



KubeCon



CloudNativeCon

India 2026

#KubeCon #CloudNativeCon

A gRPC Transport for Model Context Protocol

Pawan Bhardwaj
gRPC Maintainer
Sr. Software Engineer Google



Why gRPC for MCP?



KubeCon



CloudNativeCon

India 2026



Reliable Communication

AI agents are moving from test environments into enterprise operations, requiring reliable structured communication with external tools and services



The MCP Standard

The Model Context Protocol (MCP) is the standard that enables agents to communicate with tools.



The gRPC Gap

Many enterprises rely on gRPC for their services. MCP's default JSON-RPC transport requires transcoding gateways - adding complexity and overhead.



KubeCon



CloudNativeCon

India 2026

gRPC as Transport Backbone



Performance and Efficiency



KubeCon



CloudNativeCon

India 2026

10x

Smaller Messages

Protobuf vs JSON

Binary Encoding (Protobufs)

Shrink message size - lower latency, reduced network costs

HTTP/2



Client and server continuously send data over a persistent connection

Built-in Flow Control

Native backpressure in transport.

Enterprise grade security and Authorization



KubeCon



CloudNativeCon

India 2026



Identity & Encryption

Mutual TLS and SPIFEE Identity support ensures secure, verified communication between services.



Strong Authentication

Native hooks for JWT and OAuth provide industry-standard access control integration.

Granular Control

Method level authorization allows for fine-grained permissioning across your entire service mesh.

Operational Maturity



KubeCon



CloudNativeCon

India 2026

Unified Observability

OTEL integration for full-stack insight.

Robust Resiliency

Deadline, timeout, and built-in flow control.

Schema Validation

Protobuf strict typing rejects malformed inputs.

Polyglot Development

C++, Go, Python, Java, Node, PHP, and Rust.

Metadata Support

Extensible context for every request.

xDS Support

Proxyless service mesh integration.



KubeCon



CloudNativeCon

India 2026

Implementation



Mapping the Specification: mcp-grpc-transport-proto



KubeCon



CloudNativeCon

India 2026

Repository: <https://github.com/GoogleCloudPlatform/mcp-grpc-transport-proto>

Canonical Source of Truth

Will be used for all language implementations.

Synchronized Evolution

Updates are driven by MCP transport workgroup to maintain parity with the latest MCP Specification releases.

Native gRPC Service Definition

Implemented as a first-class service `Mcp` with dedicated RPC methods for `ListResources`, `CallTool`, and `GetPrompt`, replacing JSON-RPC 2.0 wrappers.

Semantic Parity via Direct Mapping

Direct mapping of MCP primitives to typed Protobuf messages (e.g., `Resource`, `Tool`) ensures strict contract safety.

Pluggable transports

Top layer

MCP Types

CallToolRequest, CallToolResult...

Middle layer

Message format

JSON-RPC (default) · Protobuf · ...

Bottom layer

Transport

stdio · SHTTP · gRPC · WebSocket...

"Pluggable transports" usually means swapping both bottom layers.

gRPC with protobuf over JSON-RPC makes no sense — you'd smuggle JSONRPC as a string and lose protobuf's whole value proposition.

Solution: Dispatcher pattern — swap both layers

Reference: Path to V2 for MCP SDKs - Max Isbey, Anthropic

CHANGES

The Dispatcher pattern

BEFORE

BaseSession

MCP semantics

initialize(), call_tool(), list_tools()... progress tokens, cancellation, validation
19 methods

Wire protocol tangled

JSON-RPC wrap, ID correlation, receive loop, stream management

Want gRPC? Reimplement the whole session.

extract

AFTER

python-sdk PR #2320

BaseSession

MCP semantics only

initialize(), call_tool(), list_tools()...
19 methods - unchanged

Dispatcher 5 methods

JSON-RPC (default) gRPC Protobuf ...

send_request · send_notification · send_response set_handlers · run

Implement 5 methods → all 19 MCP methods work for free

Draft PR · design settled

Reference: Path to V2 for MCP SDKs - Max Isbey, Anthropic

MCP gRPC Python



KubeCon



CloudNativeCon

India 2026

Repository: <https://github.com/GoogleCloudPlatform/mcp-grpc-transport-py>

Status: Active Development

Work in Progress: Providing a pluggable transport implementation for the MCP Python SDK.

Dispatcher-Based Architecture

Based on official transport dispatchers, facilitating the addition of MCP endpoints to existing gRPC services.

Standard Evolution

Built in lockstep with `mcp-grpc-transport-proto` and official python libs to maintain strict parity with evolving MCP specs.

Unified Developer Experience

Tool definitions remain identical. Developers simply swap `JSONRPCDispatcher` for `GrpcDispatcher` during session initialization.



KubeCon



CloudNativeCon

India 2026

What's Next



Roadmap



KubeCon



CloudNativeCon

India 2026

SEP-2598: Pluggable Transports (Under Review)

<https://github.com/modelcontextprotocol/modelcontextprotocol/pull/2598>

Language Support

Python under-development, will be followed up with others.

Expansion beyond Python

Strategic growth of the SDK ecosystem to multi-language support.

Conformance Testing

Development of comprehensive tests for gRPC Transport verification.

Migration Playbook

Guidelines for migrating existing gRPC services to tools and changing transports.



KubeCon



CloudNativeCon

India 2026

Thank You !!

Resources

Site: grpc.io

YouTube: youtube.com/@grpcio

Upcoming Events

gRPC Conf NA: 3 September, 2026

gRPC Conf Bangalore: To be announced (2026)

Join the Community

Meetup: meetup.com/grpcio

Meetup Bangalore: meetup.com/grpc-bangalore

Github

Site: <https://github.com/grpc/grpc>

Mailing List: groups.google.com/g/grpc-io

