[4] Hassan, H. B., Barakat, S. A., & Sarhan, Q. I. (2021). Survey on serverless computing. Journal of Cloud Computing, 10(1), Article 39.

[5] Singh, S., & Chana, I. (2016). A survey on resource scheduling in cloud computing: Issues and challenges. Journal of Grid Computing, 14(2), 217–264.

[6] Isah, H., Abughofa, T., Mahfuz, S., Ajerla, D., Zulkernine, F., & Khan, S. (2019). A survey of distributed data stream processing frameworks. IEEE Access, 7, 154300–154316.

[7] Giebler, C., Gröger, C., Hoos, E., Schwarz, H., & Mitschang, B. (2019). Leveraging the data lake: Current state and challenges. Big Data Research, 16, 1–12.

[8] Firmani, D., Scannapieco, M., Schirru, R., & Tosco, L. (2016). On the meaningfulness of “big data quality”. Data Science and Engineering, 1(1), 6–20.

[9] Cai, L., & Zhu, Y. (2015). The challenges of data quality and data quality assessment in the big data era. Data Science Journal, 14, Article 2.

[10] Ehrlinger, L., & Wöß, W. (2022). A survey of data quality measurement and monitoring tools. Frontiers in Big Data, 5, Article 850611.

[11] Pang, G., Shen, C., Cao, L., & van den Hengel, A. (2021). Deep learning for anomaly detection: A review. ACM Computing Surveys, 54(2), Article 38.

[12] Blázquez-García, A., Conde, A., Mori, U., & Lozano, J. A. (2021). A review on outlier/anomaly detection in time series data. ACM Computing Surveys, 54(3), Article 56.

[13] Soldani, J., & Brogi, A. (2022). Anomaly detection and failure root cause analysis in cloud applications: A survey. ACM Computing Surveys, 55(3), Article 61.
[14] Leitner, P., Cito, J., & Stöckli, E. (2016). Modelling and managing deployment costs of microservice-based cloud applications. Journal of Internet Services and Applications, 7, Article 11.
[15] Di Francesco, P., Lago, P., & Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. Journal of Systems and Software, 150, 77–97.
[16] Weyns, D., Iftikhar, M. U., Malek, S., & Andersson, J. (2016). Claims and supporting evidence for self-adaptive systems: A literature study. ACM Transactions on Autonomous and Adaptive Systems, 11(4), Article 24.