MCP 101

Model Context Protocol (MCP)
A communication protocol for apps, agents, tools, and data

Austin Python Community Meetup
Daniel Takabayashi (Taka) - 08/19/2025

Motivation?

"Models are only as good as the context provided to them" Anthropic

MCP = **M**odel **C**ontext **P**rotocol

What is it (1 of 2)

MCP is an open protocol that enables seamless integration between **AI apps & agents** and **tools & data sources.**

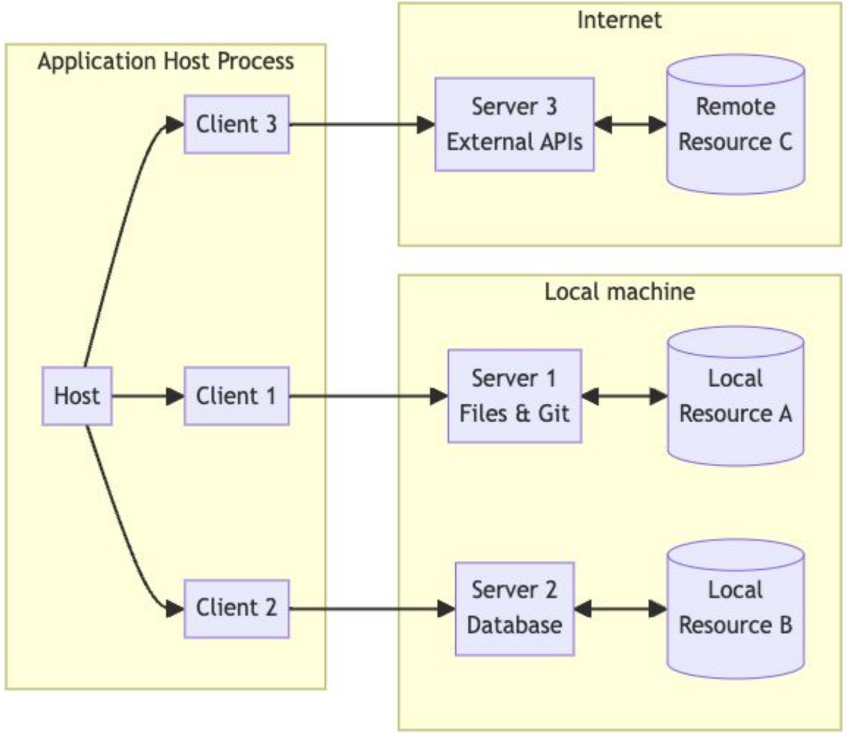
An standard for:

Share Contextual Information with Language models	Expose capabilities to AI Systems	Build Composable integrations and workflows

What is it (2 of 2)

- A communication protocol between apps, agents, tools, and data;
- A solution to make available (for AI agents) functionalities (Ex. Rest APIs) of internal systems;
- An emerging integration (and open) standard for multi-agent systems;
- A solution to help AI systems to maintain context as they move between different tools and datasets (replacement for current fragmented integrations - connectors for each datasource);
- Think of MCP like a USB-C port for AI applications…
- MCP (Anthropic Implementation) = Specification (with Schema) + SDK (python, typescript, java, kotlin) + Servers

Architecture (1 of 3)

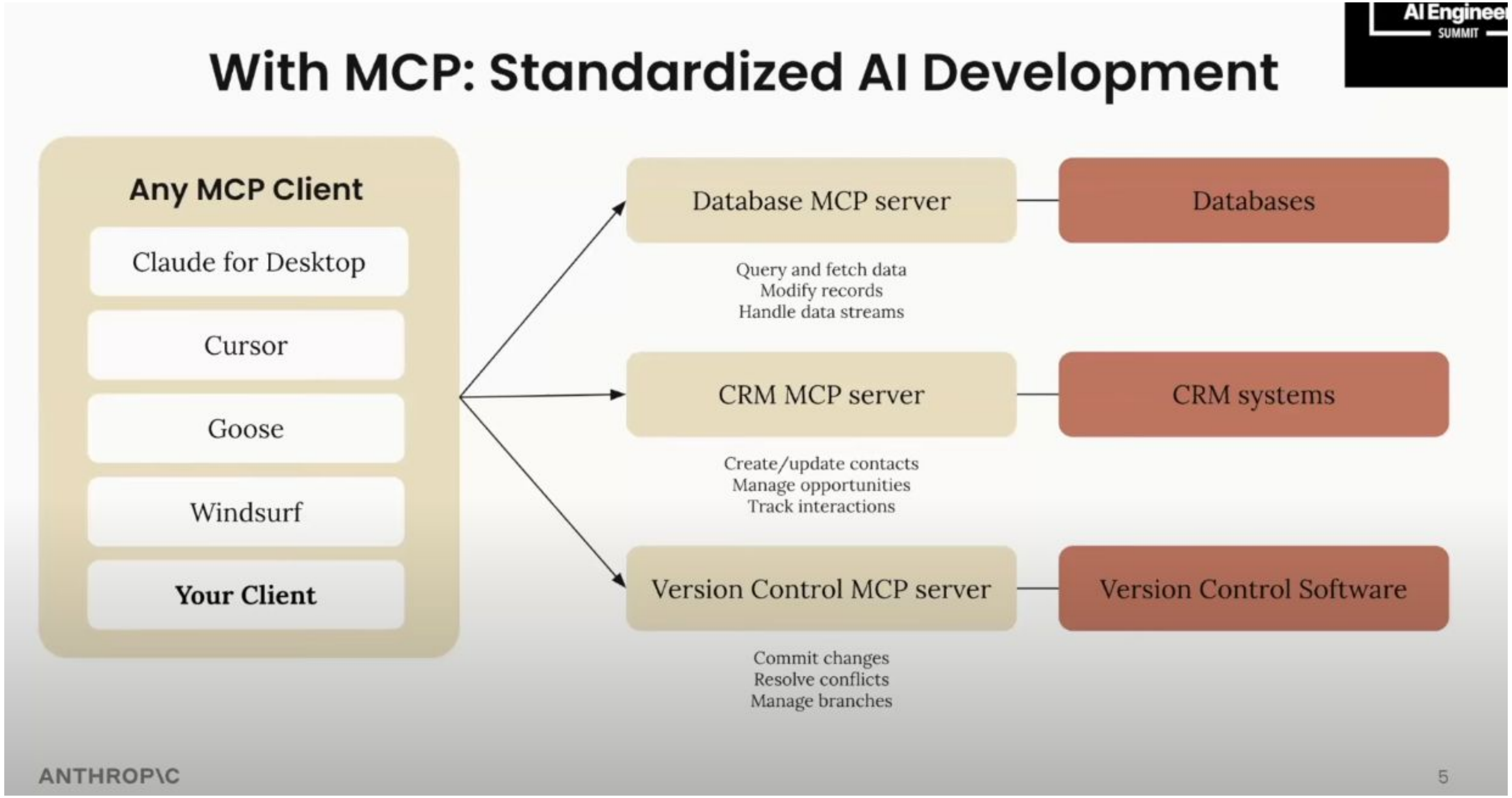


Host:: The host process acts as the container and coordinator:

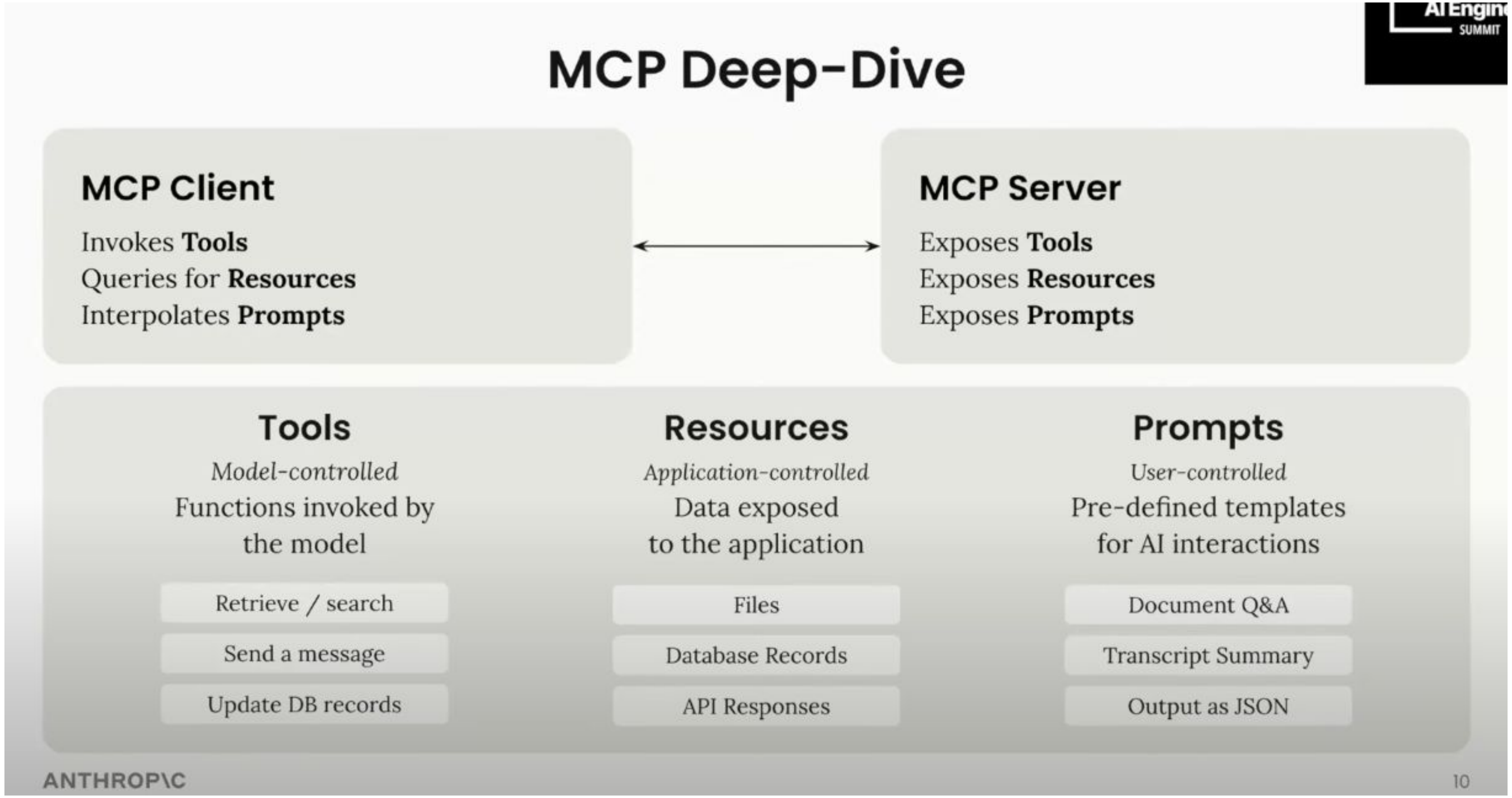
Clients:: Each client is created by the host and maintains an isolated server connection:

Servers:: Servers provide specialized context and capabilities:

Architecture (2 of 3)



Architecture (3 of 3)



MCP Deep-Dive

MCP Client

Invokes **Tools**
Queries for **Resources**
Interpolates **Prompts**

MCP Server

Exposes **Tools**
Exposes **Resources**
Exposes **Prompts**

Tools

Model-controlled

Functions invoked by the model

- Retrieve / search
- Send a message
- Update DB records

Resources

Application-controlled

Data exposed to the application

- Files
- Database Records
- API Responses

Prompts

User-controlled

Pre-defined templates for AI interactions

- Document Q&A
- Transcript Summary
- Output as JSON

ANTHROP\C

10

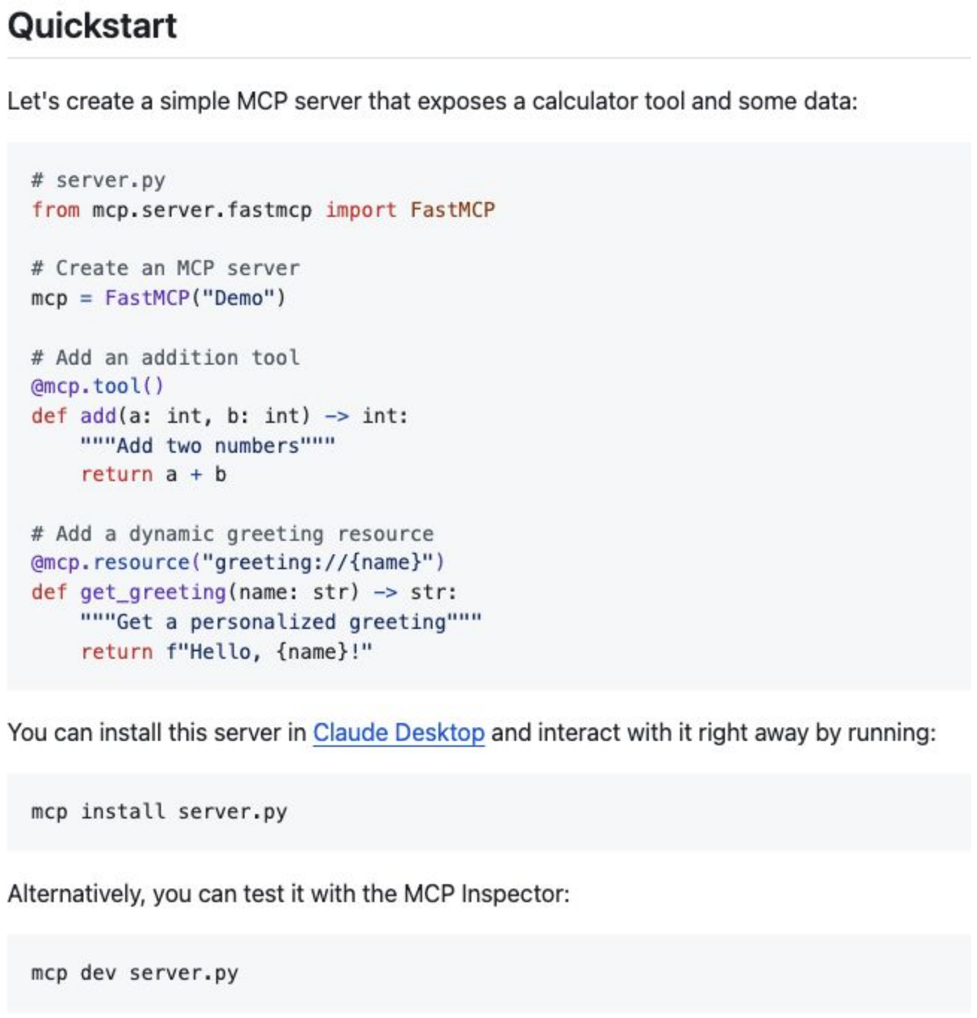
Some code…

Server :: Core interface
Connections Management;
Protocol compliance;
Message routing;

Resources :: Exposed data to LLMs
GET endpoints like;
Should Not perform (a lot of) computation;
No side effects;

Tools :: Exposed actions for AI models
POST/PUT/DELETE endpoints like;
Has side effects;

Prompts :: Templated messages and workflows
To refine outputs; to add guardrails;
To define interactions between agents;



Quickstart

Let's create a simple MCP server that exposes a calculator tool and some data:

```python
# server.py
from mcp.server.fastmcp import FastMCP

# Create an MCP server
mcp = FastMCP("Demo")

# Add an addition tool
@mcp.tool()
def add(a: int, b: int) -> int:
    """Add two numbers"""
    return a + b

# Add a dynamic greeting resource
@mcp.resource("greeting://{name}")
def get_greeting(name: str) -> str:
    """Get a personalized greeting"""
    return f"Hello, {name}!"
```

You can install this server in Claude Desktop and interact with it right away by running:

```
mcp install server.py
```

Alternatively, you can test it with the MCP Inspector:

```
mcp dev server.py
```

Example of MCP Servers

1. Cloudflare - Deploy and manage resources on the Cloudflare developer platform
2. Stripe - Interact with the Stripe API
3. Docker - Manage containers, images, volumes, and networks
4. Kubernetes - Manage pods, deployments, and services
5. Snowflake - Interact with Snowflake databases
6. Browserbase - Automate browser interactions in the cloud
7. Google Maps - Location services, directions, and place details
8. Slack - Channel management and messaging capabilities
9. Puppeteer - Browser automation and web scraping capabilities
10. GitHub - Repository management, file operations, and GitHub API integration
11. SQLite - Database interaction and business intelligence features
12. PostgreSQL - Read-only database access with schema inspection capabilities
13. Filesystem - Secure file operations with configurable access controls
14. ClickHouse - Query your ClickHouse server
15. Grafana - Search dashboards, investigate incidents and query datasources in your Grafana instance
16. Elasticsearch - MCP server implementation that provides Elasticsearch interaction.
17. JDBC - Connect to any JDBC-compatible database and query, insert, update, delete, and more. Supports MySQL, PostgreSQL, Oracle, SQL Server, sqllite and more.
18. Ticketmaster - Search for events, venues, and attractions through the Ticketmaster Discovery API
19. And much more…

References…

Anthropic blog Post | MCP Website | Github Repo

Demo!