20TH ANNIVERSARY EDITION
WSO2CON ASIA
PLATFORMLESS MODERNIZATION

Evolution of AI Agents

Malith Jayasinghe
VP of AI
WSO2

Key Innovations Driving AI Adoption

01 Enhanced Intelligence & Reasoning

- LLMs perform at expert level (High scores in SAT)
- GPT-4: 90% in US medical exam
- Enhanced reasoning capabilities (Step-by-step thinking, OpenAI's o1 or Google's Gemini 2.0)

02 **Agentic-AI**

- **AI plans and executes tasks**
- **Handles complex workflows**
- **Chat to action**

03 Multimodality

- AI understands across formats (text, speech video)
- Improved interaction quality (e.g. audio & latency
- Human-like, expressive, and emotional speech
)

04 Hardware Innovation

- Faster, scalable AI (GPUs, TPUs)
- Real-time use cases
- Cloud + Edge

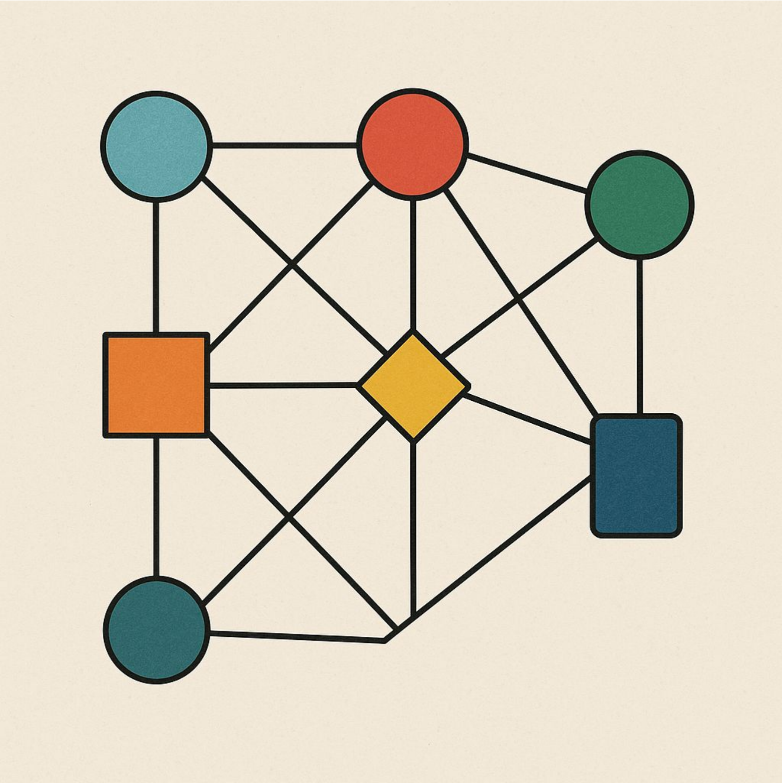
05 Transparency & Explainability

- Progress in explainability
- Transparency scores up (Anthropic's 15 -> 51)
- Traceable decisions (Link to source)
- Bias and error detection (Ongoing monitoring)

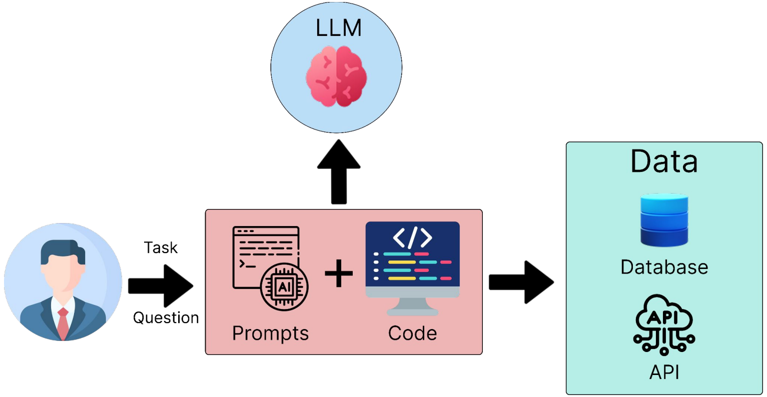
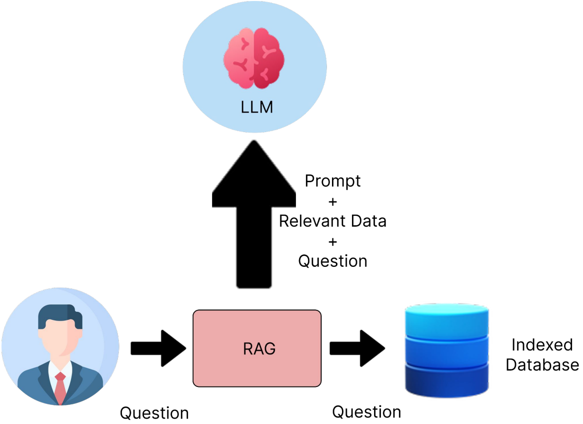
McKinsey & Company. (2025). *Superagency in the workplace*

Building Modern AI Applications

- Building modern AI applications involves
    - Developing the AI component (e.g., Agents)
    - Integrating them with other parts of the application
- Building the AI component itself solves an integration problem
- The focus has shifted from training models to integrating systems
- Success depends on applying the right architectural/ agentic patterns
- Equally important: knowing when—and when not—to use a particular pattern
- Core Patterns: Gen-AI, RAG and **Agents**



Core Patterns

Gen -AI	RAG
 LLM Task Question Prompts + Code Data Database API	 LLM Prompt + Relevant Data + Question RAG Question Question Indexed Database
Gen-AI: Application that calls a Gen-AI API Text Summarization, Sentiment Analysis, Email Drafting	Combines retrieval with generation to ground responses in data Product Documentation Assistant, HR Policy Assistant, Academic Tutor

https://icons8.com/

What AI Agents Bring

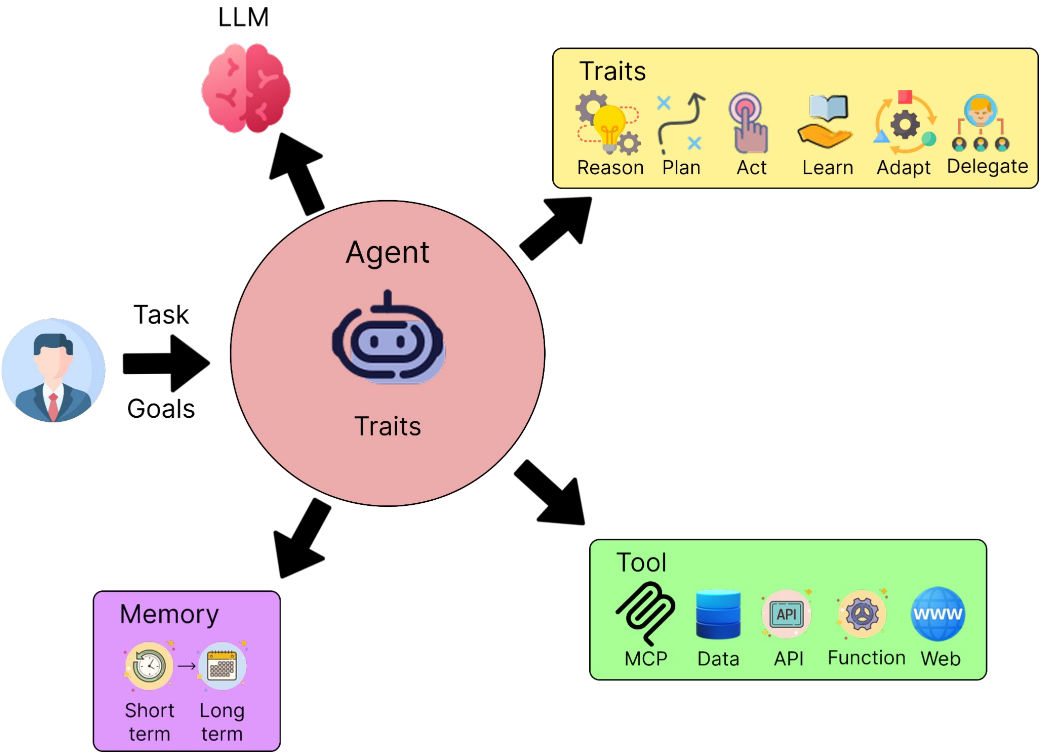
- Gen-AI and RAGs are powerful
  - But reactive
  - Great at synthesis, generation, and natural language understanding
  - Isolated from enterprise systems
- What agents bring
  - Proactive – initiate actions, not just generate content
  - Autonomous – execute tasks with minimal human intervention

Here's a summary...
Meeting scheduled. You're all set!
Reactive –
Only Generates Text
Proactive –
Takes Action Toward a Goal

AI Agent

An agent is an AI system that reasons and performs tasks by interacting with tools like functions, APIs, and databases, with the help of a large language model (LLM)

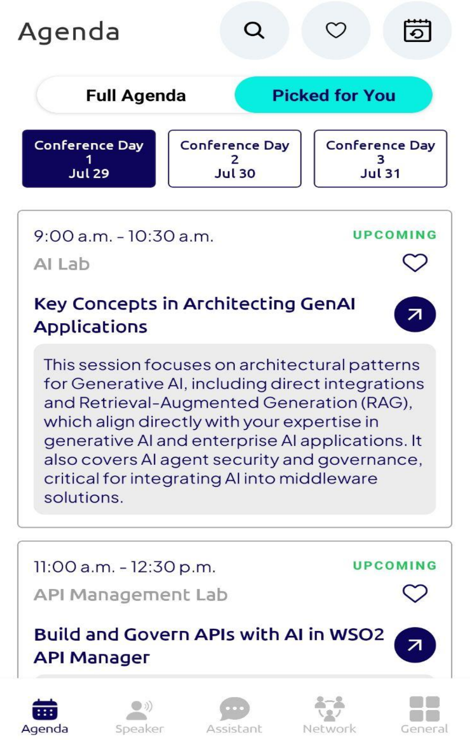
Agent in Action → Next Slides



https://icons8.com/

Building AI Apps: WSO2Con App

- WSO2con mobile App
  - Built using key AI building blocks, integrated with other system components
- Capabilities
  - WSO2Con Assistant
  - Session Advisor
  - Attendee Connections
  - Expert Finder



9:00 a.m. - 10:30 a.m. UPCOMING

AI Lab

Key Concepts in Architecting GenAI Applications

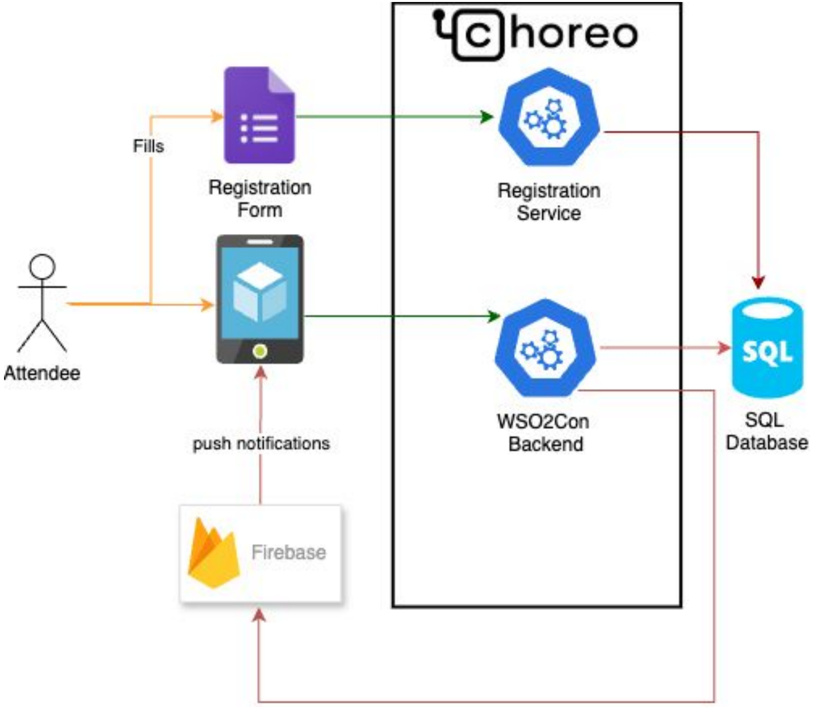
This session focuses on architectural patterns for Generative AI, including direct integrations and Retrieval-Augmented Generation (RAG), which align directly with your expertise in generative AI and enterprise AI applications. It also covers AI agent security and governance, critical for integrating AI into middleware solutions.

11:00 a.m. - 12:30 p.m. UPCOMING

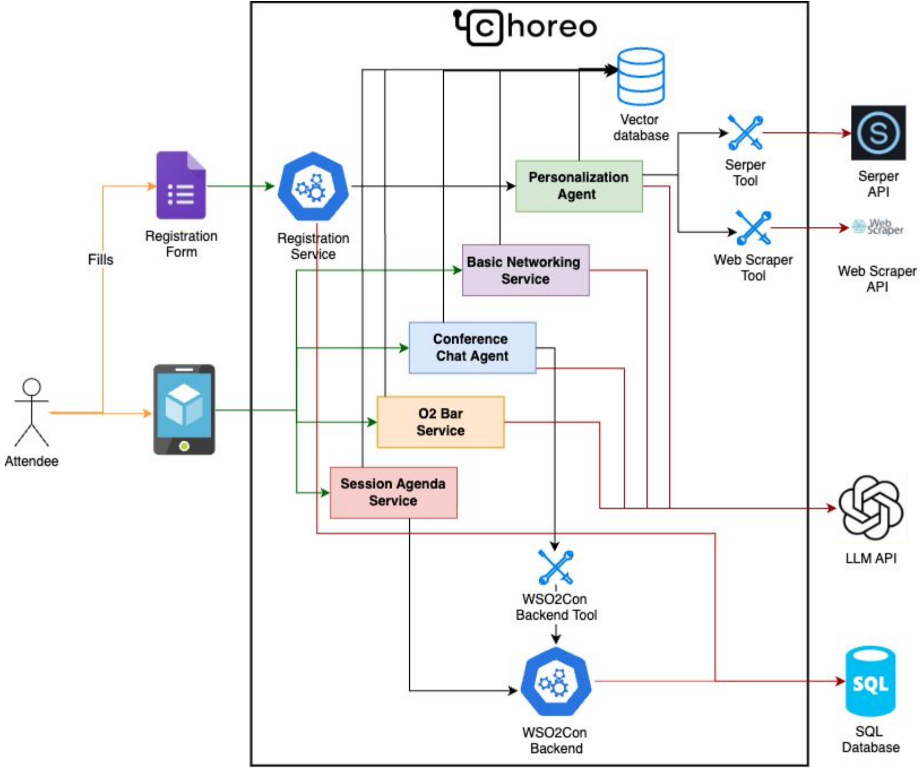
API Management Lab

Build and Govern APIs with AI in WSO2 API Manager

Buildings AI apps: WSO2con app



WSO2 Conference mobile app, 2024

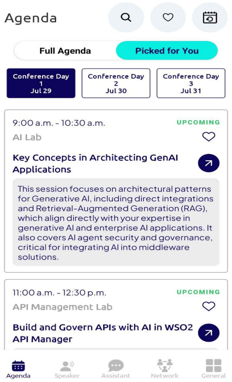


WSO2 Conference mobile app, 2025

Buildings AI apps: AI Agent

- Personalization Agent

  Search the internet and scrape information about the attendees and create a personalized profile for each attendee



9:00 a.m. - 10:30 a.m. UPCOMING

AI Lab

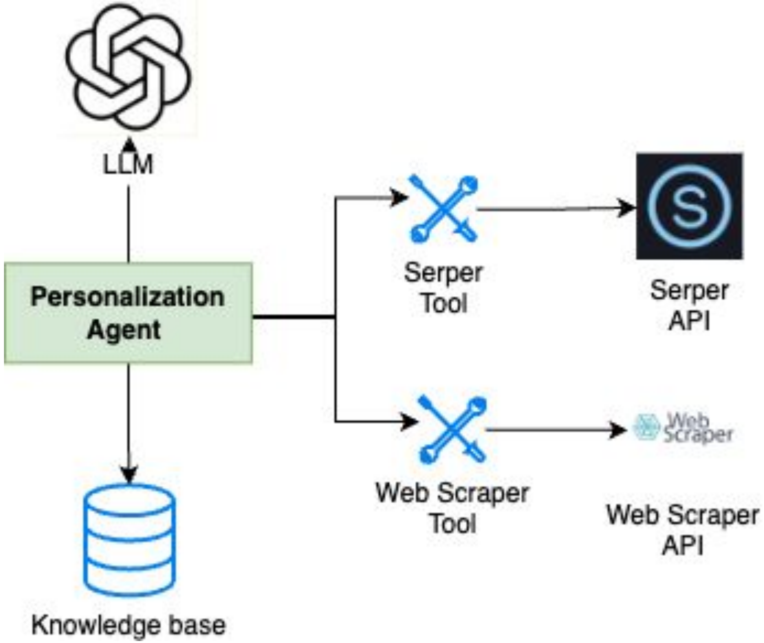
Key Concepts in Architecting GenAI Applications

This session focuses on architectural patterns for Generative AI, including direct integrations and Retrieval-Augmented Generation (RAG), which align directly with your expertise in generative AI and enterprise AI applications. It also covers AI agent security and governance, critical for integrating AI into middleware solutions.

11:00 a.m. - 12:30 p.m. UPCOMING

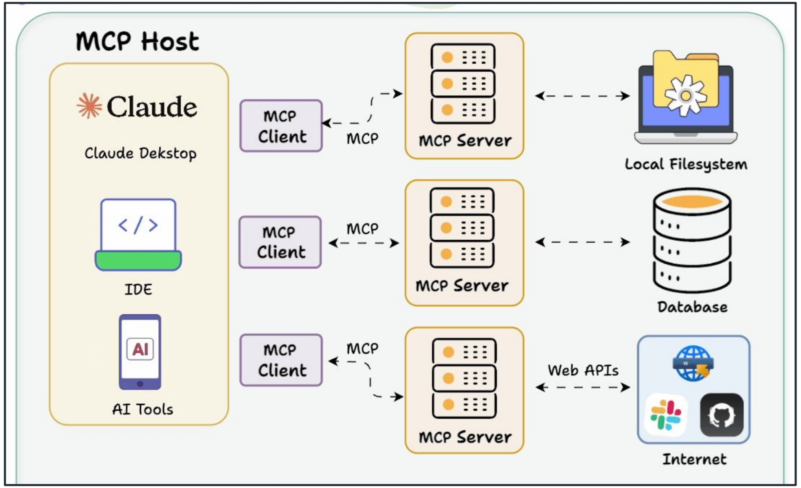
API Management Lab

Build and Govern APIs with AI in WSO2 API Manager



Model Context Protocol (MCP)

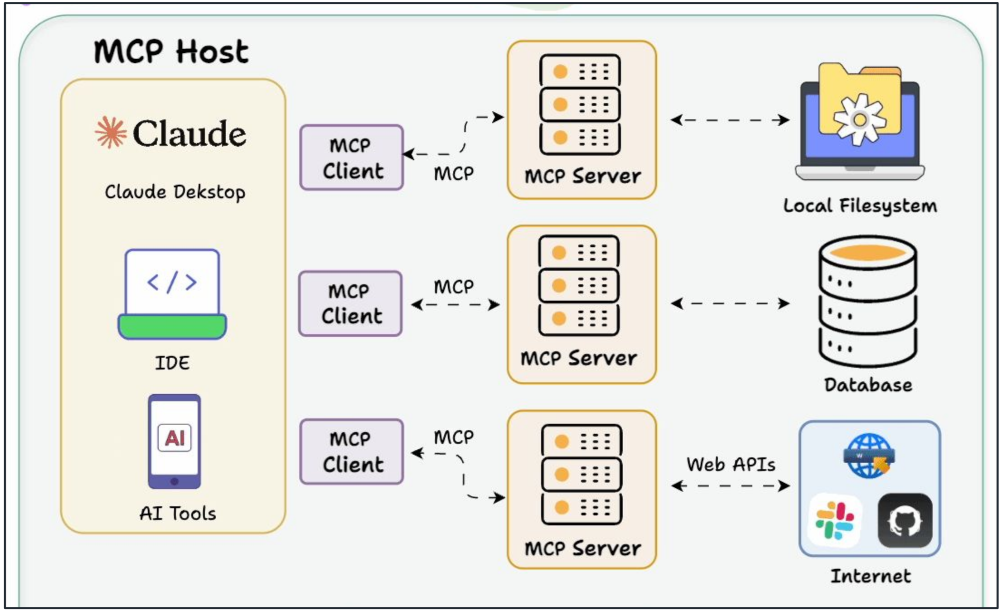
- MCP is a standardized way for AI Systems (e.g. agents) to access and interact with external data sources, tools, and services
- “MCP is an open protocol that standardizes how applications provide context to large language models”(source: modelcontextprotocol.io)
    - **Tools**: Executable functions for performing actions (e.g., API calls, DB queries)
    - **Resources**: Contextual data sources (e.g., files, databases, APIs)
    - **Prompts**: Reusable templates to structure LLM interactions (e.g., system messages, examples)



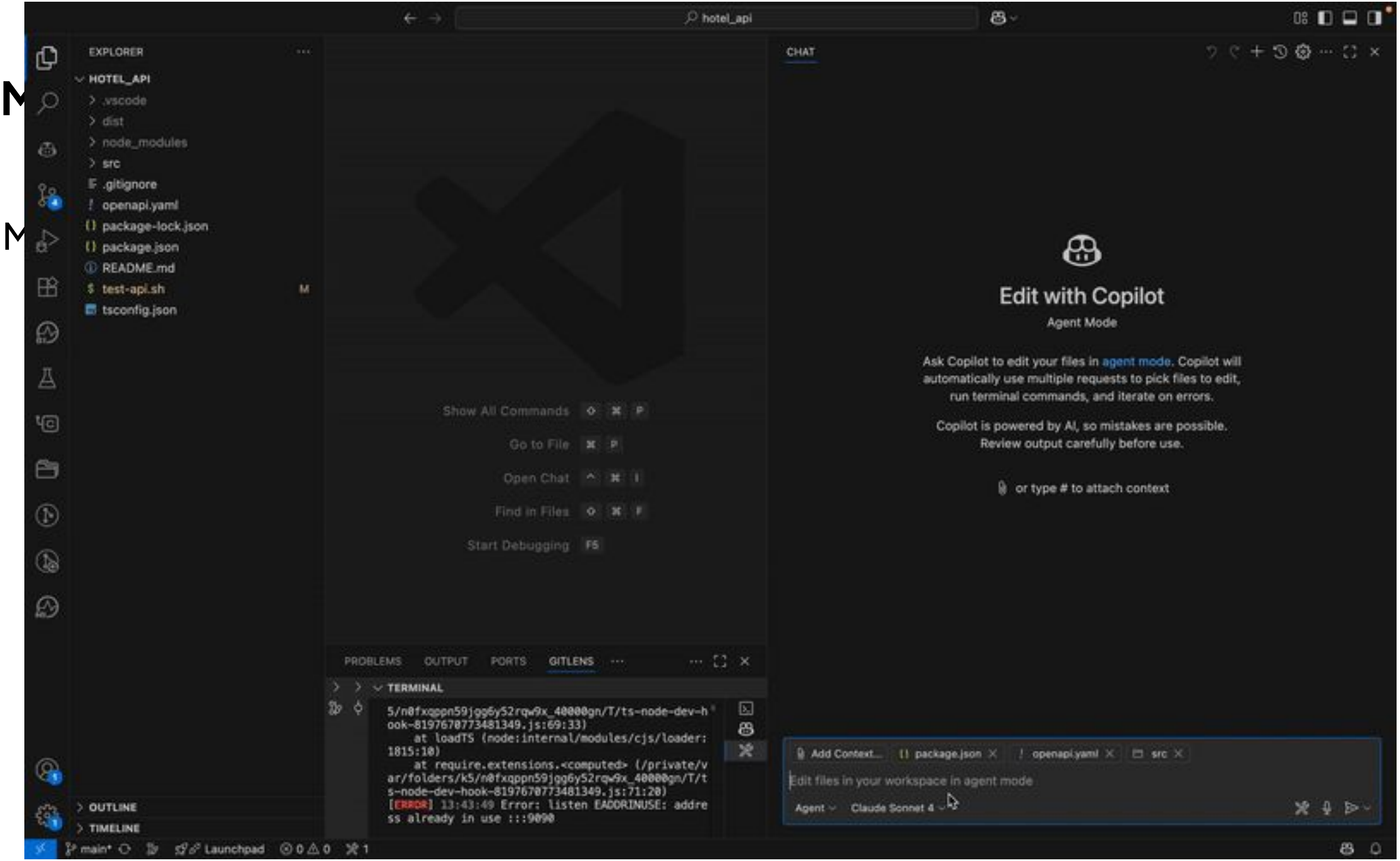
https://medium.com/design-bootcamp/model-context-protocol-mcp-a-guide-to-unlock-seamless-ai-experiences-30bc27fa47c6

What Problem MCP Solves?

- For AI App Developers
  - Focus only on implementing MCP client-side
  - No need to manually connect each API/tool
  - Typically add an MCP client SDK
- If you're exposing functionality
  - Focus only on implementing MCP server
  - No need to worry about how apps use your APIs



https://www.dailydoseofds.com/p/hands-on-building-an-mcp-server/



Edit with Copilot

Agent Mode

Ask Copilot to edit your files in agent mode. Copilot will automatically use multiple requests to pick files to edit, run terminal commands, and iterate on errors.

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

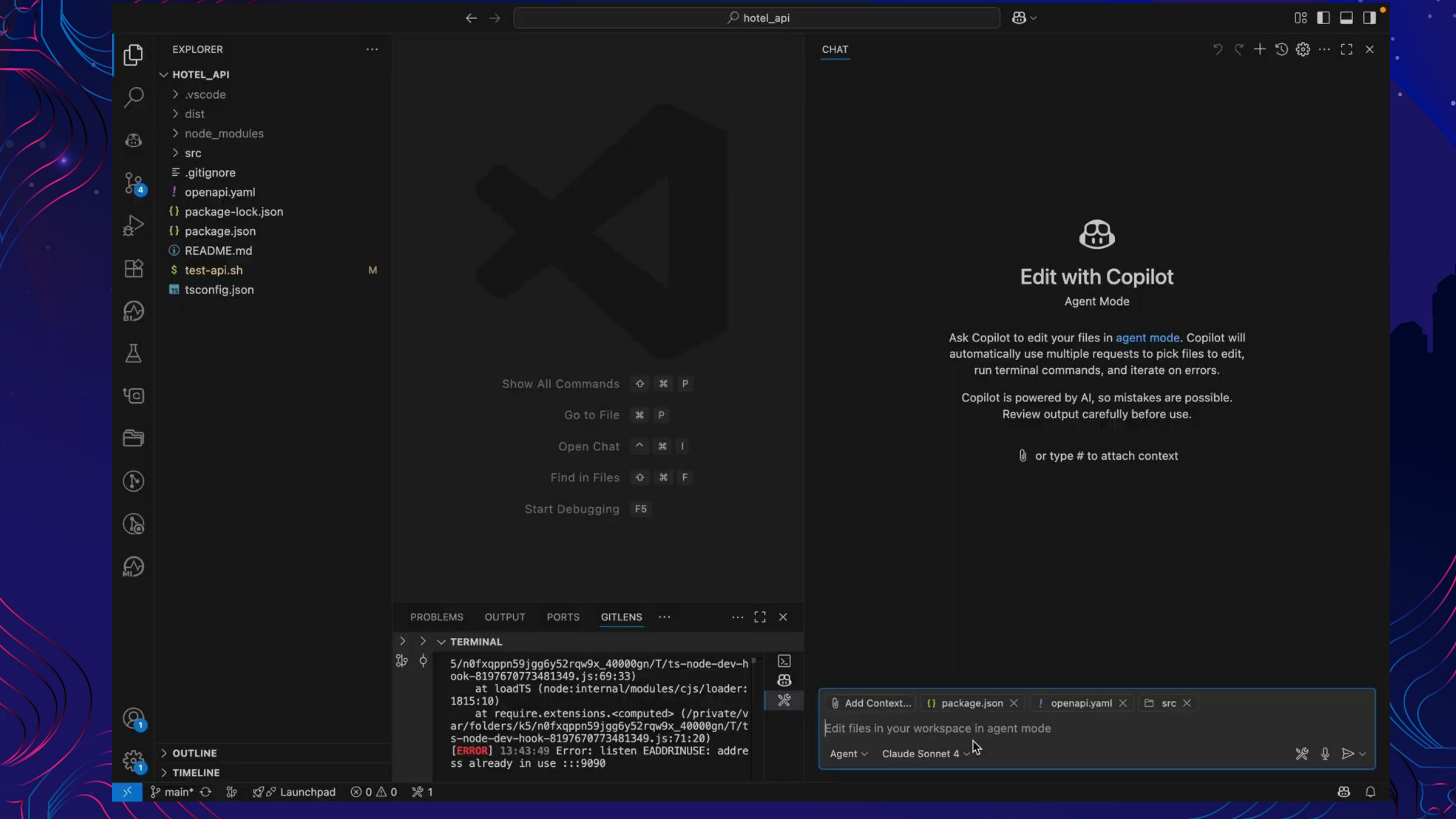
or type # to attach context

Edit files in your workspace in agent mode

Agent

Claude Sonnet 4

```
5/n0fxqppn59jgg6y52rqw9x_40000gn/T/ts-node-dev-h
ook-8197670773481349.js:69:33)
    at loadTS (node:internal/modules/cjs/loader:
1815:10)
    at require.extensions.<computed> (/private/v
ar/folders/k5/n0fxqppn59jgg6y52rqw9x_40000gn/T/t
s-node-dev-hook-8197670773481349.js:71:20)
[ERROR] 13:43:49 Error: listen EADDRINUSE: addre
ss already in use :::9090
```



Edit with Copilot

Agent Mode

Ask Copilot to edit your files in agent mode. Copilot will automatically use multiple requests to pick files to edit, run terminal commands, and iterate on errors.

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

or type # to attach context

Edit files in your workspace in agent mode

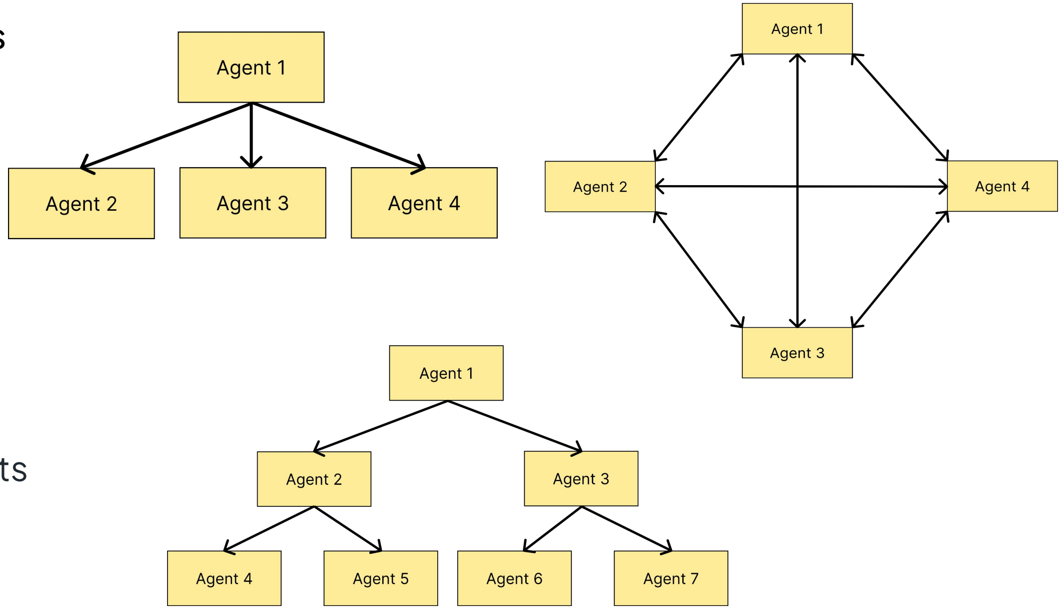
```
5/n0fxqppn59jgg6y52rqw9x_40000gn/T/ts-node-dev-h
ook-8197670773481349.js:69:33)
    at loadTS (node:internal/modules/cjs/loader:
1815:10)
    at require.extensions.<computed> (/private/v
ar/folders/k5/n0fxqppn59jgg6y52rqw9x_40000gn/T/t
s-node-dev-hook-8197670773481349.js:71:20)
[ERROR] 13:43:49 Error: listen EADDRINUSE: addre
ss already in use :::9090
```

Multi-Agents Systems: When One Isn't Enough

- Problems with a Single Agent as the System Scales:
  - **Too many tools:** The agent may struggle to choose the right tool, leading to poor decisions
  - **Diverse skill requirements:** A single agent cannot effectively handle all roles (computation, research, etc)
- Solution:
  - Break into smaller, specialized agents → form a multi-agent system
- Benefits of a multi-Agent System:
  - Specialisation: Agents focus on specific tasks or skills, improving efficiency – > higher accuracy
  - Modularity: Develop, improve, scale independently

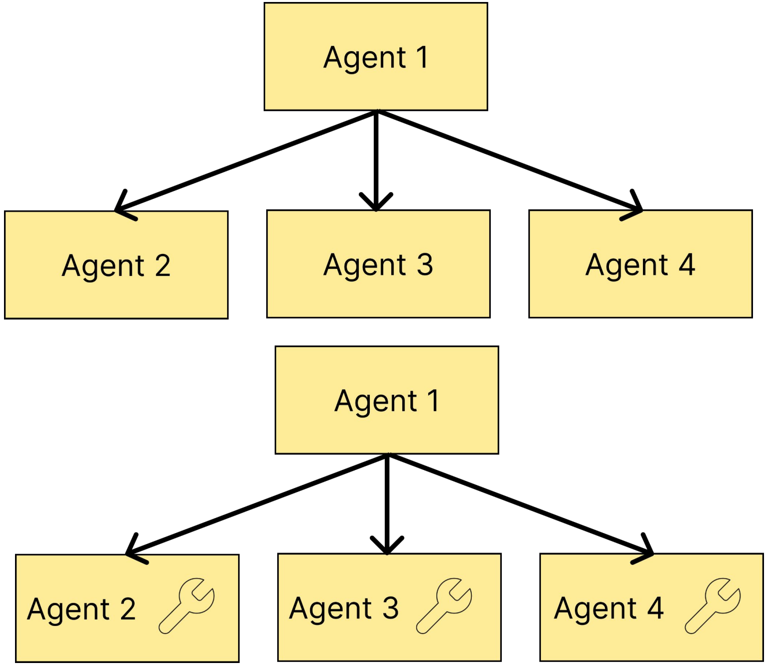
Multi-Agent Patterns

- Multi-agent patterns are emerging as common ways to organise and coordinate agents
- Different from (single) agent traits (e.g., self-reflection, planning)
- Patterns define:
  - Organization of agents (Hierarchical)
  - Communication between agents (peer-to-peer, coordinate tasks)
  - Control flow in the system
- Examples: Supervisor, Hierarchical, Network, Custom

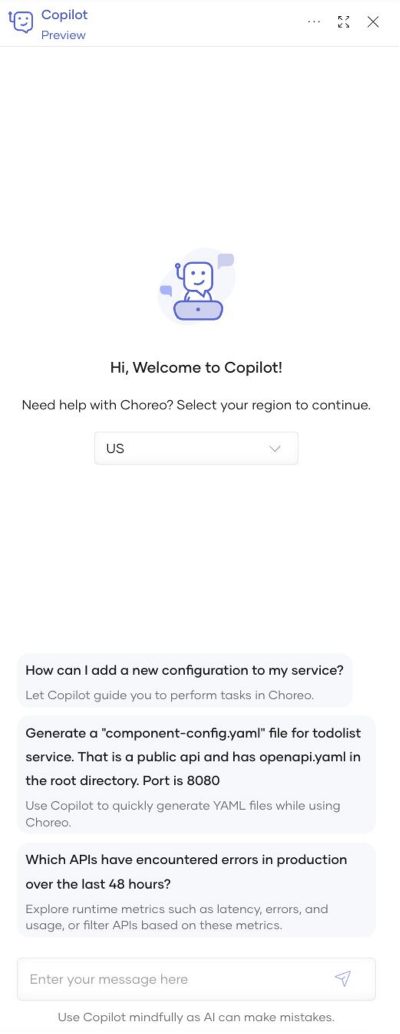


Muti-Agents and patterns

- Supervisor Pattern
- A central supervisor agent manages the flow
- Decides which agent to invoke next
- Can act as a reactive agent or a simple router
- Two Handoff Modes:
  - Agent-to-Agent Handoff
    - Supervisor passes full control
    - Entire context and history transferred
  - Tool-Style Handoff (Handoff as a Tool)
    - Supervisor retains control
    - Only task-specific input passed



https://langchain-ai.github.io/langgraph/tutorials/multi_agent/agent_supervisor/



Hi, Welcome to Copilot!

Need help with Choreo? Select your region to continue.

US

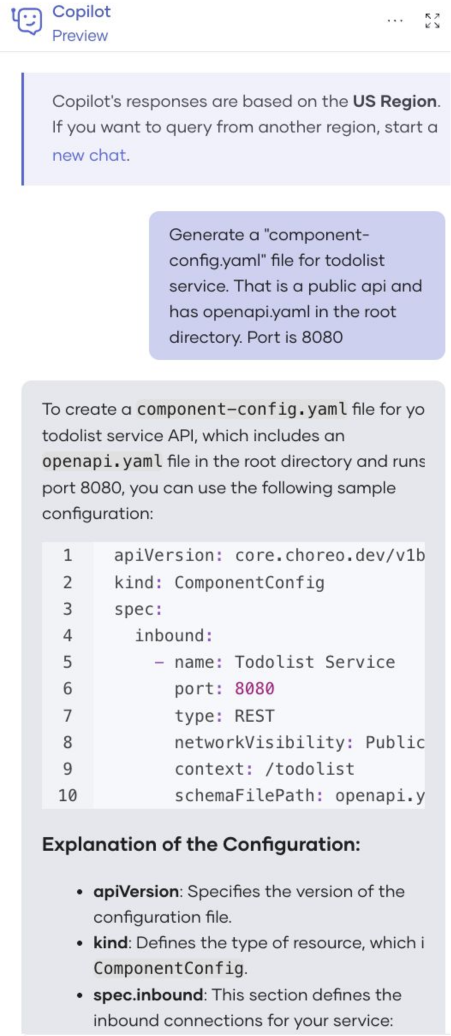
How can I add a new configuration to my service?
Let Copilot guide you to perform tasks in Choreo.

Generate a "component-config.yaml" file for todolist service. That is a public api and has openapi.yaml in the root directory. Port is 8080
Use Copilot to quickly generate YAML files while using Choreo.

Which APIs have encountered errors in production over the last 48 hours?
Explore runtime metrics such as latency, errors, and usage, or filter APIs based on these metrics.

Enter your message here

Use Copilot mindfully as AI can make mistakes.



Copilot's responses are based on the **US Region**. If you want to query from another region, start a new chat.

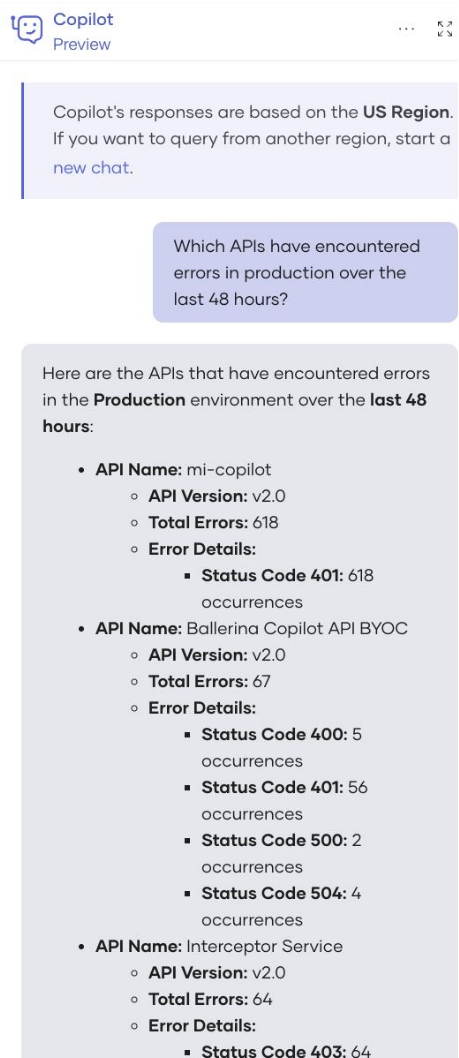
Generate a "component-config.yaml" file for todolist service. That is a public api and has openapi.yaml in the root directory. Port is 8080

To create a `component-config.yaml` file for yo todolist service API, which includes an `openapi.yaml` file in the root directory and runs port 8080, you can use the following sample configuration:

```
apiVersion: core.choreo.dev/v1b
kind: ComponentConfig
spec:
  inbound:
    - name: Todolist Service
      port: 8080
      type: REST
      networkVisibility: Public
      context: /todolist
      schemaFilePath: openapi.y
```

Explanation of the Configuration:

- **apiVersion**: Specifies the version of the configuration file.
- **kind**: Defines the type of resource, which i `ComponentConfig`.
- **spec.inbound**: This section defines the inbound connections for your service:



Copilot's responses are based on the **US Region**. If you want to query from another region, start a new chat.

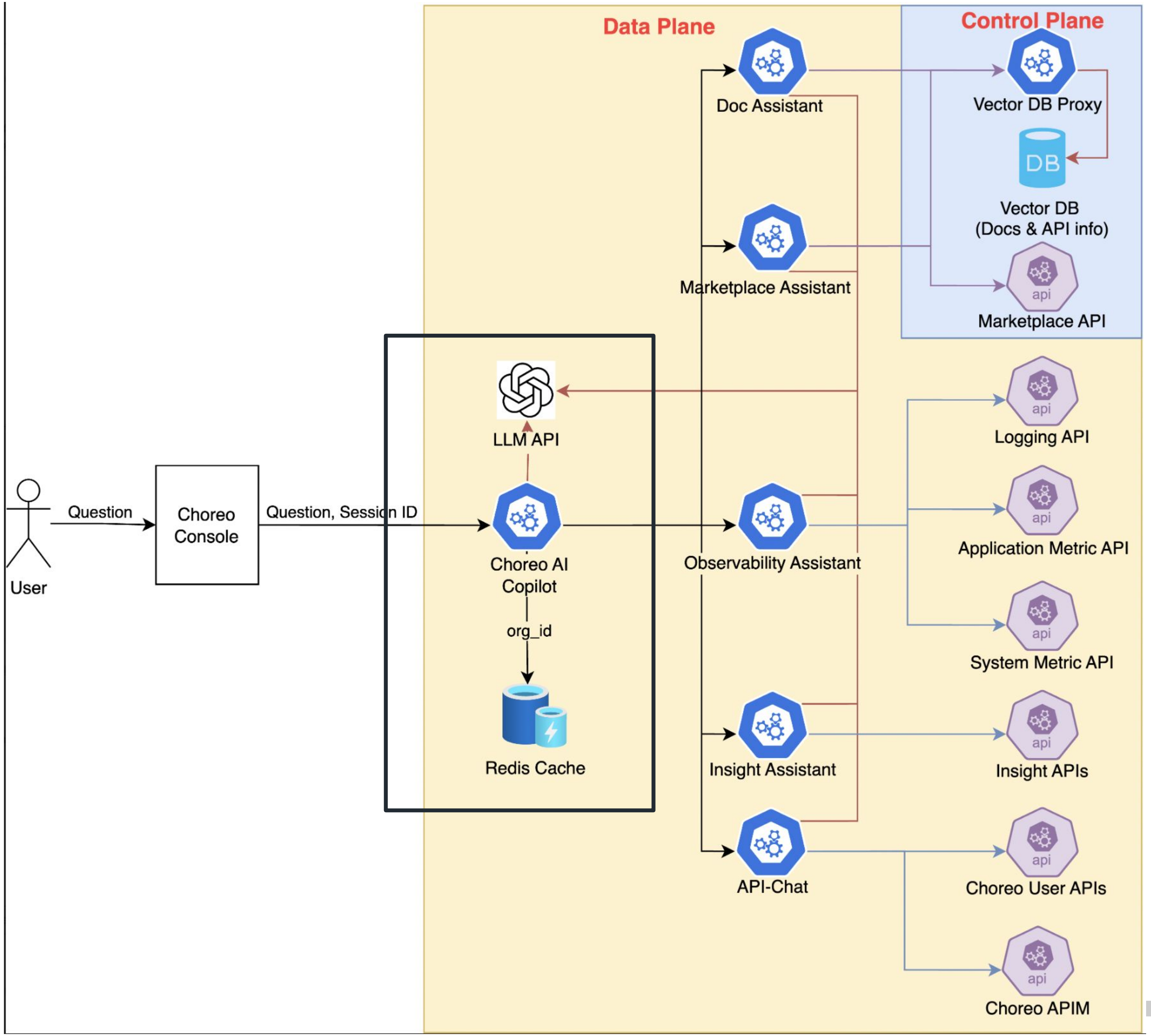
Which APIs have encountered errors in production over the last 48 hours?

Here are the APIs that have encountered errors in the **Production** environment over the **last 48 hours**:

- **API Name:** mi-copilot
  - **API Version:** v2.0
  - **Total Errors:** 618
  - **Error Details:**
    - **Status Code 401:** 618 occurrences
- **API Name:** Ballerina Copilot API BYOC
  - **API Version:** v2.0
  - **Total Errors:** 67
  - **Error Details:**
    - **Status Code 400:** 5 occurrences
    - **Status Code 401:** 56 occurrences
    - **Status Code 500:** 2 occurrences
    - **Status Code 504:** 4 occurrences
- **API Name:** Interceptor Service
  - **API Version:** v2.0
  - **Total Errors:** 64
  - **Error Details:**
    - **Status Code 403:** 64

Supervisor Pattern

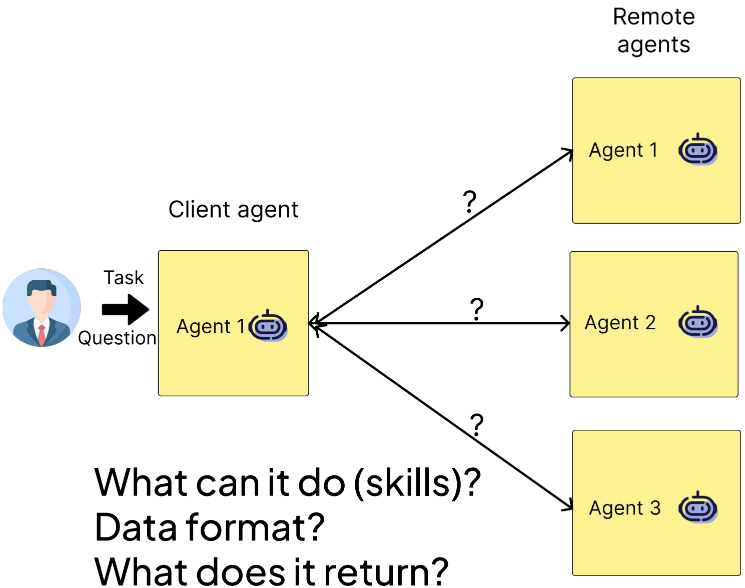
- Pattern: Supervisor
- Example Use: Choreo Copilot
    - Supervisor Agent:
    - Choreo Copilot (React Agent)
    - Supervisor agent manages the flow
    - Sub-agents function as tools
    - Control passed via hand-off as tool mechanism



Agent to Agent Communication

- MCP provides a standard way for AI systems to communicate with external systems (databases, API, etc)
- What is agent to agent communication?
  - Agent to agent refers to communication between AI agents, enabling them to collaborate, delegate, and coordinate autonomously
  - Agent to agent protocols **standardizes the agent to agent communication** (note: MCP standardizes agents to tool communication)
- Agent to Agent Protocols
  - A2A (Agent2Agent) by Google
  - ACP (Agent Communication Protocol) by IBM
  - Others (e.g. ANP by)

Without A2A Protocol



Remote agents

Client agent

Task

Question

Agent 1

Agent 1

Agent 2

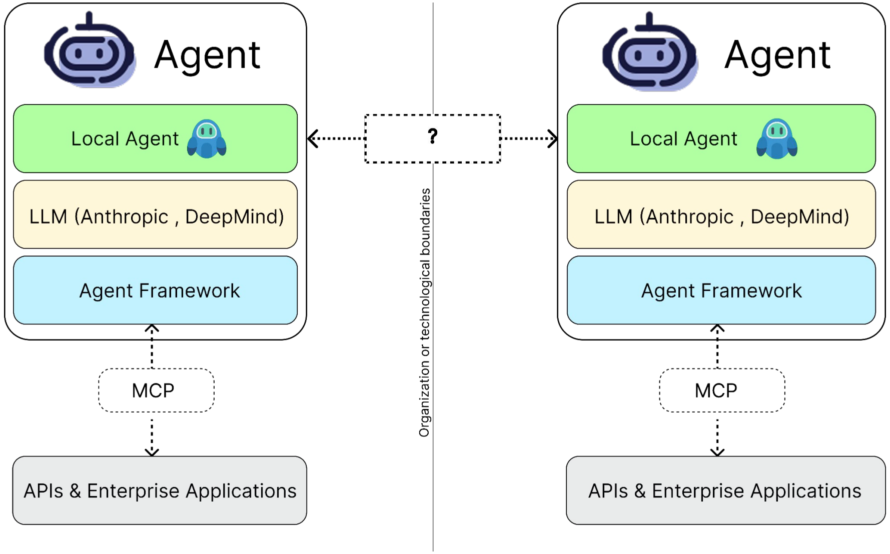
Agent 3

?

?

?

What can it do (skills)?
Data format?
What does it return?
Data type (text, ...)?



Agent

Local Agent

LLM (Anthropic , DeepMind)

Agent Framework

MCP

APIs & Enterprise Applications

?

Organization or technological boundaries

Agent

Local Agent

LLM (Anthropic , DeepMind)

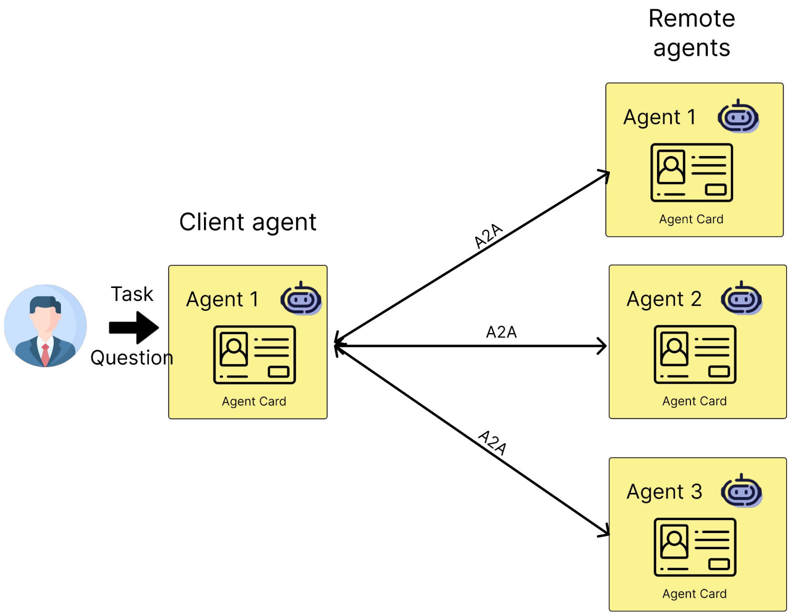
Agent Framework

MCP

APIs & Enterprise Applications

https://a2a-protocol.org/

Agent2Agent (A2A) Protocol



Remote agents

Client agent

Task

Question

Agent 1

Agent Card

A2A

A2A

A2A

Agent 1

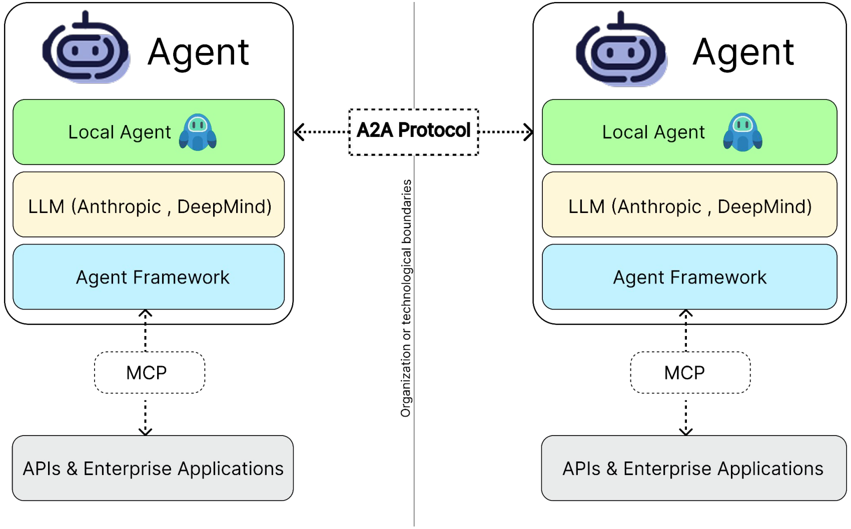
Agent Card

Agent 2

Agent Card

Agent 3

Agent Card



Agent

Local Agent

LLM (Anthropic , DeepMind)

Agent Framework

MCP

APIs & Enterprise Applications

A2A Protocol

Organization or technological boundaries

Agent

Local Agent

LLM (Anthropic , DeepMind)

Agent Framework

MCP

APIs & Enterprise Applications

https://a2a-protocol.org/

AgentToAgent (A2A) Protocol

- **Standardized Communication**: JSON-RPC 2.0 over HTTP(S).
- **Agent Discovery:** Via "Agent Cards" detailing capabilities and connection info
- **Flexible Interaction**: Supports synchronous request/response, streaming (SSE), and asynchronous push notifications
- **Rich Data Exchange:** Handles text, files, and structured JSON data
- **Enterprise-Ready**: Designed with security, authentication, and observability in mind

Agent Card

Name
Description
URL
Version
Default input / output mode
Skills: [
- AgentSkill {
  - id
  - name
  - Description
  - Tags
  - Examples
- }]

https://github.com/a2aproject/A2A

AgentToAgent (A2A) Protocol

Agent Card

Name
Description
URL
Version
Default input / output mode
Skills: [

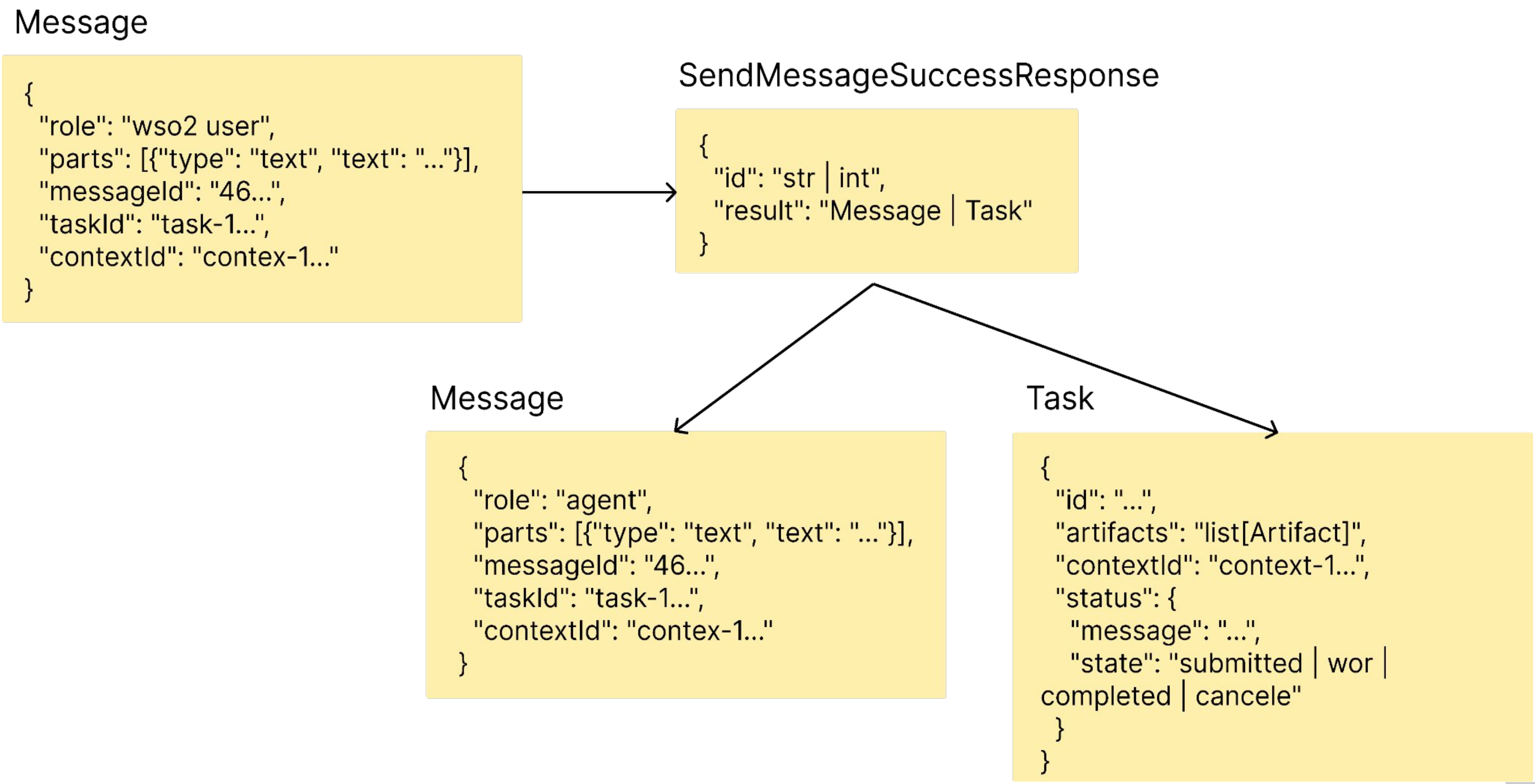
- AgentSkill {
  - id
  - name
  - Description
  - Tags
  - Examples
- }]

https://github.com/a2aproject/A2A

WSO2 Integration Assistant Agent Card

```
{
  "name": "WSO2 Integration Assistant",
  "description": "Your AI assistant for creating integrations, managing APIs, and exploring WSO2 products.",
  "url": "http://localhost:9443/agent",
  "version": "1.0.0",
  "default_input_output_mode": [
    "text/plain"
  ],
  "skills": [
    {
      "id": "create_api_proxy",
      "name": "API Proxy Creation",
      "description": "Create a new API proxy in WSO2 API Manager from a backend service endpoint.",
      "tags": [
        "api management",
        "wso2",
        "proxy",
        "api"
      ],
      "examples": [
        "Create an API for the customer service at https://api.mycompany.com/customers",
        "Proxy the service endpoint http://inventory.internal/api/v2"
      ]
    }
  ]
}
```

AgentToAgent (A2A) Protocol



Message

```
{
  "role": "wso2 user",
  "parts": [{"type": "text", "text": "..."}],
  "messageId": "46...",
  "taskId": "task-1...",
  "contextId": "contex-1..."
}
```

SendMessageSuccessResponse

```
{
  "id": "str | int",
  "result": "Message | Task"
}
```

Message

```
{
  "role": "agent",
  "parts": [{"type": "text", "text": "..."}],
  "messageId": "46...",
  "taskId": "task-1...",
  "contextId": "contex-1..."
}
```

Task

```
{
  "id": "...",
  "artifacts": "list[Artifact]",
  "contextId": "context-1...",
  "status": {
    "message": "...",
    "state": "submitted | wor |
completed | cancele"
  }
}
```

Summary

- We started with Gen-AI — smart, but static
- Then came RAG — grounding answers in real data
- Soon, we needed agents — to act, not just answer
- To connect AI apps to tools at scale and efficiently, we introduced MCP
- As tasks grew, we built multi-agent systems — specialised agents working together
- Multi-agent patterns have since evolved — e.g., Supervisor and Network
- Now, agents talk directly — A2A (Agent-to-Agent communication)
- And this evolution will continue, as agents become more capable and autonomous

Summary

- Not discussed, but important
- As agents become autonomous governance is key
  - Evaluation: Are agents doing the right thing?
  - Guardrails: What are agents allowed to do? (Covered in APIM Track and AI Lab)
  - Securing Agents (IAM Track and AI Lab)
    - How to control what agents can do and cannot do
    - Introduce concept of agent identity (agents acting on behalf of humans)
- Agents and SLMs
  - How to fine-tune? When is fine-tuning needed?
  - Domain adaptation

Question Time!

Thank you!

20TH ANNIVERSARY EDITION
WSO2CON ASIA
PLATFORMLESS MODERNIZATION