AGENTIC AI INFRASTRUCTURE

The Hidden Backbone of Agentic AI

Why Every Multi-Agent Platform Needs gRPC



Saurabh Mishra | Professional Profile & Achievements



Google Developer Expert

GCP Expert specializing in cloud infrastructure and architecture

DevOps Ambassador

Institute Ambassador and Organizer driving DevOps culture

AAIF Ambassador

Agentic AI Foundation

Connect With Me

linkedin.com/connectsaurabhmishra

medium.com/@connectsaurabhmishra

Community Leader

Neo4j Ninja, Graph DB Pune & Kong Champion

Multi-Cloud Advocate

Calico Big Cats, CDF & open-appsec AMB

Talks 300+ Globally

Sharing knowledge and presenting at major tech conferences worldwide

Evolution of AI Use Cases

STAGE 01 Predictive AI

- Regression
- Forecasting
- Sentiment
- Entity Ext.
- Object Det.

STAGE 02 Generative AI

- Content Gen
- Code Rewrite
- Summarize
- Extractive QA
- Descriptions

STAGE 03 Multimodal AI

- Image Under.
- Spatial Reas.
- Math Context
- Video QA
- Speech/Trans

STAGE 04 Agentic & Reasoning

- Chain-of-Th.
- Tool Use APIs
- Auto Workfl.
- Self-Correct
- Agent Swarms

STAGE 05 Physical & Embodied

- World Models
- Robotics
- Sensor Fusion
- Auto Nav
- Closed-Loop

AI Agent Evolution

STAGE 01



LLM + Prompt

Basic text interactions via static prompts relying purely on parametric memory.

LLM

STAGE 02



LLM + RAG

Retrieval-Augmented Generation bringing live external vector context.

LLMRAG

STAGE 03



Function & Actions

Tool execution calling APIs, databases, and structured internal services.

LLMTools

STAGE 04



Reasoning Loops

Iterative thinking (ReAct/o1/R1 style), self-correction, & multi-step planning.

System 2Plan

STAGE 05



Multi-Agent Swarms

Specialized agents collaborating & orchestrating complex workflows.

SwarmsOrchestration

NEW



Autonomous Workforces

Computer-use OS agents, self-evolving systems, and automated enterprise execution.

OS / GUIAutonomy

Experts

AGENTIC INFRASTRUCTURE

The Hidden Backbone of Agentic AI

CURRENT FOCUS

The AI Hype Stack

Everyone talks about LLMs, RAG, vector databases, and AI agents.

- **LLMs & Foundation Models**
- **RAG & Vector Databases**
- **Autonomous AI Agents**

CRITICAL QUESTION

The Missing Link

There is one critical component that often goes unnoticed:

How do these agents communicate with each other reliably and efficiently?

CORE THESIS

THE SOLUTION

Why gRPC?

Today, let's explore why gRPC could be the hidden backbone of modern Agentic AI platforms.

- ✓ **Low Latency Streaming**
- ✓ **Strict Protobuf Contracts**
- ✓ **Multi-Agent Orchestration**

AGENTIC INFRASTRUCTURE

Why REST/JSON Breaks Down in Multi-Agent Platforms

LIMITATION 01



Verbose Text Serialization

Repeated field names in massive nested JSON context objects burn CPU and bandwidth.

CPU & Bandwidth Burn

LIMITATION 02



Stateless Request-Response Friction

Forcing long-running, multi-step agent reasoning into synchronous HTTP cycles leads to timeouts and fragile polling loops.

Timeouts & Polling

LIMITATION 03



Schema Drift & Hallucinated Contracts

Loosely typed JSON payloads cause silent runtime deserialization crashes when Agent A evolves independently of Agent B.

Silent Runtime Crashes

LIMITATION 04



Lack of Native Backpressure

Floods of concurrent sub-agent tasks easily overwhelm downstream tool servers.

Tool Server Overwhelm

AGENTIC INFRASTRUCTURE

What is gRPC? (A 60-Second Refresher for AI Architects)

FEATURE 01

Built on HTTP/2

Binary framing, header compression (HPACK), and multiplexed streams over a single TCP connection.

Multiplexed Streams

FEATURE 02

Protocol Buffers

Interface Definition Language (IDL) and ultra-compact binary wire format.

Binary Wire Format

FEATURE 03

Code Generation

Auto-generates strongly typed client/server stubs across Python, Go, Rust, Java, C++, and TypeScript.

Polyglot Stubs

FEATURE 04

CNCF Standard

Battle-tested at hyperscale by Google, Netflix, and modern cloud-native infrastructure.

Hyperscale Proven

AUTONOMOUS SRE PLATFORM

Specialized Agents in Action

AGENT 01



Planner Agent

Understands incidents and creates an actionable response plan.

Plan & Strategy

AGENT 02



Kubernetes Agent

Investigates cluster resources and workload pod states.

Cluster Infra

AGENT 03



Observability Agent

Analyzes Prometheus metrics and Grafana logs.

Metrics & Logs

AGENT 04



Security Agent

Investigates security threats and vulnerability alerts.

Threat Defense

AGENT 05



Remediation Agent

Executes automated corrective actions and system fixes.

Auto-Fix Action

AGENTIC INFRASTRUCTURE

The Hidden Problem: How Do Agents Communicate?

SCENARIO



DISTRIBUTED SYSTEM

5 Independent Services

Imagine five agents running as independent services across Kubernetes.

- Separate execution contexts
- Network boundaries
- High-frequency IPC requirements



QUESTION 01 • INVOCATION

How does the Planner Agent invoke the Kubernetes Agent?



QUESTION 02 • STREAMING

How does an agent send intermediate results?



QUESTION 03 • DATA EXCHANGE

How do agents exchange structured information?



QUESTION 04 • RESILIENCE

How do we manage communication failures?



QUESTION 05 • TRACING & OBSERVABILITY

How do we track requests across multiple agents?

AGENTIC INFRASTRUCTURE

Architectural Bottlenecks: Challenge & Impact

CHALLENGE 01

High latency

Impact: Slow agent execution

CHALLENGE 02

No standard contract

Impact: Integration complexity

CHALLENGE 03

Large JSON payloads

Impact: Serialization overhead

CHALLENGE 04

Missing observability

Impact: Difficult troubleshooting

CHALLENGE 05

Unreliable communication

Impact: Failed workflows

CHALLENGE 06

Synchronous blocking

Impact: Reduced scalability

AGENTIC INFRASTRUCTURE

Introducing gRPC: The Communication Backbone

OVERVIEW

RPC FRAMEWORK

High-Performance Core

gRPC is a high-performance, open-source Remote Procedure Call framework originally developed at Google.

- **Enables services to communicate through strongly typed interfaces.**

COMPONENT 01 • SERIALIZATION

Protocol Buffers – Efficient binary serialization.

COMPONENT 02 • TRANSPORT

HTTP/2 – Multiplexed communication over persistent connections.

COMPONENT 03 • CONTRACTS

Service Definition – Defines communication contracts.

COMPONENT 04 • TOOLING

Code Generation – Generates client and server libraries.

COMPONENT 05 • DATA FLOW

Streaming – Supports continuous data exchange.

AGENTIC INFRASTRUCTURE

Protocol Buffers: The Language of Agent Communication

SCHEMA COMPARISON

JSON Request (Verbose Text Payload)

```
{
  "agent": "kubernetes-agent",
  "task": "analyze-pod-failure",
  "namespace": "production"
}
```

Equivalent Protobuf Schema (Compact Binary)

```
message AgentRequest {
  string agent = 1;
  string task = 2;
  string namespace = 3;
}
```

CONCEPT OVERVIEW

Language & platform-neutral binary serialization mechanism replacing large JSON payloads.

Scenario: Planner Agent sends diagnostic requests to a Kubernetes Agent container.

BENEFIT 01

Compact Serialization & Low Overhead

Dramatically reduces payload size, bandwidth, and parsing overhead.

BENEFIT 02

Strongly Typed Data & Schema Evolution

Strict contracts prevent runtime errors and allow seamless updates.

BENEFIT 03

Cross-Language Code Generation

Generates native client and server bindings across programming languages.

BENEFIT 04

High-Efficiency Agent Workloads

Eliminates serialization overhead in high-throughput agent environments.

AGENTIC INFRASTRUCTURE

gRPC Communication Patterns for AI Agents

PATTERN 01

Unary RPC

1 Request ⟶ 1 Response

EXAMPLE WORKLOAD

Planner Agent requests the current Kubernetes cluster status.

PATTERN 02

Server Streaming

1 Request ⟶ N Responses

EXAMPLE WORKLOAD

Monitoring Agent continuously streams diagnostic findings.

PATTERN 03

Client Streaming

N Requests ⟶ 1 Response

EXAMPLE WORKLOAD

Log Collector Agent streams multiple log batches to an analysis service.

PATTERN 04

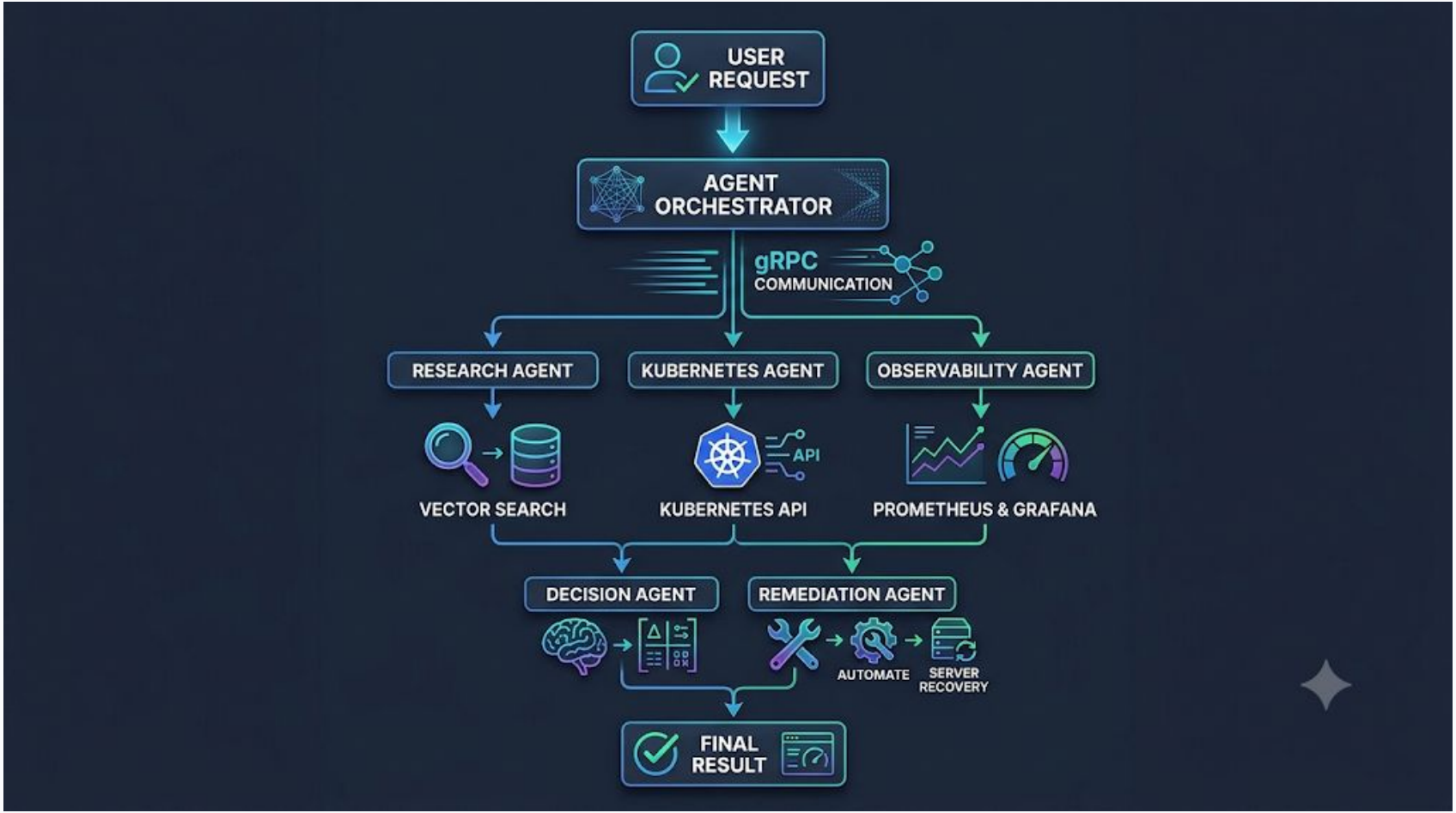
Bidirectional

Continuous Exchange

EXAMPLE WORKLOAD

Autonomous remediation agent exchanges commands and results with orchestrator.

Multi-Agent Architecture Using gRPC



KUBERNETES POD FAILURE INVESTIGATION

Deep Dive: Agent-to-Agent Communication Workflow

STEP 01

User Request

User asks: "Why is my production pod restarting?"

STEP 02

Planner Agent

Planner Agent receives the request.

STEP 03

gRPC Call

Planner invokes Kubernetes Agent through gRPC.

STEP 04

K8s Agent

Kubernetes Agent retrieves pod status and events.

STEP 05

Observability

Observability Agent analyzes metrics and logs.

STEP 06

Security Agent

Security Agent checks suspicious activity.

STEP 07

Decision Agent

Decision Agent correlates the findings.

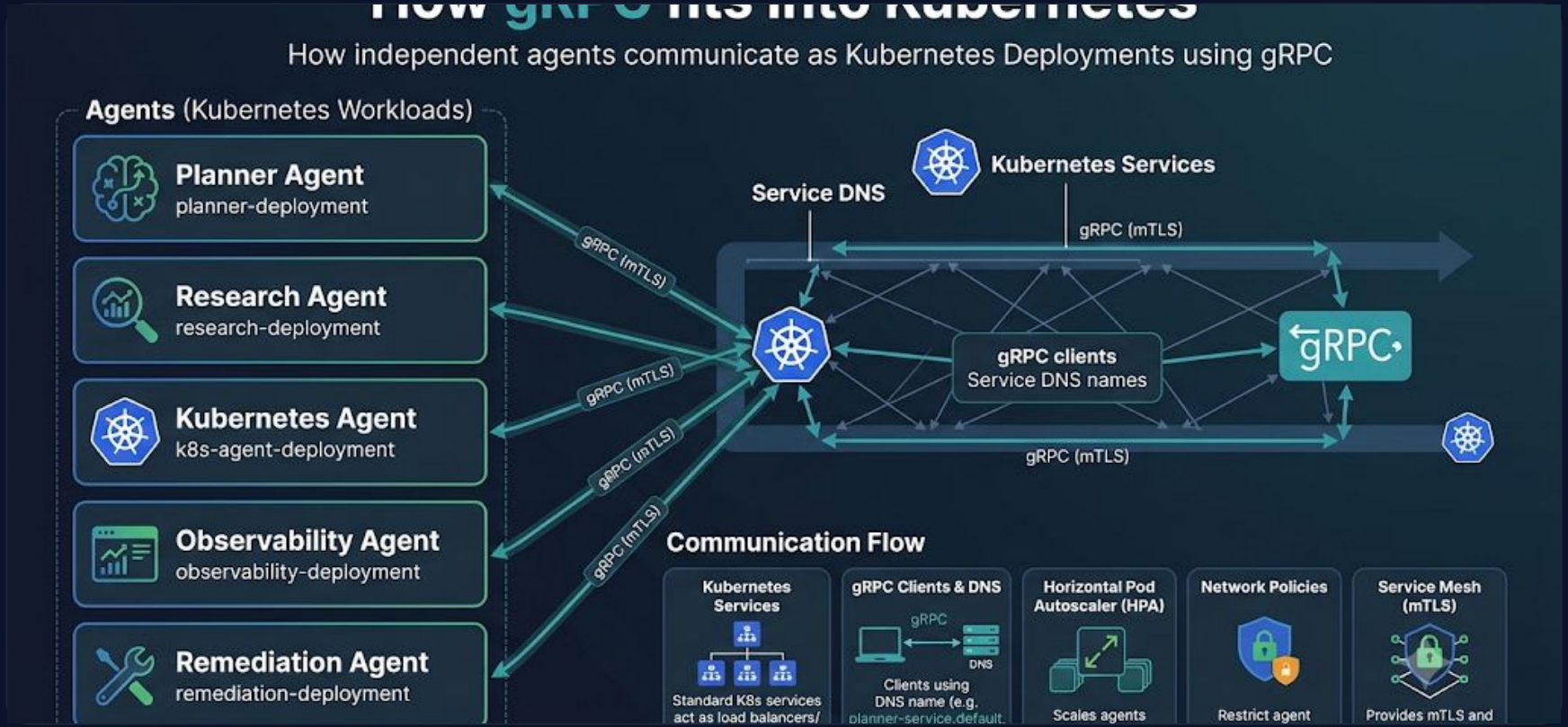
STEP 08

Remediation

Remediation Agent recommends or executes corrective actions, subject to authorization.

KUBERNETES & DISTRIBUTED AGENTS

gRPC in Kubernetes: Deploying Distributed Agents



GRPC PLATFORM ARCHITECTURE

Performance and Scalability Considerations

What makes a production-ready gRPC platform?

01
Connection Reuse
Avoid creating a new connection for every RPC.

02
HTTP/2 Multiplexing
Support concurrent streams over connections.

03
Payload Optimization
Keep messages appropriately sized.

04
Load Balancing
Distribute requests across agent replicas.

05
Backpressure
Control incoming work.

06
Resource Management
Bound concurrent requests and memory.

07
Streaming Limits
Manage long-lived streams and idle connections.

GOAL
Production Ready
Balance throughput, concurrency, and memory bounds for stability.

STEP-BY-STEP IMPLEMENTATION

Implementation: Building Your First gRPC Agent

Step 1: agent_service.proto

```
syntax = "proto3";

service AgentService {
  rpc ExecuteTask (TaskRequest)
    returns (TaskResponse);
}

message TaskRequest {
  string task_id = 1;
  string prompt = 2;
}

message TaskResponse {
  string result = 1;
  string status = 2;
}
```

02 Generate Code Generate gRPC client and server stubs from proto contract.

03 Implement Agent Server Write task execution handlers and service logic.

04 Implement Orchestrator Client Build the client to trigger tasks and consume streaming responses.

05 Deploy to Kubernetes Containerize and deploy both services into K8s cluster.

06 Test & Validate Verify communication, deadlines, failure handling, and streaming.

RECOMMENDED DEMO

Python Planner & Go/Python K8s Agent

Python-based Planner Agent communicating with a Python or Go-based Kubernetes Agent over gRPC.

FUTURE ROADMAP

Future: gRPC and the Next Generation of Agentic AI

Where the architecture is heading:

01
Autonomous Networks
Large-scale autonomous agent networks.

02
Cross-Cloud
Cross-cloud agent communication.

03
Polyglot Ecosystems
Language-independent agent ecosystems.

04
Real-Time Agents
Real-time collaborative agents.

05
Distributed Execution
Distributed AI execution environments.

06
Service Discovery
Agent-to-agent service discovery.

07
Security & Policy
Secure workload identity and policy enforcement.

VISION
Next-Gen Agentic AI
Seamless, polyglot, and hyper-scalable multi-agent intelligence platform.

Thank You

Questions & Answers

