10.48047/jocaaa.2025.34.11.60

RL for Real-Time Optimization, Green Migration, Semantic Integration via LLMs

Saravanan Palaniappan
Independent Researcher, USA

Abstract

Cloud migration and data integration workflows face growing challenges related to increasing infrastructure complexity, sustainability imperatives, and semantic interoperability across heterogeneous environments. Traditional orchestration frameworks have relied on static resource allocation and manual schema mapping processes, rather than adapting at runtime to performance variations, carbon intensity fluctuations, or semantic ambiguities of heterogeneous data structures. The paper introduces the design of an integrated architecture that leverages Reinforcement Learning algorithms for dynamic pipeline optimization, Large Language Model capabilities for automated semantic integration, and carbon-aware orchestration mechanisms considering environmental impact. Reinforcement Learning agents perceive multi-dimensional state vectors capturing throughput rates, resource utilization patterns, network latency distributions, and regional emissions data to execute orchestration actions balancing completion time, cost efficiency, and carbon footprint reduction. Large Language Models interpret schema definitions, business glossaries, and metadata repositories to generate semantic mappings between disparate data sources, thereby automating expert tasks in traditional settings. Carbon-aware scheduling components interface with real-time emission data sources to implement temporal shifting strategies that route workloads to low-carbon regions and time windows. Implementation considerations deal with training data based on a digital twin simulation environment, governance through explainable decision logging mechanisms, and practical integrations with established cloud management platforms. This architectural framework demonstrates pathways toward self-sustaining, sustainable, and semantically intelligent migration systems that reduce operational complexity while enhancing technical and environmental results.

Keywords: Reinforcement Learning, Semantic Integration, Carbon-Aware Computing, Large Language Models, Cloud Migration, Multi-Agent Systems

Introduction

Cloud-driven digital transformation has ushered in a new paradigm in data architecture: one typified by rapid scaling demands, heterogeneity of systems, and rising sustainability pressures. Global expenditure on cloud infrastructure has shown exponential growth trajectories; enterprise data migration workloads account for substantial portions of cloud operational costs and carbon emissions during enterprise transformation phases. Traditional approaches to data migration and integration rely on static resource allocation, manual configuration of mappings, and cost-first optimization models that overlook carbon impact and operational adaptability. In large projects for data integration, manual schema mapping processes consume considerable portions of overall project engineering hours; error rates in semantic interpretation further prolong validation cycles and delay production. Going forward, organizations are looking to transition from monolithic cloud adoption patterns toward dynamic, intelligent, and sustainability-aware architectures and urgently require a form of AI-enabled automation that exceeds the capability of traditional rule-based orchestration.

Recent Reinforcement Learning advances demonstrate that the potential for self-optimizing data pipelines now exists, capable of adapting throughput, compute allocation, and transfer routes against real-time performance and sustainability signals. Resource allocation techniques in cloud computing systems have

10.48047/jocaaa.2025.34.11.60

evolved from threshold-based approaches to sophisticated learning mechanisms that elastically adjust to workload characteristics and infrastructure constraints [1]. Dynamic voltage and frequency scaling, virtual machine consolidation strategies, and intelligent workload placement algorithms consider both performance requirements and energy consumption patterns. By using policy gradient methods and actor-critic architectures, continuous adaptation to dynamic workload patterns can be achieved. Convergence behaviors are further improved by transfer learning across similar topologies. Integrating energy awareness into resource management frameworks constitutes a paradigm shift from reactive to proactive optimization, where algorithmic decision-making attempts to foresee future resource demand while minimizing the carbon footprint [1].

Meanwhile, Large Language Models are a breakthrough for semantic understanding of data structures, allowing automated comprehension of schemas, business glossaries, and lineage relationships that until now had been interpreted by an expert human. Latest studies on scalable schema mapping using large language models demonstrate remarkable capabilities to understand structural and semantic relationships across heterogeneous data sources [2]. Application of transformer-based architectures to the task of schema matching gave especially good results regarding the identification of correspondences between attributes in different data models, based on contextual embeddings as a means to express semantic similarity that is not confined by syntactic matching. Fine-tuned models, covering enterprise data catalogues and metadata repositories, achieve better results in the identification of semantic equivalencies and the derivation of transformation logic than general-purpose language models [2]. Their ability to process natural language descriptions, data type specifications, and example values simultaneously supports holistic interpretation of the schema elements in the greater context of organizational information.

Parallel to this, the concept of Green Migration has gained traction amongst enterprises and regulatory bodies, wherein energy efficiency and carbon footprint become first-class metrics. Data centre energy consumption accounts for a measurable proportion of global electricity demand, with cloud migration activities contributing observable spikes in resource utilization during peak transfer windows. The variation in regional carbon intensity across cloud availability zones differs significantly due to differences in the spread of renewable energy sources in their respective electricity grids. This presents an opportunity to optimize carbon-aware workload scheduling. However, most tooling doesn't integrate sustainability metrics into the orchestration decisions. Therein comes the gap that AI-driven, carbon-aware orchestration can fill by balancing cost, performance, and environmental considerations [1]. Finally, temporal shifting of non-critical data transfers to low-carbon time windows could considerably decrease migration-related emissions, depending on the characteristics of the workload as well as the dynamics of the regional energy mix.

This paper proposes an architectural model integrated with AI, combining RL-based optimization, LLM-driven semantic integration, and carbon-aware migration orchestration. The goal is to demonstrate how AI can reshape the migration lifecycle from reactive execution to proactive, self-learning orchestration, reducing manual engineering overhead while aligning with governance and ESG mandates. The proposed architecture will tackle three critical gaps in today's migration frameworks: the lack of adaptive optimization mechanisms that can respond to runtime conditions; the persistent manual bottleneck in semantic mapping and schema alignment; and the absence of operationalized carbon metrics in informing orchestration decision-making. By integrating these capabilities within a unified framework, this research points to pathways toward the development of autonomous, sustainable, and semantically intelligent data migration systems that lessen operational burden while improving technical and environmental outcomes.

10.48047/jocaaa.2025.34.11.60

Background and Problem Statement

Limitations of Traditional Migration Frameworks

Contemporary cloud migration frameworks follow prescriptive patterns based on static orchestration templates that require extensive manual configuration. These frameworks lack adaptability to runtime situations, which include network congestion, compute saturation, or fluctuating power expenses throughout local data centres. Migration pipelines typically operate on predetermined schedules without feedback loops that allow path correction based on actual-world performance degradation or sustainability alerts. Static resource provisioning models allocate compute capacity based on peak load estimations, resulting in substantial idle resources during off-peak intervals and resource contention during unexpected demand spikes. The inflexibility inherent in template-driven migration procedures necessitates human intervention when environmental conditions deviate from planned parameters, creating operational bottlenecks that extend project timelines and increase costs. Traditional orchestration engines execute predefined workflows sequentially, lacking the capacity to parallelize tasks dynamically or reroute data transfers via alternative network paths when primary routes experience congestion. Energy-efficient allocation of virtual machines in cloud environments requires dynamic consolidation strategies that respond to workload variations, yet conventional migration frameworks operate with fixed resource assignments that persist regardless of actual utilization patterns [3]. The absence of dynamic VM placement algorithms in migration orchestration results in inefficient resource distribution, where computational capacity remains allocated even during idle periods, contributing to unnecessary energy consumption. Virtual machine consolidation techniques that migrate workloads to fewer physical hosts during low-utilization periods can substantially reduce power consumption, yet migration frameworks rarely incorporate such adaptive mechanisms into their orchestration logic [3]. This rigidity manifests in suboptimal resource utilization patterns, with compute instances remaining allocated throughout migration windows regardless of actual processing requirements. Extended migration timelines result from conservative scheduling that prioritizes data integrity over operational efficiency, often leading to prolonged periods of dual system operation where both source and target environments consume resources simultaneously. The absence of real-time performance feedback mechanisms prevents orchestration systems from detecting and responding to throughput degradation, network latency variations, or storage I/O constraints that emerge during execution. Unnecessary carbon emissions from inefficient compute allocation stem from the failure to consider temporal variations in grid carbon intensity or regional differences in renewable energy availability when scheduling migration activities.

Semantic interoperability challenges data integration across heterogeneous cloud environments, presenting continual challenges in schema alignment, metadata reconciliation, and semantic mapping. Enterprises maintain disparate data models with inconsistent naming conventions, data type representations, and business terminology that evolved independently across departmental boundaries and legacy system generations. Schema heterogeneity manifests at multiple levels, such as structural variations in entity relationships, semantic variations in attribute meanings despite syntactic similarity, and conceptual mismatches in data granularity and aggregation levels. Manual schema mapping efforts consume substantial engineering time, with domain specialists required to examine source and target schemas, identify corresponding elements, and specify transformation rules that preserve semantic integrity during data transfer. The cognitive burden of understanding complex data models spanning hundreds or thousands of attributes across multiple tables introduces significant risk of misinterpretation, particularly when documentation is incomplete or business logic is embedded implicitly within

10.48047/jocaaa.2025.34.11.60

application code rather than explicitly documented in metadata repositories. Schema matching problems involve discovering both direct matches, where single source elements correspond to single target elements, and indirect matches, where complex mappings involve multiple elements or require transformation logic to establish semantic equivalence [4]. Direct correspondences represent straightforward attribute-to-attribute mappings where naming conventions and data types align sufficiently for automated detection, whilst indirect matches necessitate deeper semantic analysis to identify relationships that span multiple schema elements or require computational transformations. The discovery of indirect schema matches poses particular challenges, as these relationships may involve aggregation functions, concatenation operations, or derived calculations that are not immediately apparent from structural inspection alone [4]. Errors in field semantic interpretation propagate through integration pipelines, manifesting as data quality issues that require expensive remediation cycles and potentially compromising analytical insights derived from integrated datasets. Schema mapping projects in enterprise environments commonly encounter scenarios where identical business concepts are represented through different naming conventions, data types, and structural patterns across systems, necessitating intricate transformation logic to achieve semantic equivalence. Existing metadata management tools provide cataloguing capabilities through automated discovery and lineage tracking, enabling technical metadata collection regarding table structures, column names, and data types. However, these tools lack the intelligent interpretation of contextual relationships between data elements across source and target systems; they depend on manual annotation and business glossary maintenance to capture semantic meaning. Absence of automated semantic inference creates persistent integration bottlenecks, especially in scenarios involving real-time data pipelines where schema evolution in source systems requires equivalent updates of transformation logic and target schemas.

The Carbon Footprint Gap

While cloud providers increasingly report sustainability metrics through environmental dashboards and carbon footprint calculators, migration and integration workflows rarely incorporate carbon awareness into operational decision-making. Cloud infrastructure operates across geographically distributed data centres with significantly different carbon intensity profiles determined by regional electricity grid composition, renewable energy penetration, and temporal variations in generation sources. Data transfers scheduled during peak renewable energy availability periods or routed through low-carbon regions could significantly reduce environmental impact compared to migrations executed during fossil fuel-dependent generation periods or through high-carbon availability zones. Dynamic resource allocation strategies that consolidate virtual machines onto fewer physical servers during periods of reduced workload can achieve measurable energy savings, yet such techniques remain underutilized in migration orchestration contexts [3]. Temporal carbon intensity patterns showcase predictable diurnal variations, with solar generation contributing substantially during daylight hours in regions with significant photovoltaic capacity, while wind generation patterns follow region-specific meteorological conditions. Regional carbon intensity variations across cloud provider availability zones can vary substantially, with data centres powered predominantly through renewable sources demonstrating markedly lower emissions per unit of compute compared to centres reliant on coal or natural gas generation. The absence of carbon-aware orchestration represents a missed opportunity to align technical operations with organizational sustainability commitments and regulatory requirements for emissions reporting under frameworks such as Greenhouse Gas Protocol standards and emerging climate disclosure mandates. Migration workloads characterized by flexible completion timelines present unique opportunities for carbon-conscious scheduling, wherein deferring non-critical transfers to periods of lower grid carbon intensity achieves emissions reductions

10.48047/jocaaa.2025.34.11.60

without compromising operational requirements. Modern orchestration platforms lack integration with real-time carbon intensity data sources, preventing automated decision-making that considers environmental impact alongside traditional performance and cost metrics. The computational overhead of large-scale data migrations, including data extraction, transformation, network transfer, and target system ingestion, translates directly to energy consumption and associated carbon emissions that remain largely unquantified and unoptimized in current migration practices.

Dimension	Traditional Frameworks	AI-Augmented Frameworks
Resource Allocation	Static provisioning, fixed capacity	Dynamic RL-based adjustment to workload patterns
Network Management	Predetermined sequential routes	Adaptive rerouting based on congestion detection
Schema Mapping	Manual expert interpretation, error-prone	Automated LLM semantic analysis
Carbon Management	Cost-focused, ignores emissions	Carbon-intensity-driven temporal and regional routing
Performance Monitoring	No real-time feedback loops	Continuous telemetry with policy refinement
Adaptability	Template-driven, requires human intervention	Self-learning through environmental interaction

Table 1. Traditional Versus AI-Augmented Migration Frameworks [3, 4].

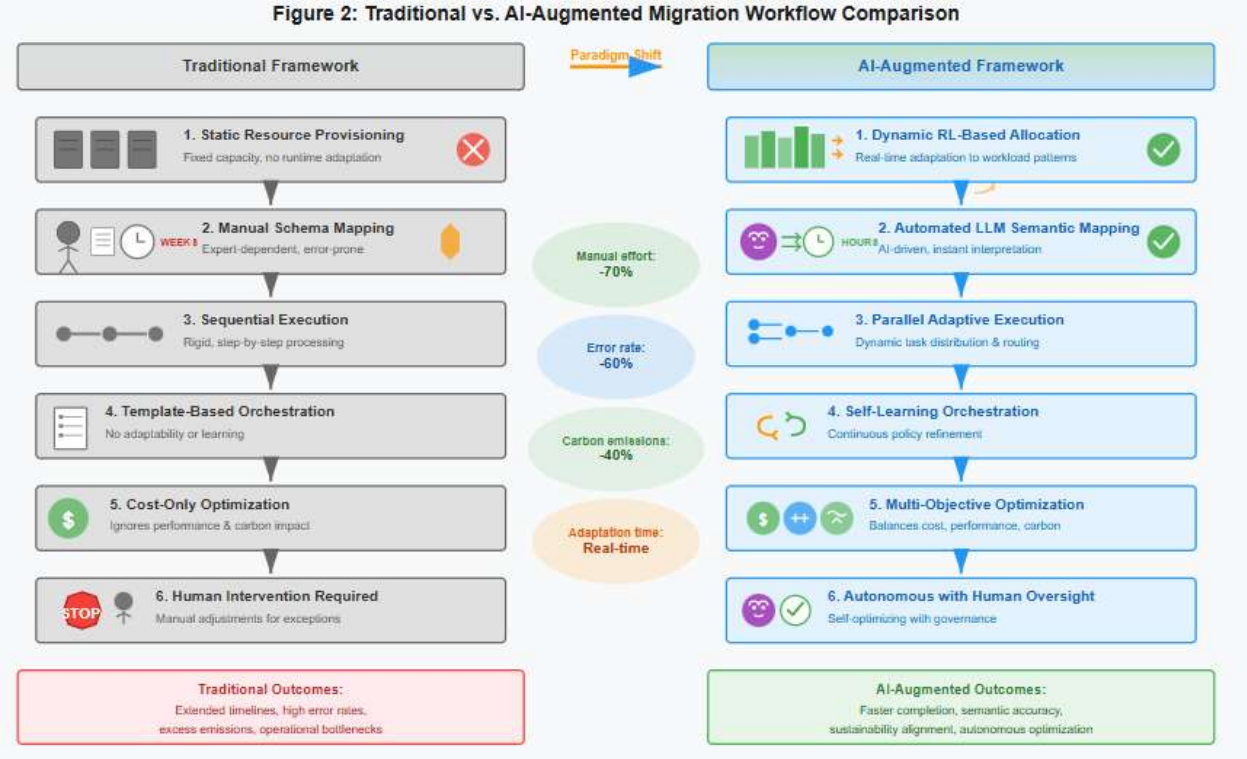

Fig 1. Traditional vs. AI-Augmented Migration Workflow Comparison [3, 4].

10.48047/jocaaa.2025.34.11.60

Proposed AI-Augmented Architecture

Reinforcement Learning for Dynamic Optimization

The proposed architecture deploys RL agents observing the pipeline state vectors of current throughput rates, queue depths, compute utilization patterns, network latency distributions, and regional carbon intensity levels. Such agents interact with the migration environment through actions that range from tuning parallelism levels and rerouting data flows across alternative network paths to scaling compute resources or deferring noncritical transfers to periods of low carbon intensity. State representation includes multi-dimensional feature vectors that express temporal patterns in resource consumption, instantaneous performance metrics, and environmental considerations influencing orchestration decisions. Deep reinforcement learning for job scheduling and resource management in cloud computing corresponds to a number of methodological categories, including value-based approaches such as Deep Q-Networks and their variants, policy gradient methods such as actor-critic architectures, and hybrid techniques combining multiple learning paradigms. The reward function involves multi-objective optimization, balancing completion time, resource cost efficiency, and carbon impact reduction; weighted combinations allow flexible prioritization based on organizational objectives and operational constraints. Model-free reinforcement learning methods prove particularly apt for cloud resource management scenarios characterized by hard-to-model environmental dynamics, enabling agents to learn optimal policies directly from interactions without predetermined system models.

The RL framework relies on the continuous state spaces of pipeline telemetry and discrete action spaces of orchestration decisions. By iteratively experimenting in simulated and production environments, policy networks learn optimal decision policies, and the more migration scenarios are encountered, the decision policies get progressively refined. In actor-critic architectures, the representation of the policy is separated from the value estimation to allow stable learning dynamics where policy gradients guide exploration and value functions provide variance reduction for gradient estimates. Transfer learning underpins generalization across diverse data types and cloud configurations by agents that have been trained on specific workload patterns, allowing them to exploit shared representations of resource dynamics and performance characteristics persistent across heterogeneous environments. Additionally, at the algorithm level, deep reinforcement learning involves selecting neural network architectures for function approximation, designing the reward shaping strategy to yield informative learning signals, and implementing experience replay mechanisms that enhance sample efficiency during training [5]. Conducting pre-training on synthetic workloads generated through simulation environments accelerates policy convergence when deploying agents to production systems, while fine-tuning allows adaptation to organization-specific infrastructure characteristics and operational requirements. Multiple-agent coordination frameworks decompose complex orchestration problems into hierarchical decision structures where strategic objectives are set by high-level agents, while specialized agents manage tactical resource allocation within constrained solution spaces.

LLM-Based Semantic Integration

Large Language Models are semantic interpretation engines, capable of processing schema definitions, data dictionaries, and business glossaries to produce contextual mappings of source to target systems. The LLM component analyzes field names, data types, sample values, and descriptive metadata to infer semantic equivalence and transformation requirements. This approach goes beyond simple matching of names for understanding conceptual relationships, hierarchies of structures, and detection of embedded business rules in data models. In fact, pattern-exploiting training methods have shown that much smaller language models can achieve competitive few-shot learning performance by effectively leveraging task

10.48047/jocaaa.2025.34.11.60

demonstrations contained in the input context [6]. Prompt engineering techniques within the Semantic Integration pipeline instruct the LLMs to create structured mapping specifications, SQL transformation logic, and validation rules. Carefully phrased prompts provide examples of desired output formats, constrain the mapping generation, and include domain knowledge that contextualizes schema elements within organizational business processes. Their few-shot learning capability lets the language models adapt to new schema matching tasks by conditioning upon small numbers of example mappings given inside the prompt without the need for any gradient-based fine-tuning or parameter updates [6]. Chain-of-thought reasoning enables the model to express its rationale about how this interpretation was reached, thus giving transparency to the human validator while reviewing the mappings generated through intermediate reasoning steps, explaining how semantic correspondences were found. Pattern-Exploiting Training formulates natural language understanding tasks as cloze-style questions where the model has to fill positions that have been masked, relying on the surrounding context for the answer. This allows the much better leveraging of pre-trained knowledge in semantic matching downstream applications [6]. Domain-specific examples tune the model to organizational data modeling conventions and business terminology and provide concrete examples of schema pairs and their correct mappings inside prompt contexts.

Carbon-Aware Orchestration Layer

A carbon orchestration component interfaces with regional carbon intensity APIs to pull real-time and forecasted emissions for cloud regions. This component coordinates with the RL agent to integrate carbon metrics into scheduling decisions, at the same time ensuring that workloads are routed preferably to regions with higher renewable energy availability when latency constraints permit. Carbon-aware scheduling applies temporal shifting strategies, deferring non-urgent transfers to time windows with lower grid carbon intensity while maintaining service level agreements for time-sensitive migrations. Integration with forecasting services of the electricity grid allows for predictive scheduling based on patterns of anticipated renewable generation, solar irradiance predictions, and wind availability forecasts. The orchestration layer maintains up-to-date mappings from cloud provider availability zones to their corresponding electrical grid region, taking into consideration power purchase agreements and renewable energy credits impacting the calculation of effective carbon intensity. Multi-objective optimization strikes a balance between carbon reduction and performance requirements, exploring Pareto frontier solutions for optimal trade-offs across conflicting objectives.

10.48047/jocaaa.2025.34.11.60

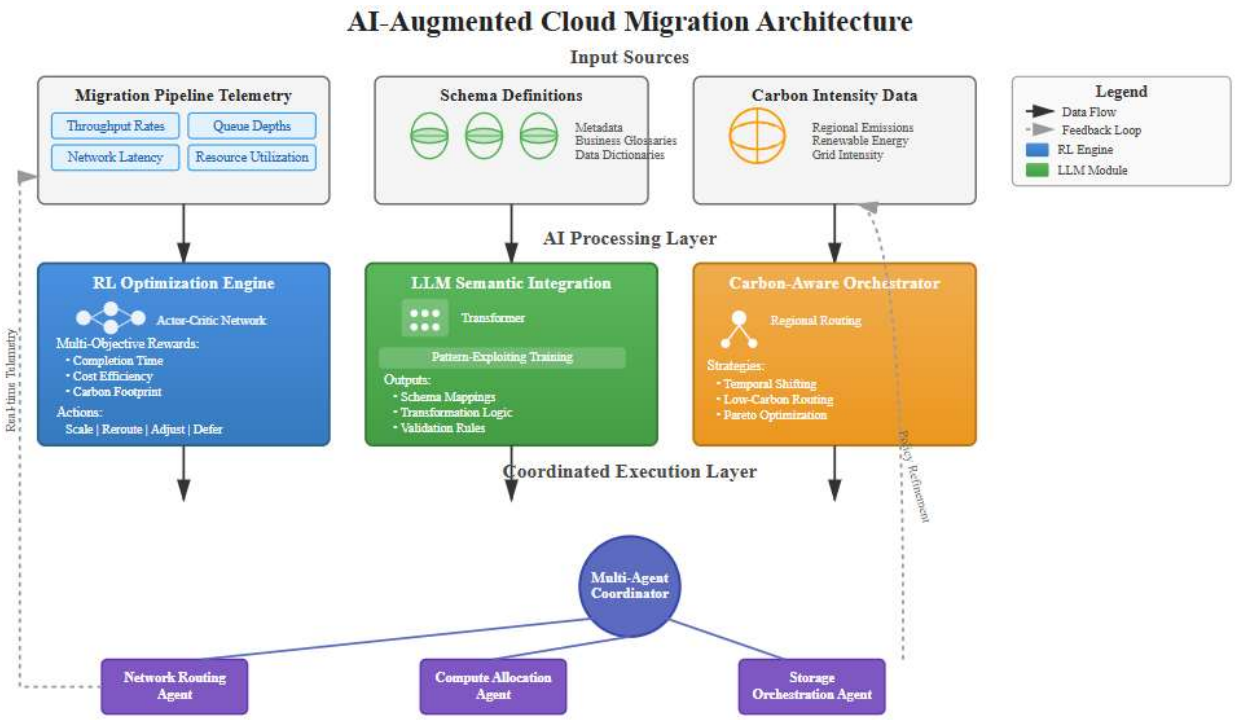

Fig 2. AI-Augmented Cloud Migration Architecture [5, 6].

Component	Key Functions	Implementation Approach
RL Optimisation	Balances completion time, cost, and carbon impact	Actor-critic networks with multi-objective rewards
LLM Integration	Automates schema mapping and semantic interpretation	Transformer models with prompt engineering
Carbon Orchestration	Routes workloads to low-carbon regions and time windows	Real-time carbon intensity API integration
Multi-Agent System	Coordinates specialised agents for parallel decisions	Hierarchical agents with message-passing protocols

Table 2. Core Architectural Components [5, 6].

Technical Components and Integration

Multi-Agent Coordination Framework

The architecture implements a hierarchical multi-agent system where specialized RL agents manage distinct aspects of the migration pipeline. A coordinator agent maintains global optimization goals whilst delegating tactical decisions to subordinate agents in charge of network routing, compute allocation, and storage orchestration. This decomposition enables parallel decision-making whilst retaining coherent system-wide behavior thanks to shared reward signals and inter-agent communication protocols. Hierarchical multi-agent reinforcement learning frameworks address scalability challenges inherent in complex cloud orchestration scenarios by partitioning the decision space across multiple specialized agents, each operating within constrained action domains aligned with specific infrastructure components. Partially observable environments characterize many cloud migration scenarios wherein individual agents

10.48047/jocaaa.2025.34.11.60

lack complete visibility into the global system state, therefore requiring algorithms capable of learning effective policies despite incomplete information with respect to environmental conditions and other agents' states [7]. The coordinator agent aggregates local observations from subordinate agents to construct comprehensive system state representations that enable strategic decisions accounting for dependencies and interactions across pipeline components. RL algorithms designed for partially observable Markov decision problems employ techniques such as memory-augmented architectures and recurrent neural networks to maintain internal state representations that capture relevant historical information, compensating for observational limitations [7]. Communication protocols among agents appoint message-passing mechanisms that propagate state information, coordination signals, and reward feedback through the agent hierarchy, ensuring temporal synchronization and causal consistency in distributed decision-making processes. Shared reward functions align individual agent incentives with system-level objectives, preventing suboptimal local optimization that neglects global performance impacts. Subordinate agents function semi-autonomously within their specific domains, leveraging local observations and action spaces tailored to specific resource types, while respecting constraints and coordination signals issued by higher-level agents.

Observability and Feedback Mechanisms

Comprehensive instrumentation captures the fine-grained telemetry of the pipeline components, consisting of data ingestion rates, transformation processing latencies, network transfer speeds, and resource consumption patterns. This feeds both the RL state observation space and a separate monitoring system tracking carbon attribution for each stage of the pipeline. Instrumentation of data flows across pipeline stages with distributed tracing frameworks captures timing information, data volume metrics, and resource utilization at detailed levels, enabling thorough performance analysis and bottleneck identification. Aggregation of telemetry streams from heterogeneous infrastructure components in time series databases provides queryable repositories for historical performance data supportive of agent decision-making in real time and offline policy evaluation. Feedback loops provide an avenue for continuous policy refinement as agents observe the consequences of their actions and adjust their strategies accordingly—essentially implementing online learning paradigms where policies evolve incrementally throughout production operation rather than requiring separate phases of training. Carbon attribution mechanisms enable the decomposition of total emissions across the pipeline stages by correlating resource consumption telemetry with regional carbon intensity data, thus allowing the identification of high-impact operations susceptible to optimization through adjustments in the scheduling or architecture.

Semantic Mapping Validation Pipeline

The LLM-generated mappings automatically go through various validation processes: constraint checking, referential integrity verification, and sample data transformation testing. Human-in-the-loop review mechanisms are applied by presenting mapping proposals along with confidence scores and supporting rationale, which enable domain experts to validate or correct interpretations. Active learning frameworks select instances informative for humans to annotate strategically, while query strategies identify samples that maximize the expected information gain optimally, balancing model performance improvement against the effort of labeling [8]. Active learning strategies thereby prioritize human review efforts toward mappings characterized by high uncertainty or low confidence scores; they maximize validation efficiency by focusing expert attention on ambiguous cases while automatically accepting high-confidence predictions. Uncertainty sampling, query-by-committee, and expected model change—established query strategies—select candidate instances most likely to enhance model performance when

10.48047/jocaaa.2025.34.11.60

labeled by domain experts [8]. Accepted mappings add knowledge to a knowledge base that benefits future generations of LLMs, for example, through retrieval-augmented generation patterns in which similar historical mapping examples retrieved based on schema similarity serve as additional context during inference. Similarly, interactive machine learning frameworks with iterative refinement cycles allow models to be adjusted through human feedback on initial mapping proposals as the semantic interpretation accuracy improves gradually through domain expert corrections. Confidence scoring builds upon model uncertainty estimates as calculated from output probability distributions, attention weights, and disagreement measures of ensembles to quantify prediction reliability. Similarly, automated testing protocols will be executed against sample data sets in validation workflows.

Component	Functionality	Key Mechanisms
Multi-Agent Coordination	Hierarchical decision-making across pipeline aspects	Shared reward signals, inter-agent communication
Observability System	Captures telemetry from all pipeline stages	Time-series databases, distributed tracing, carbon attribution
Validation Pipeline	Verifies LLM-generated mappings	Automated testing, human-in-the-loop review, confidence scoring
Knowledge Base	Stores validated mappings for future reference	Retrieval-augmented generation, interactive refinement

Table 3. Technical Integration Components [7, 8].

Implementation Considerations and Challenges

Training Data Requirements and Simulation Environments

RL agent training requires diverse scenarios encompassing various data volumes, schema complexities, and environmental conditions. Digital twin environments simulate multi-cloud infrastructure with realistic network topologies, compute heterogeneity, and carbon intensity variability. Synthetic workload generation produces representative migration scenarios whilst avoiding exposure of sensitive production data during training phases. Digital twin technology creates virtual representations of physical systems that enable real-time monitoring, simulation, and optimization across diverse industrial domains, including manufacturing, healthcare, energy systems, and infrastructure management [9]. The application of digital twins to cloud computing environments extends these capabilities to virtualized infrastructure, enabling simulation of migration workflows under varying operational conditions without risking production system stability or data integrity. Digital twin implementations integrate data from multiple sources, including sensor networks, operational databases, and external environmental monitors, to construct comprehensive virtual models that mirror physical system behaviors with high fidelity [9]. Simulation fidelity represents a critical consideration in digital twin development, as discrepancies between simulated and production environments can lead to sim-to-real transfer gaps where policies optimized in simulation perform suboptimally when deployed to actual infrastructure. High-fidelity simulations incorporate stochastic models of network latency distributions, failure probability functions for hardware components, and temporal patterns in workload arrival rates derived from historical telemetry analysis. Synthetic workload generators employ statistical models and generative techniques to produce representative data migration patterns spanning diverse characteristics, including record counts, schema complexities, data type distributions, and referential integrity constraints. Training curriculum design structures the progression of scenarios encountered during agent learning, starting with simplified

10.48047/jocaaa.2025.34.11.60

environments and gradually introducing complexity through increased data volumes, tighter performance constraints, and more dynamic environmental conditions. Transfer learning from simulated to production environments frequently requires domain adaptation techniques that account for distributional shifts between training and deployment contexts, with fine-tuning phases using limited production telemetry to calibrate policies for real-world operational conditions.

Governance and Explainability Requirements

Autonomous decision-making systems must satisfy governance requirements for auditability and explainability. The architecture maintains decision logs documenting RL agent actions, reward signals, and policy selection rationale. LLM-generated mappings include natural language explanations of semantic interpretations, supporting compliance reviews and knowledge transfer to human operators. Policy constraints enforce regulatory requirements and organizational standards as hard boundaries on the RL action space. Explainable artificial intelligence addresses fundamental challenges in understanding and interpreting decisions made by complex machine learning models, particularly deep neural networks whose internal representations remain opaque to human inspection [10]. The taxonomy of explainability approaches encompasses pre-model interpretability through transparent algorithm selection, in-model interpretability via architectural modifications that enhance transparency, and post-model interpretability through explanation generation techniques applied to trained models [10]. Regulatory frameworks increasingly mandate transparency in automated decision systems, particularly where algorithmic choices impact financial outcomes, environmental compliance, or data governance obligations. Decision logs capture complete audit trails documenting the sequence of states, actions, and rewards encountered during policy execution, enabling post-hoc analysis of agent behavior and forensic investigation of unexpected outcomes or policy failures. Explanation methods for machine learning models range from local techniques that interpret individual predictions to global approaches that characterize overall model behavior across the entire input space [10]. Model-agnostic explanation techniques such as LIME and SHAP provide interpretable approximations of complex model decisions through locally faithful surrogate models or Shapley value computations that quantify feature contributions. Constrained reinforcement learning incorporates safety specifications and regulatory requirements directly into the learning process through constraint satisfaction formulations, ensuring learned policies respect hard boundaries on permissible actions regardless of potential reward maximization opportunities. Natural language explanation generation for LLM outputs leverages the model's capacity for textual reasoning to articulate the logical basis for generated mappings, describing semantic relationships identified between schema elements and transformation logic required to achieve equivalence.

Integration with Existing Infrastructure

Realistic deployment requires integration with installed cloud management platforms, data pipeline frameworks, and monitoring systems. The architecture exposes standard APIs for pipeline definition, status monitoring, and manual override capabilities. Backward compatibility with existing orchestration templates enables gradual adoption while organizations build confidence in AI-driven automation. RESTful API designs provide language-agnostic interfaces to enable integration with heterogeneous technology stacks deployed across enterprise environments, while GraphQL endpoints offer flexible query capabilities to support complex observability and monitoring use cases. Manual override mechanisms preserve human agency in autonomous systems, allowing operators to intervene where agent decisions conflict with operational knowledge or contextual factors not represented in the state space. Gradual adoption strategies implement phased rollouts where AI-augmented orchestration operates initially in advisory mode, recommending actions for human approval before progressing to semi-

10.48047/jocaaa.2025.34.11.60

autonomous operation with human oversight and ultimately to fully autonomous execution for well-characterized scenarios. Integration middleware translates between native cloud provider APIs and standardized orchestration interfaces, abstracting infrastructure heterogeneity while preserving access to provider-specific capabilities that enable optimal resource utilization.

Challenge	Problem	Solution	Governance Approach
Training Data	Limited diverse scenarios without exposing production data	Digital twin simulation with synthetic workloads	Curriculum-based scenario progression
Simulation Accuracy	Performance gaps between simulation and production	High-fidelity stochastic models with domain adaptation	Fine-tuning using production telemetry
Explainability	Neural network opacity for compliance	Decision logs with natural language explanations	Attention visualisation, feature attribution
Integration	Heterogeneous platform compatibility	Standard APIs (REST, GraphQL) with middleware	Manual overrides, phased rollout strategy

Table 4. Implementation Challenges and Solutions [9, 10].

10.48047/jocaaa.2025.34.11.60

Conclusion

The architectural framework presented herein addresses fundamental limitations in the state-of-the-art cloud migration and integration workflows via intelligent automation that combines Reinforcement Learning optimization, Large Language Model semantic interpretation, and carbon-aware orchestration. Traditional migration frameworks based on static templates and manual configuration are inadequate for a dynamic multi-cloud environment characterized by fluctuating performance conditions, heterogeneous data models, and increasing sustainability accountability. Reinforcement Learning agents enable adaptive orchestration to respond continuously to runtime telemetry, learning optimal strategies balancing multiple objectives such as throughput, resource efficiency, and environmental impact. The application of policy networks to migration orchestration presents a radical departure from rule-based automation; systems are allowed to develop sophisticated decision strategies through experiential learning rather than predetermined logic. Large Language Model integration addresses the persistent semantic interoperability challenges that automate schema mapping tasks that consume substantial engineering effort while introducing interpretation errors through manual processes. Pattern recognition capabilities inherent to transformer architectures allow for contextual understanding of data structures that surpass syntactic matching, identifying semantic equivalencies across disparate naming conventions and structural representations. Carbon-aware orchestration operationalizes sustainability metrics as first-class optimization criteria, allowing organizations to align technical operations with environmental commitments via intelligent workload scheduling that takes into consideration regional carbon intensity variations and temporal patterns related to renewable energy supply. The convergence of advanced machine learning capabilities with cloud infrastructure orchestration fundamentally transforms migration workflows from reactive execution paradigms to proactive self-optimizing systems. Avenues for future work include empirical validation across diverse operational contexts, investigation of federated learning approaches that enable collective policy improvement across organizations, and exploration of hybrid human-AI collaboration models optimized for the balance between autonomous efficiency and human oversight requirements.

10.48047/jocaaa.2025.34.11.60

References

[1] Abdul Hameed et al., "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems," Springer, 2014. Available: https://d1wqtxts1xzle7.cloudfront.net/66171106/A_survey_and_taxonomy_on_energy_efficien20210403-27222-1hkelij.pdf?1738410120=&response-content-disposition=inline%3B+filename%3DA_survey_and_taxonomy_on_energy_efficien.pdf&Expires=1762167078&Signature=YytYRBFJh2j-i~wP7KJ4BNfmSE5J9qj5MRQHd04yqi7Z~EaBbjbcb1umv9iITV2I9RHNXULT1oMqH75RhF-RVbGVHUFIqgt-qgP3jKksnoYwagctH1tDCpHu-ORtSeqRp-RdFHx~ZlhjH8ma090It0-whUB8Oacxke5Ro5HEo75KbxlUcgd9uFCVI4Ii5b1ryP3PCq2oVt5Q7vF4VeQJ6IIpZwb~X2lM-sXbTmDVS4hTVp7YGrMBZkxMAA5-1Kr1~g8Km2R7VYphjxa-mPDWEhqw9g0YimhT6FXLnsxuEFzTgguiLTmQFfbrl2yPq-gB7qVroKXNPpU4HZQBaKaxCw__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

[2] Christopher Buss, "Towards Scalable Schema Mapping using Large Language Models," ACM, 2025. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3737412.3743490

[3] Anton Beloglazov and Rajkumar Buyya, "Energy Efficient Allocation of Virtual Machines in Cloud Data Centers," IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010. [Online]. Available: https://www.researchgate.net/profile/Anton-Beloglazov/publication/220941227_Energy_Efficient_Allocation_of_Virtual_Machines_in_Cloud_Data_Centers/links/0fcfd5092eebb2b8f7000000/Energy-Efficient-Allocation-of-Virtual-Machines-in-Cloud-Data-Centers.pdf

[4] Li Xu and David W. Embley, "Discovering Direct and Indirect Matches for Schema Elements," ResearchGate. [Online]. Available: https://www.researchgate.net/profile/Li-Xu-5/publication/4011022_Discovering_direct_and_indirect_matches_for_schema_elements/links/557e32b308aec87640dc6447/Discovering-direct-and-indirect-matches-for-schema-elements.pdf

[5] Yan Gu et al., "Deep Reinforcement Learning for Job Scheduling and Resource Management in Cloud Computing: An Algorithm-Level Review," arXiv, 2025. [Online]. Available: https://arxiv.org/pdf/2501.01007

[6] Timo Schick and Hinrich Schütze, "It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners," arXiv, 2021. [Online]. Available: https://arxiv.org/pdf/2009.07118

[7] Tommi Jaakkola et al., "Reinforcement Learning Algorithm for Partially Observable Markov Decision Problems," NeurIPS. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/1994/file/1c1d4df596d01da60385f0bb17a4a9e0-Paper.pdf

[8] Alaa Tharwat and Wolfram Schenck, "A Survey on Active Learning: State-of-the-Art, Practical Challenges and Research Directions," MDPI, 2023. [Online]. Available: https://www.mdpi.com/2227-7390/11/4/820

[9] Maulshree Singh et al., "Applications of Digital Twin across Industries: A Review," MDPI, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/11/5727

[10] AMINA ADADI AND MOHAMMED BERRADA, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," IEEE Access, 2018. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8466590