CAZE LABS
ignite innovations, harness open source!

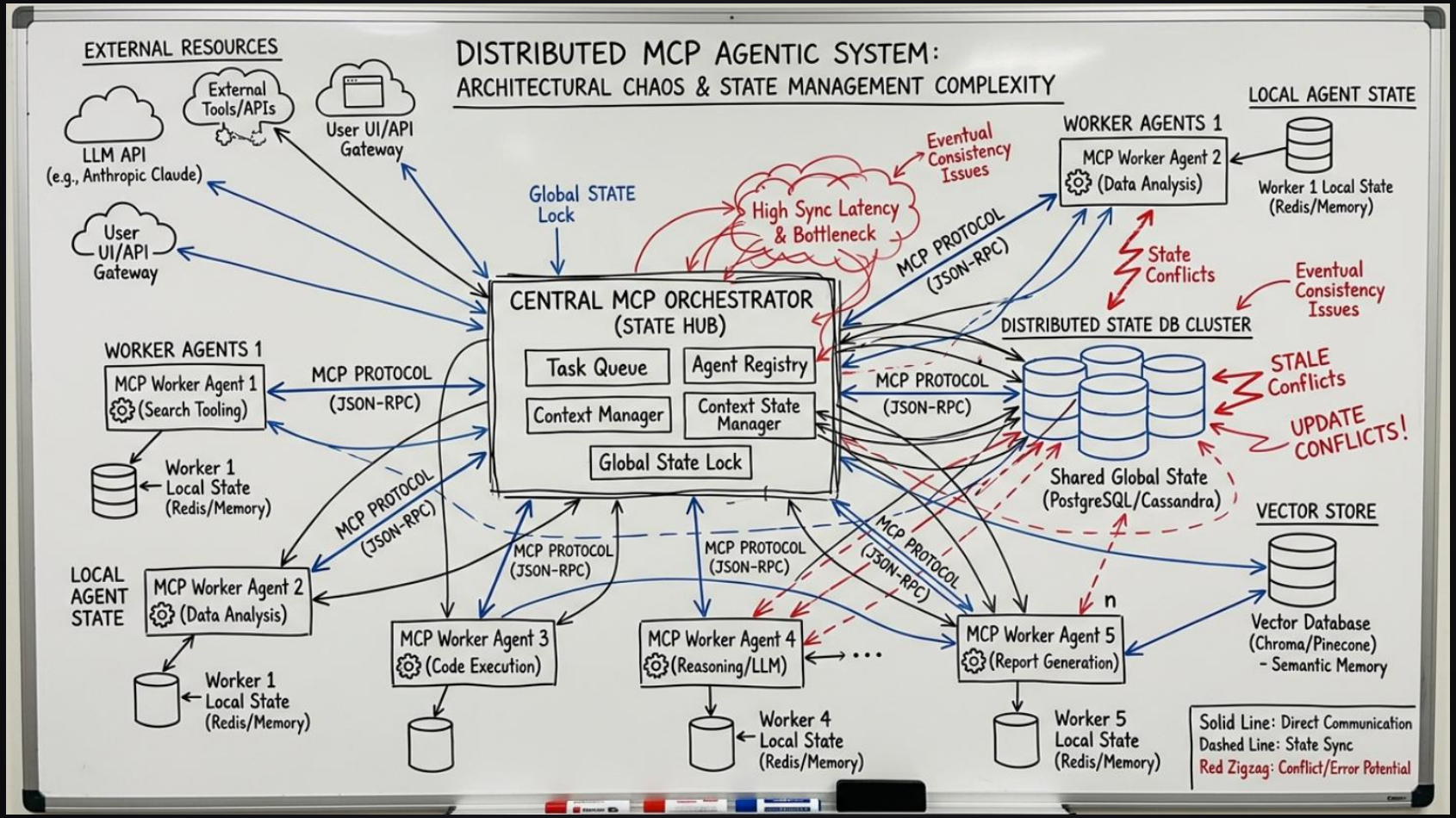
soda foundation

The State Sidecar

Solving the MCP Stateless Paradox

Advaith Sanil Kumar, Joval Kuruvila

MCP Dev Summit Mumbai

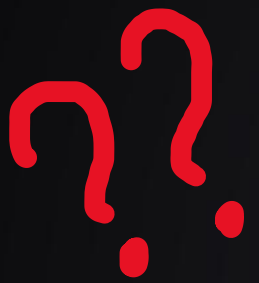

DISTRIBUTED MCP AGENTIC SYSTEM:
ARCHITECTURAL CHAOS & STATE MANAGEMENT COMPLEXITY
EXTERNAL RESOURCES
LLM API
(e.g., Anthropic Claude)
External Tools/APIs
User UI/API Gateway
User UI/API Gateway
Global STATE Lock
Eventual Consistency Issues
High Sync Latency & Bottleneck
CENTRAL MCP ORCHESTRATOR
(STATE HUB)
Task Queue
Agent Registry
Context Manager
Context State Manager
Global State Lock
WORKER AGENTS 1
MCP Worker Agent 1
(Search Tooling)
MCP PROTOCOL
(JSON-RPC)
Worker 1
Local State
(Redis/Memory)
LOCAL AGENT STATE
MCP Worker Agent 2
(Data Analysis)
Worker 1
Local State
(Redis/Memory)
MCP Worker Agent 3
(Code Execution)
MCP Worker Agent 4
(Reasoning/LLM)
Worker 4
Local State
(Redis/Memory)
MCP Worker Agent 5
(Report Generation)
Worker 5
Local State
(Redis/Memory)
WORKER AGENTS 1
MCP Worker Agent 2
(Data Analysis)
LOCAL AGENT STATE
Worker 1 Local State
(Redis/Memory)
State Conflicts
Eventual Consistency Issues
DISTRIBUTED STATE DB CLUSTER
STALE Conflicts
UPDATE CONFLICTS!
Shared Global State
(PostgreSQL/Cassandra)
VECTOR STORE
Vector Database
(Chroma/Pinecone)
- Semantic Memory
n
Solid Line: Direct Communication
Dashed Line: State Sync
Red Zigzag: Conflict/Error Potential

The Stateless Paradox

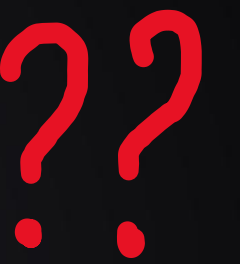
MCP Inherently Stateful! · Persistent connections and context

Stateless Infrastructure · Ephemeral, serverless scaling

The Gap · Externalise state, but no standard coordination layer



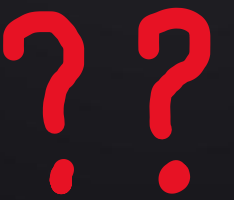
The Stateless Paradox



MCP Inherently Stateful! · Persistent connections and context

Stateless Infrastructure · Ephemeral, serverless scaling

The Gap · Externalise state, but no standard coordination layer



CAZE LABS
ignite innovations, harness open source!

soda foundation

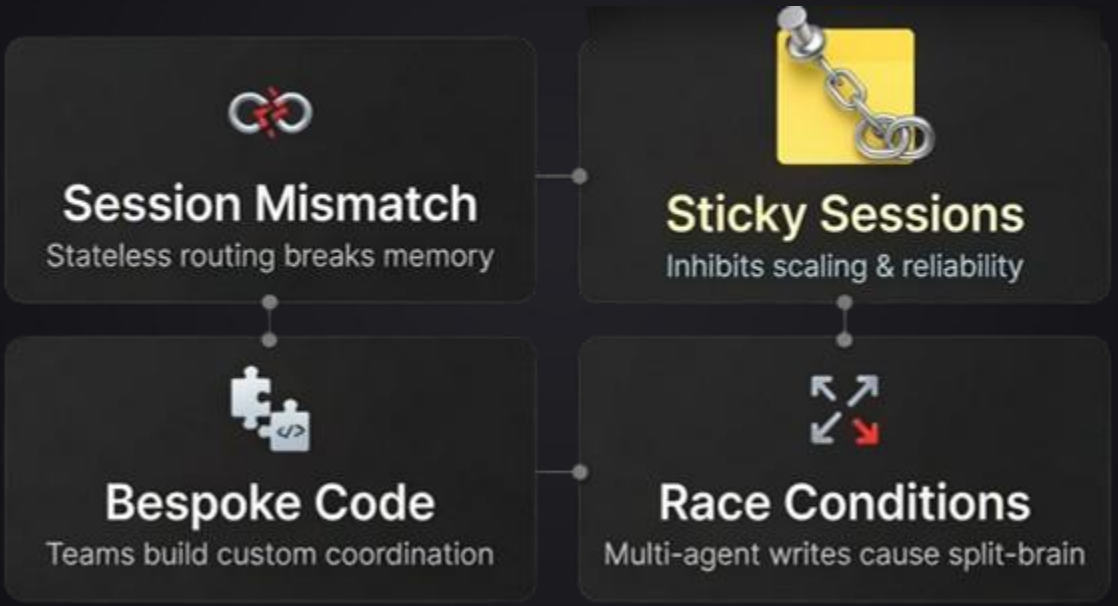
The NEW! Stateless Paradox

MCP Inherently Stateless! (soon) · No handshake, no sticky sessions

Scalability · Right architecture for scalable agentic systems

The Gap · **Applications STILL need a coordination layer**

What Breaks in Production



The Current Landscape

How does the ecosystem solve agent application state today?

Existing Approaches

Framework Lock-in

Coupled to specific runtimes (e.g., LangGraph, Letta)

Semantic Focus

Focused on memory recall, not workflow progress

No Protocol Standards

Not accessible as native MCP tool calls

The Gap

What if the coordination layer was just another MCP server?

Any agent, any framework, same interface.

Our Solution: The State Sidecar

An MCP-native coordination layer for state management in distributed environments; **EXACTLY the state backend the RC needs**

The Core Idea

Standalone Server · **19 tools across 5 groups of capabilities**

Universal Access · Any agent, any framework; just a standard tool call!

Distributed Coordination · Lock mechanisms, TTL leases, and atomic claims engineered for distributed resilience

Key Principles

Decoupled Lifecycle · State survives agent recycling

Concurrency-Safe · TTL leases and atomic claims

Resilient Execution · Checkpoint and resume from anywhere

The Comparison

The State Sidecar is the only solution that incorporates:

MCP-Native state management

Framework-Agnostic workflows

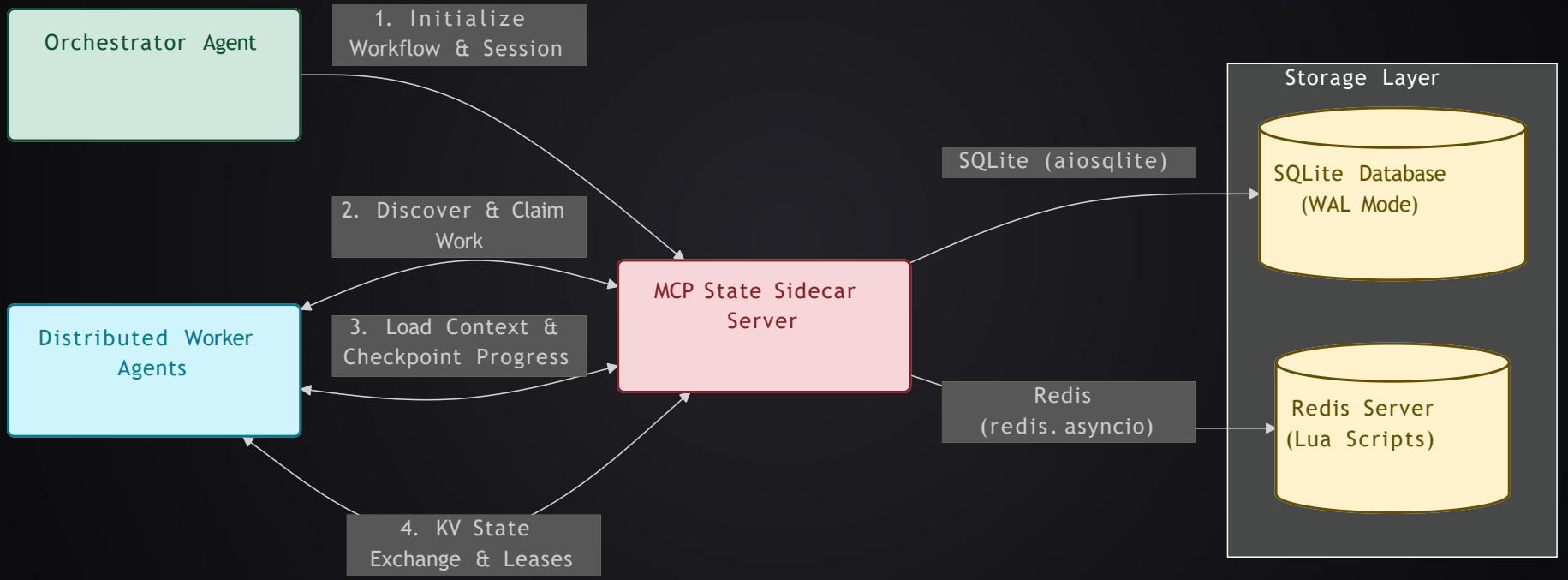
Provides an **external state backend** for the Tasks primitive

Distributed Suitability · Built-in leases, atomic claims, and checkpoint recovery

Technical Deep Dive

Let’s get into the nitty gritty!

Architecture Overview



Five Groups of Tools

1. **Key-Value Store:** Simple CRUD state
2. **Workflow Lifecycle:** Durable pipelines
3. **Lease Concurrency:** TTL locks
4. **Session & History:** Snapshots + audit
5. **Observability:** Diagnostics & health

Key-Value Store

CRUD operations with TTL-based state and associated agent metadata.

- **Store and auto-expire** with TTL
- Info **tagged with the agent** that wrote it
- Essentially acts as a **shared whiteboard**

Workflow Lifecycle

Orchestrate work using atomic claim mechanisms and step-level checkpoints.

Orchestration · Registers jobs with tags

Discovery · Workers find unclaimed work matching tags

Execution · First worker to claim the job wins atomically

Lease-Based Concurrency

Acquire exclusive TTL-based locks to coordinate shared resources.

Acquire · Get lease with TTL

Heartbeat · Renew lease while working

Auto-Release · Crash → TTL expires → lease freed

Sessions & History

Snapshot conversational context and trace execution history.

Save Context · Snapshot current session/conversation state

Trace History · View all logs for a given key

The Persistence Layer

Do whatever you want! You have a unified interface :)

Persistence Layer

SQLite · WAL mode, zero config, local development

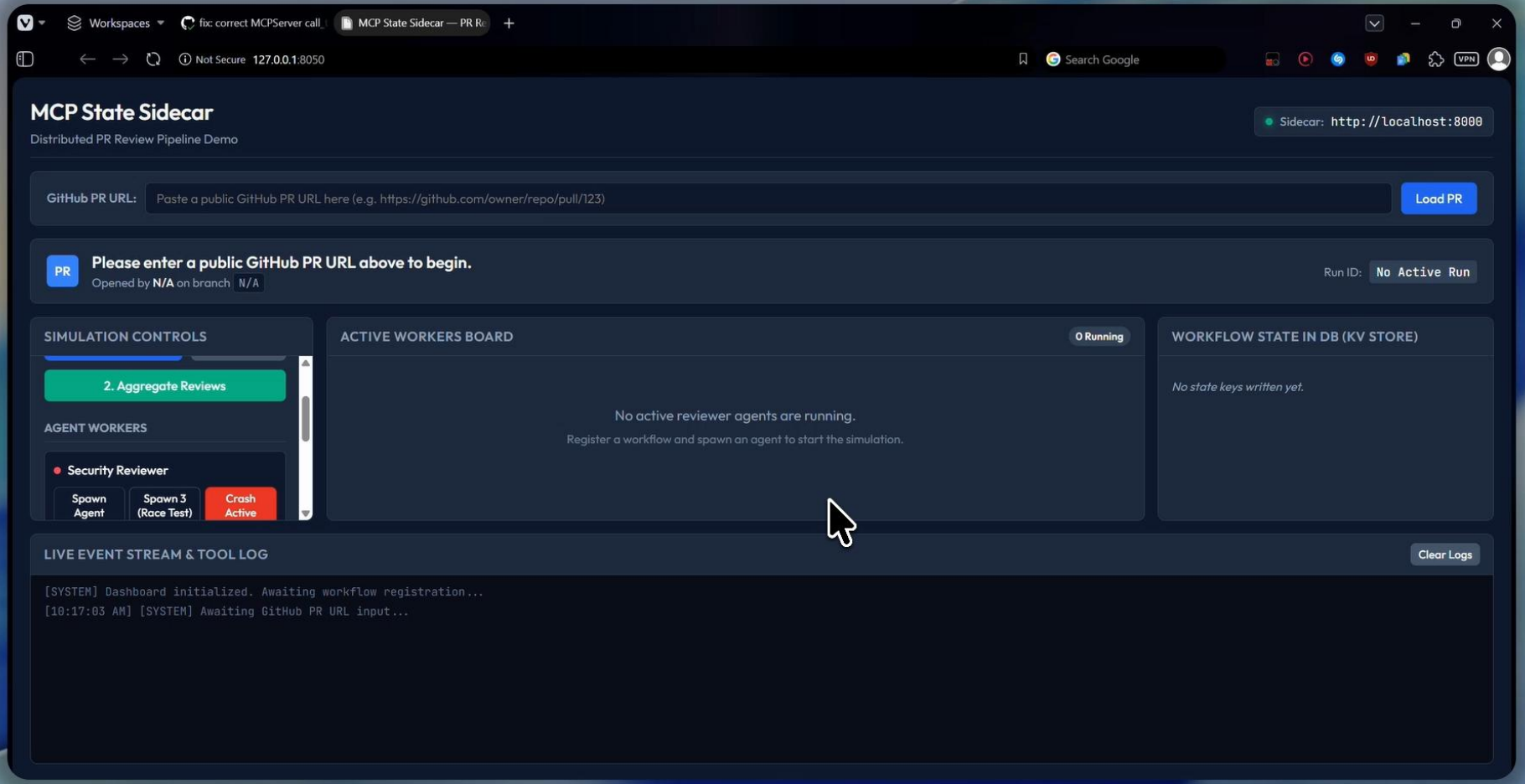
Redis · Native TTL, atomic Lua lock validations, production scaling

> In our implementation, simply an environment variable!

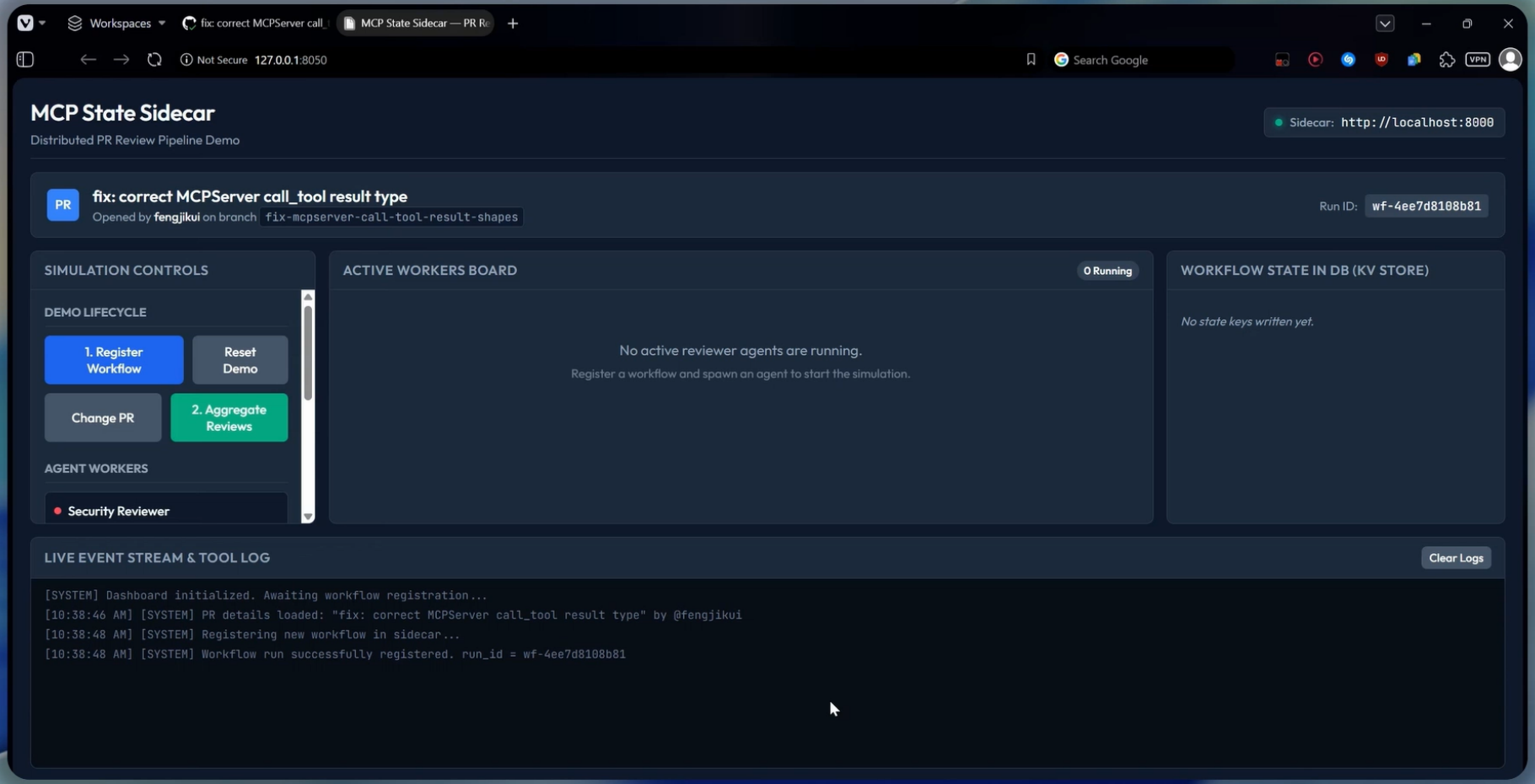
Demo

The Setup

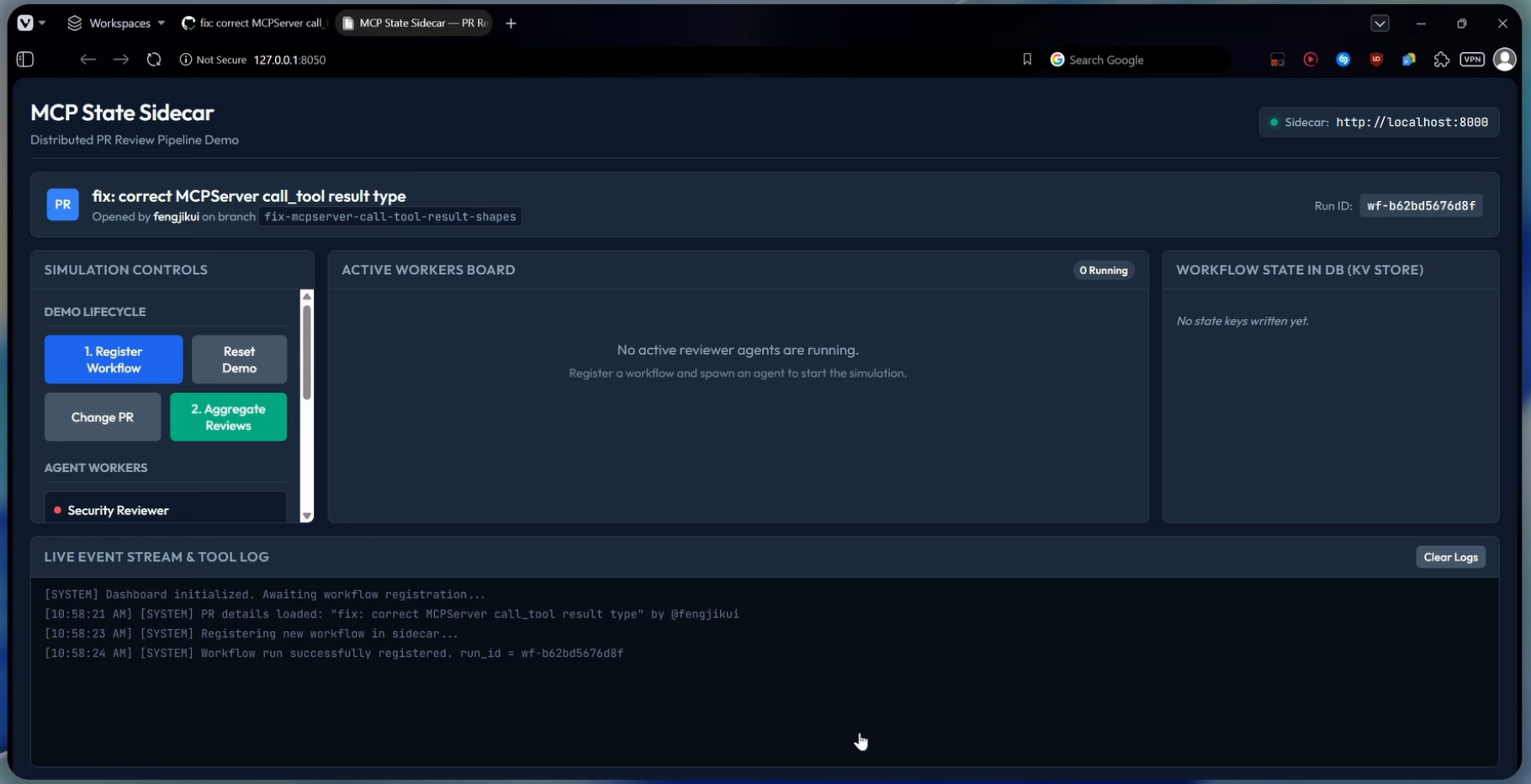
- Distributed agentic GitHub PR reviewer
- Three types of agents: Security, Performance, Style
- Entirely independent from one another; application coordinates through the sidecar!


Workspaces
fix: correct MCPServer call
MCP State Sidecar — PR Re
Not Secure 127.0.0.1:8050
Search Google
MCP State Sidecar
Distributed PR Review Pipeline Demo
Sidecar: http://localhost:8000
GitHub PR URL:
Paste a public GitHub PR URL here (e.g. https://github.com/owner/repo/pull/123)
Load PR
PR
Please enter a public GitHub PR URL above to begin.
Opened by N/A on branch N/A
Run ID: No Active Run
SIMULATION CONTROLS
2. Aggregate Reviews
AGENT WORKERS
Security Reviewer
Spawn Agent
Spawn 3 (Race Test)
Crash Active
ACTIVE WORKERS BOARD
0 Running
No active reviewer agents are running.
Register a workflow and spawn an agent to start the simulation.
WORKFLOW STATE IN DB (KV STORE)
No state keys written yet.
LIVE EVENT STREAM & TOOL LOG
Clear Logs
[SYSTEM] Dashboard initialized. Awaiting workflow registration...
[10:17:03 AM] [SYSTEM] Awaiting GitHub PR URL input...

Now what about crashes?


Workspaces
fix: correct MCPServer call_
MCP State Sidecar — PR Re
Not Secure 127.0.0.1:8050
Search Google
VPN
MCP State Sidecar
Distributed PR Review Pipeline Demo
Sidecar: http://localhost:8000
PR
fix: correct MCPServer call_tool result type
Opened by fengjikui on branch fix-mcpserver-call-tool-result-shapes
Run ID: wf-4ee7d8108b81
SIMULATION CONTROLS
DEMO LIFECYCLE
1. Register Workflow
Reset Demo
Change PR
2. Aggregate Reviews
AGENT WORKERS
Security Reviewer
ACTIVE WORKERS BOARD
0 Running
No active reviewer agents are running.
Register a workflow and spawn an agent to start the simulation.
WORKFLOW STATE IN DB (KV STORE)
No state keys written yet.
LIVE EVENT STREAM & TOOL LOG
Clear Logs
[SYSTEM] Dashboard initialized. Awaiting workflow registration...
[10:38:46 AM] [SYSTEM] PR details loaded: "fix: correct MCPServer call_tool result type" by @fengjikui
[10:38:48 AM] [SYSTEM] Registering new workflow in sidecar...
[10:38:48 AM] [SYSTEM] Workflow run successfully registered. run_id = wf-4ee7d8108b81

Let’s add concurrency to the mix!


Workspaces
fix: correct MCPServer call_
MCP State Sidecar — PR Re
Not Secure 127.0.0.1:8050
Search Google
VPN
MCP State Sidecar
Distributed PR Review Pipeline Demo
Sidecar: http://localhost:8000
PR
fix: correct MCPServer call_tool result type
Opened by fengjikui on branch fix-mcpserver-call-tool-result-shapes
Run ID: wf-b62bd5676d8f
SIMULATION CONTROLS
DEMO LIFECYCLE
1. Register Workflow
Reset Demo
Change PR
2. Aggregate Reviews
AGENT WORKERS
Security Reviewer
ACTIVE WORKERS BOARD
0 Running
No active reviewer agents are running.
Register a workflow and spawn an agent to start the simulation.
WORKFLOW STATE IN DB (KV STORE)
No state keys written yet.
LIVE EVENT STREAM & TOOL LOG
Clear Logs
[SYSTEM] Dashboard initialized. Awaiting workflow registration...
[10:58:21 AM] [SYSTEM] PR details loaded: "fix: correct MCPServer call_tool result type" by @fengjikui
[10:58:23 AM] [SYSTEM] Registering new workflow in sidecar...
[10:58:24 AM] [SYSTEM] Workflow run successfully registered. run_id = wf-b62bd5676d8f

So what can you build with this?

Unlocked Patterns

Multi-session pipelines

Asynchronous agent handoffs

Fan-out/Fan-in queues

Serverless session continuity

Basically any complex distributed setup!

Flexible Orchestration

- Deterministic pipelines
- Dynamic pipelines
- Both!

Key Takeaways

The Stateless Paradox · Scaled agentic systems need coordination, regardless of MCP being stateful or stateless

Universal Interface · Decoupled, framework-agnostic MCP tools

Distributed Primitives · TTL leases, atomic claims, checkpointing, and crash recovery

Going Forward…

CAZE LABS
ignite innovations, harness open source!

- AI Native, Agentic and MCP
- Identified state management challenge from our own product dev
- This led to the “State Sidecar”!
- Next: Formal open-source release under Caze, potentially integrating RC changes

soda foundation

- SODA Contexture: Open Context Specification (OCS) (https://github.com/sodafoundation/contexture)
- Next: AI Observability and Benchmarking

CAZE LABS
ignite innovations, harness open source!

soda foundation

Thank You!

Any Questions?

linkedin.com/advaithsanil

Scan to view on GitHub

linkedin.com/joval-kuruvila