Week 6 → MCP (Model context Protocol)

→ MCP is a standardized framework for AI agents to interact with external systems and data, enabling them to operate with enhanced memory reasoning, and multi-task execution.

What MCP is not →

1. A framework for building agent
2. A fundamental change to how agents works
3. A way to code agent

What MCP is actually →

1. A protocol - a standard
2. A simple way to integrate tools, resources, promps
3. "A USB-C port for AI application"

Reason not be excited →

1. It's just a standard, it not a tool themselves.
2. Langchains already has big Tool ecosystem
3. You can already make any function into any tool

Reason to be excited →

1. Makes it easy to intergate tools.
2. It's taking off! Exploding ecosystem.
3. HTML was a standard too.

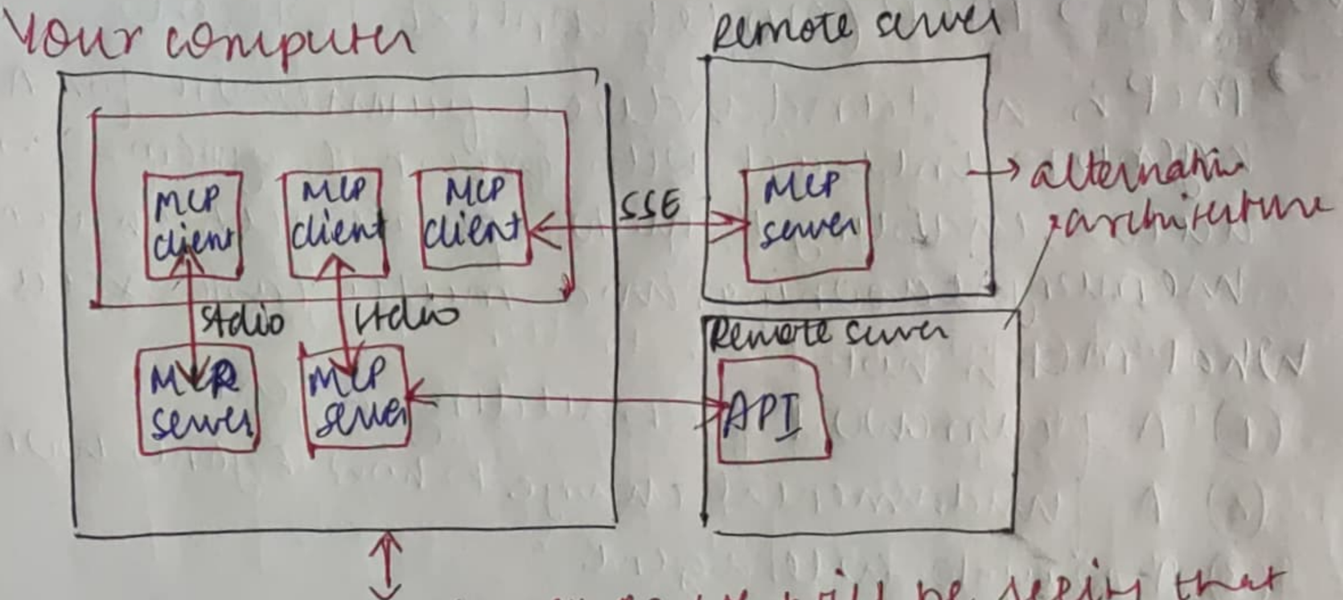
MCP core concept:→

The three components →

1. Host is an LLM app like claude or our Agent architecture.
2. MCP client lives inside Host and connects 1:1 to MCP server.
3. MCP server provides tools, context and promps'

Example →

Fetch → MCP server that searches the web via a headless browser.



→ In most of example or cases we will be seeing that Mcp server are running outside the host and your connecting it, it still running on your machi

→ Mcp servers can takes advantages of functionality over internet.

Two different transport machenism→

① stdio → (standard input output) → The AI runs and mcp server as a local program, communicates via standard input/output.

② SSE → (Server-Sent-Events (SSE)) → The AI connects to a remote server via HTTP and exchanges message asynchronously.

Commands →

python based

① fetch-params = {"command": "uvx", "args": {"mcp-server-fetch"}}

This also uses playwright

This commands is going to create an Mcp client running within OpenAI agents SDK.

→ for running mcp server by using JS node npm

playwright-param = {"command": "npx", "args": {"@playwright/mcp@latest"}}

Distributed agent Runtime:

→ A distributed agent runtime handles life cycle and communication and comminication lifecycle and communication accross process boundaries, consisting of:

ⓐ host service → A host service connected to worker runtimes, handling message delivery and sessions for direct message

ⓑ worker runtime → A worker runtime advertises agent to the host service and handles executing their code.

→ Grpc is an host to run and send message on local host according to the port no specified.

Which framework to select?

→ It's not the most important question and it doesn't really matter!

→ Pick the framework that suits your styles and the skill of your team.

~~→ my go-to OpenAI~~

→ Other framework → Google ADK, Huggingface smolAgent and Pydantic AI - but they will seem familiar.

What matters →

① Start with the problem, not the solution.

② Have a metric to evaluate success

③ Favour workflow over autonomy initially. (do try to do the code by handcoding)

④ Work bottom up, not top down (do not start with a big problems, instead start with a small thing).

⑤ Start simple, then add [small piece → big problem]

⑥ Start with large frontier model, then reduce

⑦ Think context rather than memory.

(8) [try giving LLM the right context to be able to answer its question].

(8) most problems are solved with better prompts.

(9) Look at the traces.

(10) Be a scientist, no shortcut to R&D.