LTM BlueVerse

POV

MCP Is Not Enough: Why Agentic AI Needs an AI Control Plane

Model Context Protocol (MCP) solved the connectivity problem. The control problem remains. This POV maps the four gateway roles enterprises must evaluate to govern Agentic AI at scale.

Dr. Joyita Chakraborty
Data Scientist, BlueVerse, LTM

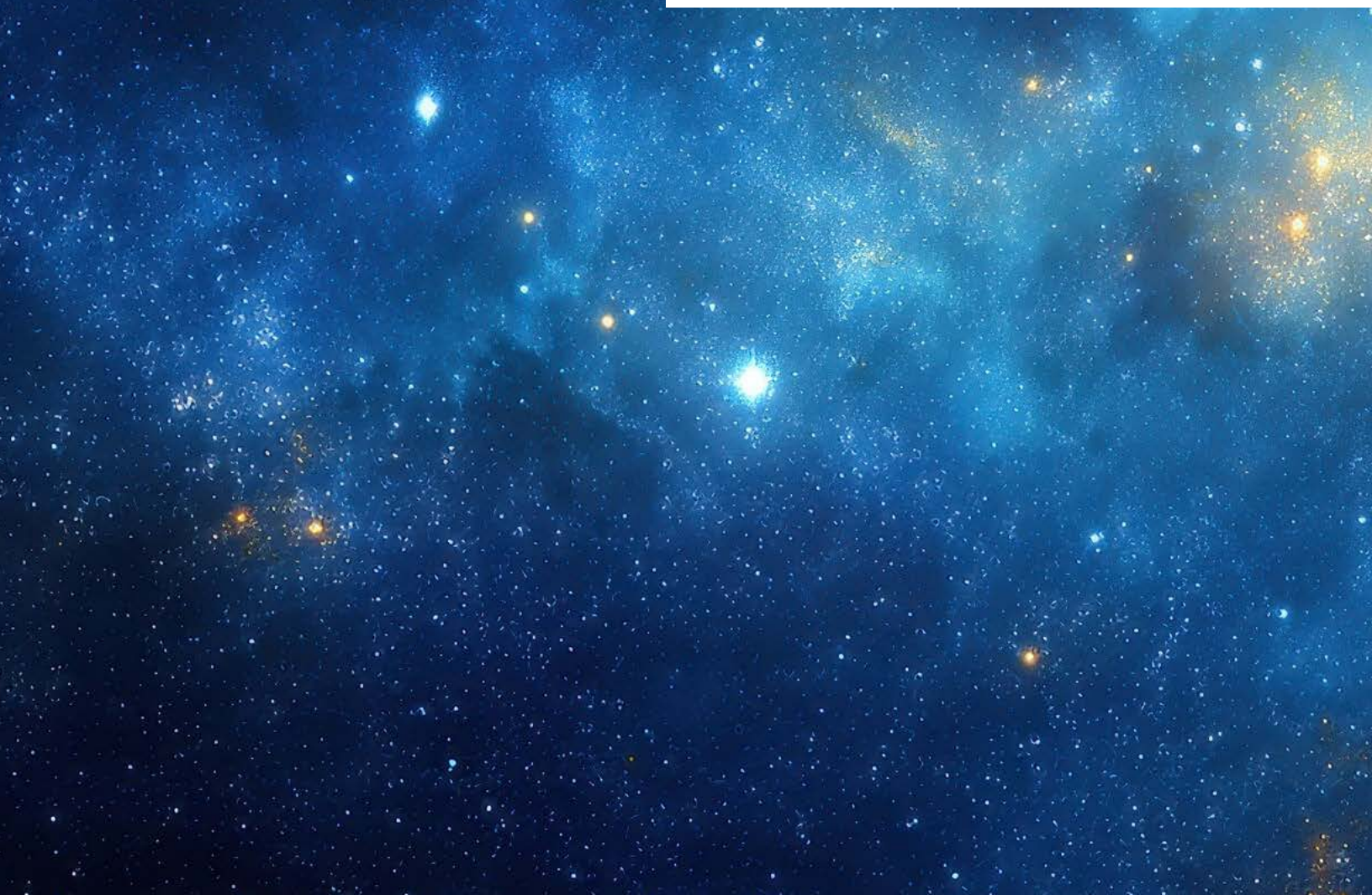

Table of Contents

When Every Team Builds it's Own Agent, Nobody Controls Any of Them

Enterprise AI has entered a phase that looks remarkably familiar. A decade ago, microservices promised modularity and speed. What followed was sprawl, duplication, and years of building platforms to tame the chaos. Agentic AI is heading down the same path.

Organizations aren't experimenting with isolated copilots anymore. They're building interconnected agent ecosystems that automate complex workflows, call tools, and connect across dozens of external systems. A new standard has emerged for this interaction: Model Context Protocol (MCP). It allows large language models to discover and invoke tools dynamically rather than rely on hard-coded integrations. One protocol, one interface for APIs, databases, applications, and services.

MCP solved the communication problem. It also exposed a bigger one.

Hundreds of MCP servers now expose overlapping tools. Different transport methods (STDIO, HTTP, WebSocket, streaming) fragment the ecosystem. Agents invoke other agents with no central governance. Security responsibilities are scattered across teams, each implementing authentication differently. And when something breaks, enterprises can't trace what an agent did, why it did it, or who authorized the action.

The industry figured out how agents talk to tools. It hasn't figured out how enterprises control what agents actually do at scale. The challenge has shifted from connectivity to coordination: enforcing policies, building trust, ensuring accountability, and maintaining operational clarity across regions and clusters.

The protocol exists. The control plane doesn't.

What We Believe: The Gateway is the Real Infrastructure Play

Here's the thesis, stated plainly: the MCP gateway is the enterprise control plane for deploying agentic systems at scale.

Not a nice-to-have middleware layer. Not a routing convenience. The control plane.

Agent systems carry a fundamentally different operational risk profile than traditional software. Their decisions are probabilistic. Their actions chain across systems in ways no single team fully anticipates. Without a gateway layer sitting above MCP, security logic stays trapped inside individual agents. Every tool implements authentication in its own way. Deployments become deeply customized, nearly impossible to standardize, and painful to audit.

We've seen this firsthand. IBM's ContextForge MCP Gateway represents one of the earliest architectures treating the gateway not as a simple tool router, but as a federated control plane capable of managing tools, agents, policies, identity, and observability across distributed environments.

One distinction matters more than any other: MCP standardizes communication. The gateway standardizes reliable communication.

The long-term architectural direction, then, isn't "deploy more MCP servers." It's deploying an agent control plane above MCP that governs how those servers behave, who accesses them, and what gets logged when they act.

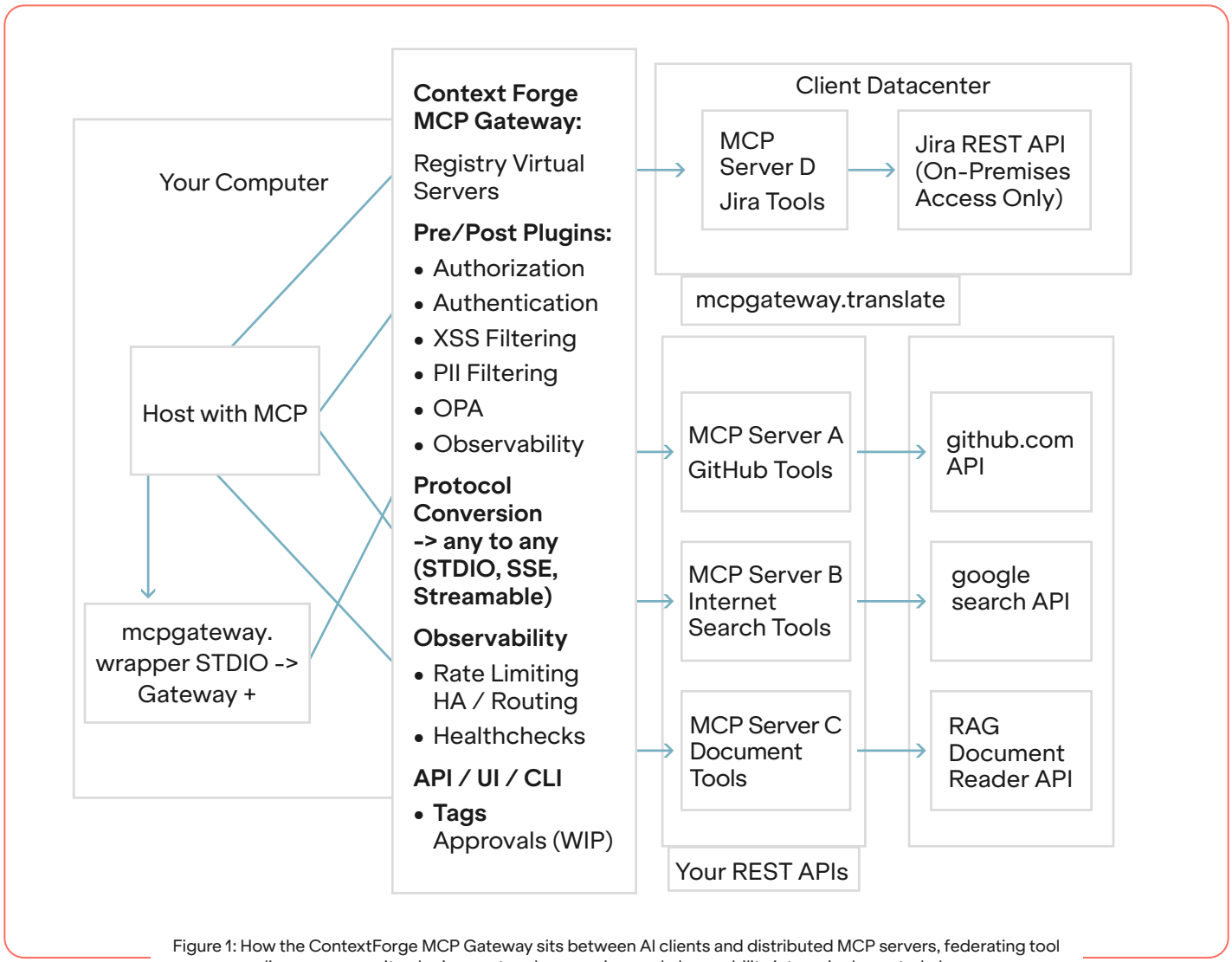

Figure 1: How the ContextForge MCP Gateway sits between AI clients and distributed MCP servers, federating tool discovery, security plugins, protocol conversion, and observability into a single control plane.

Four Gateway Roles Every Enterprise Should Understand

The industry is converging on four distinct gateway categories. Each solves a different problem. Each demands a different architectural choice. Rather than picking tools based on brand preference, organizations should start with a question: What exactly do we need to control?

We've evaluated these gateways across key dimensions: composition, governance, performance, and interoperability. What follows is drawn from our research and hands-on work with enterprise agentic deployments.

1. Composition — Building One AI Environment from Many

Start with the most common scenario. Multiple domain teams, each building their own agents, each connecting to overlapping MCP servers, each implementing tool calls with different schemas, authorization models, and logging standards. No shared catalog. No discovery mechanism. Dozens of agents are doing similar things in slightly different ways.

This is where the IBM ContextForge MCP Gateway operates. It functions as a gateway, registry, and proxy that sits in front of any MCP server, A2A server, or REST API, exposing a unified endpoint for all AI clients. Its core capabilities include:

- Federation across multiple MCP and REST services into a single interface
- A2A (Agent-to-Agent) integration for external AI agents (OpenAI, Anthropic, custom)
- Virtualization of legacy APIs as MCP-compliant tools and servers
- Transport flexibility across HTTP, JSON-RPC, WebSocket, SSE, STDIO, and streamable HTTP
- Built-in authentication, retries, rate-limiting, and Redis-backed caching
- OpenTelemetry observability with Phoenix, Jaeger, Zipkin, and other OTLP backends

We saw this play out clearly in a regulated financial services engagement. Risk, Fraud, and Operations had each built custom agents using the LangChain framework. All three teams connected to overlapping APIs and identical tools, but with different authorization logic, schemas, and logging configurations. Nobody knew what already existed. Agents randomly invoked the same tools through different paths. It was classic microservices sprawl, just dressed up in AI language.

What we did was straightforward. We registered all required MCP servers and REST APIs with the ContextForge Gateway. Tool definitions got standardized overnight. Teams could browse, discover, and reuse tools through a shared catalog instead of rebuilding what already existed. Security teams gained a single point to verify and mitigate risks. The shift from fragmented to federated was visible almost immediately.

2. Governance — Proving What Your Agents Did and Why

Composition solves the duplication problem. It doesn't solve the accountability problem.

In that same financial services engagement, we hit a wall that no tool catalog could fix. Every agent technically worked. But from a compliance perspective, no one could reliably prove which agent accessed which dataset, under whose authority, and for what reason. Auditors asked questions. Nobody had clean answers. The agents were doing the right things. Proving it was a different story entirely.

This is the domain of governance-focused gateways. Two stand out.

- Agent Gateway is an open-source proxy and control plane built for agentic AI workflows across MCP and A2A protocols. It federates MCP servers, centralizes tool discovery, and enforces security policies from a single layer.

- TrueFoundry Agent Gateway routes all agent tool calls through registered MCP servers while enforcing OAuth2, RBAC, policy controls, telemetry, and audit logging. It acts as an intelligent reverse proxy that centralizes tool registration and discovery with enterprise-grade guardrails.

What changed once we introduced a governance gateway in that engagement? RBAC got enforced consistently across every tool call. OAuth2 token validation happened per invocation. Audit logs became centralized, reliable, and auditor-ready. The agents themselves didn't change. The accountability around them did. That distinction matters more than most teams realize until they're sitting across from a regulator.

3. Performance — Keeping Agents Fast Under Pressure

Not every problem is about governance or composition. Sometimes, the problem is pure speed.

We worked on a consumer-facing customer support system embedded in a real-time application serving millions of daily users. Latency expectations were brutal. Anything above 500 milliseconds felt unresponsive to end users. The system ran multiple LLMs across providers, and any single point of failure cascaded into visible user impact. This wasn't an internal tool where a two-second delay gets shrugged off. These were real users, real frustration, real churn.

Bifrost by Maxim AI addresses exactly this. It's a high-performance AI gateway with integrated MCP support and provider-agnostic access to multiple LLMs through a single API. It handles failover, caching, cost-optimized routing, and combined tool/LLM orchestration in self-hosted environments. Built for high-throughput production where every millisecond matters.

By introducing Bifrost as the inference gateway, we could dynamically route requests across providers, optimize for cost and speed, and protect the system with seamless failover below the application layer. The user never noticed. That's the whole point of a well-placed performance gateway. When it works, nobody knows it's there.

4. Translation — Making MCP Work Where it Can't

Some environments simply don't speak MCP. Legacy platforms built around REST and OpenAPI won't be rewritten overnight. The tools work. The protocol doesn't match.

MCPO (MCP-to-OpenAPI Proxy) handles this cleanly. It's a lightweight open-source proxy that translates MCP tool servers into standard REST/OpenAPI endpoints. Not a full gateway with governance. A compatibility bridge.

We encountered this in the same financial services engagement. One consuming platform only supported REST and OpenAPI, making direct MCP integration impractical. Placing MCPO in front of the MCP server exposed the tools as standard REST endpoints with OpenAPI specifications. No rewrite. No heavy infrastructure. The integration happened in days rather than months. Sometimes the most impactful architectural decision is the simplest one.

Managed Cloud Alternatives

For organizations that want zero infrastructure ownership, managed cloud gateways offer a different trade-off:

- **AWS Bedrock AgentCore Gateway:** Fully managed agent runtime within the AWS ecosystem
- **Azure API Management + Foundry Agent Service:** Native Azure integration for agent orchestration
- **Google Apigee:** API management extended to MCP and agent ecosystems

These options trade flexibility for operational simplicity and native cloud integration. In our experience, they work best when the organization has already committed deeply to a single cloud platform and values managed operations over architectural control.

What Separates the Teams That Scale from Those That Stall

We've worked with organizations at very different stages of agentic AI maturity. Some move fast and build durable infrastructure. Others invest heavily and still end up with fragmented, ungovernable systems six months later. The difference is rarely about technology. It's about how teams think about the problem.

Here are the patterns we keep seeing on the wrong side:

Many teams treat MCP as a replacement for APIs rather than what it actually is:	Others assume that because agents reason, they reason safely.	Then there's the organizational blind spot.
A communication standard. The result is tool sprawl instead of tool governance. They build more servers, expose more endpoints, and call it progress. Nobody steps back to ask whether the tools already exist somewhere else in the organization.	They don't. MCP-connected agents expand the attack surface precisely because they chain actions across systems. Our research indicates that MCP ecosystems introduce vulnerability patterns that traditional software environments don't. Autonomous decision-making without centralized guardrails is a risk most security teams haven't fully mapped yet.	Platform teams manage infrastructure. Security teams manage identity. AI teams manage prompts. Agent systems cut across all three domains. Without a gateway layer, no single team owns governance. Everyone assumes someone else is handling it.

Now, the other side. The teams that scale share a few recognizable habits.

They establish an internal AI platform team early and treat tools as shared infrastructure, not team-specific utilities.	From our research across enterprise agentic deployments, these patterns mirror what happened with API management a decade ago.
Tools get versioned, cataloged, and certified before any agent can invoke them. The gateway becomes the single enforcement point for PII filtering, access policies, rate limits, and agent restrictions. Enterprises with regional footprints run regional gateways but synchronize metadata globally.	The organizations that built API platforms early gained compounding advantages. The ones that treated APIs as point solutions spent years untangling the mess. Agentic AI is following the same arc. The teams applying Business Creativity to this challenge, bringing human judgment and intelligent systems together rather than letting agents operate unchecked, are the ones building infrastructure that lasts.

The gateway isn't the finish line. It's the starting point for everything that comes after.

The AI Control Plane isn't Coming. It's Already Late.

MCP gave agents a shared language for talking to tools. That was the easy part.

The hard part is everything above: who gets access, what gets logged, how failures get traced, and where accountability sits when an autonomous system makes a decision that affects real people and real money.

We've spent months inside these architectures. Testing gateways. Watching organizations get this right and get this wrong. The pattern is always the same. Teams that treat the gateway as a proxy keep patching. Teams that treat it as a control plane build something they can actually stand behind when the pressure comes.

Four responsibilities

Composition

Governance

Performance

Interoperability

Every enterprise will eventually need all four. Trying to solve them simultaneously is how projects die in committee. Pick the one that's bleeding. Build around it. Expand later.

One thing stands out from this work more than anything else. The organizations pulling ahead aren't just deploying smarter agents. They're pairing human judgment with intelligent systems, designing governance structures that keep pace with autonomy rather than chasing it.

That's how enterprises Outcreate the complexity of agent governance: not through faster models, but through infrastructure that enables autonomous systems to earn trust before they gain freedom.

Build the control plane now. Or spend the next two years explaining why you didn't.

About the Global AI Unit — Research Team

The Global AI Unit at LTM brings together deep research capabilities and real-world implementation expertise to help enterprises unlock the full potential of Artificial Intelligence. Our Research Team focuses on emerging AI paradigms, including Agentic AI, Large Language Models (LLMs), and AI Observability, to develop actionable insights and strategic recommendations.

We specialize in translating complex AI advancements into client-ready solutions, empowering organizations to scale AI responsibly, efficiently, and with measurable impact.

References

1. *ContextForge MCP Gateway — GitHub Repository, Mihai Criveti,* IBM, 2025, https://github.com/IBM/mcp-context-forge [github.com
2. *ontextForge MCP Gateway Documentation, Mihai Criveti,* IBM, 2025, https://ibm.github.io/mcp-context-forge/ [ibm.github.io]
3. *Available Now: ContextForge MCP Gateway 1.0.0 Beta, Mihai Criveti, IBM Developer,* December 2025, https://developer.ibm.com/blogs/context-forge-mcp-gateway-now-available/ [developer.ibm.com
4. *ContextForge MCP Gateway: The Missing Proxy & Registry for AI Tools, Mihai Criveti, Medium,* June 2025, https://medium.com/@crivetimihai/mcp-gateway-the-missing-proxy-for-ai-tools-2b16d3b018d5 [medium.com]
5. *Taming MCP Chaos with Mihai Criveti from IBM/ContextForge, David Jones-Gilardi and Mihai Criveti, Langflow (The Flow Podcast),* December 2025, https://www.youtube.com/watch?v=qx6vCtlshro [youtube.com]
6. *ContextForge: MCP Gateway — AI Alliance Community Office Hours, Mihai Criveti, The AI Alliance,* August 2025, https://www.youtube.com/watch?v=S-DlCid50Zo [youtube.com]
7. *Agent Gateway: A Unified Control Plane for AI Workflows, TrueFoundry,* TrueFoundry, 2025, https://www.truefoundry.com/agent-gateway [truefoundry.com]
8. *Agent Gateway — Next-Generation Agentic Proxy for MCP and A2A, Solo.io and Linux Foundation,* agentgateway.dev, 2025, https://agentgateway.dev/ [agentgateway.dev]
9. *Bifrost: The Fastest Enterprise AI Gateway, Maxim AI,* GitHub, 2025, https://github.com/maximhq/bifrost [github.com]
10. *MCPO: A Simple, Secure MCP-to-OpenAPI Proxy Server, Open WebUI,* GitHub, 2025, https://github.com/open-webui/mcpo [github.com]
11. *Amazon Bedrock AgentCore, Amazon Web Services,* Amazon, 2025, https://aws.amazon.com/bedrock/agentcore/
12. *Azure AI Foundry API via API Management, Microsoft,* Microsoft Learn, 2024, https://learn.microsoft.com/en-us/azure/api-management/azure-ai-foundry-api
13. *Apigee and MCP: Bridging Enterprise APIs with AI Agent Ecosystems, Google Cloud Apigee Team,* Google Cloud, 2024, https://cloud.google.com/apigee

About the Author

Dr. Joyita Chakraborty

Data Scientist, BlueVerse, LTM

Dr. Joyita Chakraborty is a Data Scientist at LTM with 7+ years of experience in applied data science, machine learning, and GenAI. She has published research on evolving networks, time-series analytics, anomaly detection, and the evaluation of agentic GenAI systems.