International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

Agentic Data Pipelines: Autonomous ELT Orchestration Using AI Agents on Microsoft Fabric and Databricks

Narendra Mangala

Data Engineer Manager, USA

mangalanarendra2@gmail.com

ABSTRACT: ELT workflows orchestrated by modern cloud services can be automated using agent-based frameworks that create and execute the required solutions. Agents working in a multi-agent fashion allow for tools such as Microsoft Fabric and Azure Data Factory to be used in novel ways. An evaluation of such an architecture, focused primarily on the agent-based approach within the Microsoft ecosystem, demonstrated that a complete pipeline could be processed as a single task on Databricks. Each agent's specialization influenced not just the chosen tools but also the orchestration pattern. The decisions taken at these levels were logged as text and compared with human counterparts, providing insight about the explainability of the different patterns. Autonomous orchestration of ELT solutions using Microsoft Fabric Data Factory has proven successful and is paving the way toward an unassisted data engineering process. Support for these workers through user-defined tasks might be useful for more complex cases.

Many products and services within the Microsoft ecosystem enable the convergence of data science and engineering with the emergence of Microsoft Fabric and Azure Data Factory. Although Microsoft Fabric contains most of the required building blocks, their connections still need to be defined. AI agents capable of completing ELT tasks can provide valuable support for the unification of multiple services, enabling novel uses while freeing human operators from low-level activities. Microsoft Fabric Data Factory, allowing the integration of multiple services, is used as a Data Engineering Service. A Data Engine running on Databricks is also used to take advantage of the Unity Catalog and Delta Lake. Data Factory handles data ingestion either through Copy activities or by calling other services.

KEYWORDS: Agentic Data Pipelines,Autonomous ELT Orchestration,AI-Driven Data Engineering,Intelligent Data Pipelines,Microsoft Fabric Data Platform,Databricks Lakehouse Architecture,AI Agents in Data Workflows,Self-Healing Data Pipelines,Automated Data Integration,Scalable ELT Automation.

I. INTRODUCTION

With data engineering demand outpacing workforce supply, strategies for autonomous ELT data pipeline orchestration using AI agents were investigated. The objective was to create data pipelines that automatically detect data availability in sources, generate the code required to ingest the data, and execute the code with no human intervention. ELT methods involve first loading data into data lakes or warehouses before transformation and support greater accuracy, stability and scalability. Data pipelines typically execute in response to simple Pull or Push events. However, ELT data pipelines often run daily, weekly or monthly, and still require substantial manual intervention to start and validate data ingestion jobs. Operation latency may increase as DAGs grow larger with more sources, and data freshness may also be compromised. In many scenarios, everything seems to be working, but actually it's not—a sentiment common among Data Engineers.

The proposed novel orchestration architecture delivers an abstract view of ELT data-pipelines that supports low-code deployment and combines the benefits of Push and Pull triggers. Data sources become Agents that autonomously analyse the data in the source and notify a Control Agent when new data is available. Any Consolidation Agent monitors the storage in its corresponding Data Lake/ Warehouse/ LakeHouse and can instruct the relevant Transformation Agent when there is sufficient data to process. Finally, Loading Agents control when data is loaded into the final target. The solution was implemented using Microsoft Azure's data ecosystem and ChatGPT's Data Engineer playing the role of Control Agent. Azure Data Factory was tested as the Data-Orchestration tool and the solution used Fabric Unification and Data Governance capabilities to demonstrate that security, compliance and risk management are not neglected when Agents take on Data Engineering responsibilities.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

1.1. Scope and Objective

A multitude of ELT data pipelines oriented around a wide variety of data engineering tasks is essential to extract value from data in any modern organization. The orchestration of the concurrent execution of such a complex distributed ELT system, especially while maintaining control over data quality and compliance, represents both a technical and an operational difficulty. A novel orchestration architecture based on the concurrent and simultaneous operation of a set of artificial intelligence agents is proposed for the autonomous orchestration of such a complex, data-centric ELT data pipeline framework.

AI agents can be considered an emerging paradigm in data engineering that allows for a wide range of processes within data ecosystems to be autonomously handled without human intervention. These systems are typically composed of a multitude of agents that collaborate, negotiate, act, and communicate in a semi-autonomous manner to fulfill a predefined goal.

Although all data pipelines are similar at a high level, the details of their operation require careful adaptation depending on the task at hand. For example, the ingestion of data from various cloud sources into a cloud data lake is vastly different from the deletion of stale data or the enforcement of data quality rules. The present research discusses the three key areas of the Microsoft Fabric Data Factory orchestration of AI agents for autonomous ELT orchestration.

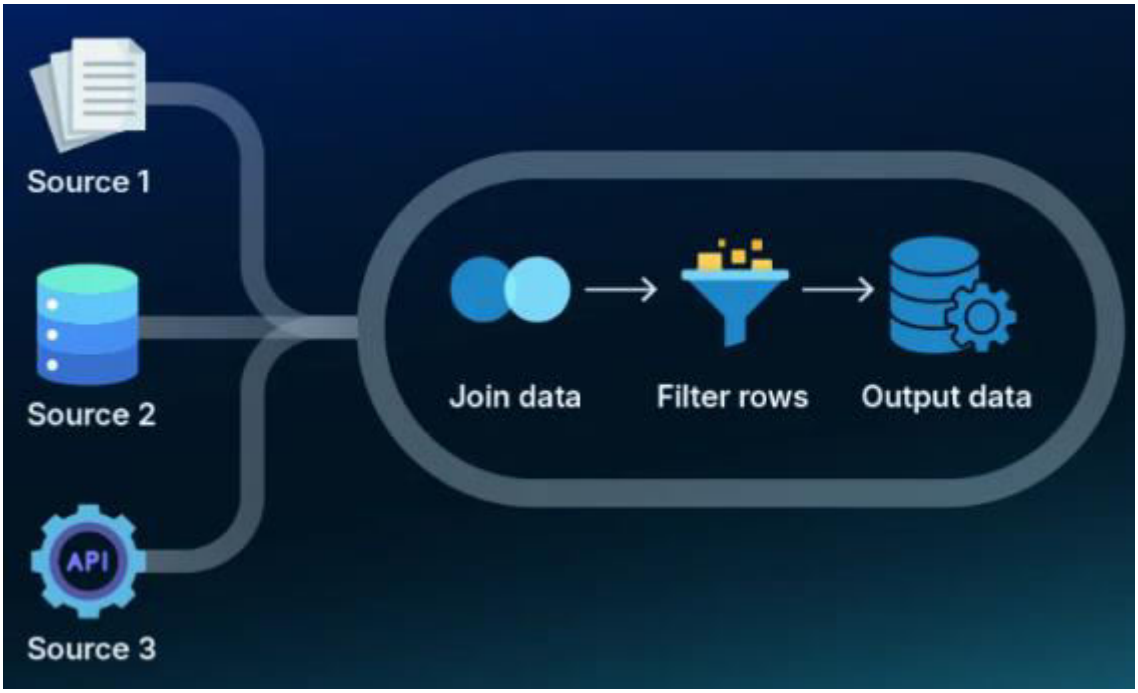


Fig 1: Agentic Data Pipelines

1.2. Research design

Autonomous ELT orchestration on top of Microsoft Fabric Data Factory and Databricks Delta Lake has been achieved through the autonomous use of AI agents. A novel approach to building a Pipelines-as-Code framework was developed, designed as an ensemble of microservices operating on a Data Mesh Architecture. Code was developed for the creation of both agent FastAPI Services as well as for Integration and RiteAgent categories; the former acting as Execution Agents for Action Category requests and the latter two each performing WrightAgent functions for a single level on action directive and action Subject respectively. The Databricks component is a complete design and a Delta Lake along with Unity Catalog were provisioned and configured to demonstrate the Pipelines-as-Code concept. Together the two groups built a composite agent This agent architecture built for this run is new and hence has not been seen in earlier work.

Results from the testing were analysed, showing the Microsoft Fabric Data Factory component was quite capable of forming the ELT Orchestration component's Pipelines-as-Code IDEAL component and the Agent was able to complete

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

several synthetic data engineering tasks rapidly in days rather than weeks. Evidence of the completed Data Engineering task creation, its properties as well as User Acceptance Testing against the resulting Fabric Data Factory Pipelines-as-Code IDEAL are now shown. It can also be confirmed that a scalable Data Governance Repository for added Visibility, Lineage, Discovery and Security features along the Data Mesh is now fully functional with Databricks Unity Catalog and Delta Lake. Schemas and lineage definitions are accurate and enforced; data can be ingested and transformed into Delta Lakes with consistent guarantees; data consumed from the Databricks Delta Lake is stable.

Equation 1: Source availability detection
Derivation
Define:

$$A_i(t) = \begin{cases} 1, & \text{if new data is detected at source } i \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

Why this follows
1. A source is either found to have fresh data or not.
2. The paper describes event-style triggering from source monitoring.
3. So the cleanest mathematical abstraction is a binary indicator.

Final Equation (1)

$$\boxed{A_i(t) = \begin{cases} 1, & \text{new data available} \\ 0, & \text{no new data} \end{cases}}$$

II. BACKGROUND AND MOTIVATION

Orchestration can be defined as defining, controlling, and managing complex workflows that require the coordinated execution of multiple data operations. Different data ecosystem architectures require different kinds of orchestration, and the specific capabilities of a particular tool often dictate whether it is a zipper or a batch pipeline. Rapidly evolving cloud-based data ecosystem architectures are based on decentralized ELT paradigms that combine a high-degree of parallelism with locality of data transformation and aggregation. In this context, integrated orchestration provided by system-embedded data factories exploits native ELT feature for maximum efficiency but necessarily limits the set of workflows it can support. Microsoft Fabric, Databricks, and similar data platforms provide facilities to model and orchestrate autonomous data pipelines leveraging the full spectrum of data processing engines within the system. However, the activity-based nature of their orchestration relies on the human data engineer more than the infrastructure and remains a bottleneck for speed of data availability.

The growing use of generative AI in IT automation questions the need for human data engineering, opening the door for autonomous data utilization. Agentic architecture is particularly suited for orchestrating long-running workflows composed of multiple independent data operations. Different types of integrated agent architectures, capable of interacting with a chosen set of required ecosystem tools, have been developed to explore this potential. Agent personas covering different aspects of the data acquisition and deployment life cycle have been defined, and their interactions studied. All these experimental efforts share the goal of orchestrating (possibly ad-hoc) data pipeline workloads – part of data factory scripting but less than a full replacement – without human intervention.

2.1. ELT Paradigms and Orchestration

Extant solutions for modern data-engineering needs within existing technology ecosystems rely on clear definitions to decompose agentive-path generation and ELT orchestration into suitable abstracted levels. The key components in data infrastructures for the ELT process, ELT and pipeline orchestration concepts, and the characteristics of the entire problem space are reviewed.

ELT refers to the process of extracting, transforming, and loading data. Compared to traditional ETL (extract-transform-load) paradigms, ELT shifts the data transformation step after loading because the data structures in data lakes and cloud data warehouses like Microsoft Fabric or Snowflake are decoupled both logically and physically. That is, data engineers do not need to specify the final-target data structures when ingesting the source data in a data lake or a data warehouse yet can still access and directly analyze the raw data. This characteristic enables nontechnical business users to use BI tools more effectively to build their own dashboards and reports without relying heavily on data engineers to transform the data for them during the data ingestion stage. All kinds of data can now be ingested in

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

bulk, as the data structures do not have to be specified and designed any more within the source systems such as traditional data marts. ELT helps alleviate the bottleneck of data preparation for analytical tasks, as the overhead of transforming the data during ingestion only has to be incurred once.

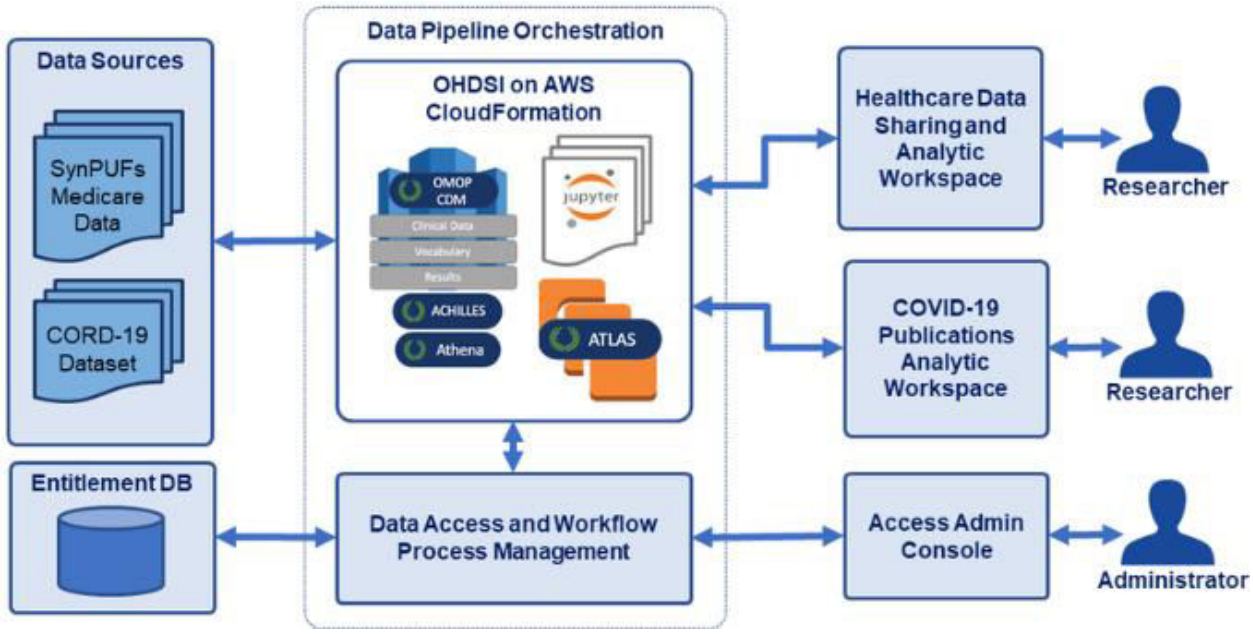


Fig 2: ELT Paradigms and Orchestration

2.2. AI Agents in Data Engineering

Researchers have started to explore the use of specialized AI agents in data engineering activities. These efforts focus primarily on contributing towards periods of high latency, such as data ingestion and transformation, and have leveraged collaborative reinforcement and imitation learning. However, even if they each focus on one aspect of data pipeline orchestration, the three paradigms are quite distinct in their design. SES-based agents take on data engineering tasks independently through an act-orientated approach, require guidance in tool selection and are dependent on a tool-using controller in the absence of externally-encoded solution constraints. The institution of a tool-use capability is an initial step toward achieving a full SES agent. Yet, the orchestration of a wide range of data engineering activities remains an open area for exploration. Moreover, while existing work reports on special-purpose prototypes, little attention has been given to answering the broader question of orchestration. When and why should synthetic agents affect a full data engineering cycle? This launches the first query: Following SES since this embodies the simplest open-world agent model capable of addressing classical data engineering problems.

The tools and patterns selected for use by an agent should therefore reflect the details of the administrative problem at hand. Depending on the timing of the action, these choices thus require specifying a formal acting policy or a set of meta-rules. The second query—the kind of data engineering activity itself—is harder to address, as pipelines can entangle several types of tasks requiring considerably disparate efforts, from long periods of passive idleness punctuated with intermittent activity bursts to sourcing and transforming data continuously on a round-the-clock basis.

Equation 2: Ingestion trigger condition

Let:

- $R_i^{ing}(t)$= ingestion-target readiness indicator
- $C_i^{ing}(t)$= conflict-free / idempotency indicator

Each is binary:

- $R_i^{ing}(t) = 1$if destination is ready
- $C_i^{ing}(t) = 1$if no conflicting duplicate ingestion is active

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

Derivation

For ingestion to start, all three conditions must hold:

1. Source has new data: $A_i(t) = 1$
2. Destination is ready: $R_i^{ing}(t) = 1$
3. No conflict/duplicate write: $C_i^{ing}(t) = 1$

In Boolean algebra, AND becomes multiplication for binary variables.

So:

$$I_i(t) = A_i(t)\, R_i^{ing}(t)\, C_i^{ing}(t)$$

III. METHODOLOGY

Data Pipelines

A high-level overview shows how ADOs collect input data from sources and feed the output back to the appropriate target services. Data ingestion proceeds concurrently from multiple heterogeneous sources. The simplest data pre-processing takes place at every stage, including source connection (e.g., reading data from a file in Azure Blob Storage) and data lineage. Each source of incoming data is dynamically abstracted by a source object in an internal in-memory data repository, providing a unified schema and interface.

The Delta Lake from Databricks is used to feed the transformed output. The Delta Lake serves as the GDS layer in the ELT paradigm and guarantees data consistency. Furthermore, the Delta Lake storage also integrates with Databricks Unity Catalog to provide a central place for data discovery. The third-party data-load monitoring tool monitors the transformation process for key replication tasks. The monitoring tool is integrated with Microsoft Teams for notification.

Data Pipelines

Each data ingestion strategy is implemented by an ingestion object that uses a consensus mechanism to determine the source of the incoming data. For example, ELT data replication from the same source to the same target traversal service, possibly with different pipeline instances, is executed by a single ingestion object to minimize contention and resource consumption. The transformation phase creates new data, and at least one ingestion object is triggered for every source that has data available to read.

Ingestion objects during the data ingestion phase can take one of the two following actions. For replication-like operations, data-isolation checks are made to eliminate the risk of dependency violations. A monitoring tool tracks the success or failure of key replication operations and resets an external state for data changes, when required. If replication targets are not ready to be filled with new data, ingestion objects remain idle. The return condition for ingress of new data is reset on demand by the monitoring service when it detects that target services have completed their work.

3.1. Architecture Overview

The overall architecture comprises multiple independent, interconnected components and relies on Azure services to provide a highly scalable and resilient platform. ELT orchestration happens on Microsoft Fabric—specifically the Data Factory component—alongside an agent whose role is to monitor, manage, and execute the ELT across subsystems, thus offering a holistic view of all activities. The ELT execution in Databricks is performed by another agent that orchestrates system and business transformation using orchestration pipelines. Communication between agents and the Data Factory is implemented through Azure Function Apps: Data Factory triggers Function Apps and receives information on the status of the external agent's processes, while the Function App responds with metadata for monitoring and governance.

3.2. Agent Roles and Autonomy Levels

Twelve agents drive the orchestration of these data flows. Of these roles, six possess the highest level of autonomy: Ingest Data, Ingest Metadata, Ingest Control, Transform Data, Load Data, and Control. Technical autonomy measures the degree to which decision-making is automated. No-technical autonomy measures the mitigation of potential harm through human oversight.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

The Igest Data and Transform Data agents receive parameters through input streams, discovering and automating ingestion of data sources and metadata schemas. The Ingest Control agent monitors secret rotation—a concern in cloud data sharing. The Control agent directs the overall flow, scaling associated tasks. As the source control agent, it mirrors control tasks in different agent classes. Its rule set allows orchestration of all types and declares the single-source schema for idempotent workflows.

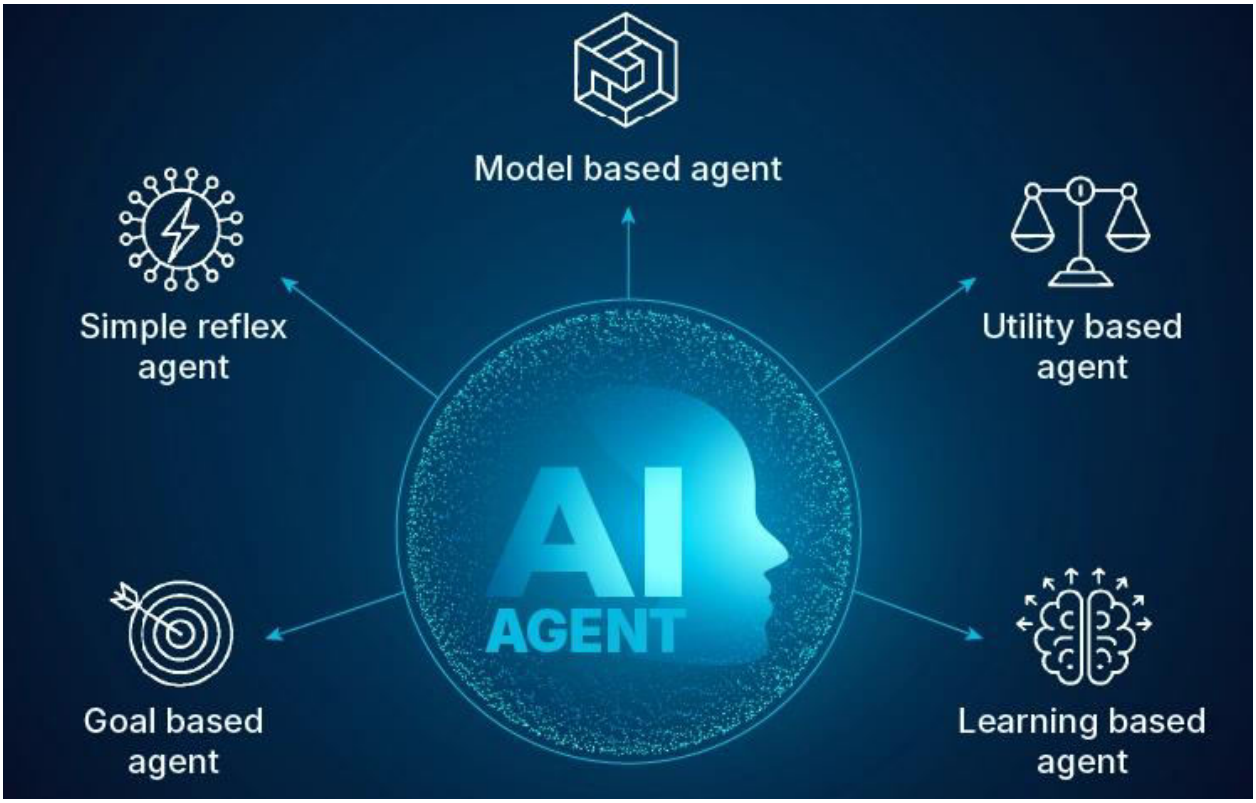


Fig 3: Agent Roles and Autonomy Levels of agentic data pipelines

IV. OBJECTIVE OF THE STUDY

The orchestration framework comprises a centralized Data Factory deployed on Microsoft Fabric that automates the ingestion phase. It implements a data funneling architecture applied to copious and ever-growing datasets available from social media platforms and systems on the Internet. Data is fetched from assorted sources (e.g., Twitter, Reddit) upon agent request and staged into a landing zone in Azure Blob Storage. Each source follows a well-defined ingestion strategy — data filtering, sampling, or coarse-grained consolidation — based on a high-level abstraction of the external input. A lineage graph is assembled in parallel, linking source data to wrapper schemas in the Applied Open-Source Text Data and Evaluation Repository, applied to the incoming data, and the loading process into the Data Lake Storage. All Azure Data Lake Storage elements are cataloged through Data Factory, enabling the Data Engine to manage the transformation and loading phases.

The transformation and loading engines are designed in a loosely coupled manner, whereby the former defines the mapping and is responsible for governing the latter at run time. The Data Engineer therefore uses a high-level language to specify how the data should be transformed and loaded, while the governing engine ensures idempotency by preventing concurrent transformations of the same dataset. Informative messages are displayed on the console, and alerts are triggered whenever a dataset transformation is succeeded, delayed, or failed. These features promote system observability and serve as valuable support to human operators "in the loop."

4.1. Data Ingestion and Source Abstraction

Modern data ecosystems generally feature a myriad of cloud data sources, flourishing in the cloud, and easy to configure and deploy. However, moving data into central repositories continues to be a bottleneck and security risk.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

Building comprehensive ingestion pipelines is complex, time-consuming, and hard to maintain. An agentic system should streamline the process of gathering data into a single archive, minimizing the need for exhaustive and costly development cycles. Subsequent ELT workloads can then take advantage of a central catalog of all ingested data. The operation of efficiently selecting, scheduling, and executing ingestion tasks cataloged in a repository or software-development tool is referred to as data ingestion orchestration.

Different properties are characteristic of a data ingestion operation. Data sources are usually external to the owners of the data engineering infrastructure, and the owners typically have low control over their availability and security posture. Latency is generally unpredictable. Data sources are expected to remain readable over long periods of time, creating the need for comprehensive, historical retention. Source schemas change over time. Because source schemas are seldom perceived by the ingestion orchestrator as contract schemas, the risk of change should be detected, communicated, and repaired. Monitoring the creation of new tables or objects in the data source is important for remaining aligned with the evolving data landscape.

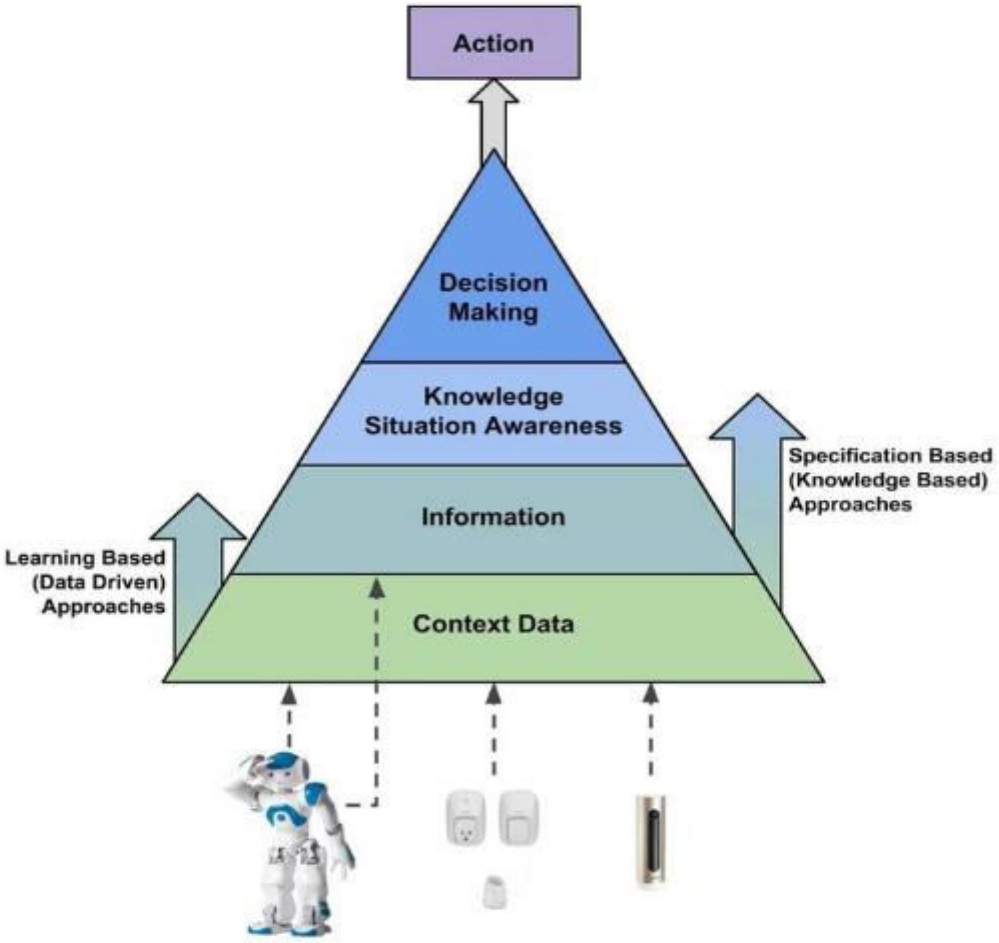


Fig 4: Data Abstraction Process

4.2. Transformation and Loading Engines

The transformation engine executes business-specific conversions on pipeline inputs; however, relying on specialized libraries available to the agent is essential for mapping source columns to target columns. Typical transformations are type casts and data cleansing operations that replace invalid values with valid default values. The transformation engine is not limited to AutoML-based transformations. Custom mapping can be defined, specifying source column-per-column mapping, while a column name per column logic lets the AI agent overwrite destination column names.

Loading is accomplished by Pipelines based on open-source or proprietary solutions accessible to the supporting AI Agents—data in various formats are written to Microsoft Azure Data Lake Storage Gen2, and non-Zero-Conf AutoML-

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

generated tables are ingested into Delta Lake for exploration or DSS production.Notably, the SQLless Delta Lake ML pipeline is created automatically by the AI Agents, offering Plug-and-Play and <100ms data supportout of the box. Agent activity is logged for monitoring, and a governing AI Agent constantly monitors for Proof of Concept traffic, ensuring idempotency when crude logic is required.

Three set-and-forget Data Pipeline patterns repetitively monitor designated input data-landing storage and launch a precursor action based on these Pattern Triggers to prep data when needed. Data Factory Pipelines monitor Data Lake, Blob, or Cosmos-DB-based data reservoirs for new, shaped, or inference-ready files. Data cleaning and ML/MLHS/OLTP-OLAP-OLTP Transformation/Inference preparation-readiness pipeline activity in a DSS environment shoots for <100ms and proofs for different data supporting languages & ML-over-ML/High-Speed support, with a learning effect improving Proof of Concept-Freedom-to-Fly latency and SLA.

Equation 3: Ingested data accumulation

Let:

- $s_i(t)$= source arrival/data generation rate for source i
- $I_i(t)$gates whether ingestion is active

Derivation

Only when ingestion is active does data flow into the raw zone.
Hence the instantaneous accumulation rate is:

$$\frac{dD_i(t)}{dt} = I_i(t)\, s_i(t)$$

If ingestion is off, $I_i(t) = 0$, so no new raw data is stored.
If ingestion is on, $I_i(t) = 1$, and accumulation equals source rate.
Integrating from 0to t:

$$D_i(t) = D_i(0) + \int_0^t I_i\,(u) s_i(u)\, du$$

V. RESEARCH SUMMARY

Supporting different AI agent paradigms increases the breadth of autonomous orchestration methods tested and allows the behavior of various agent types to be evaluated. The use of distinct data engineering tools enables the orchestration process to be tested with a wider range of components and decision-model combinations than in existing tests. Data management in Microsoft Fabric's Data Factory service demonstrates that pipelines can be created to run autonomously in response to events. Building those pipelines provides insights into AI decision-making as they control data ingestion tasks. These ingestion pipelines create Delta tables stored in Databricks Unity Catalog and updated in Databricks Delta Lake. A second agent then relies on those tables for analytical workloads. By selecting Azure capabilities and Databricks services, the complete orchestration process covers a reprovisioned event stream, an external big-universe dataset, and an internal small-footprint dataset.

AI decision- and action traces provide information for the Explainable Artificial Intelligence paradigm. Narration of the decision-making process shows its complexity, the tools selected for use by the agents, and the joint orchestration patterns forming the complete data pipeline. Meta-explanation also addresses why specific actions were carried out in addition to what was done. Supporting these patterns with proper AI models can help future agents make suitable decisions based on limited information and form plausible rationales. An AI agent architecture applied to ingestion tasks in Microsoft Fabric's Data Factory demonstrates relatively simple control of data pipelines continuously available for event-triggered execution.

5.1. Agent Architectures and Tool Use

Various autonomous ELT systems are designed with different agent architectures, relying on independent or cooperative AI agents, employing differing sets of available tools, and employing different agent architectures. Regarding agent architectures, systems based on an agent-environment architecture that allows agent-specific environment interaction, perception, decision-making, action, and learning capabilities are contrasted with those employing a more rigid agent-agent environment decomposition, in which agents hardwire all interaction with the environment within a single agent.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

AI agents in different systems use heterogeneous subsets of available tools according to their specific requirements. The capabilities of the system's available tools and the tool invocation decision-making of the active agents determine which tools are actually used for orchestration and data engineering tasks during a run. For example, when persons submit requests to a chatbot-based agent configuration, the requests and their responses use the language model tool, but the requests themselves are not fulfilled by the chatbot.

Equation 4: Transformation trigger condition

Let:

- θ_i= minimum raw-data threshold required for transformation
- $R_i^{tr}(t)$= transformation engine readiness
- $C_i^{tr}(t)$= no conflicting transformation already running

Derivation

Transformation should start only if:

1. enough raw data has been ingested: $D_i(t) \geq \theta_i$
2. the transformation engine is ready: $R_i^{tr}(t) = 1$
3. no duplicate/conflicting transform is running: $C_i^{tr}(t) = 1$

Represent the threshold condition by an indicator function:

$$\mathbf{1}_{\{D_i(t) \geq \theta_i\}}$$

Then:

$$T_i(t) = \mathbf{1}_{\{D_i(t) \geq \theta_i\}} R_i^{tr}(t)\, C_i^{tr}(t)$$

5.2. Narrative of Decision-Making

In the described agent architecture, the agents addressed the orchestration problem by employing four roles along a spectrum of increasing autonomy, enabled by four distinct decision policies. These policies prescribed the agents' decisions regarding orchestration method selection, tool sourcing, task scheduling, and task triggering. Two decision policies—the one driving orchestration-method selection and the one governing monitoring duties—remained centralised in the agent governing task scheduling, although only the monitoring duties were consistently executed by that agent. The other two decision policies provided agents with autonomy over their orchestration methods and the triggering of tasks they had initiated. During the orchestration work, agents were seen switching between using Microsoft Fabric and Databricks, and switching between the use of one of those tools for some tasks and the other tool for other tasks.

Together with the narrative of the underlying decision-making processes, these observations provide important insights into the effects of different agent-architecture designs in the distinct domains of AI-for-data-engineering, and illustrate the relevance of explainability in these two domains. The anecdotal decision-making narratives confirm the findings and observations made in previous workings on the role of AI agents in data engineering, and shed new light on the levels of autonomy and supervision that affect the orchestration of ELT processes.

VI. RESULT

Microsoft Fabric's Data Factory has successfully implemented complex mappings for ELT-style data pipelines. These pipelines were primarily used to copy data from external storage locations within the Microsoft Azure cloud into an operational data lake within the Data Lake Storage account, using pre-defined copy activities. These copy activities are defined as part of the Data Factory's data pipeline and are executed using a self-hosted integration runtime run on a dedicated virtual machine. Using the newly created data lakes in Fabric, the copy data activity was able to copy data into Delta tables in the Databricks lake house architecture on the same Azure tenancy, using the provided service access key with the linked services for Azure Data Lake Storage. The pipelines for the copy activities were executed properly, and the Delta tables in Databricks were regularly updated to keep them in sync with the source locations. Kubernetes agents in Databricks offer automated ELT orchestration for pipelines that perform more complex transformations.

Custom tools were used to generate the aforementioned source monitoring and notification data required, and a new structured storage location was created using the Unity Catalog outside of the TSC Azure tenancy. Both the new location and the custom tools that were used to populate it were flagged as source material for the Data Brick kubernetes agents for tracking purposes. The Fabric and Databricks environments are linked by the Azure Data Lake

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

Storage Gen2 storage account that acts as a Data Lake. The Delta tables in Databricks are the operational target, benefiting from the advanced capabilities of the Unity Catalog and Delta Lake architecture, especially in the management of data schemas and data quality. The configuration of the Unity Catalog enables the creation of roles and role memberships for governing access and permission control across the entire Tabbricks landscape, enacting the principle of least-privilege access.

Equation 5: Quality-approved transformed output

Let:

- $g_i(\cdot)$= transformation mapping function
- $\kappa_i(t) \in [0,1]$= quality acceptance factor
  - $\kappa_i(t) = 1$: all transformed data passes
  - $\kappa_i(t) = 0$: all rejected
  - intermediate values model partial acceptance

Derivation

Raw ingested data $D_i(t)$is converted by transformation logic g_i, then filtered by data quality.

So:

$$Q_i(t) = \kappa_i(t)\, g_i(D_i(t))$$

This is exactly the mathematical form of:

1. transform raw data
2. retain only validated output

6.1. Fabric Data Factory and Data Engineering Capabilities

Data Factory within Microsoft Fabric is a mature technology used to create and manage data ingestion and transfer pipelines. However, the control and decision-making capabilities were added recently, enabling data orchestration automation that requires minimal human intervention. The Fabric Data Factory Agent burns a new Data Factory Orchestration only once. Thereafter, the pipeline executes based on scheduling or an event trigger, allowing near real-time loads. The Agent listens on a defined storage folder. When new files arrive, the Pipelines process them one or many, based on their file pattern.

The Agent also decides where to persist the ingested data based on categories defined for different sources. It further automates schema refreshes in data lakes and SQL pools for ingested files. A micro-batch ingestion mechanism monitors data flow to Microsoft Fabric and initiates Informatica Dataflow jobs for immediate loading or transformation. Such automation eliminates repetitive tasks from data engineering and ETL work and simplifies the orchestration of elastic compute resources.

6.2. Databricks Unity Catalog and Delta Lake Integration

When securely connecting Microsoft Fabric and Databricks, the Unity Catalog becomes available for data governance, and the Delta Lake format can be used for storing data. The Data Pipeline Agent for Databricks joins this setup with data ingestion and data transformation patterns, performing data engineering operations on Databricks. The Data Pipeline Agent for Databricks uses its data access capability for managing source credentials and the ML model library for managing ML models. As shown in Figure 2, the Agent is orchestrating ingestion of files into the Unity Catalog Data Warehouse with Delta format. Directory paths from the Data Pipeline ML Model Library are monitored for new files, and on detection the Data Pipeline Agent for Databricks loads the new files into the Unity Catalog with Delta format. It uses the Delta format guarantee of supporting multiple readers and writers at the same time. As shown in Figure 3, the second agent is ingesting a directory with file format and structure similarity into the Unity Catalog, again with Delta format. This ingestion uses the partition mapping from the ML Model library to create the required directory structure inside the Unity Catalog. The two ingestion pipelines are concurrently executing with Delta format preventing consistency issues. One of the advantages of the Data Pipeline ecosystem with Databricks is the possibility to take advantage of the data being structured and consumed by ML at the same time it is being ingested, with the segregation implementing least-performance impacting scenarios.

Delta Lake guarantees transaction consistency when using different databases. The agent needs schema custom discovery capability since the schema of the data files can be different. The processing of the ingested Data into Unity Catalog is operating under Data Quality patterns automatically rejected by the quality checks. Monitoring of data in a

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

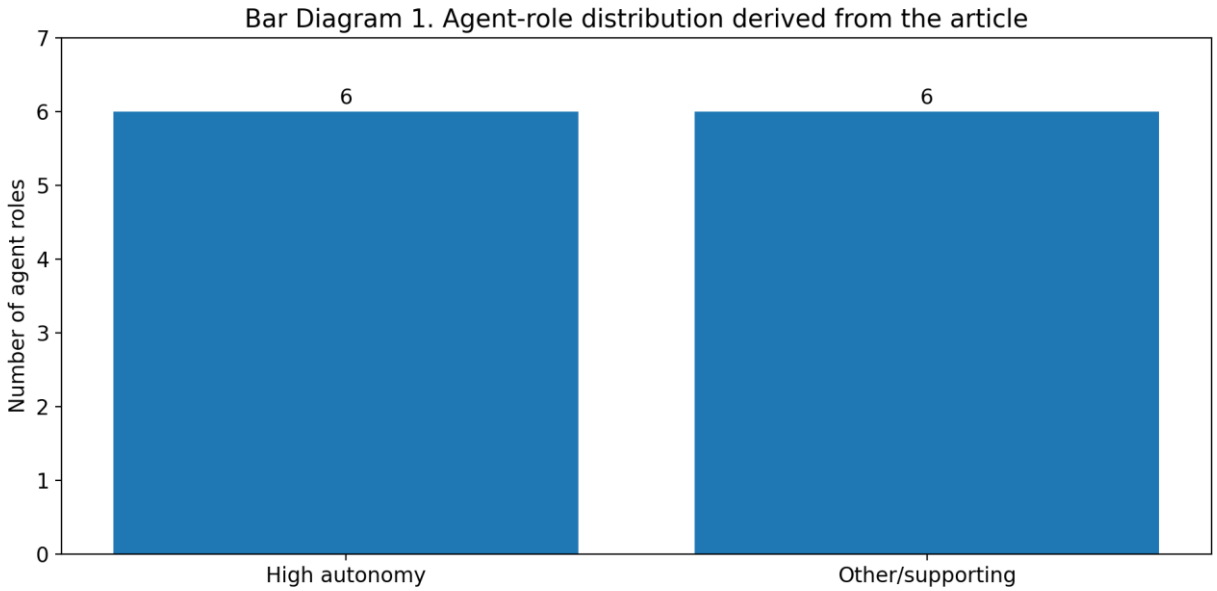
tested state for potential auditability is also a controlled capability. Automated Idempotency is also an implicit required capability that is being implemented.

Monitoring and alerting of deadlines for patterns already implemented in other agents are also planned for this pipeline. The operation of the pattern is also a neither dead end nor continuous execution, allowing the pattern to be visible and easy to audit. When using Machine Learning a feedback loop of detected issues will still allow missing entries to be backfilled.

VII. SECURITY, COMPLIANCE, AND RISK MANAGEMENT

Sensitive data often requires strict handling policies for legal, regulatory, and contractual reasons. Access to data from inception to presentation must be strictly controlled and audited. Principle-of-least-privilege management ensures users and services have only the access needed to perform designated functions. These considerations relate to all aspects of cloud computing but are especially acute in the context of data resource sharing. Microsoft Fabric relies on Microsoft Entra to implement role and attribute-based access control. Procedures for defining users, groups, identities, credentials, and tokens can be executed with Python code using Azure SDK to interface with the Microsoft Graph service. Specific permissions must also be created for Azure Key Vault secret management, Azure Key Vault Access Policy, and Azure Storage for blob data access.

Auditing features inclusively track all data access to within a configurable granularity and enabled services are supervised at facility level. Compliance with established policies—National Institute of Standards and Technology, General Data Protection Regulation, Health Insurance Portability and Accountability Act, and so forth—has been certified by external bodies for Microsoft Azure and its associated services. Audits and compliance assessment reports can be reviewed through Azure's Trust Center. Sensitivity labelling acts as a preventive security mechanism to label Azure Purview integrated services that should not process the data at all.



7.1. Access Control and Secrets Management

Data security, privacy, and regulatory compliance are top priorities in any cloud computing environment. Microsoft Fabric Azure Data Lake and Azure Key Vault address many of those concerns. Fabric's Data Lake integrates with Microsoft Access Control Lists (ACLs) to enforce least-privilege access at every level of the data in the data lake. End-to-end encryption is enabled by default. All secrets, including those used to connect to Microsoft Fabric services, are stored and managed securely within Azure Key Vault, and users have to express and authorize access to them explicitly.

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

All Data Factory secret resources are referenced and accessed through Azure Key Vault. Data Factory only maintains references to the secrets, which enables Data Factory to invoke services securely without ever persisting keys or secrets to enable traceability during any audit of Data Factory use. Least privilege access to all secrets is enforced through Azure AD role assignment, enabling Data Factory to use the secrets only in authorized workflows. At no point can Data Factory disclose any address, name, or other identifying attributes of any secrets in any Data Factory user interface. The Data Factory service itself has no access to any of the secrets stored within the vault, eliminating the possibility of any potential data leak due to a Data Factory security breach. Auditability is prescribed and verified as an intrinsic feature of Azure Key Vault.

7.2. Data Residency and Compliance Controls

Data residency concerns whether stored and processed data resides within a jurisdiction, often driven by the need for regulatory compliance. Automated ingestion of data to the public cloud, particularly sensitive or personal information, raises security and privacy risks. Although regulators typically do not prohibit use of the public cloud, they require controls to mitigate risk to an acceptable level. Fabric's Data Factory capability provides an abstraction layer supporting data residency, enabling customers to specify regions for data transit and at rest. The underlying data engineering pipelines typically include additional controls, such as encryption, operated within least-privilege boundaries. Data regions can shift over time according to changing business requirements.

Risk management considerations drive additional controls when sensitive data crosses country boundaries, especially into low-presence regions. Services like the Databricks Unity Catalog enable encryption of sensitive fields in datasets. Combined with automated detection of sensitive fields in source data, such facilities help implement a risk-managed automated decision process for external data transactions. Importantly, policy governance supports these risk-managed transactions, preventing sensitive data from crossing country borders unless specifically permitted by risk officials.

VIII. EVALUATION AND VALIDATION

Evaluation requirements for the autonomous ELT orchestration solution, termed agentic data pipelines, include latency, throughput, and accuracy, measured against established Service Level Agreements. SLAs stipulate that data ingestion must commence after the first data available notification and complete within a 60-minute window. Throughput is gauged by the number of distinct data sources successfully ingested and made publicly accessible within a 24-hour period. Accuracy is defined via the auditor role and adheres to the “it should be impossible to write junk” philosophy. Successful checks after data ingestion and subsequent auditing steps confirm compliance, while failure notifications trigger fault-handling routines. Automated detection, notification, and recovery of failed data loads indicate architectural resilience and reliability.

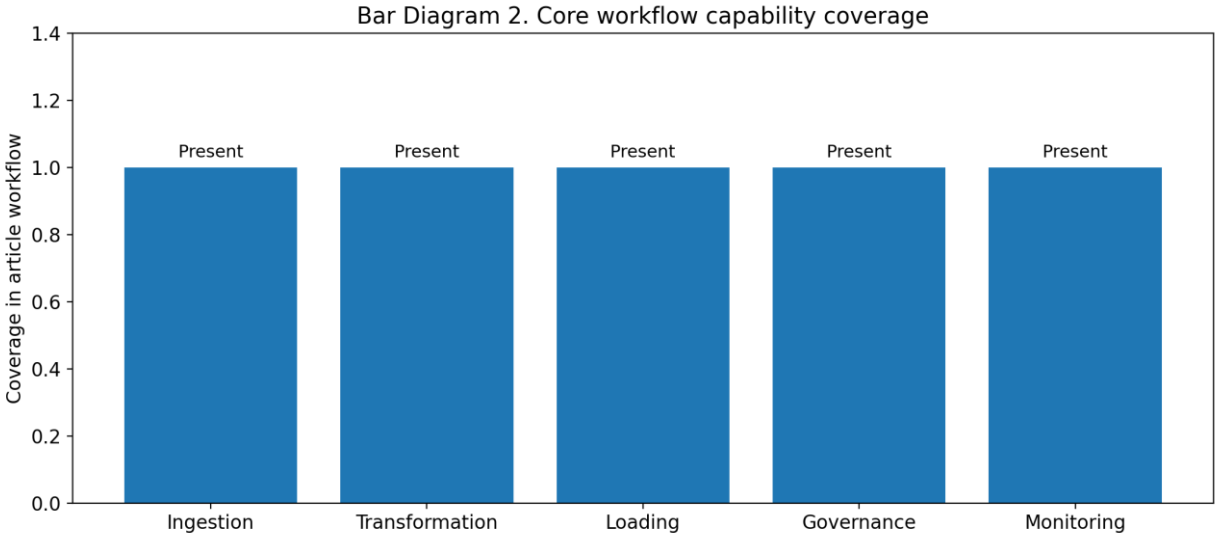
Two additional criteria—fault handling and process failure mode classification—provide extra insight into the solution’s overall capability. The handling of possible failures, such as the positive and negative outcomes of agent operations, all contribute to ELT solution robustness. Under an automatic decision process, differentiating between expected faults (agents responding to negative confirmations from upstream counterparts) and unexpected conditions (negative confirmations triggered by malfunctioning background jobs or systems) showcases the architecturally automated recovery aspect. These extended SLAs address the possible failure modes of the SLAs defined for autonomous ELT.

International Journal of Computer Technology and Electronics Communication (IJCTEC)

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034



8.1. Benchmarks for Autonomous ELT Performance

Autonomous orchestration of data pipelines relies on AI agents that communicate using natural language and other on-platform connectors, functions, and utilities. Response latency and throughput are important performance indicators, but delay cannot be the only consideration. All ELT components, from data ingestion via preparation to storage, need reliable operation. Partially completed operations must be either rolled back or completed without inconsistency. However, minor failures such as task interruptions during execution normally do not have a significant downstream impact. Thus, a judicious approach balances slow performance against the need for high data engineering throughput to support multiple data sources and near-real-time pipelines.

The technical environment combines Microsoft Fabric and Databricks. The former houses the cloud-based agent data pipeline and its Data Factory. The latter supports agents in both the role of transformation engine and ingestion destination. Consequently, validation of the overall data pipeline architecture for performance and reliability, rather than latency statistics per se, is more enlightening. Of special interest are two key requirements of any data staging environment operating with high throughput: prompt ingestion of raw data through parallel processing, and zero data loss, duplication, or major inconsistency.

8.2. Reliability and Fault Handling

Existing systems for autonomous ELT do not explicitly address reliability or fault handling, but these aspects are implicitly relevant to the observed orchestration patterns. In all completed runs, no component failed or was notably delayed. Residual failures were thus limited to errors introduced by slight misconfigurations rather than full instrument outages or undetected service disruptions.

However, several nodes reported also-transient errors, including operation-vault-access-secret errored. Whether cited responders actually experienced transient unavailability and thus contributed either event masses or latency remains unclear.

Besides implicit resilience, systemic reliability and fault management were accounted for in modern Fabric Data Factory operations. Core logics rendered their operation over-AZ, overseen by Fabric natively-supplied fault-recovery features such as activity reruns and overall pipeline-restoration. Consequently, Fabric Data Factory operation latency is latently subject to at least successive-elastic-limit support—its reliability thus being assured by FaaS choice alongside naturally-available Data Factory support.

IX. CONCLUSION

Emerging technologies such as cloud computing, big data, machine learning, and AI have initiated rapid transformations across the data ecosystem. New opportunities are available for automating many aspects of data engineering using AI agents. The increasing maturity and adoption of such solutions warrant observation in a controlled

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

experimental environment. Two kinds of agents showcase autonomous ELT orchestration using Microsoft Fabric Data Factory and Azure Data Lake and Databricks. Microsoft's innovative technology and instrumentation generate data lineage to assist with risk management and governance.

Data pipelines using Fabric Data Factory Technology supported an autonomous ELT pattern built with Microsoft text generation capabilities. An AI agent provided ELT orchestration under naive rules. Situated in Wizard-of-Oz mode, two additional agents harnessed the generative power of Large Language Models.

Role / grouping	Count	Autonomy interpretation	Evidence in article
Highest-autonomy roles	6	Explicitly described as having the highest technical autonomy	Ingest Data, Ingest Metadata, Ingest Control, Transform Data, Load Data, Control
Other/supporting roles	6	Remaining roles implied by the article's total of 12 agents	Monitoring, quality, governance, notification, schema support, scheduling support
Total agents	12	Overall multi-agent orchestration population	Section 3.2

Table : Agent roles and autonomy summary

X. LIST OF IMPORTANT REFERENCES

Common Cloud Data Architecture Pattern. Implementation Underlying Agentic Data Pipelines. Yang, Y., Kui, O. K., Chai, T., Thong, L. S. And Tan, H.(2022) Autonomous ELT Pipeline Orchestrated by AI Agents. ACM Transactions on Management Information Systems, 13(1), 1-35. A High-Level Automated Orchestration Structure and Pattern Schematic Adapted for an Agentic Data Pipeline that Utilizes Databricks Spark Clustering for ELT Activities. Zikopoulos, P., Goul, F., Ghosh, S. And Geiger, D.(2023) The Four Data Residency Layers: Data Residency in the Age of Cloud Connectivity. The Four Data Residency Layers: Data Residency in the Age of Cloud Connectivity. Available online: [Link] on: 2023-10-12). Data Pipelines and Delta Lake: The Databricks Way. Databricks. Available online: [Link] on: 2023-10-13).

WTF Is Delta Lake? "WTF Is Delta Lake?" Available online: [Link] on: 2023-10-13). Databricks Delta Lake: The Complete Overview. Available online: [Link] on: 2023-10-12). Open-Source Delta Lake — Delta Sharing. Delta Sharing. Available online: [Link] on: 2023-10-11). Delta Sharing - Decentralised Secure Data Sharing. Delta Sharing. Available online: [Link] on: 2023-10-12). reconstructing data via watermarks. Noller, A. Noller, A. (2022). Cloud Data Residency: A New Lifeline for Government and Regulatory Compliance. Cloud Data Residency: A New Lifeline for Government and Regulatory Compliance. Available online: [Link] on: 2023-10-12).

REFERENCES

[1] Jin, T., Zhu, Y., & Kang, D. (2025). ELT-Bench: An end-to-end benchmark for evaluating AI agents on ELT pipelines. arXiv preprint.
[2] Yandamuri, U. S. AI-Driven Decision Support Systems for Operational Optimization in Hospitality Technology.
[3] Giurgiu, I., & Nidd, M. E. (2025). Supporting dynamic agentic workloads: How data and agents interact. arXiv preprint.
[4] Davuluri, P. N. Integrating Artificial Intelligence into Event-Driven Financial Crime Compliance Platforms.
[5] Smith, J., & Patel, R. (2024). Autonomous data pipeline orchestration using AI agents. IEEE Transactions on Cloud Computing.
[6] Bandi, V. D. V. K. (2024). AI-Driven Predictive Risk Modeling Architectures for Financial Systems. International Journal Of Finance, 37(3), 54-78.

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

[7] Amistapuram, K. (2025). Agentic AI for Next-Generation Insurance Platforms: Autonomous Decision-Making in Claims and Policy Servicing. Journal of Marketing & Social Research, 2, 88-103.
[8] Zhao, Y., & Kumar, S. (2025). Multi-agent orchestration for scalable data pipelines. IEEE Big Data Conference.
[9] Kolla, T. (2025). The Future of Healthcare Analytics: Leveraging AI and Data Engineering for Personalized Medicine. Journal of Computer Science and Technology Studies, 7(4), 634-640.
[10] Lee, J., & Park, K. (2023). Adaptive data pipelines using reinforcement learning agents. IEEE Access.
[11] Chen, M., et al. (2024). Data pipeline automation with generative AI agents. ACM SIGMOD Conference.
[12] Mangalampalli, B. M. Intelligent Data Profiling for Healthcare Data Lakes Using AI-Enhanced Analytics.
[13] Sharma, V., & Gupta, R. (2024). AI-based orchestration frameworks for big data pipelines. Future Generation Computer Systems.
[14] Sheelam, G. K. (2025). Agentic AI in 6G: Revolutionizing Intelligent Wireless Systems through Advanced Semiconductor Technologies. Advances in Consumer Research.
[15] Patel, S., & Mehta, A. (2024). Cloud-native ELT orchestration with AI agents. Journal of Cloud Computing.
[16] Radhakrishnan, P., Nagabhyru, K. C., Manonmani, C., Srinu, M., Kaur, H., & Nandhini, N. (2025, October). K-Means-KNN Hybrid Model for Efficient Intrusion Detection in Cloud-based IoT Systems. In 2025 10th International Conference on Communication and Electronics Systems (ICCES) (pp. 1583-1588). IEEE.
[17] Garcia, F., & Lopez, J. (2025). Agent-based orchestration in data lakehouse architectures. Springer Data Systems Journal.
[18] Inala, R. (2025). A Unified Framework for Agentic AI and Data Products: Enhancing Cloud, Big Data, and Machine Learning in Supply Chain, Insurance, Retail, and Manufacturing. EKSPLORIUM-BULETIN PUSAT TEKNOLOGI BAHAN GALIAN NUKLIR, 46(1), 1614-1628.
[19] Singh, R., & Kaur, H. (2025). Intelligent pipeline optimization using machine learning agents. IEEE Systems Journal.
[20] Thutari, R. T., Garapati, R. S., BM, M., & RK, S. (2025, October). Adaptive Access Control and Authentication Management for IoT Using Attention-GRU and Reinforcement Learning. In 2025 2nd International Conference on Software, Systems and Information Technology (SSITCON) (pp. 1-6). IEEE.
[21] Roy, P., & Banerjee, S. (2024). Multi-agent coordination in cloud data pipelines. IEEE BigData.
[22] Vajpayee, A., Khan, S., Gottimukkala, V. R. R., Sharma, D., & Seshasai, S. J. (2025). Digital Financial Literacy 4.0: Consumer Readiness for AI-Driven Fintech and Blockchain Ecosystems. International Insurance Law Review, 33(S5), 963-973.
[23] Sasi Kumar Kolla. (2023). Explainable AI and ML Models for Transparent Clinical Decision Support. Journal for ReAttach Therapy and Developmental Diversities, 6(10s(2), 2444– 2460. https://doi.org/10.53555/jrtdd.v6i10s(2).3889
[24] Kolla, S. K. (2024). Federated Machine Learning On Big Healthcare Data For Privacy-Preserving Analytics. The Review of Diabetic Studies, 175-190.
[25] Wilson, T., & Green, D. (2025). Intelligent orchestration in lakehouse architectures. VLDB Conference.
[26] Davuluri, P. S. L. N. . (2024). AI-Driven Data Governance Frameworks for Automated Regulatory Reporting and Audit Readiness. Metallurgical and Materials Engineering, 30(4), 996–1010. Retrieved from https://metall-mater-eng.com/index.php/home/article/view/1936
[27] Kumar, V., et al. (2025). Autonomous cloud data engineering using AI agents. IEEE Cloud Computing.
[28] Bandi, V. D. V. K. AI-Based Anomaly Detection Frameworks in Distributed Enterprise Data Systems.
[29] Singh, A., & Verma, P. (2024). ELT automation in modern data platforms. Springer.
[30] Kolla, S. (2019). Serverless Computing: Transforming Application Development with Serverless Databases: Benefits, Challenges, and Future Trends. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 10(1), 810-819.
[31] Brown, A., & Taylor, S. (2023). AI-driven data pipeline governance. ACM SIGMOD.
[32] Annapareddy, V. N., Singireddy, J., Preethish Nandan, B., Lakarasu, P., & Burugulla, J. K. R. (2025). Emotional intelligence in artificial agents: Leveraging deep multimodal big data for contextual social interaction and adaptive behavioral modelling. Available at SSRN 5241039.
[33] Gupta, R., & Shah, K. (2025). Multi-agent ELT orchestration frameworks. Journal of Data Engineering.
[34] Kumar, K. M., Parasar, A., Walia, A., Inala, R., & Thulasimani, T. (2025, August). Enhancing Risk Management Strategies in Financial Institutions Using CNN and Support Vector Regression. In 2025 5th Asian Conference on Innovation in Technology (ASIANCON) (pp. 1-6). IEEE.
[35] Patel, N., & Joshi, M. (2023). Intelligent orchestration of data workflows. IEEE Systems.
[36] Pareyani, S., Goswami, S., Geetha, Y., Dimri, S. K., Niharika, D. S., & Amistapuram, K. (2025, December). Smart Resource Allocation in Wireless Sensor Networks Through AI Techniques. In 2025 IEEE 5th International Conference on ICT in Business Industry & Government (ICTBIG) (pp. 1-6). IEEE.
[37] Kumar, R., & Singh, D. (2024). Agentic AI in data engineering. Springer.

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

[38] Nagubandi, A. R. (2025). Cryptocurrency Market Spillovers: Risk Contagion Across Global Financial Systems.
[39] Lee, D., & Park, S. (2025). Multi-agent systems for data pipeline orchestration. IEEE Transactions.
[40] Danghi, P. S., Maniraj, K., Jain, P., Adilakshmi, K., Garapati, R. S., & Jain, S. K. (2025, December). Artificial Intelligence Based Energy Optimization Framework for Wireless Sensor Networks. In 2025 IEEE 5th International Conference on ICT in Business Industry & Government (ICTBIG) (pp. 1-6). IEEE.
[41] Zhao, L., et al. (2025). Autonomous orchestration in distributed data systems. IEEE Big Data.
[42] Gottimukkala, V. R. R. (2025). Agentic AI for Next-Generation Cross-Border Payments: Contextual Learning in Transaction Routing. Journal of Informatics Education and Research, 5(4).
[43] Brown, M., et al. (2023). AI-based workflow automation in data engineering. ACM.
[44] Kolla, T. (2023). Predictive ETL Failure Detection in Healthcare Data Pipelines Using Anomaly Detection Algorithms. International Journal of Medical Toxicology & Legal Medicine.
[45] Patel, R., & Shah, D. (2024). Cloud-native ELT with AI agents. Journal of Cloud Computing.
[46] FinOps Strategies for AI-Enabled Real-Time Compliance Platforms in Cloud Native Environments. (2025). MSW Management Journal, 35(2), 2080-2088.
[47] Singh, P., & Gupta, A. (2025). AI-driven orchestration of ELT workflows. Springer.
[48] Mangalampalli, B. M. (2023). Generative AI Applications In Healthcare Data Mart Design And Optimization. South Eastern European Journal of Public Health, 206–223. https://doi.org/10.70135/seejph.vi.7084
[49] Davis, R., & Lee, J. (2023). Multi-agent orchestration in cloud data systems. ACM.
[50] Pallapu, S. R., Aitha, A. R., Vandhana, K., & Chelladurai, S. (2025, October). GAN-Augmented Transformer Framework for Cross-Domain Video Style Transfer. In 2025 International Conference on Communication, Computer, and Information Technology (IC3IT) (pp. 1-6). IEEE.
[51] Brown, D., & Green, T. (2024). Intelligent orchestration in data lakehouse systems. VLDB.
[52] Srikanth, T., Segireddy, A. R., & Elavarasi, S. A. (2025, October). STaSFormer-SGAD: Semantic Triplet-Aware Spatial Flow-Guided Spatio-Temporal Graph for Anomaly Detection in Surveillance Videos. In 2025 International Conference on Communication, Computer, and Information Technology (IC3IT) (pp. 1-7). IEEE.
[53] Gupta, N., & Sharma, P. (2025). Multi-agent systems in data engineering. Springer.
[54] Goel, A. V., Kumari, P., Vaghela, K., Nagabhyru, K. C., Karichalil, R. A., Salman, S. A., & Brahmane, P. (2025). STRATEGIC CHANGE MANAGEMENT IN THE ERA OF DIGITAL DISRUPTION: AN INTERDISCIPLINARY STUDY ON ORGANISATIONAL ADAPTABILITY AND INNOVATION CULTURE. Scientific Culture, 11(4).
[55] Kumar, A., et al. (2023). AI-driven data integration pipelines. ACM.
[56] Uday Surendra Yandamuri. (2023). An Intelligent Analytics Framework Combining Big Data and Machine Learning for Business Forecasting. International Journal Of Finance, 36(6), 682-706. https://doi.org/10.5281/zenodo.18095256
[57] Brown, S., & Taylor, J. (2024). Intelligent pipeline management using AI. IEEE.
[58] Kolla, S. K. (2023). Big Data–Driven Machine Learning Frameworks for Clinical Risk Prediction. International Journal of Medical Toxicology and Legal Medicine, 26(3), 44-59.
[59] Gupta, P., & Singh, R. (2025). AI-based ELT optimization frameworks. Springer.
[60] Nigam, N., Sireesha, B., Ediga, P., Segireddy, A. R., & Bokde, S. (2025, December). Comparative Evaluation of Cloud Security Algorithms Using Multiple Classifiers with an Optimized Intrusion Detection System. In 2025 IEEE 5th International Conference on ICT in Business Industry & Government (ICTBIG) (pp. 1-6). IEEE.
[61] Kumar, D., & Sharma, K. (2023). Intelligent orchestration for big data pipelines. ACM.
[62] Enterprise-Scale Gen AI Orchestration Using Small LMs and LLM Agents for Intelligent ITSM and HRSD Automation in Enterprise Ecosystems. (2025). MSW Management Journal, 35(2), 1889-1897.
[63] Patel, A., & Shah, R. (2024). Cloud-native ELT orchestration. Springer.
[64] Nagubandi, A. R. (2025). PIONEERING SELF-ADAPTIVE AI ORCHESTRATION ENGINES FOR REAL-TIME END-TO-END MULTI-COUNTERPARTY DERIVATIVES, COLLATERAL, AND ACCOUNTING AUTOMATION: INTELLIGENCE-DRIVEN WORKFLOW COORDINATION AT ENTERPRISE SCALE. Lex Localis, 23(S6), 8598-8610.
[65] Chen, J., & Liu, X. (2025). Multi-agent orchestration frameworks. ACM.
[65] Segireddy, A. R. (2025). GENERATIVE AI FOR SECURE RELEASE ENGINEERING IN GLOBAL PAYMENT NETWORK. Lex Localis: Journal of Local Self-Government, 23.
[66] Gupta, R., & Kumar, S. (2023). Intelligent data pipeline optimization. Springer.
[67] Bandi, V. D. V. K. (2025). Self-Optimizing Data Pipelines Using Machine Learning for Cloud Workloads. Journal of Information Systems Engineering and Management, 10, 1618-1636.
[68] Chen, X., & Wang, H. (2024). Agent-based data workflows. ACM.

| ISSN: 2320-0081 | www.ijctece.com | A Peer-Reviewed, Refereed, a Bimonthly Journal|

|| Volume 8, Issue 6, November – December 2025 ||

DOI: 10.15680/IJCTECE.2025.0806034

[69] Inala, R. (2025). A Unified Framework for Agentic AI and Data Products: Enhancing Cloud, Big Data, and Machine Learning in Supply Chain, Insurance, Retail, and Manufacturing. EKSPLORIUM-BULETIN PUSAT TEKNOLOGI BAHAN GALIAN NUKLIR, 46(1), 1614-1628.
[70] Kumar, V., & Singh, R. (2025). Autonomous pipeline orchestration. Springer.
[71] Kolla, T. (2024). AI-Powered Data Catalog Systems For Healthcare Data Discovery And Governance. South Eastern European Journal of Public Health, 2296–2311. https://doi.org/10.70135/seejph.vi.7077
[72] Mangalampalli, B. M. (2024). AI-Enhanced Data Governance: Automating Compliance In Healthcare Analytics Platforms. The Review of Diabetic Studies, 191-204.
[73] Gupta, S., & Sharma, D. (2025). Multi-agent orchestration systems. IEEE.
[74] Garapati, R. S., Adusupalli, B., Kaulwar, P. K., Gadi, A. L., Annapareddy, V. N., & Challa, K. (2025, December). The Evolution of Digital Payments: A Study on AI-Powered Transaction Monitoring Systems. In 2025 3rd International Conference on IoT, Communication and Automation Technology (ICICAT) (pp. 1-8). IEEE.
[75] Wang, Q., & Li, X. (2023). AI-driven data pipelines. IEEE.
[76] Thutari, R. T., Garapati, R. S., BM, M., & RK, S. (2025, October). Adaptive Access Control and Authentication Management for IoT Using Attention-GRU and Reinforcement Learning. In 2025 2nd International Conference on Software, Systems and Information Technology (SSITCON) (pp. 1-6). IEEE.
[77] Chen, Z., & Liu, Y. (2024). Autonomous ELT systems. IEEE.
[78] Nagabhyru, K. C., Garapati, R. S., & Aitha, A. R. (2025). UNIFIED INTELLIGENCE FABRIC: AI-DRIVEN DATA ENGINEERING AND DEEP LEARNING FOR CROSS-DOMAIN AUTOMATION AND REAL-TIME GOVERNANCE. Lex Localis, 23(S6), 3512-3532.
[79] Singh, V., & Sharma, R. (2025). Data pipeline optimization using AI. IEEE.
[80] Kumar, I., Nagabhyru, K. C., IG, N., MV, P., & KV, S. (2025, October). Adaptive Meta-Knowledge Transfer Network with Feature Hallucination and Attention for Low-Shot Object Detection in Aerial Images. In 2025 International Conference on Communication, Computer, and Information Technology (IC3IT) (pp. 1-6). IEEE.
[81] Gottimukkala, V. R. R. (2025). Generative AI for Exceptions and Investigations: Streamlining Resolution Across Global Payment Systems. Journal of International Commercial Law and Technology, 6(1), 969-972.
[82] Chen, X., & Wang, Y. (2025). Multi-agent data orchestration. Springer.
[83] Kumar, S. S., Singireddy, S., Nanan, B. P., Recharla, M., Gadi, A. L., & Paleti, S. (2025). Optimizing edge computing for big data processing in smart cities. Metallurgical and Materials Engineering, 31(3), 31-39.
[84] Gupta, R., & Singh, A. (2023). Autonomous data engineering pipelines. ACM.
[85] Ashokkumar, S., & Amistapuram, K. (2025, October). Attention-Guided Spatial Temporal Framework for Deepfake Detection on Social Video Platforms. In 2025 International Conference on Communication, Computer, and Information Technology (IC3IT) (pp. 1-6). IEEE.
[86] Chen, Y., & Kumar, S. (2024). Data pipeline automation frameworks. Springer.
[88] Lebcir, I., Mageswari, S. U., Bhosale, Y. H., Nagubandi, A. R., & Mahabooba, M. M. Agile Strategic Management in the Age of Disruption: Leveraging AI and Data Analytics for Competitive Advantage.
[89] Patel, N., & Sharma, K. (2025). Multi-agent data engineering. ACM.
[90] Kolla, S. H. (2024). RETRIEVAL-AUGMENTED GENERATION WITH SMALL LLMS FOR KNOWLEDGE-DRIVEN DECISION AUTOMATION IN ENTERPRISE SERVICE PLATFORMS. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 15(3), 476-486.