Position Paper

In Concert: Bridging AI Systems & Network Infrastructure through MCP

How to Build Network-Aware Intelligent Applications

THE LINUX FOUNDATION

CAMARA
THE TELCO GLOBAL API ALLIANCE

Executive Summary

Artificial intelligence (AI) and network infrastructure often operate as separate systems today, each performing well independently, but unable to communicate. It's like having talented musicians each playing their own instrument, and vocalists each singing their own scales, but unable to perform a concert because they lack the aligned sheet music needed to perform together.

The current separation prevents AI applications from accessing real-time network intelligence to address connectivity challenges or enhance digital experiences.

This paper describes how an AI application can integrate with network infrastructure by connecting network-exposed APIs with AI technology through Model Context Protocol (MCP).

Contents

Executive Summary 1

Chapter 1: Setting the Stage 2

1.1 The Opportunity: Harmonizing APIs and AI 2
1.2 What's Needed: Combining AI technology with network APIs 2
1.3 In Concert: CAMARA + MCP 2

Chapter 2: Instrumental Interfaces 3

2.1 CAMARA 3
2.2 Model Context Protocol (MCP) 3
2.3 Perfect Harmony: How They Work Together 4

Chapter 3: Example Usage in AI Applications 5

3.1 Intelligent Video Streaming: AI-Powered Quality Optimization 5
3.2 Banking Fraud Prevention: Network-Verified Security 5
3.3 Local/Edge Optimized AI Deployment 6

Chapter 4: Value for Players Across the Combined Ecosystem 7

4.1 AI Developers: Enrich AI Applications with Trusted Network Capabilities 7
4.2 CAMARA API Providers: Transform into AI Partners 7
4.3 CAMARA + MCP Ecosystem: Common Solutions for Common Growth 8

Chapter 5: CAMARA Objectives for MCP Support 8

5.1 Objective 1: Security Guidelines 8
5.2 Objective 2: MCP Tools for CAMARA APIs 9
5.3 Objective 3: Define Quality Requirements 9
5.4 Implementation Success Factors 9

Chapter 6: Finale 10

Composing the Bridge for Network-Aware AI 10

Chapter 1: Setting the Stage

1.1 The Opportunity: Harmonizing APIs and AI

AI applications today can analyze data and generate responses, but they cannot directly interact with the network infrastructure that powers our digital world. This technical isolation limits their ability to solve real-world connectivity problems or optimize user experiences based on actual network information and conditions. Enabling AI applications to interact with network APIs opens an immense opportunity for AI developers.

1.2 What's Needed: Combining AI technology with network APIs

Breaking through the current isolation requires covering two critical aspects:

1. **Network capabilities,** accessible through uniform APIs and data models, allow applications to use the same code regardless of the network provider and are offered as operator-agnostic, intent-based interfaces.
2. **Bridging protocols** enable AI systems to access external data sources seamlessly. Such protocols translate between AI systems and external data sources, using tool abstraction layers.

This paper describes how to combine two complementary technologies, CAMARA APIs and Model Context Protocol (MCP), to enable network infrastructure to serve as a data source for AI applications.

1.3 In Concert: CAMARA + MCP

This section addresses how CAMARA APIs can be integrated with AI applications. The integration is done through an MCP server that acts as a translator: it takes a CAMARA API and translates it into a format that AI systems can understand and use. This translated version is called an MCP tool and contains descriptive information about the network capability(s) offered through the CAMARA API, which is understandable by an AI application.

By combining the strengths of CAMARA with those of MCP, AI application developers can:

- **Deliver more intelligent, context-aware services powered by dynamic network capabilities.** These include real-time capabilities like Quality on Demand for bandwidth optimization, Device Location for verification, and Edge Discovery for compute resource allocation.
- **Reduce delivery complexity, accelerate time-to-market, and expand global reach with CAMARA-defined APIs.** CAMARA APIs eliminate the need for operator-specific integration flows and custom response handling for each network provider.
- **Enable AI models to make real-time, policy-compliant decisions that optimize user experience and resource utilization across diverse operator environments.** These decisions shall be consent-aware and privacy-compliant.

Connecting AI applications to network capabilities requires a common layer that maps CAMARA APIs to MCP, eliminating the isolation that has prevented AI and networks from interoperating and communicating - that is, from performing together effectively.

In the following, we'll look more closely at each of these two foundational instruments: CAMARA network APIs and MCP.

Chapter 2: Instrumental Interfaces

2.1 CAMARA

CAMARA is an open-source initiative hosted by the Linux Foundation, created to solve a fundamental challenge in telecommunications: making advanced network capabilities easily accessible to developers in a consistent and telco-operator-neutral way. Before CAMARA, each telco operator had its own approach, forcing developers to build custom integrations for each. CAMARA set out to resolve the fragmented ecosystem of network APIs so developers could write code once and have it work across different telco networks.

Launched in early 2022, CAMARA's mission is to define and validate network APIs that abstract the complexity of network infrastructure. Like sheet music, which provides a common language for musicians from different backgrounds, CAMARA empowers developers to request services such as prioritized connectivity, device status, or optimal edge resource discovery without needing in-depth knowledge of the underlying network technologies.

CAMARA APIs are designed around real-world application needs, with early use cases focusing on areas like:

- **Quality on Demand (QoD):** Request better network performance when it's needed. Technically, this dynamically allocates bandwidth and reduces latency based on application or user context.
- **Device Location:** Get verified location data while respecting privacy. This retrieves verified device location data from the network in a privacy-respecting manner, enabling location-based services with user consent.
- **Edge Discovery:** Find the best nearby computing resources. This identifies optimal edge computing resources based on proximity, latency, and network performance metrics.
- **Anti-fraud APIs:** Identify unusual patterns for the device at the time of an end-user activity, e.g., a financial transaction.

With a developer-friendly approach and alignment with the GSMA Open Gateway initiative (a global effort to expose telco network capabilities), CAMARA APIs are becoming a critical interface between global network infrastructure and the broader application ecosystem.

2.2 Model Context Protocol (MCP)

MCP is a protocol that enables seamless integration between LLMs and external data sources and tools. MCP provides a universal language that lets AI systems access different capabilities without learning custom protocols for each.

It solves a communication problem: AI applications want to access external capabilities (like network APIs), but they don't speak the same 'language' as those APIs. MCP acts as an abstraction layer that enables external data sources to describe their diverse capabilities in a way that AI systems can understand, select, and access, providing the mechanism that allows AI and network systems to perform together.

Whether building an AI-powered Integrated Development Environment (IDE), enhancing a chat interface, or creating custom AI workflows, MCP provides a common way for these AI applications to access the context they need. Since Anthropic developed and announced MCP in November 2024, it has gained rapid industry adoption. Major players include OpenAI, Microsoft Azure, and other cloud providers that have integrated the protocol into their AI systems. The protocol marked its first anniversary with a major specification update (version 2025-11-25), demonstrating continued momentum toward enterprise readiness.

2.3 Perfect Harmony: How They Work Together

The relationship between MCP and CAMARA represents a powerful convergence of efforts in complementary domains. Just as CAMARA has defined network APIs to eliminate fragmentation across network providers, MCP is specifying the interface between AI applications and data sources. Together, they address both sides of the integration challenge.

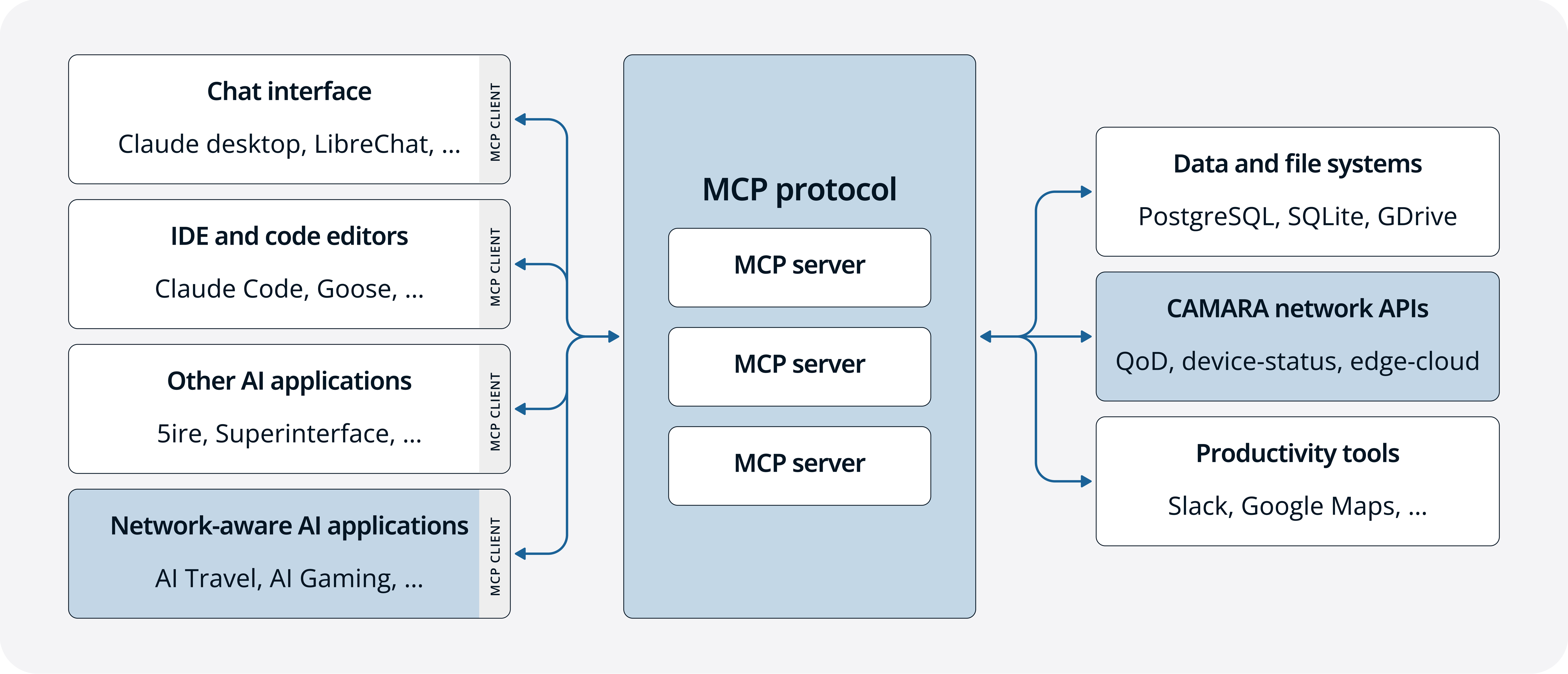
This combined approach creates a natural resonance. When CAMARA APIs are made accessible through MCP servers, network capabilities become immediately accessible to any MCP-compatible AI application. The MCP term for this is "tool abstraction," in which CAMARA network APIs are represented as callable tools that AI systems can use via the MCP protocol.

The result: AI applications or LLMs can make decisions using real-time network and user context delivered through CAMARA APIs.

Simple Integration Architecture

The integration follows a straightforward pattern:

- The AI applications use an MCP Client to communicate with external sources
- The MCP Client uses the MCP protocol to connect to an MCP Server that exposes CAMARA APIs as tools
- The MCP Server uses the authorization token obtained by the AI application to call a CAMARA API for real-time network capabilities and data
- The CAMARA API delivers the network capabilities and data to the MCP Server
- The MCP Server provides the information to the MCP Client of the AI application using the MCP protocol



In practical terms, an MCP server provides an adaptation layer that allows any MCP-compatible AI system to access the capabilities of any CAMARA API provider.

The power of the combination lies in its simplicity: existing CAMARA implementations can add MCP server capabilities as a lightweight layer, while AI applications can add network awareness by implementing standard MCP clients. No fundamental changes are required to either ecosystem, just an adaptation layer that enables both systems to perform together seamlessly.

Note: Technical details about authentication flows, security implementation, and detailed API mappings continue to evolve. The latest MCP release introduced additional security and authorization features, providing a stronger foundation for CAMARA integration work going forward.

Chapter 3: Example Usage in AI Applications

The integration of CAMARA and MCP enables the development of new categories of intelligent applications that access real-time network information.

The following use cases demonstrate the practical value of what is possible across different industries.

3.1 Intelligent Video Streaming: AI-Powered Quality Optimization

The Challenge

Degraded video streams often leave users with generic error messages and limited resolution options, such as checking connections or trying again later. When network conditions cause buffering, users are faced with either manual troubleshooting or accepting poor quality, leading to frustration and service abandonment.

The AI-Enabled Solution

CAMARA APIs, accessible through MCP, enable streaming applications to diagnose network conditions in real time and deliver targeted solutions.

When a user experiences video buffering, the AI assistant follows a verification process. First, it confirms the user's device identity and location using CAMARA network verification APIs. Next, it checks the type of network they're using and its performance. Then it reviews the customer's account history to offer optimized pricing. Finally, it can offer to activate enhanced network quality directly on the operator's network - Quality of Service (QoS) improvements via QoD APIs.

The user receives a specific offer, such as ***"Activate streaming optimization for $3.50 for 2 hours,"*** rather than generic troubleshooting advice.

Multi-stakeholder value

- **Users:** Instant problem resolution with transparent pricing and control
- **Streaming Service Provider:** Reduced churn and enhanced user experience through network intelligence
- **CAMARA API Providers:** New revenue streams from network capability monetization
- **AI applications:** Practical demonstration of real-world problem-solving capabilities

With a developer-friendly approach and alignment with the GSMA Open Gateway initiative (a global effort to expose telco network capabilities), CAMARA APIs are becoming a critical interface between global network infrastructure and the broader application ecosystem.

3.2 Banking Fraud Prevention: Network-Verified Security

The Challenge

Financial fraud detection systems often generate false positives when customers travel or change devices, blocking legitimate transactions and forcing time-consuming verification calls. Traditional systems rely on approximate location data and transaction history (geographic estimates and behavioral patterns) that can incorrectly flag normal customer activity.

For example, a customer who flew from New York to Barcelona for vacation may find their card declined at a restaurant because the fraud system flagged the international location change as suspicious, even though they're using their registered iPhone and the SIM card shows valid roaming status.

The Network-Aware Solution

CAMARA network APIs can provide additional, real-time context to AI fraud detection systems, including customer identity and location verification directly from authoritative network data.

When the Barcelona transaction appears suspicious, the banking AI agent initiates a verification process. First, it verifies the customer's registered device to confirm it's their iPhone 16e. Next, it confirms the actual location through network positioning to validate they’re in Barcelona. Then it validates roaming status to verify legitimate international use from their US carrier. Finally, it checks for recent SIM swap activity to ensure no security breaches. This comprehensive verification occurs in real time, allowing legitimate transactions to proceed while maintaining security controls.

Key benefits:

- **Enhanced Accuracy:** Network-verified data reduces false-positive rates compared to third-party location services
- **Improved Security:** Detection of actual device and location spoofing attempts
- **Customer Experience:** Seamless transactions without unnecessary friction

3.3 Local/Edge Optimized AI Deployment

Open-source AI inference frameworks can be deployed locally on edge infrastructure, AI PCs, on-premises, and in the cloud, enabling integration with MCP servers and AI applications. Inference frameworks, such as OpenVINO, efficiently execute LLMs, including Llama3, Phi4, Qwen3, and Hermes3, on CPUs, GPUs, and NPUs (Neural Processing Units).

Benefits for local/edge AI deployment are as follows:

- **Performance Gains:** Edge nodes with NPUs or accelerators achieve lower latency than distant cloud centers (e.g., for AR vision processing).
- **Control over Data:** Local inference keeps business-critical data private without external sharing.
- **Cost Efficiency:** Open-source frameworks enable cost-efficient AI applications that can run locally, avoiding additional cloud-provider inference fees.

In related 5G Multi-access Edge Computing (MEC) scenarios, AI applications can use CAMARA Edge Discovery APIs to find optimal edge nodes and deploy models on them.

Continue to next page for Chapter 4: Value for Players Across the Combined Ecosystem.

Chapter 4: Value for Players Across the Combined Ecosystem

4.1 AI Developers: Enrich AI Applications with Trusted Network Capabilities

LLM-Agnostic Development and AI Vendor Flexibility

MCP provides a tool abstraction layer, a common way for AI systems to access capabilities, that works consistently across different AI models and providers. Applications can switch their underlying LLM without losing the ability to use CAMARA network APIs through the available MCP tools. For example, a developer could move from OpenAI GPT-4 to Anthropic Claude, or to open-source models running locally or on-premises, and the network API access would continue to work because MCP servers support network API access in a model-agnostic way. This also ensures that AI developers aren't locked into a single AI ecosystem.

Network Capabilities Without Specialized Network Expertise

MCP's abstraction of CAMARA APIs enables developers to build network-aware AI applications without needing specialized network knowledge. Developers in banking, retail, or other industries typically lack network domain expertise but can now access network functions as easily as any other API. AI systems can even generate these calls through natural language, with the heavy lifting of translating intent into correct network API calls handled by MCP servers and CAMARA APIs.

Authoritative Network Data for AI Decisions

CAMARA network APIs provide data from network providers, such as network-verified device location, subscriber status, and congestion information. AI applications access such authoritative, possibly real-time (ground-truth) data from the user's network provider, rather than relying on third-party estimates or outdated information. For applications where accuracy is critical, such as banking applications detecting SIM swap fraud, autonomous vehicles determining edge compute locations, and ride-sharing apps verifying location authenticity, this authoritative data source enables confident decision-making.

4.2 CAMARA API Providers: Transform into AI Partners

Expanding Beyond Connectivity Services

Network operators can position themselves as essential infrastructure players in the AI era by exposing capabilities through CAMARA APIs. Offering easily programmable network services (those that applications can control via API calls), such as bandwidth-on-demand, device status, and edge computing, means that AI-driven applications can leverage operator networks for advanced functionality. This transforms the operator's role into a trusted AI partner that enterprises view as strategically valuable.

AI as High-Volume API Consumer

AI systems generate orders of magnitude more transactions than human users. When network APIs become accessible to AI applications, API providers may expect an increase in API calls per second, as individual application users trigger them less frequently.

4.3 CAMARA + MCP Ecosystem: Common Solutions for Common Growth

Privacy-First Design with Granular Consent

Both CAMARA and MCP emphasize privacy and consent controls. Industry-standard security protocols are used for controlling MCP tool access, requiring explicit authorization for every network API request. CAMARA API calls must include the scope and purpose of each requested data item to ensure user consent is obtained when required. In addition, AI applications may also require fine-grained consent for specific purposes—approving navigation assistance while denying marketing analytics. A combined consent approach enables enterprises to comply with privacy regulations for their AI applications while fostering consumer trust, as indicated in section 5.1, below.

Accelerating CAMARA + MCP Adoption

Both MCP and CAMARA have open approaches that target interoperability for use by the broadest possible developer ecosystem. MCP works across AI systems, from OpenAI to Microsoft, and open-source communities, meaning tool integrations reach a broad audience. CAMARA APIs work across network operators globally, maintaining consistent semantics (meaning they work the same way). Multiple language Software Development Kits (SDKs) and sandbox environments lower barriers to entry, enabling startups to prototype network-aware AI features risk-free. It is expected that openness creates positive feedback loops, where more adoption brings more contributors, solidifying a complementary approach and accelerating ecosystem growth.

Chapter 5: CAMARA Objectives for MCP Support

At the request of the CAMARA Board, the Technical Steering Committee (TSC) established the MCP activity to provide concrete examples of interworking between AI applications, MCP, and CAMARA APIs. More details on the MCP activity are available here.

The MCP activity has identified the following objectives that the CAMARA project needs to address:

5.1 Objective 1: Security Guidelines

Security and Interoperability Considerations

While CAMARA's OAuth 2.0/OpenID Connect-based security profile aligns conceptually with MCP's authorization framework, implementation challenges remain before MCP tools for CAMARA APIs can be used in production networks. MCP implementations must manage authentication tokens for AI systems. Three-Legged Access Tokens involve multiple steps across three parties: the user, the application, and the authorization server. Because AI systems cannot participate directly in the OAuth authentication flow, the MCP layer must handle this complexity. Extension of the CAMARA security guidelines will be needed to ensure consistent security handling across MCP and CAMARA APIs. The latest MCP specification introduces improvements to authorization flows but does not eliminate the integration work required for alignment with CAMARA's security profile.

User consent

CAMARA's explicit purpose declaration requirements align well with MCP's transparency principles. However, these declarations need seamless integration into how AI systems will make API calls. Most critically, the MCP layer must abstract the complexities of consent management while maintaining full compliance with CAMARA’s stringent requirements.

5.2 Objective 2: MCP Tools for CAMARA APIs

API Design Guidelines

CAMARA's guidelines provide a solid foundation for consistent API design across the project. Enhancements may be needed for optimal MCP integration. Examples may include making parameter naming consistent across all APIs. This consistency facilitates automated MCP functions, reducing manual mapping work. Furthermore, additional error responses may be needed to enhance the reliability of AI interpretation. If an API call fails, AI systems require clear, consistent error messages that they can understand and act upon.

Additionally, API documentation must evolve beyond technical specifications. Rich natural language descriptions help AI systems understand what each API does and when to use it. Comprehensive input/output examples demonstrate proper usage patterns, making it easier for AI to generate correct API calls.

CAMARA-defined MCP Tools

Creating MCP function descriptions (“MCP tools”) for CAMARA APIs presents a crucial opportunity to create industry-wide consistency. Rather than leaving interpretation to individual ecosystem players, CAMARA may provide authoritative MCP function definitions as core deliverables alongside API specifications.

This approach ensures consistency across implementations, significantly lowers barriers for AI developers, and maintains quality control over how network capabilities are exposed to AI systems. These CAMARA-defined descriptions should be versioned in lockstep with corresponding CAMARA API specifications, ensuring synchronized evolution.

Provider Implementations

While CAMARA doesn't operate infrastructure, as with APIs, CAMARA members may provide their example MCP server implementations in dedicated repositories

5.3 Objective 3: Define Quality Requirements

Conformance and Quality Assurance

Ensuring consistent functionality across diverse MCP implementations requires a comprehensive approach. This includes testing systems that verify implementations work correctly (conformance test suites), detailed specifications for how systems should behave in unusual situations (behavioral specifications for edge cases), and documentation of common AI interaction patterns. Establishing minimum requirements will be essential for any implementation claiming CAMARA and MCP compatibility.

Ecosystem Governance and Sustainability

Success requires the convergence of multiple parties working together to create standardized, interoperable systems. On the MCP side, AI companies contribute integration expertise. On the network side, CAMARA API providers bring knowledge of network capabilities. Both communities collaborate through CAMARA's governance structures to ensure standards meet real-world operational needs while maintaining interoperability.

5.4 Implementation Success Factors

Successfully integrating CAMARA and MCP requires balancing technical adaptations with ecosystem development. By providing design and security alignment guidelines, CAMARA can create a robust framework that enables AI systems to reliably leverage network capabilities while maintaining security, privacy, and operational excellence.

Chapter 6: Finale

Composing the Bridge for Network-Aware AI

The combination of CAMARA and MCP represents more than technical advancement. It establishes the layer that links AI applications with network infrastructure. This combination eliminates the isolation that is preventing these technologies from working together effectively.

Like musicians and vocalists, who can read the same sheet music, AI applications using CAMARA's network APIs through the MCP protocol can now seamlessly access data across network providers, while AI systems gain the real-world context needed for truly intelligent responses.

This leads beyond technical capability to a fundamental business opportunity. CAMARA API providers transition to become trusted AI partners. Developers can create network-aware AI applications once and implement them everywhere, and leverage network data without specialized network expertise, thereby creating competitive advantages.

The use cases presented in this document, intelligent video streaming, network-verified fraud prevention, and local/edge optimized AI deployment, represent just the beginning. As CAMARA APIs mature and adoption grows, entirely new categories of network-aware AI applications will emerge, limited only by imagination rather than integration complexity.

We invite network and AI communities to recognize the CAMARA + MCP combination as an excellent opportunity to unlock the power of network-aware AI applications. The concert of network-aware AI is ready to begin.

Link to CAMARA Project Website: www.camaraproject.org

Link to CAMARA Project Linkedin: www.linkedin.com/company/camara-project

---

References

CAMARA

Project Documentation: https://camaraproject.org
GitHub Repositories: https://github.com/camaraproject

MCP

What is the MCP?: https://modelcontextprotocol.io
MCP Specification 2025-11-25: https://modelcontextprotocol.io/specification/2025-11-25

Acknowledgements

The CAMARA Project and The Linux Foundation gratefully acknowledge the following individuals for their contributions to the research, authorship, technical review, and community feedback that informed this paper.

Lead Authors

- Herbert Damker
- Nick Venezia
- Tanja de Groot

Editorial & Design

- Andrew Sheridan
- Kristen Stelzer
- Evangelos Kouvdis

Technical Review & Engineering Contributions

- Diego Gonzalez
- Jesús Peña García-Oliva
- Ludovic Robert
- Markus Kümmerle
- Rafal Artych

Linux Foundation Support

- Casey Cain
- Jill Lovato

Community Reviewers & Participants

- Ali Iqbal
- Artur Krukowski
- Ben Hepworth
- Chris Howell
- Jan Friman
- Jose Luis Urien Pinedo
- Jorge Garcia Hospital
- Lyle Bertz
- Mahesh Chapalamadugu
- Mark Cornall
- Megan Collins
- Pierre Close
- Randy Levensalor
- Ramesh Shanmugasundaram
- Surajj Jaggernath
- Toshi Wakayama