Partha Sarathi Samal, 2025, 13:6
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Bridging The Gap: A Model Context Protocol (MCP) Framework for AI Agents in Media Supply Chains

Partha Sarathi Samal[1], Suresh Kumar Palus[2], Sai Kiran Padmam[3], Bhavan Kumar B R[4]

[1]Independent Researcher Connecticut, USA, [2]Independent Researcher Pennsylvania, USA
[3]Independent Researcher New Jersey, USA, [4]Independent Researcher Colorado, USA

Abstract- Media supply chains currently suffer from severe fragmentation. Tools for transcoding, quality control (QC), and media asset management (MAM) exist in isolated silos. Connecting these systems requires custom, brittle API integrations that fail when vendor specifications change. AI Agents offer a solution through autonomous planning and execution. However, agents struggle to connect to diverse legacy systems without a standardized interface. This paper proposes a unified framework based on the Model Context Protocol (MCP). MCP acts as a universal transport layer. It allows AI agents to discover, query, and manipulate media assets securely across heterogeneous environments. We outline a "Media-MCP-Agent" architecture that enables autonomous agents to parse technical metadata, trigger transcode jobs and rectify QC failures without hard-coded scripts. We provide a rigorous analysis of the security implications, specifically regarding prompt injection, and discuss the future of hierarchical agent swarms in broadcast operations.

Keywords: Model Context Protocol, MCP, AI Agents, Media Supply Chain, Automation, Metadata, Quality Control, SMPTE ST 2110, Prompt Injection, Agentic Workflows.

I. INTRODUCTION

The modern media factory operates as a tangle of incompatible tools. A typical studio employs distinct vendors for cloud storage, transcoding, metadata logging, and playout. Humans act as the "glue" layer. They download files, check emails, and move assets manually between buckets. This manual orchestration slows down production and introduces human error.

AI Agents promise to automate these cognitive tasks. Unlike rigid scripts, agents can plan dynamic workflows. They can "see" a video file's technical properties and "decide" to normalize the audio. But agents face a connectivity bottleneck. Every tool exposes a different API schema. Building a single agent that interoperates with ten distinct media systems is computationally expensive and challenging to maintain. We need a standard interface. The Model Context Protocol (MCP) provides this standardization [1]. It defines how AI models connect to data and tools, functioning analogously to a USB-C port for AI applications [3]. This paper explores how MCP can unify the media supply chain by replacing point-to-point integrations with a scalable, agent-centric architecture.

II. UNDERSTANDING CORE BACKGROUND AND RELATED WORK

Traditional The API Integration Bottleneck

Media workflows rely on fragile "point-to-point" integrations. A developer writes specific code to connect a MAM system to a Transcoder. If the Transcoder API changes, the integration breaks. This maintenance burden stifles innovation. AI projects frequently fail because the necessary data remains locked behind proprietary interfaces.

Agentic AI and Orchestration

Generative AI creates content, but Agentic AI performs actions. An agent operates with a goal (e.g., "Prepare this movie for release"). It decomposes this goal into discrete steps and uses "tools" to execute them. Early frameworks like LangChain defined proprietary tool formats, creating a new layer of fragmentation [2]. Agents built for one framework could not efficiently utilize tools designed for another. Recent research on multi-agent systems for supply chain management demonstrates the efficacy

of this approach for achieving consensus and automated negotiation [12].

The Model Context Protocol (MCP) Architecture

Anthropic introduced MCP as an open standard in late 2024. It decouples the AI model (Client) from the data source (Server). It defines three core primitives:

- **Resources:** Data the agent can read (e.g., a video file manifest, a subtitle track, a QC report).
- **Prompts:** Templates that guide the agent's interaction (e.g., "Analyze this scene for compliance violations").
- **Tools:** Executable functions the agent can call (e.g., delete_asset, start_transcode, update_metadata). This architecture allows the agent to query the MCP Server for available capabilities without prior knowledge of the underlying system logic [5].

III. PROPOSED MULTIMODAL AI FRAMEWORK ARCHITECTURE

We propose an architecture in which each media component exposes a lightweight MCP Server. The AI Agent runs as an MCP Client. This standardizes the communication layer using JSON-RPC over secure transports like SSE (Server-Sent Events) or Stdio.

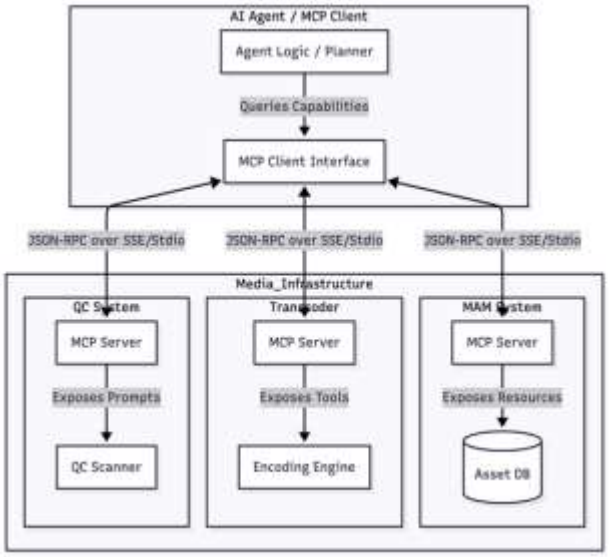

Figure 1: The Media-MCP-Agent Architecture. The Agent connects to diverse media systems via standardized MCP Servers, abstracting the underlying API complexities.

Figure 1 illustrates this topology. The Agent connects to multiple servers simultaneously. It reads metadata from the MAM and triggers a job on the Transcoder using the same protocol primitives

The MCP Server Layer

- Each media component hosts a specialized MCP Server.
- **The MAM Server:** Exposes assets as "Resources." The agent reads mcp://mam/asset/123/metadata to fetch technical parameters. It exposes tools like search_assets(query) to find content.
- **The Transcoder Server:** Exposes tools for manipulation. The agent calls transcode (source_id, profile) and subscribes to progress updates via the protocol's notification system.
- **The QC Server:** Exposes domain-specific prompts. It offers a check_compliance prompt that formats the request for the specific QC engine (e.g., detecting photosensitive epilepsy triggers).

The AI Agent Layer

The agent serves as the orchestration brain. It utilizes a Large Language Model (LLM) to reason about the supply chain state. Upon startup, it connects to all available MCP servers and lists their tools. When a human operator commands, "Fix the audio levels on Project X," the agent plans the execution path:

1. **Discovery:** Search Project X on the MAM (using the MAM Tool).
2. **Analysis:** Analyze Audio Loudness (using the QC Tool).
3. **Remediation:** Normalize Audio to -24 LKFS (using the Transcoder Tool). The agent generates the necessary glue logic dynamically. This eliminates the need for hard-coded, brittle scripts.

Workflow Example: Automated QC Repair

Figure 2 details a comprehensive repair workflow. The agent identifies a compliance failure. It uses MCP to locate the source file. It triggers a re-encode with specific parameters. It then verifies the fix before notifying the human operator.

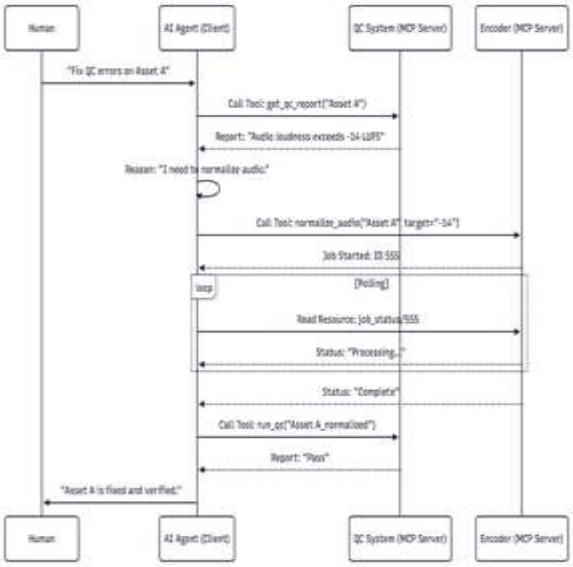

Figure 2: Sequence Diagram of an Agent autonomously fixing a QC issue using MCP tools.

IV. TECHNICAL CHALLENGES AND MITIGATION

We examine a set of practical tools and strategies that help reduce hallucinations at various stages of the AI development and deployment pipeline. These methods range from simple prompt adjustments to sophisticated real-time verification systems and user feedback integration.

Security: Prompt Injection and Sandboxing

MCP allows agents to execute code and modify data. This capability introduces significant risk. A "Prompt Injection" attack occurs when malicious instructions are embedded within a media file's metadata (e.g., a subtitle track containing hidden commands). If the agent reads this metadata without sanitization, it might execute the attacker's instructions [7].

- **Mitigation:** We must implement strict input validation and sandboxing. MCP tools should operate with "Least Privilege." We implement "Human-in-the-Loop" authorization for all destructive actions (e.g., delete_asset). The MCP Client must prompt the user for confirmation before executing high-risk tools [10].

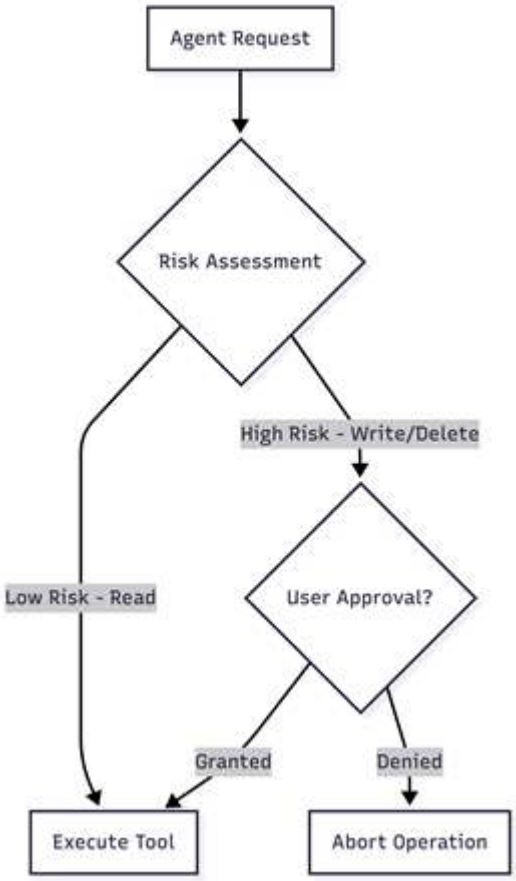

Figure 3: Security Flow Authorization. High-risk actions require explicit human approval.

Future Research

Media metadata is voluminous. A technical QC report for a feature film can exceed megabytes of text. Passing this entire context to the LLM for every interaction is computationally prohibitive and induces high latency [5].

- **Mitigation:** We utilize "Model Context Optimization." The MCP Server exposes concise summaries as the default Resource. The agent requests the full, granular log only if the summary indicates an anomaly. This hierarchical data access keeps the context window small and reduces token costs.

Probabilistic vs. Deterministic Conflicts

LLMs are probabilistic engines; media supply chains require deterministic outcomes. An agent might hallucinate a file format or invent a frame rate. This

is unacceptable in broadcast environments governed by SMPTE standards.

- **Mitigation:** We implement "Chain-of-Verification" workflows [13]. The agent treats its own outputs as hypotheses. It uses MCP tools to fetch ground truth (e.g., running ffprobe) to validate its assumptions before acting. We anchor the agent's reasoning in structured data rather than free text [15].

V. FUTURE WORK

Neuromorphic Edge Agents

Future implementations will push agents to the edge. We envision lightweight agents running directly on cameras or ingest stations, powered by neuromorphic hardware. These agents will use MCP to perform real-time metadata tagging and QC at the point of capture, reducing cloud egress costs and latency [14].

Hierarchical Agent Swarms

As workflows complexity increases, a single agent becomes a bottleneck. We propose a "Hierarchical Swarm" architecture. A "Manager Agent" decomposes high-level goals. It delegates tasks to specialized "Worker Agents" (e.g., a dedicated Audio Repair Agent, a Subtitle Sync Agent). These agents communicate via MCP, coordinating their actions to achieve the production objective [11].

Semantic Standardization

We must map MCP resources to existing media ontologies, such as EBUCore and SMPTE ST 2067. This ensures that an agent's understanding of "Frame Rate" is mathematically consistent across all tools in the chain.

VI. CONCLUSION

The Model Context Protocol solves the connectivity crisis in media. It replaces brittle, custom APIs with a standardized, secure conversation layer. It empowers AI agents to traverse the supply chain autonomously. We have moved from rigid, imperative scripts to adaptive, agentic workflows. The Media-MCP-Agent framework reduces manual labor, accelerates content delivery, and improves error handling. While security and determinism remain critical challenges, the path to fully autonomous media operations is now open.

ACKNOWLEDGMENTS

We thank the open-source community for their contributions to the Model Context Protocol ecosystem. We also acknowledge the valuable insights provided by the SMPTE working groups on AI and Media Supply Chain standards. Special thanks to the engineering teams at major broadcast facilities who provided feedback on early prototypes of the Media-MCP-Agent architecture.

REFERENCES

1. Anthropic. "Model Context Protocol Specification." Anthropic Technical Documentation, 2024.
2. LangChain. "LangChain: Building applications with LLMs through composability." LangChain Docs, 2023.
3. David, L. "What is Model Context Protocol (MCP) and How does it work?" Medium Data Science, 2025.
4. Andrenacci, G. "Creating Your First MCP Server: A Hello World Guide." AI Bistrot, 2025.
5. Anthropic. "Code execution with MCP: building more efficient AI agents." Anthropic Engineering Blog, 2025.
6. OWASP. "LLM01:2025 Prompt Injection Vulnerabilities." OWASP Gen AI Security Project, 2025.
7. Palo Alto Networks Unit 42. "New Prompt Injection Attack Vectors Through MCP Sampling." Palo Alto Networks Research, 2025.
8. eSentire. "Model Context Protocol Security: Critical Vulnerabilities Every CISO Should Address in 2025." eSentire Threat Intelligence, 2025.
9. Jannelli, V., et al. "Agentic LLMs in the Supply Chain: Towards Autonomous Multi-Agent Consensus-Seeking." arXiv preprint arXiv:2411.10184, 2024.
10. Tahir, B. "How to Secure Model Context Protocol (MCP)." Medium Cybersecurity, 2025.

11. Chen, et al. "From Unstructured Communication to Intelligent RAG: Multi-Agent Automation." arXiv preprint arXiv:2506.17484, 2025.
12. SAP. "Agentic AI in the global supply chain." SAP Insights, 2025.
13. Amazon Web Services. "Reducing hallucinations in large language models with custom intervention using Amazon Bedrock Agents." AWS Machine Learning Blog, 2024.
14. MIT News. "A smarter way for large language models to think about hard problems." Massachusetts Institute of Technology, 2025.
15. arXiv. "Blueprint First, Model Second: A Framework for Deterministic LLM Workflow." arXiv preprint arXiv:2508.02721, 2025.