Agent-Based SRE: Automated Diagnosis and Mitigation in K8s

Marcello Martini

Problem

Modern cloud-native infrastructures (e.g., Kubernetes) are complex and prone to faults.

Site Reliability Engineers (SREs) face challenges in rapidly diagnosing and mitigating incidents in distributed systems.

There is a need for intelligent systems that can support SREs with actionable insights and automated mitigation plans

Goal

Develop a prototype of an autonomous agent that observes the overall architecture, identifies issues, and proposes (or initiates) mitigation steps in Kubernetes environments.

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.

Tools

Microsoft AIOpsLab

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

LangChain & LangGraph

LangChain and LangGraph are used to create workflows and agents.

LLMs

Large Language Models are used as engines for parsing and processing metrics.

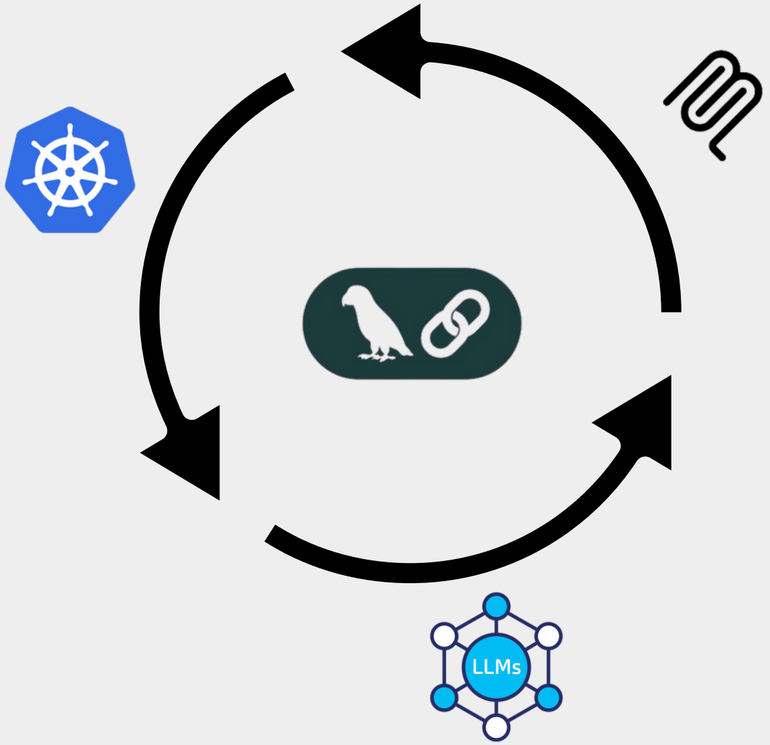
MCP servers

Model Context Protocol servers are used to interact with Kubernetes and other external resources.

Solution Architecture

Agent-based system with modular design.
Langchain agent connects to multiple MCPs:

- Kubernetes MCP: Cluster state and events
- Prometheus MCP: Metrics and alerts
- RAG MCP: Knowledge retrieval
- (expandable)



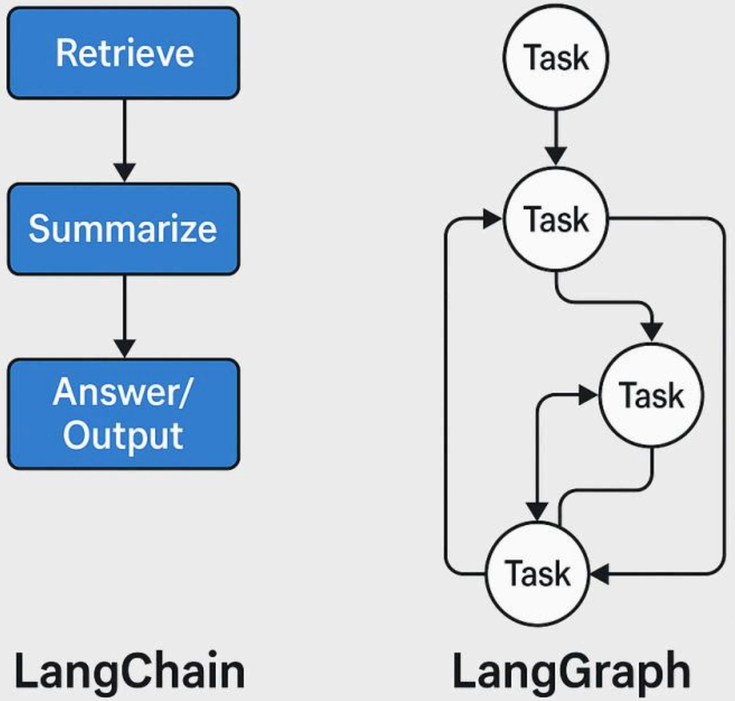
Requirements

- Agent must autonomously detect and diagnose issues.
- Provide clear explanations for each step taken.
- Suggest and/or execute mitigation actions.
- Evaluation framework to assess agent performance.

LangChain and LangGraph

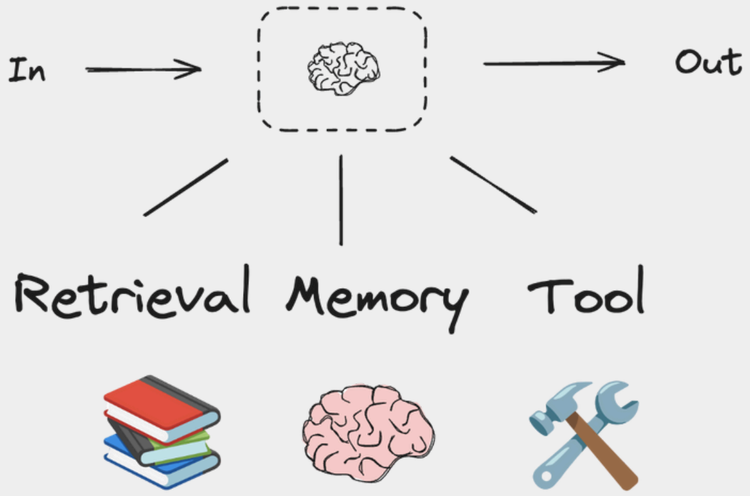
LangChain is a framework that simplifies the development of powerful applications using large language models by connecting them with data sources, tools, and workflows.

LangGraph builds on LangChain, enabling developers to create dynamic, multi-step conversational agents with flexible, **graph-based workflows**.



LangChain and LangGraph

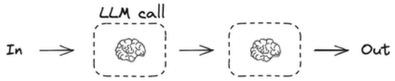
- **Modular LLM Workflows**: Build complex applications by chaining together language models and tools.
- **Dynamic Conversational Agents**: Create agents capable of multi-step reasoning.
- **Rich Ecosystem Integrations**: Connect easily with databases, APIs, document loaders etc.
- **State Management**: Maintain conversation history and manage state across interactions.



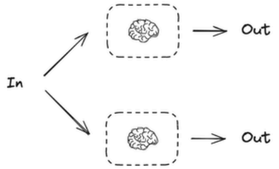
LangChain and LangGraph

Workflows

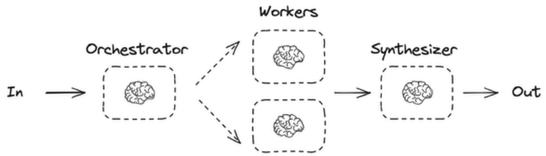
Prompt Chaining



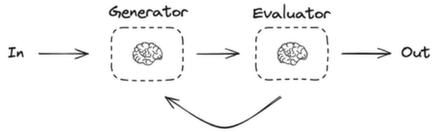
Parallelization



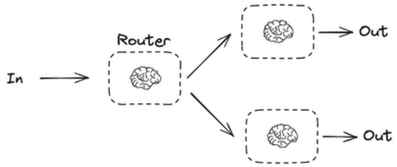
Orchestrator-Worker



Evaluator-optimizer



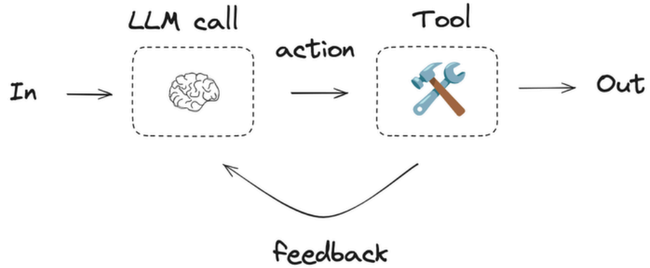
Routing



LLM is embedded in predefined code paths

LLM directs control flow through predefined code paths

Agent



LLM directs its own actions based on environmental feedback

MCP servers

Flux159/**mcp-server-kubernetes**

MCP Server for kubernetes management commands

23 Contributors · 7 Used by · 903 Stars · 140 Forks

Flux159/mcp-server-kubernetes: MCP Server for kubernetes management commands

MCP Server for kubernetes management commands. Contribute to Flux159/mcp-server-kubernetes development by creating an account on GitHub.

GitHub

chroma-core/**chroma-mcp**

A Model Context Protocol (MCP) server implementation that provides database capabilities for Chroma

4 Contributors · 4 Issues · 241 Stars · 44 Forks

chroma-core/chroma-mcp: A Model Context Protocol (MCP) server implementation that provides database...

A Model Context Protocol (MCP) server implementation that provides database capabilities for Chroma - chroma-core/chroma-mcp

GitHub

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

pab1it0/**prometheus-mcp-server**

A Model Context Protocol (MCP) server that enables AI assistants to query and analyze Prometheus metrics through standardized interfaces.

6 Contributors · 1 Issue · 157 Stars · 32 Forks

pab1it0/prometheus-mcp-server: A Model Context Protocol (MCP) server that enables AI assistants to que...

A Model Context Protocol (MCP) server that enables AI assistants to query and analyze Prometheus metrics through standardized interfaces. - pab1it0/prometheus-mcp-server

GitHub

Firecrawl

mendableai/**firecrawl**

🔥 Turn entire websites into LLM-ready markdown or structured data. Scrape, crawl and extract with a single API.

98 Contributors · 144 Used by · 39 Discussions · 43k Stars · 4k Forks

mendableai/firecrawl: 🔥 Turn entire websites into LLM-ready markdown or structured data. Scrape, crawl and...

🔥 Turn entire websites into LLM-ready markdown or structured data. Scrape, crawl and extract with a single API. - mendableai/firecrawl

GitHub

Firecrawl is an API that crawls websites to extract clean, structured content for AI use.

Large Language Models

OpenAI

OpenAI provides access to the advanced **GPT-4.1** model, offering 250,000 free tokens daily

Google AI Studio

Google AI Studio enables experimentation with **Gemini models** and includes a free tier for select versions (flash)

Ollama

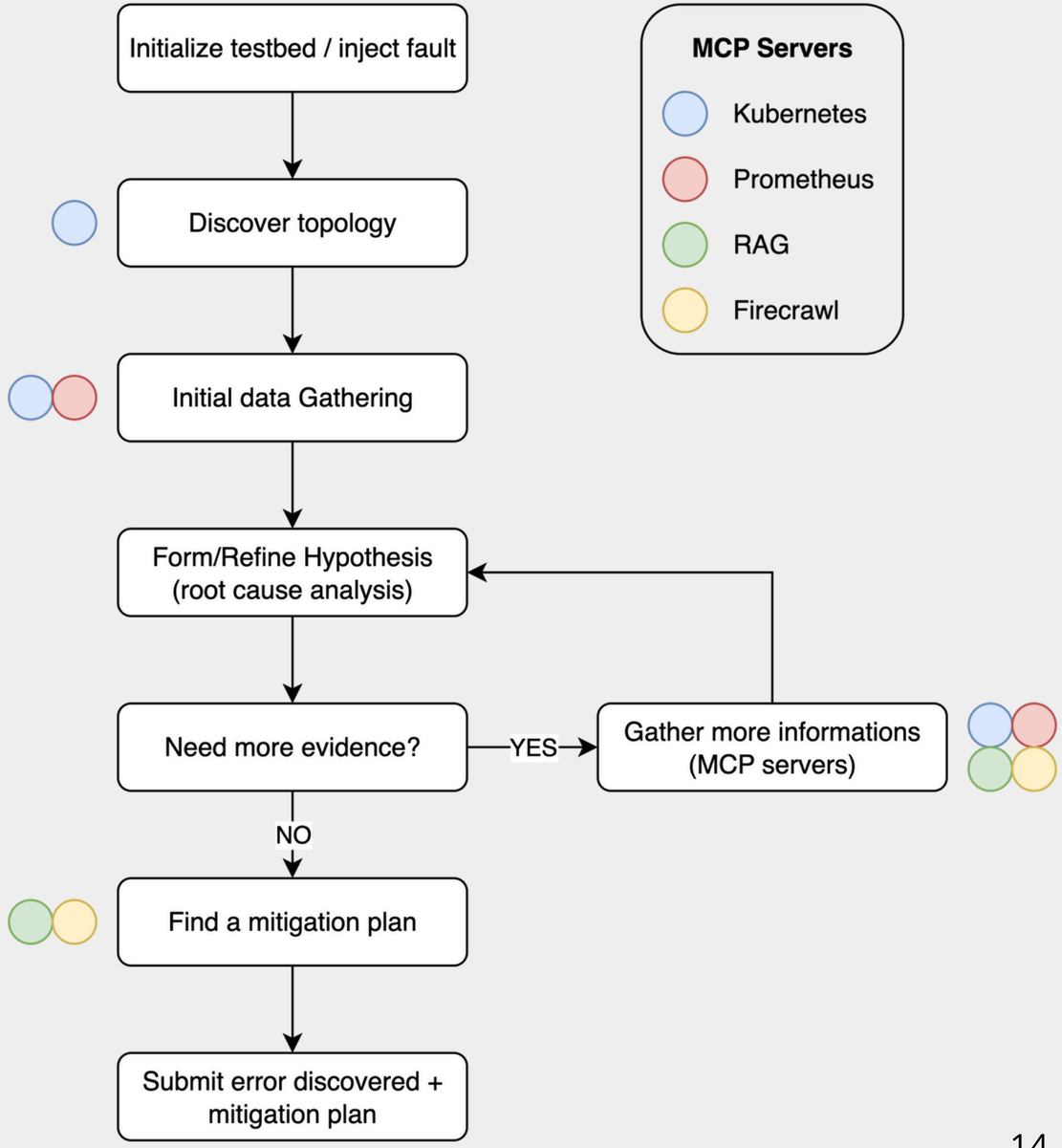
Ollama runs models **locally**, but not viable here due to hardware constraints.

Workflow

How to design the workflow?

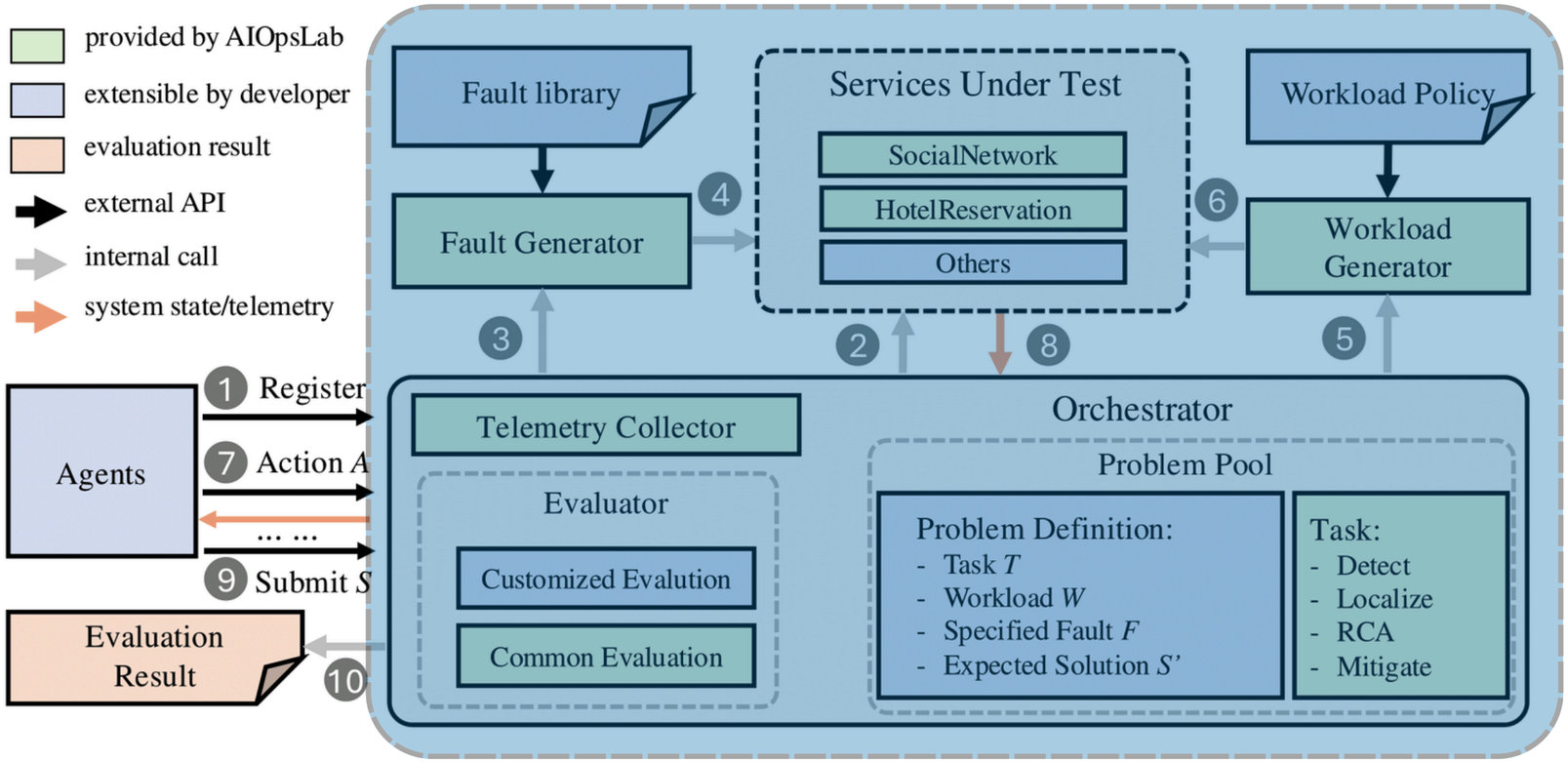
Need to find a trade off between effectviness and semplicity.

Critical point: how to implement the feedback loop without human in the loop



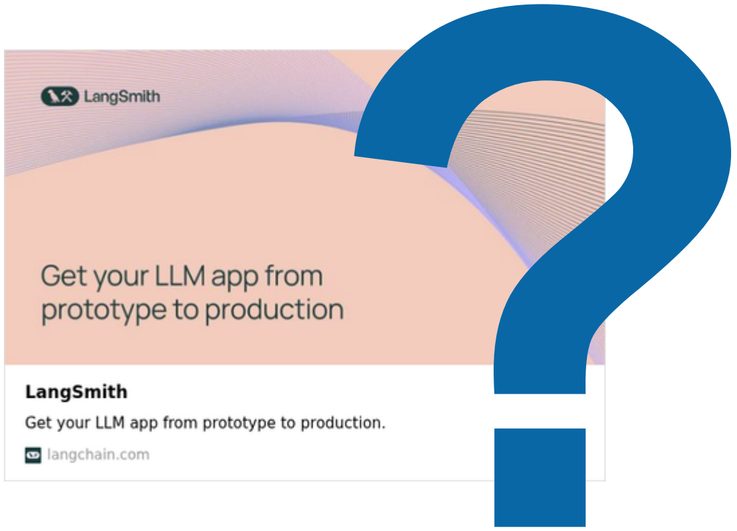
Testbed & Benchmarking

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



Performance evaluation

- Root cause detected (correctness)
- Total tokens consumed to solution
- Total steps executed to reach solution
- Elapsed time to solution
- Estimated cost per simulation



Open questions

- How to scale agent reasoning for large-scale clusters?
- How to optimize retrieval and RAG integration?
- Can the agent autonomously adapt to novel fault types?
- Integration with vector DBs for richer context

Potential thesis topics

- Can small models (SLMs) perform as well as larger ones if the workflow is organized differently?
- Are LLMs sufficient for understanding and detecting faults in microservices architectures?
- Can model distillation be effective for specific tasks in IT operations?

Thank you

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