Agent-Based SRE: Automated Diagnosis and Mitigation in Kubernetes

Marcello Martini

12/09/2025 - NECSTLab

The Problem

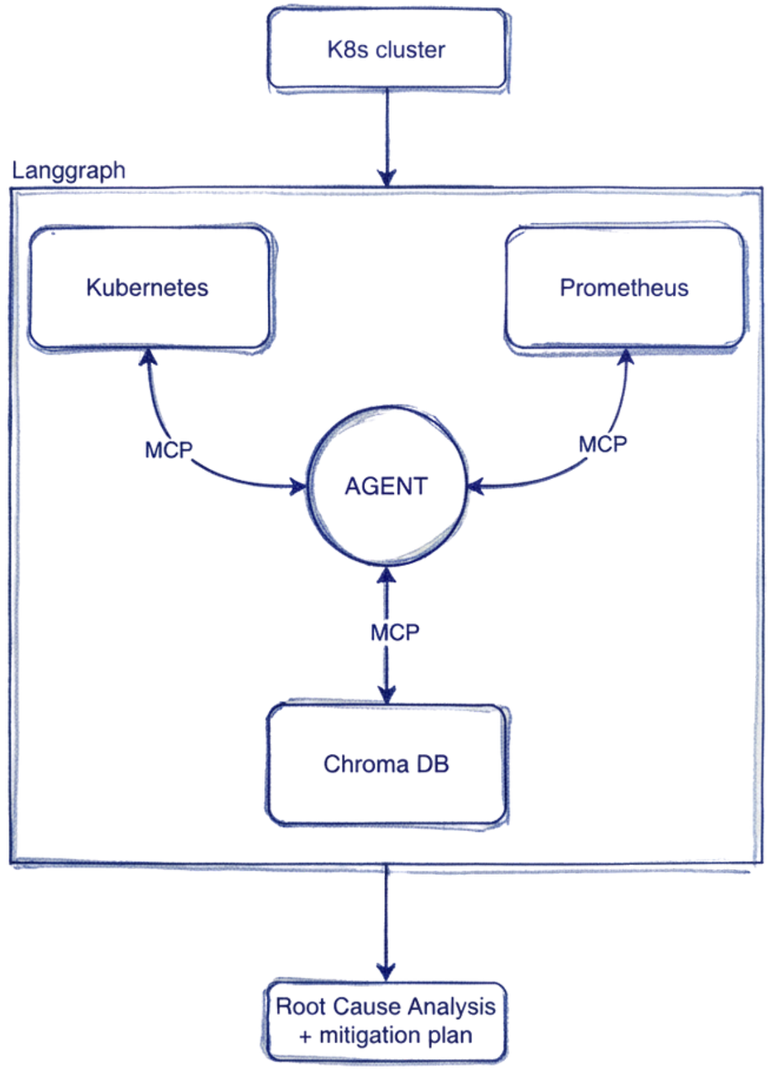
Increasing complexity of cloud-native systems.

Site Reliability Engineering teams face **heavy operational load**.

Need for **fast diagnosis**, **automatic mitigation**, and an **holistic view** of system health to address issues rapidly.

Solution

An autonomous **SRE agent** that inspects Kubernetes cluster signals, pinpoints root cause, and proposes fix using **MCP tools**, **structured reasoning**, and **incident memory** to cut time and cost.



Key components

LangChain & LangGraph

Agent built with LangGraph (stateful graph) on top of LangChain.

MCP servers

Standard integration via Model Context Protocol (MCP): Kubernetes, Prometheus, and ChromaDB.

LLMs

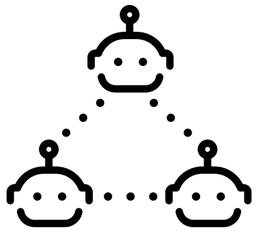
LLM used as reasoning engine for the agent.

Microsoft AIOpsLab

Microsoft AIOpsLab is used to set up the testbed (i.e., the distributed infrastructure) and inject faults.

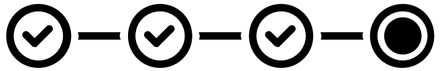
Main contributions

End-to-end agentic SRE for Kubernetes



Cost/efficiency optimization

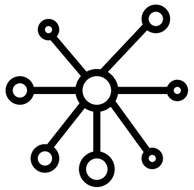
Explainability and observability by design



Standardized, safe integrations via MCP

Incident memory with RAG (ChromaDB)

Reproducible PoC on a realistic testbed



Agent configurations

Diagnostic agent

- ReAct agent (baseline)
- ReAct + reduced context window
- ReAct + structured schema

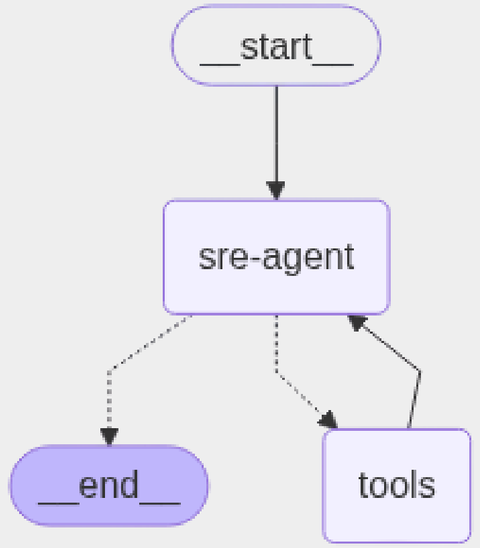
End-to-end SRE agent (diagnostic + mitigation)

- Structured schema
- Structured schema + RAG

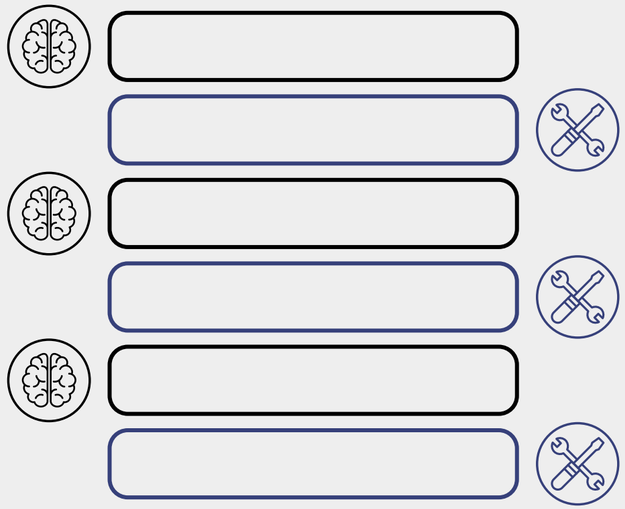
ReAct baseline

- **ReAct loop**: reason → call tool → observe → iterate.
- Nodes: **LLM node** (decide next step), **Tool node** (route to K8s/Prometheus).
- Works but **expensive**: full chat history grows per iteration
- No explicit explainability

Agent graph



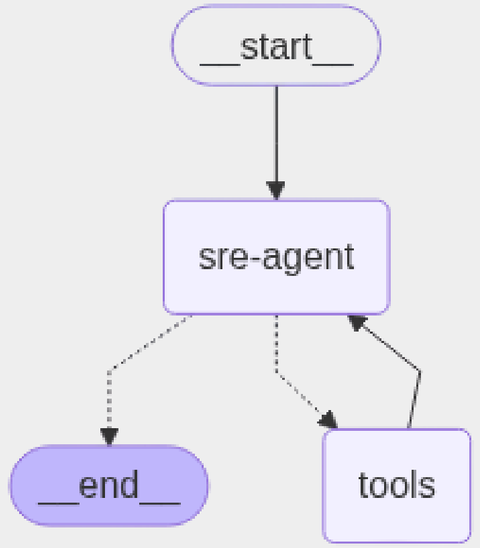
Graph state



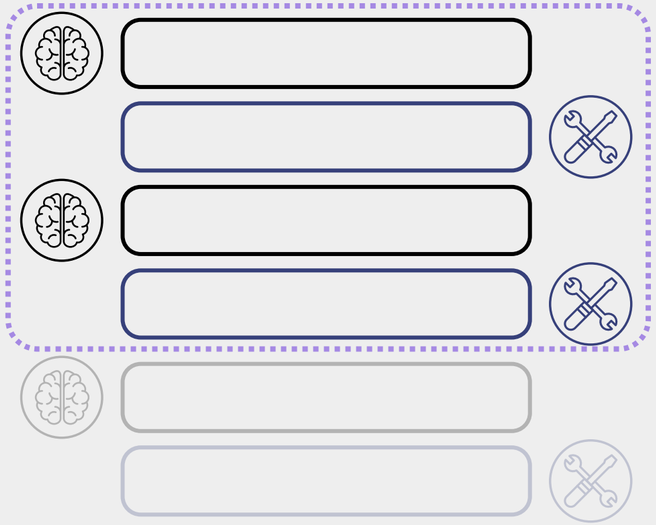
Reduced context window

- Keep only **last ~7 messages** in prompt.
- Pros: **reduces tokens** per step.
- Cons: may **drop early clues** → repeated tool calls.
- No explicit explainability

Agent graph



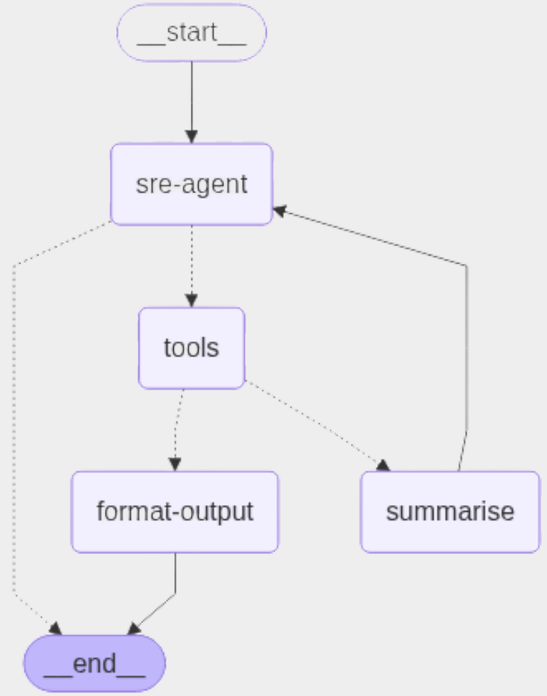
Graph state



Structured graph schema

- Replace raw chat history with typed state:
  - Insights (key findings)
  - Previous steps (tool calls)
  - Response (final structured output)
- New node: Summarise extracts key insights each step.
- **Big token savings** without losing salient context and explainability improved.
- Explicit explainability

Agent graph



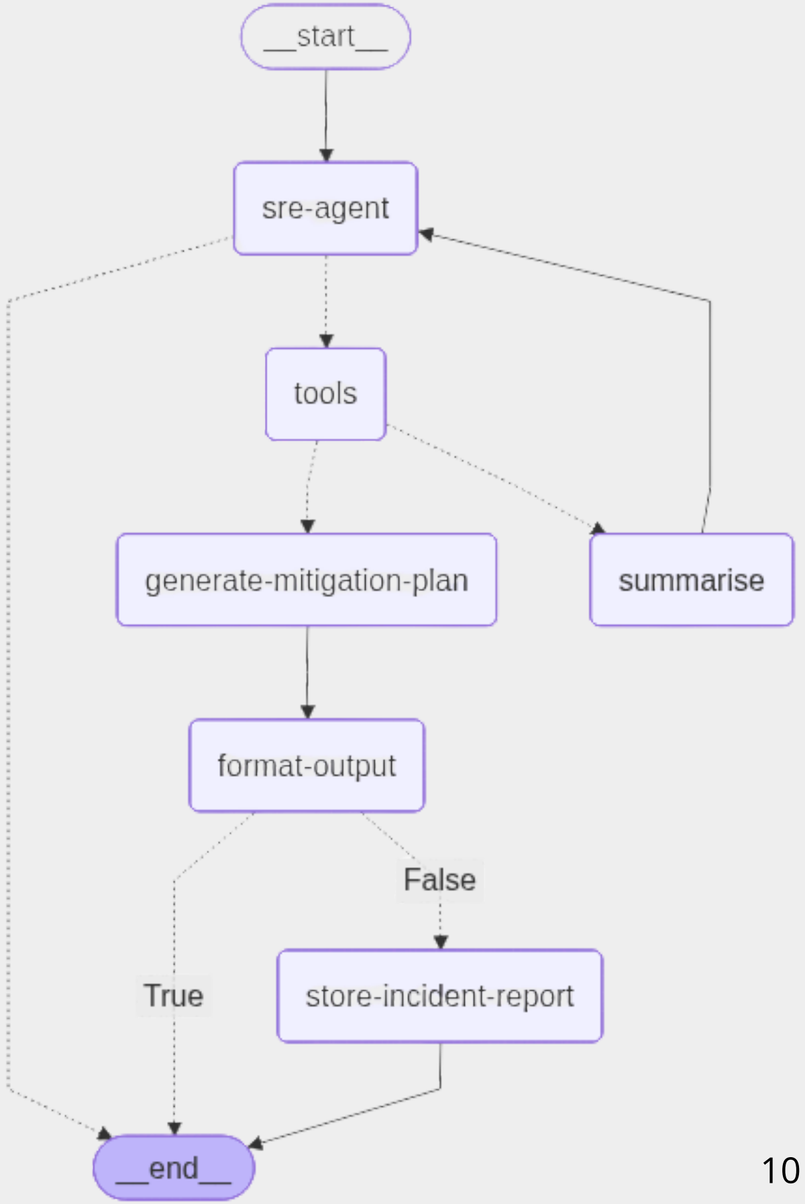
Graph state

```python
class SREAgentState(TypedDict):
    messages: Annotated[list[AnyMessage], add_messages]
    app_summary: str
    insights: Annotated[list[str], operator.add]
    prev_steps: Annotated[list[str], operator.add]
    response: str
    final_output: str
```

Mitigation plan + RAG incident reuse

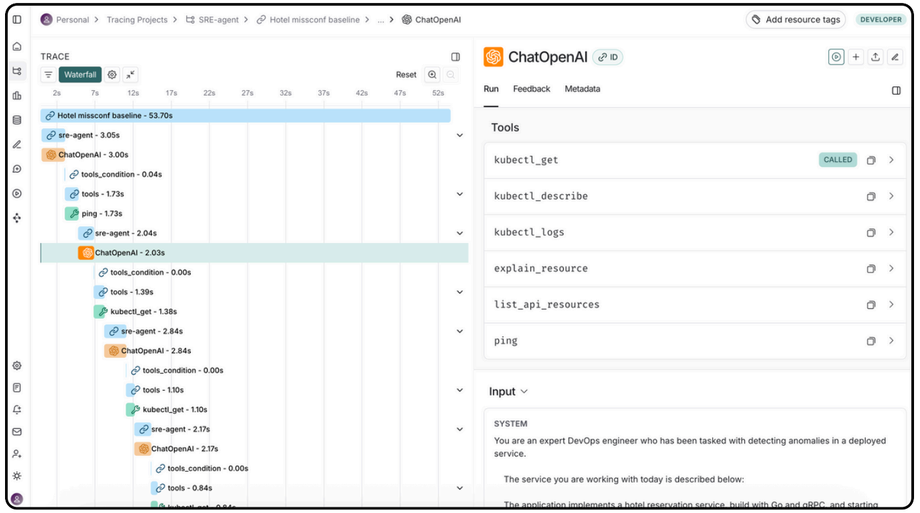
- Separate ReAct agent **proposes mitigation**; checks feasibility via K8s tools.
- **RAG** on incident history (**ChromaDB**): reuse known diagnoses/plans (based on symptoms).
- Retrieval layer acts as a cache: faster, cheaper on repeat incidents.

Agent graph



Experimental setup

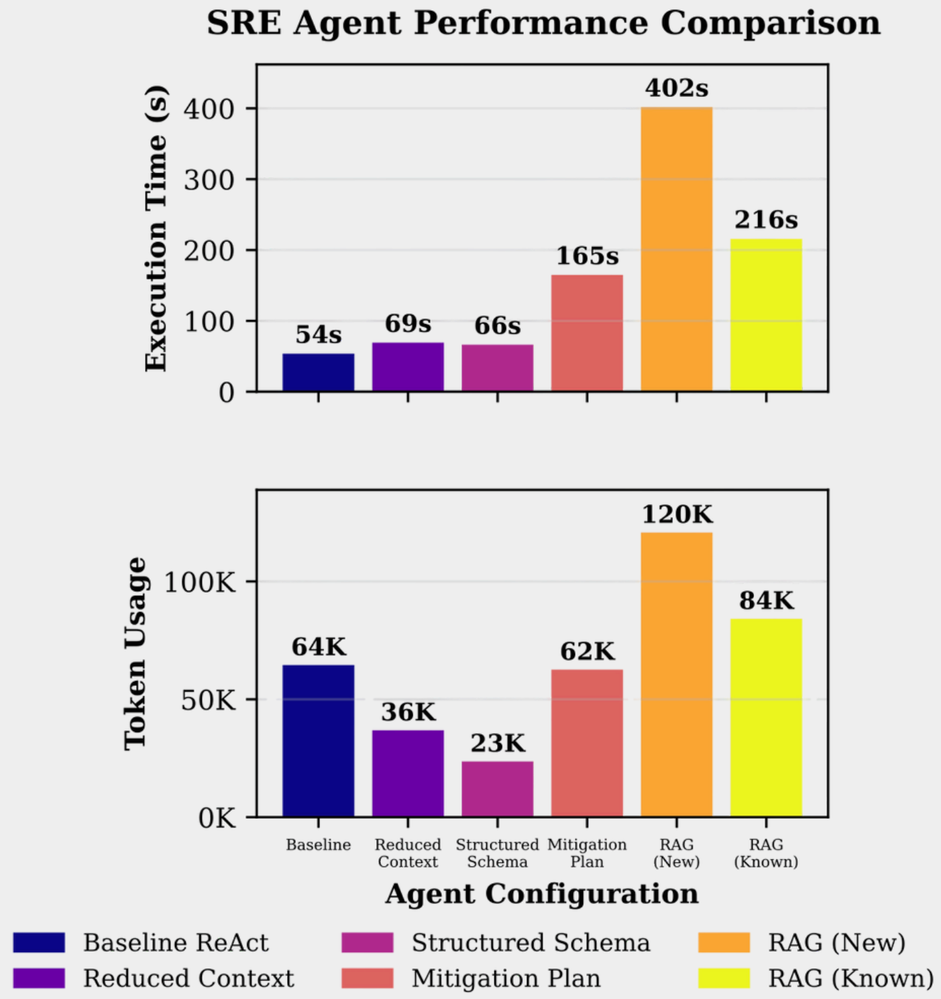
- **Kind cluster** on Mac M1, **AIOpsLab** to deploy **DeathStarBench** (hotel-reservation).
- **Chaos Mesh** faults; **wrk2** workload; **Prometheus** for metrics.
- **LangSmith** for tracing tokens, tool calls, intermediate states.



Langsmith platform

Results

- **Diagnosis**: consistently found root cause in injected scenario.
- **Token savings**: ReAct baseline 64k → Structured schema 23k.
- **Latency tradeoff**: 54s → 66s (added summarisation steps).
- **RAG reuse**: ~2× faster and −36k tokens on repeated incidents.



Limitations & Future Works

Limitations

- Tested only one fault scenario due to hardware limitations.
- Prometheus occasionally fails to collect metrics due to resource constraints.

Future Works

- Run the agent across the **full AIOpsLab suite**.
- Connect **Loki** for logs and **Firecrawl** for web scraping.
- **Human-in-the-loop** for safe actions.
- Use **fine-tuned SLMs** instead of LLMs for reasoning.

Thank you

github.com/martinimarcello00/SRE-agent

Marcello Martini - marcello.martini@mail.polimi.it

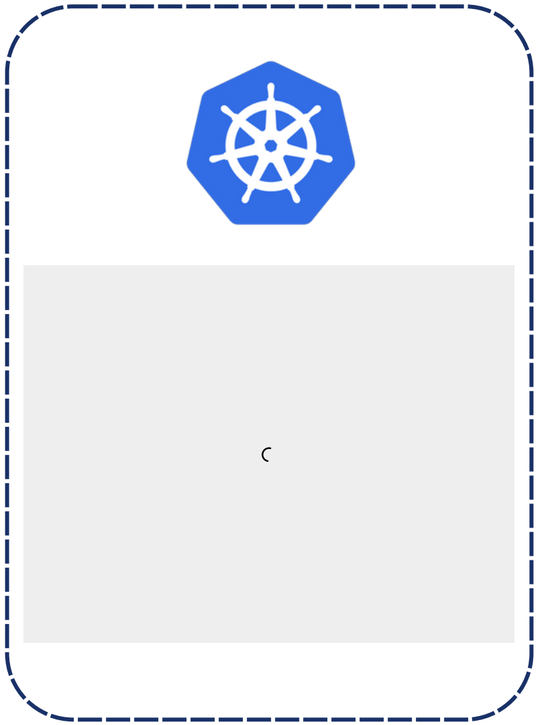
State of the Art

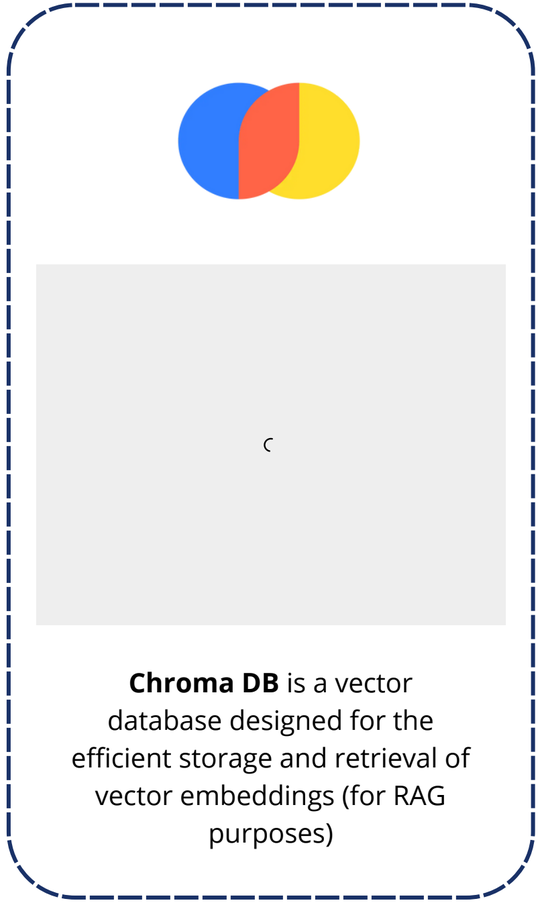
AIOpsLab - Microsoft *(https://doi.org/10.48550/arXiv.2501.06706)*
AIOpsLab presents a benchmark suite for evaluating AI agents in autonomous cloud management, comparing LLM-based and traditional AIOps solutions using fault-injected microservices scenarios.

ITBench *- IBM (https://doi.org/10.48550/arXiv.2502.05352)*
ITBench introduces an open-source framework for benchmarking AI agents on diverse, real-world IT automation tasks, supporting multiple IT personas and offering comprehensive observability and alerting.

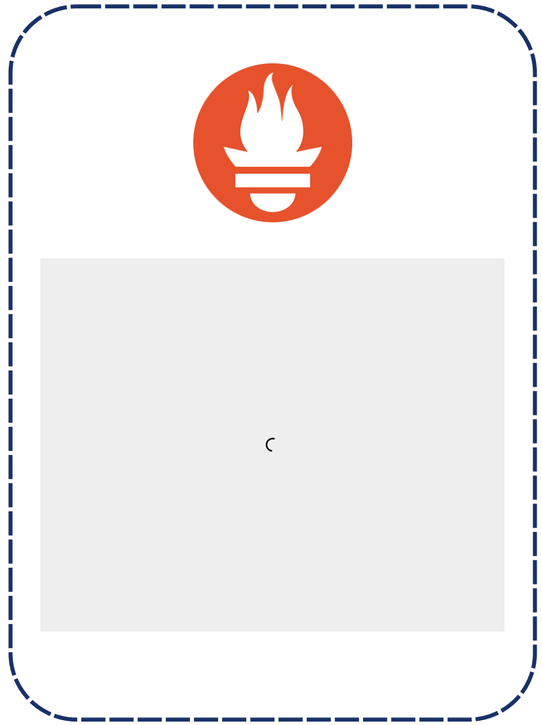
MonitorAssistant - Microsoft (*https://doi.org/10.1145/3663529.3663826*)
MonitorAssistant proposes an end-to-end system that leverages large language models to simplify cloud service monitoring, providing automated configuration recommendations, practical anomaly detection, and interactive troubleshooting guidance.

MCP Servers



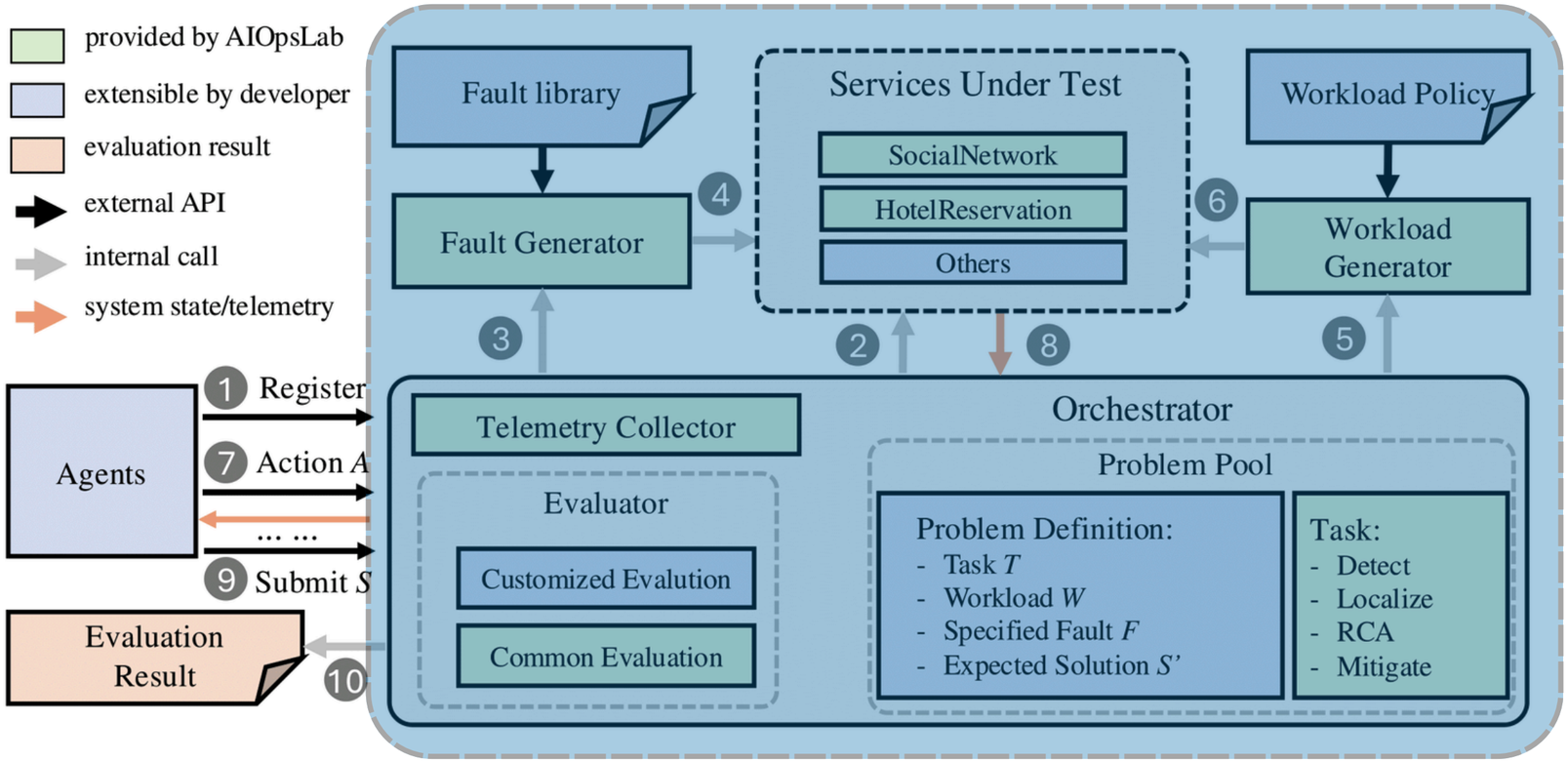


Chroma DB is a vector database designed for the efficient storage and retrieval of vector embeddings (for RAG purposes)

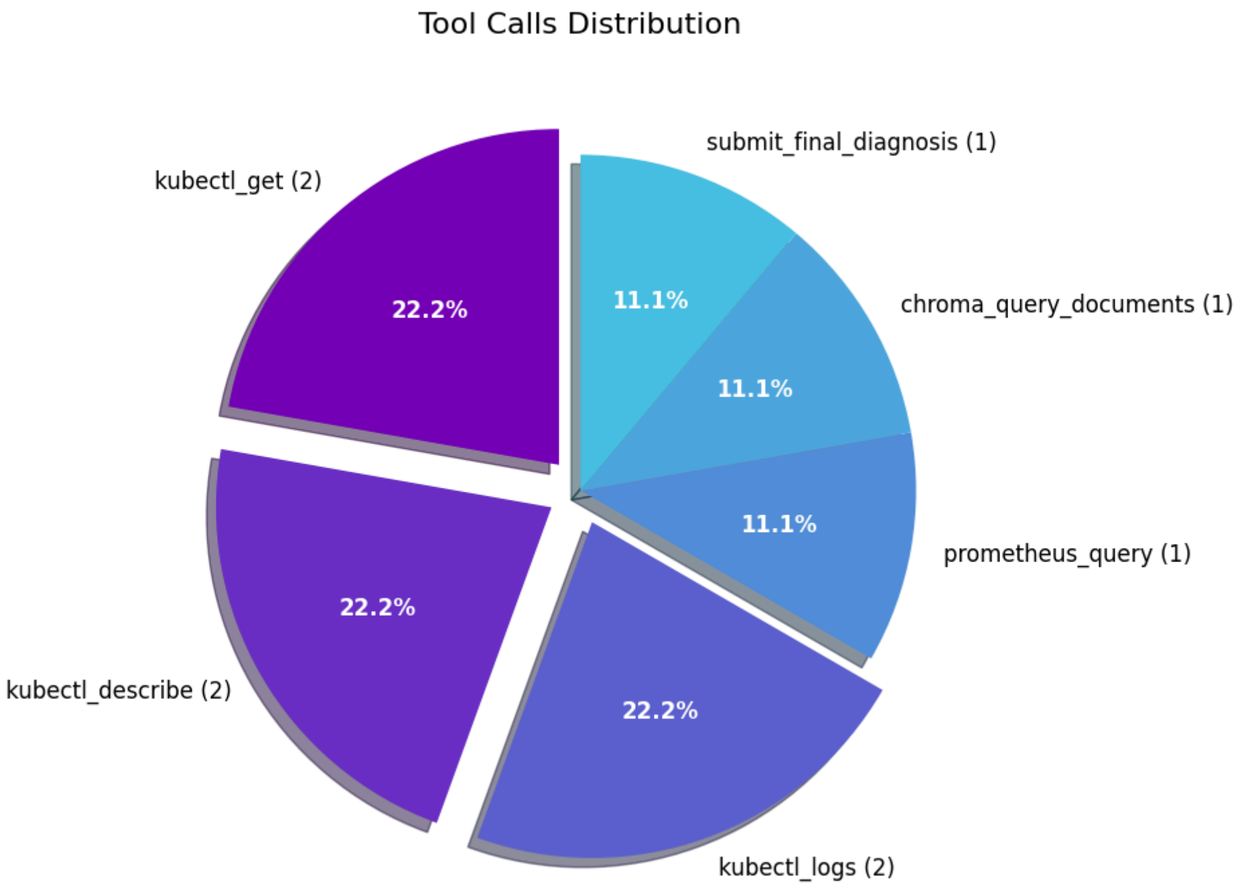


AIOpsLab

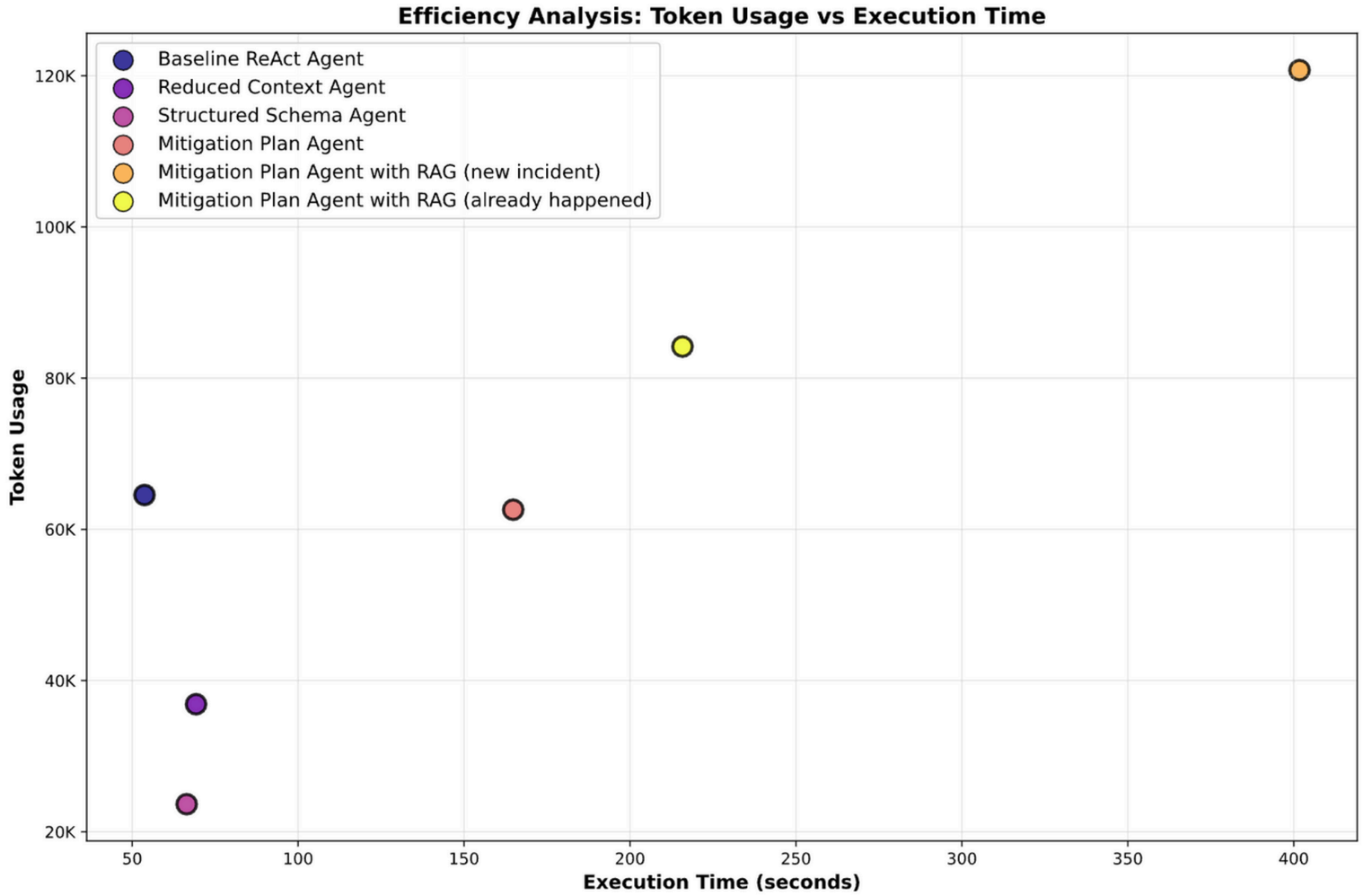
Microsoft AIOpsLab used to generate realistic infrastructure and inject faults (e.g., misconfigurations, DoS)



Tool calls analysis

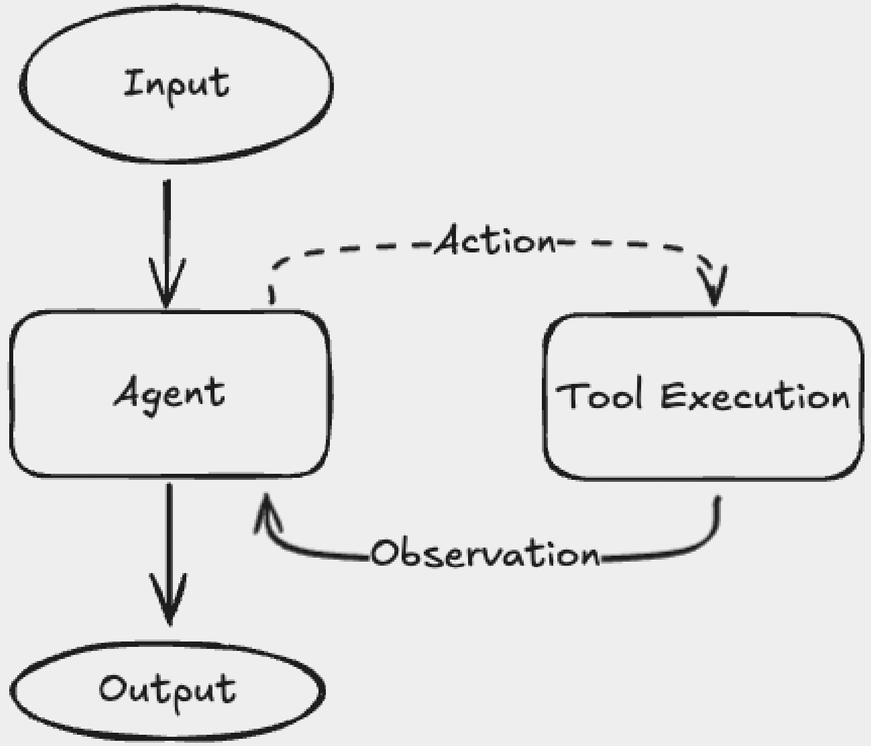


Efficiency analysis



ReAct pattern

- Combines **Reasoning + Acting** by interleaving **chain-of-thought reasoning** with **external tool calls**.
- Operates in a **loop**: Thought → Action → Observation, allowing **dynamic information retrieval** and **step-by-step problem solving**
- Enables LLMs to go beyond static knowledge by integrating tools to gather real-time data and perform intermediate computations before final answer



Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023, January). React: Synergizing reasoning and acting in language models. In International Conference on Learning Representations (ICLR).