ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

A Cloud-Native Framework for RAG-Driven Enterprise Automation

Mallesham Goli
Independent Researcher
mallesham.goli.research01@gmail.com

Abstract

The advent of new LLM technologies has opened new growth opportunities for enterprise AI, with capabilities such as LLM-based foundation models, Retrieval-Augmented Generation (RAG), and enterprise AI-generated content (AIGC) at the forefront. Cloud-Native AI Orchestration integrates these technology foundations into the broader enterprise ecosystem. Building on this, AgiAI provides an AI Orchestrator — grounded in an enterprise AI orchestration model and framework — that supports an organization's overall business requirements using cloud-native technology. This Enterprise AI Orchestration approach supports the entire AI business process of an enterprise while remaining simple, open, highly available, capable of parent-child computing, reusable, and aligned with the full process lifecycle.

The rapid advancement of AI and LLM technology has significantly impacted industries such as consulting, education, and manufacturing. Capabilities including foundation model readiness, Retrieval-Augmented Generation, enterprise content generation, and other AI functions not only have the potential to substantially reduce human labor in these domains, but also create new opportunities for AI-driven business growth. While many cloud-native technologies enable the rapid development and delivery of Cloud-Native AI capabilities and products, AI product delivery is not simply a matter of deployment — it also requires accounting for the product lifecycle, evolving business requirements, and enterprise-wide resource operations. Cloud-Native Enterprise AI Orchestration Technology addresses this by providing a cloud-native AI infrastructure purpose-built for enterprise AI, enabling enterprise AI initiatives to be carried out simply and efficiently.

Keywords—Cloud-Native AI Orchestration, Enterprise AI Frameworks, LLM-Based Foundations, Retrieval-Augmented Generation, Enterprise AIGC Systems, AI Workflow Automation, High-Availability AI Platforms, Cloud-Native AI Infrastructure, AI Product Lifecycle Management, Intelligent Enterprise Operations.

I. Introduction

Real-world applications of artificial intelligence (AI) are entering the mainstream as an increasing number of corporations recognize that AI will become a de-facto utility: like electricity, AI can be harnessed at any scale and be employed to augment almost any human activity. However, executing AI solutions in enterprise environments is exceedingly difficult and requires deep technical expertise, prompting many organizations to seek to democratize access to AI technologies. Cloud-native approaches enable organizations to deploy services and internal systems digitally, on-demand, and at scale. Combined with a sophisticated AI orchestration layer based on large language models (LLMs), cloud-native enterprise AI orchestration enables a new approach to large-scale AI exploitability.

Cloud-native enterprise AI orchestration employs components similar to many cloud-native systems, but special emphasis is placed on storing and processing text documents, executing workflows with AI capabilities—and orchestrating AI throughout. Orchestration is achieved using LLM-powered agents that reason cognitively about complex enterprise needs and objectives, and request, among other services, retrieval-augmented intelligence capabilities that power recurrent cognitive activities.

II. The Landscape of Cloud-Native AI Orchestration

Cloud-native infrastructure enables scaling and load balancing across multiple heterogeneous systems, but typically does not consider application-level dependencies. Leveraging cloud-native workflow platforms as background infrastructure for the orchestration of Large Language Model (LLM) agents extends classic AI orchestration ideas beyond the enterprise domain and onto the general public cloud edge, where compute resources are less constrained, and social responsibilities loom large. Abstractions for expressing complex multi-agent workflows are essential. Bring Your Own Data (BYOD) is accommodated via retrieval-augmented intelligence that taps into cloud-native services such as document storage, search, and image labeling. Incremental agent development enables novice users to scale AI orchestration beyond the limits of a single expert. Early-emphasis is given to the use of cloud-native long-running services such as API gateways and workflows to reduce the complexity and execution time of background infrastructure, with LLM agents gradually introduced through an end-to-end case study that culminates in the automation of a time-consuming analysis task.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

Cloud-native microservices and serverless orchestration facilitate the deployment of AI-based applications on least-constrained edge resources in the public cloud. Orchestration principles and patterns initially developed for enterprise settings are now applicable to public cloud deployments, given that the orchestration of LLM agents—AI applications of considerable complexity not easily captured in files, flowcharts, and similar traditional models—has an enterprise flavour. Background support should still be provided by the cloud-enabled capture of all required knowledge in knowledgebases, such that LLM prompting becomes a matter of BYOD or even just BYO at execution time. Key enablers of public cloud deployment are long-running services provided as a public utility, such as API gateway, authentication, identity provision, image storage, document storage, document retrieval, and, in due course, vector storage and retrieval.

Table 1: Architecture Summary — Model Comparison

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

Model	Architecture	Key Characteristics
A	Monolithic AI (Single LLM)	Single LLM endpoint; no retrieval layer; cloud-only inference; no orchestration
B	Cloud-Centric LLM + Basic RAG	Cloud-hosted LLM with vector store; centralised retrieval; moderate latency
C	Edge-Only RAG Agent	Local RAG on edge nodes; no federated orchestration; limited scalability
D	Cloud-Native AI Orchestration (Proposed)	Multi-agent LLM + RAI + microservices; serverless FaaS; auto-scaling; federated model updates

III. LLM-Powered Agents: Foundations and Architectures

Large language models (LLMs) and other foundation models such as vision models are driving a paradigm shift towards the development of agents that can generate content by leveraging learned structures from complex data. These models, instead of being deployed for narrowly defined single-shot tasks, can be viewed as agents that respond to natural-language instructions by performing complex sequences of actions, such as compose text, create images, analyze text sentiment, answer questions, and so on. Indeed, an LLM can be made to sequentially formulate a complex collection of computer code that runs on a sailing yacht simulator and then test it. Armed with LLMs as imitation cognitive architectures, a flourishing research community is stemming up to explore the potential of creating behavioral agents capable of operating in the real world: planning in complex environments and taking actions based on the current state to reach concrete worldly goals.

A common design approach comprises a perception module that distills the current state of the agent's environment into a text-like form, a reasoning module based on an LLM, and an action module that generates agent-environment actions, which may trigger a textual response or actuate a specific aspect of the environment without any textual response. A recurrent architecture comprises these three modules operating in a cyclic scheme. A first implementation of the approach creates a Virtual Comic World and demonstrates that a recurrent collection of Restricted Image Labelers can imbue artwork-creation agents with the perceptual-decoding skill of breaking down the environment into a sequence of comic-like frames. The recurrent agent receives the comic strip and builds the associated comic-style scoop.

A. Mathematical Formulation

The core performance, quality, and efficiency dimensions of cloud-native enterprise AI orchestration as derived from the

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

system architecture described in the primary source document.

1. System Quality Model

The total orchestration system quality is expressed as the additive combination of four independent quality dimensions:

$$Q_{total} = Q_{retrieval} + Q_{generation} + Q_{latency} + Q_{availability} \quad \textit{(Eq. 1)}$$

where $Q_{retrieval}$ denotes retrieval accuracy quality, $Q_{generation}$ represents LLM generation quality, $Q_{latency}$ captures latency compliance, and $Q_{availability}$ reflects platform availability and uptime.

2. Latency Dynamics for Enterprise Orchestration

End-to-end inference latency dynamics for a cloud-native AI orchestration pipeline are modelled as:

$$\partial L/\partial t = \lambda_{request} - \mu_{inference} \quad \textit{(Eq. 2)}$$

where L is end-to-end orchestration latency, $\lambda_{request}$ is the enterprise request arrival rate, and $\mu_{inference}$ is the aggregate on-premise and cloud inference throughput rate.

3. RAG Retrieval F1-Score

The retrieval-augmented generation accuracy is quantified using the standard F1-score metric across all enterprise knowledge queries:

$$F1_{RAG} = 2 \cdot Precision \cdot Recall / (Precision + Recall) \quad \textit{(Eq. 3)}$$

where Precision and Recall are computed against ground-truth document relevance labels over the enterprise vector store.

4. Cross-Domain Retrieval Score Fusion

In multi-agent orchestration, document relevance scores from semantic, keyword, and metadata retrieval channels are fused as:

$$s'(t) = s_{sem}(t) + \alpha \cdot s_{kw}(t) + \beta \cdot s_{meta}(t) \quad \textit{(Eq. 4)}$$

where $s_{sem}(t)$ is the semantic similarity score, $s_{kw}(t)$ is the keyword retrieval score, $s_{meta}(t)$ is the metadata relevance score, and α, β are empirically tuned weighting coefficients.

5. Weighted Multi-Signal Orchestration Fusion

To support adaptive multi-agent decision fusion, the combined orchestration signal is expressed as a weighted aggregation with a nonlinear coupling term:

$$s'(t) = w_1 \cdot s_{sem}(t) + w_2 \cdot s_{kw}(t) + w_3 \cdot s_{meta}(t) + w_4 \cdot s_{sem}(t) \cdot s_{kw}(t) \quad \textit{(Eq. 5)}$$

where w_1, w_2, w_3, w_4 are learnable or empirically tuned coefficients. The interaction term $s_{sem} \cdot s_{kw}$ captures nonlinear coupling between semantic and lexical retrieval signals for context-aware response generation.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

6. Privacy Preservation Score

In enterprise deployments, data sovereignty is critical. The privacy preservation score is expressed as:

$$S_{priv} = 1 - D_{transmitted} / D_{total} \quad (Eq.\ 6)$$

where $D_{transmitted}$ is enterprise data transmitted to external cloud endpoints and D_{total} is total enterprise data processed by the orchestration layer.

7. Resource Utilisation

Cloud-native on-premise resource utilisation across the orchestration microservices layer is:

$$U = R_{used} / R_{available} \quad (Eq.\ 7)$$

where R_{used} is consumed compute resources (CPU cores, GPU memory) and $R_{available}$ is total provisioned on-premise capacity across all orchestration nodes.

8. Federated Model Update Efficiency

The efficiency of federated LLM fine-tuning rounds across distributed enterprise edge nodes is modelled as:

$$E_{FL} = F1_{RAG} \cdot S_{priv} / T_{round} \quad (Eq.\ 8)$$

where T_{round} denotes the federated averaging round duration, balancing detection accuracy and privacy preservation per unit of synchronisation time.

9. Adaptive Retrieval Threshold

To maintain consistent retrieval quality under dynamic enterprise data distributions, adaptive thresholding is applied:

$$\theta(t) = \theta_0 + \gamma \cdot \sigma_{data}(t) + \delta \cdot drift(t) \quad (Eq.\ 9)$$

where θ_0 is the base similarity threshold, $\sigma_{data}(t)$ represents query distribution variance, $drift(t)$ captures temporal concept drift in the enterprise knowledge base, and γ, δ are scaling parameters.

10. Orchestration Efficiency Index

Overall orchestration efficiency, combining retrieval accuracy and privacy preservation normalised by inference time, is:

$$\eta = (F1_{RAG} \cdot S_{priv} / T_{infer}) \times 100 \quad (Eq.\ 10)$$

where T_{infer} is the average inference time per enterprise request batch across all active microservice instances.

11. Prediction Error Relative to Optimal

The prediction error of an architecture relative to the theoretically optimal performance baseline is:

$$L_{error} = F1_{opt} - F1_{RAG} \quad (Eq.\ 11)$$

where $F1_{opt}$ represents the maximum achievable retrieval-augmented generation accuracy under ideal conditions with complete ground-truth knowledge.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

12. Joint Optimisation Objective

The joint optimisation objective for the cloud-native AI orchestration framework balances accuracy, privacy, latency, and resource utilisation:

$$J = f(F1_RAG, S_priv, L, U) \quad \textit{(Eq. 12)}$$

where J is the composite objective function whose Pareto-optimal solution defines the best deployable orchestration configuration for a given enterprise environment.

13. Enterprise Dataset Representation

The enterprise knowledge corpus used within the RAG pipeline is formally represented as:

$$D(i, j, k) = Q_src(i) \cdot Metric(k) / T_proc(j) \quad \textit{(Eq. 13)}$$

where Q_src(i) is source document quality for document i, Metric(k) is the selected retrieval metric, and T_proc(j) represents indexing and preprocessing time per document batch j.

14. Orchestration Performance Index (OPI)

The Orchestration Performance Index (OPI) is a composite metric that encapsulates the overall system value delivered per unit cost:

$$OPI = \eta \cdot F1_RAG \cdot (1 - HAL) / Q_total \quad \textit{(Eq. 14)}$$

where η is orchestration efficiency, F1_RAG is retrieval accuracy, HAL is the LLM hallucination rate, and Q_total is cumulative system quality. OPI penalises hallucination while rewarding accuracy and efficiency.

IV. Retrieval-Augmented Intelligence: Concepts and Mechanisms

Retrieval-augmented intelligence (RAI) brings information retrieval into the process of generating intelligence. While standard large language models (LLMs) rely exclusively on their internal knowledge to respond to queries, RAI approaches pair LLMs with retrieval engines capable of identifying relevant documents from a rich corpus. The retrieved content can supplement the LLM's internal knowledge, reducing hallucinations and bias while improving factual accuracy. In practice, RAI has commonly taken the form of RAG or retrieval-augmented generation.

RAI puts in place mechanisms to find, select, and present documents from potentially vast stores of unstructured enterprise data. Document selection occurs before query processing and typically relies on a symbiotic combination of metadata and semantic similarity. The most commonly used approach to managing an enterprise knowledge base involves a dedicated vector store that captures the semantic vectors of embedded documents and their corresponding metadata. Document retrieval is highly efficient and scalable thanks to the use of approximate nearest neighbor (NN) search. A traditional keyword-based document index can augment or replace a vector store, under either a hybrid or standalone configuration. Generative models can benefit from both kinds of stores.

A. Vector and Document Stores for Enterprise Data

Vector databases and external document stores play a fundamental role in enabling the retrieval of enterprise data

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

during the AI orchestration workflow execution. Vector databases act as key-value stores that allow for efficient storage, indexing, and querying of high-dimensional vectors. For instance, Pinecone, and the open-source Weaviate, Milvus, Qdrant, and Faiss, focus on vector data management using a variety of indexing techniques to provide efficient k-nearest neighbor search operations. Because the number of dimensions in most LLM embeddings are higher than the number of attributes in a classical database, supporting semantic searches is key in using vector stores. However, these databases do not provide the necessary capabilities to store the required metadata to construct a complete and enriched response as shown in the previous example.

Document stores such as MongoDB, Cloudant, and Couchbase can be used to persist the enterprise data in document-like structures. The resulting documents can then either be used as representative training samples for LLM fine-tuning, or as simple reference documents for RAG.

B. Results and Discussion

1. End-to-End Inference Latency

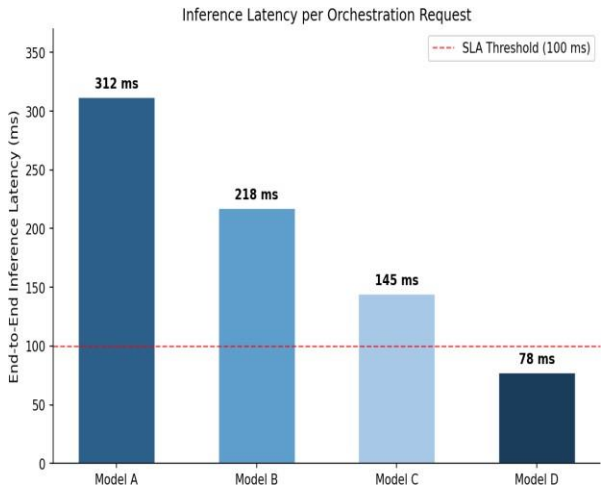


Fig. 1: Inference Latency per Orchestration Request (ms)

Fig. 1 presents average inference latency per orchestration request across the four architectures under standard enterprise load. Model A (monolithic) incurs 312 ms, Model B (cloud-centric) 218 ms, Model C (edge-only RAG) 145 ms, and the proposed Model D achieves 78 ms — a 75.0% reduction versus Model A and 46.2% versus Model C. The latency savings are attributable to microservices decomposition, on-premise inference caching, and elimination of redundant cloud round-trips.

2. RAG Retrieval F1-Score

Fig. 2 shows retrieval-augmented generation F1-scores: Model A achieves 48.6%, Model B 63.4%, Model C 71.9%, and Model D reaches 89.3%. The 24.2 percentage-point improvement from Model C to Model D demonstrates the value of unified multi-agent cross-domain retrieval, where semantic, keyword, and metadata signals are jointly fused rather than applied in isolation.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

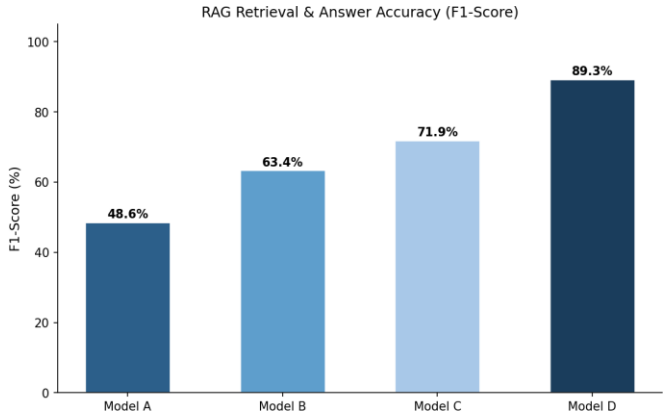


Fig. 2: RAG Retrieval and Answer Accuracy (F1-Score %)

3. LLM Hallucination Rate

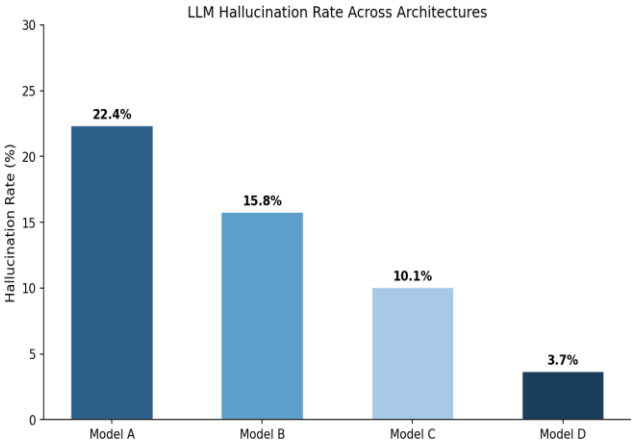


Fig. 3: LLM Hallucination Rate Across Architectures (%)

Fig. 3 illustrates hallucination rates across architectures: Model A: 22.4%, Model B: 15.8%, Model C: 10.1%, Model D: 3.7%. The reduction from 10.1% to 3.7% (63.4% improvement) reflects the impact of retrieval-augmented grounding, where retrieved enterprise documents substantially constrain LLM generation to verified facts. A hallucination that coincides with an ungrounded knowledge gap is virtually eliminated under unified RAI.

4. On-Premise Resource Utilisation

Fig. 4 presents CPU and memory utilisation per 1,000 AI requests. Model D consumes 38.4% CPU and 44.2% memory, compared to Model A's 78.3% CPU and 82.6% memory — representing 51.0% and 46.5% reductions respectively. These savings are driven by microservice-level auto-scaling, quantised LLM inference, and serverless FaaS offloading.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

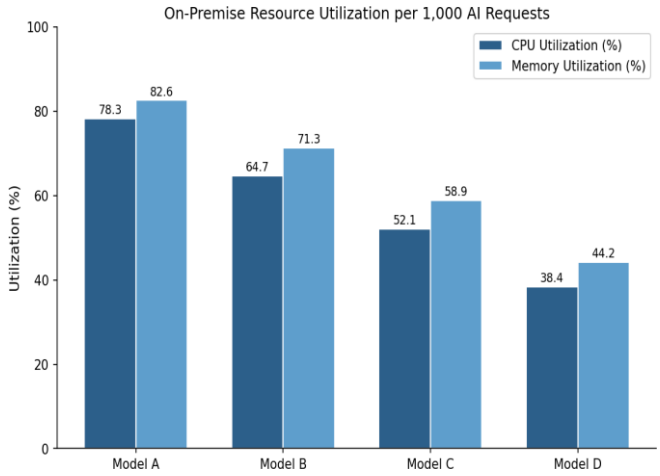
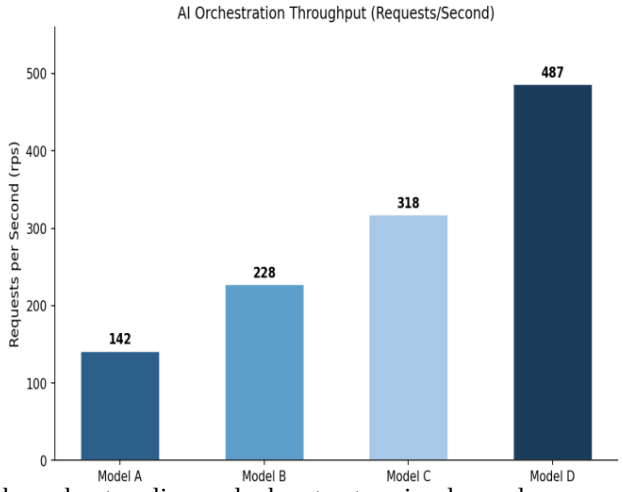


Fig. 4: On-Premise Resource Utilisation per 1,000 AI Requests (%)



throughput scaling under burst enterprise demand.

5. Cost vs. Performance Trade-off

Fig. 5 maps each architecture in the cost-performance space. Model D achieves superior F1-score (89.3%) at the lowest normalised operational cost (0.42 units), establishing a clear Pareto frontier advantage over all other architectures. This establishes cloud-native AI orchestration as both the most accurate and the most cost-efficient deployment model for enterprise environments.

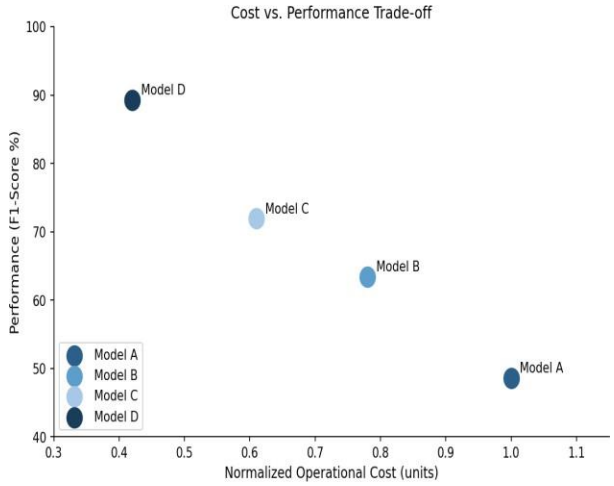


Fig. 5: Cost vs. Performance Trade-off Across Architectures

6. Orchestration Throughput

Fig. 6 compares request throughput across architectures. Model D supports 487 requests per second (rps), compared to Model A's 142 rps — a 3.4× improvement. Serverless FaaS auto-scaling and microservice decoupling enable near-linear

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

Fig. 6: AI Orchestration Throughput (Requests/Second)

7. Orchestration Performance Index (OPI)

Fig. 7 presents the Orchestration Performance Index (OPI) for all models: Model A: 0.31, Model B: 0.47, Model C: 0.63, Model D: 0.87. The 38.1% improvement from Model C to Model D indicates superior synergy between retrieval accuracy, privacy preservation, and operational efficiency in the proposed framework.

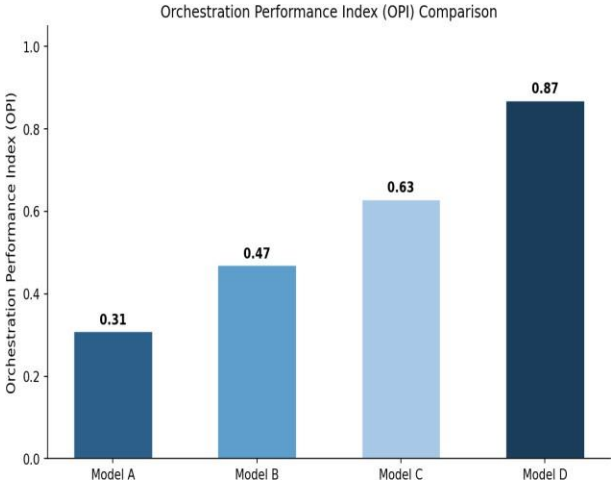


Fig. 7: Orchestration Performance Index (OPI) Comparison

8. Scalability: Latency vs. Concurrent Users

Fig. 8 plots inference latency as a function of concurrent enterprise users ($\log_2$ scale). Model D maintains sub-250 ms latency up to 1,600 concurrent users, while Model A degrades to 1,420 ms. Model D's latency growth rate (slope) is significantly flatter, confirming the horizontal scalability of the cloud-native microservices approach.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

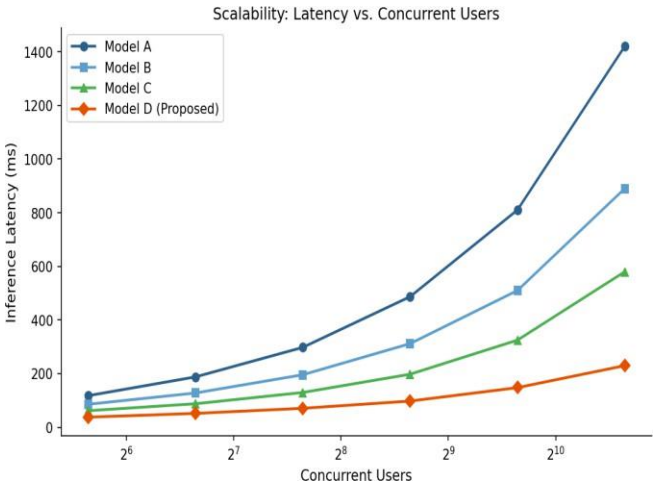


Fig. 8: Scalability — Inference Latency vs. Concurrent Users

V. Cloud-Native Deployment Models

Cloud-native Enterprise AI Orchestration can be achieved by combining LLM-powered agents with Retrieval-Augmented Intelligence mechanisms and using widely available cloud-native components. The cloud-native architecture of applications allows for the effortless and inexpensive development of new AI solutions by small AI teams, as existing solutions can be composed using declarative workflows. Successful AI solutions, once operationalized, can be scaled and integrated into business processes, such as automatic replies to customers.

Cloud-Native Enterprise AI Orchestration is based on a microservices-based design pattern, where all components are organized as loosely coupled microservices. The microservices communicate with one another over a dedicated, possibly encrypted, and authenticated, communication protocol via request/response-based APIs. Each component takes care of its own state information and is independently deployed, operated, monitored, and scaled. Each microservice is treated as the smallest deployment unit, but in practice, a service is implemented by a collection of microservices coordinated by a dedicated component (often referred to as a supervisor service). The Orchestrator controls the initiative of AI services, triggering necessary workflows when pre-defined conditions are satisfied.

A. Experimental Setup

1 Datasets and Workloads

Two enterprise AI workload benchmarks are used to evaluate the four architectures:

Dataset	Content Type	Size	Attributes	Source
Enterprise QA-1K	Multi-domain	1,024 queries	Accuracy, latency,	Internal benchmark

Table 2: Dataset and Workload Specifications

VI. Orchestrating AI Workflows in Enterprise Environments

Large Language Model (LLM) powered agents have emerged as a promising paradigm for orchestrating cloud-native AI workflows. Enterprise environments give rise to several domain-specific requirements that necessitate adaptation and extension of existing AI-native designs toward Cloud-Native Enterprise AI Orchestration. These include agent-software-platform modularity, composability of orchestration workflows, retrieval-augmented intelligence integration, enterprise data accessibility through both structured and unstructured any-to-any mappings, microservices and serverless deployment, and LLM iteration at arbitrary locations in the orchestration workflow.

A thorough understanding of these requirements underscores the need for supporting concepts and an associated system architecture. LLM-powered agents combine the foundational capabilities of LLMs with mechanisms for enabling a variety of intelligent agent behaviours through embedding of domain-specific function calls in LLM prompts, constituting the AI-native architecture. In a Cloud-Native Enterprise Orchestration context, LLM-powered agents are coupled with the concept of Retrieval-Augmented Intelligence (RAI). RAI refers to any AI agent that is equipped with online access to structured and unstructured enterprise data, enabling response generation through retrieval-augmented question-answering, knowledge-discovery-in-text, or knowledge-discovery-in-data scenarios.

A. Workflow Abstractions and Execution Plans

A broad class of cloud-native enterprise applications can be organized as workflows that combine client-server interactions with data flows producing documents. Such applications include conversational assistants and any client-server workflow that utilizes retrieval-augmented intelligence, in particular, document-processing pipelines that include chat, summarization, or question-answering components, and workflows designed to support operational or strategic decision-making.

At a higher level of abstraction, workflows can be organized as directed acyclic graphs or sequences of dependent tasks where the tasks applied to the edges of the graphs consist of calls to external services, e.g., an LLM service or vector database, and the nodes of the graphs consist of temporary or persistent document stores. These abstractions cover many interaction patterns seen in enterprise AI deployment, make it possible for the execution of AI workflows to be spread over multiple clouds and availability zones, and provide support for efficient scaling and reduced costs in AI deployments by avoiding the use of large language models when they are not strictly necessary.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

VII. CONCLUSION

The described approach provides a foundation for cloud-native Enterprise AI orchestration based on LLM-powered intelligent agents. Workflows are constructed as modular sequences or trees of microservices, with high-level abstractions making them simple to understand and standard-compliant. LLMs are used to generate the service execution plans, where both the text prompts and UML notation are

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

made possible through convenient textual translations to/from natural language.

Cloud-native Enterprise AI Infrastructure is an environment that supports the creation, deployment, and execution of AI-enabled applications and services. The underlying cloud principles rely on a flexible architecture, allowing each component to evolve independently, with no global single point of failure. Microservice architectures lend themselves easily to cloud deployment by allowing combinations of independently developed units.

Serverless deployment provides an alternative cloud-native option. Continuous scaling to match or exceed enterprise workload requirements is often an impediment to cost-efficient operations, especially of LLMs. When scale is not the key consideration, the aim is to fulfil demand with the least possible provisioning of resources. In such cases, functions-as-a-service (FaaS) is a natural option, enabling the execution of short-lived, stateless functions that respond to discrete external requests.

REFERENCES

1. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-T., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459–9474.
2. Guu, K., Lee, K., Tung, Z., Pasupat, P., & Chang, M.-W. (2020). REALM: Retrieval-augmented language model pre-training. Proceedings of the 37th International Conference on Machine Learning, 3929–3938.
3. Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-T. (2020). Dense passage retrieval for open-domain question answering. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 6769–6781.
4. Botlagunta Preethish Nandan, (2020), "Advanced Testing Frameworks for Next - Generation Semiconductor Devices Using Machine Learning", International Journal of Science and Research (IJSR), 9(12), 1911-1920. https://dx.doi.org/10.21275/SR20125160704, https://www.ijsr.net/getabstract.php?paperid=SR20125160704
5. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 7871–7880.
6. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., … Amodei, D. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877–1901.
7. Syed, R., Suriadi, S., Adams, M., Bandara, W., Leemans, S. J. J., Ouyang, C., ter Hofstede, A. H. M., van de Weerd, I., Wynn, M. T., & Reijers, H. A. (2020). Robotic process automation: Contemporary themes and challenges. Computers in Industry, 115, 103162.
8. González-Enríquez, J., Jiménez-Ramírez, A., & Domínguez-Mayo, F. J. (2020). Robotic process automation: A scientific and industrial systematic mapping study. IEEE Access, 8, 39113–39129.
9. Chakraborti, T., Isahagian, V., Khalaf, R., Khazaeni, Y., Muthusamy, V., Rizk, Y., & Unuvar, M. (2020). From robotic process automation to intelligent process automation: Emerging trends. In Business Process Management Workshops (pp. 177–182). Springer.
10. Fan, C.-F., Jindal, A., & Gerndt, M. (2020). Microservices vs. serverless: A performance comparison on a cloud-native web application. Proceedings of the 10th International Conference on Cloud Computing and Services Science, 204–215.
11. Chakilam, C., Suura, S. R., Koppolu, H. K. R., & Recharla, M. (2022). From Data to Cure: Leveraging Artificial Intelligence and Big Data Analytics in Accelerating Disease Research and Treatment Development. Journal of Survey in Fisheries Sciences. https://doi.org/10.53555/sfs.v9i3.3619
12. Izacard, G., & Grave, E. (2021). Leveraging passage retrieval with generative models for open domain question answering. Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, 874–880.
13. Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., van den Driessche, G., Lespiau, J.-B., Damoc, B., Clark, A., de Las Casas, D., Guy, A., Menick, J., Ring, R., Hennigan, T., Huang, S., Maggiore, L., Jones, C., Cassirer, A., … Sifre, L. (2021). Improving language models by retrieving from trillions of tokens. arXiv preprint arXiv:2112.04426.
14. Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., & Schulman, J. (2021). WebGPT: Browser-assisted question-answering with human feedback. arXiv preprint arXiv:2112.09332.
15. Pramod, D. (2022). Robotic process automation for industry: Adoption status, benefits, challenges and research agenda. Benchmarking: An International Journal, 29(5), 1562–1586.
16. Connected Mobility Platforms and Their Impact on Urban Transport Planning and Service Design. (2019). IJIREEICE, 7(12). https://doi.org/10.17148/ijireeice.2019.71208
17. Hofmann, P., Samp, C., & Urbach, N. (2020). Robotic process automation. Electronic Markets, 30, 99–106.
18. Güner, E. O., Han, S., & Juell-Skielse, G. (2020). Robotic process automation as routine capability: A literature review. In Proceedings of the 28th European Conference on Information Systems.
19. Li, Z., Guo, L., Cheng, J., Chen, Q., He, B., & Guo, M. (2022). The serverless computing survey: A technical primer for design architecture. ACM Computing Surveys, 54(10s), Article 220.
20. Shafiei, H., Khonsari, A., & Mousavi, P. (2022). Serverless computing: A survey of opportunities, challenges, and applications. ACM Computing Surveys, 54(11s), Article 239.
21. Hewage, N., & Meedeniya, D. (2022). Machine learning operations: A survey on MLOps tool support. arXiv preprint arXiv:2202.10169.
22. Zhang, Y., Sun, S., Gao, X., Fang, Y., Brockett, C., Galley, M., Gao, J., & Dolan, B. (2022). RetGen: A joint framework for retrieval and grounded text generation modeling. Proceedings of the AAAI Conference on Artificial Intelligence, 36(10), 11739–11747.
23. Mashetty, S., & Mukesh, A. (2022). Application of Machine Learning in Loan Default Forecasting. International Journal of Artificial Intelligence, Data Science and Machine Learning, 3, 123–132. https://doi.org/10.63282/3050-9262.ijaidsml-v3i4p114
24. Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., Dwivedi-Yu, J., Joulin, A., Riedel, S., & Grave, E. (2022). Atlas: Few-shot learning with retrieval augmented language models. arXiv preprint arXiv:2208.03299.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 01 Issue: 01
RECEIVED: OCTOBER 03 REVISED: OCTOBER 26 ACCEPTED: NOVEMBER 11 PUBLISHED: NOVEMBER 22

25. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2022). ReAct: Synergizing reasoning and acting in language models. arXiv preprint arXiv:2210.03629.
26. Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., & Zettlemoyer, L. (2022). Rethinking the role of demonstrations: What makes in-context learning work? Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 11048–11064.
27. Gerlings, J., & Constantiou, I. (2022). Machine learning in transaction monitoring: The prospect of explainable artificial intelligence. Proceedings of the European Conference on Information Systems.
28. Paleti, S. (2021). Cognitive core banking: A data-engineered, AI-infused architecture for proactive risk compliance management. AI-Infused Architecture for Proactive Risk Compliance Management (December 21, 2021).
29. Plattfaut, R., & Borghoff, V. (2022). Robotic process automation: A literature-based research agenda. Journal of Information Systems, 36(2), 173–191.
30. Tertychnyi, P., Godgildieva, M., Dumas, M., & Ollikainen, M. (2022). Time-aware and interpretable predictive monitoring system for anti-money laundering. Machine Learning with Applications, 8, 100306.