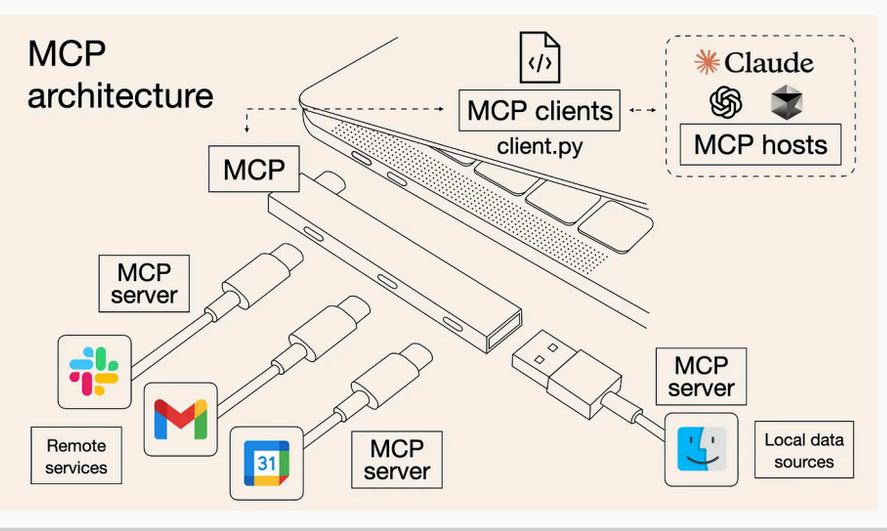
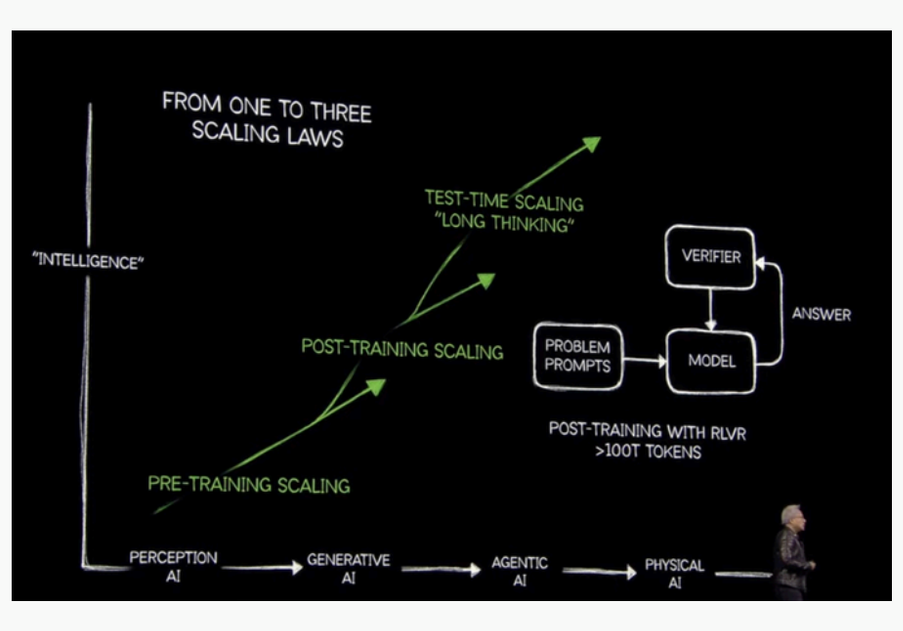
MCP简介 (by @oracleblog)

- 什么是mcp
  - mcp，Model Context ProtocolNo，协议
    - 主要解决什么问题
      - 有思考无行动
        - 解决方法：调用工具
          - 是工具集
          - 是插件包
      - 多个AI模型获取多个数据源：M×N问题
        - 解决方法：通用接口
          - 类似 USB-C

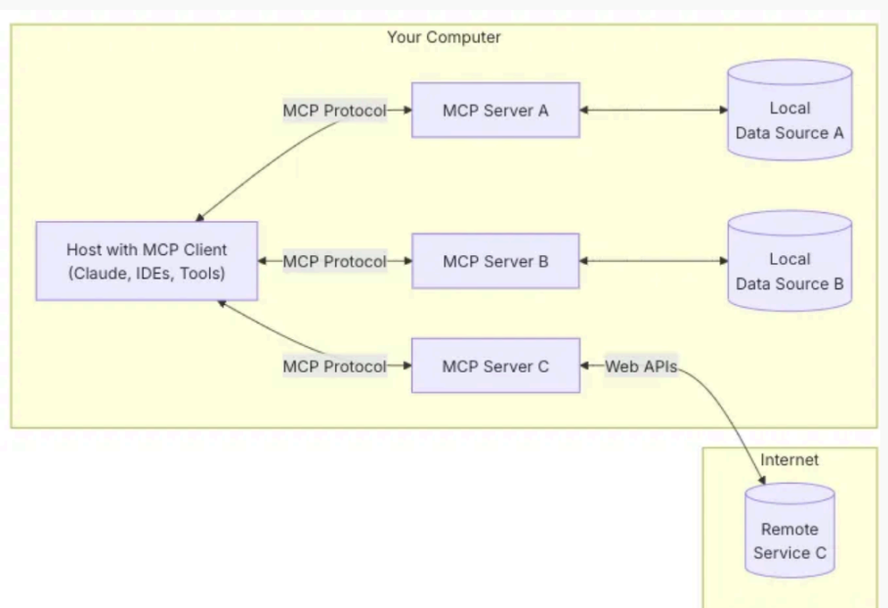




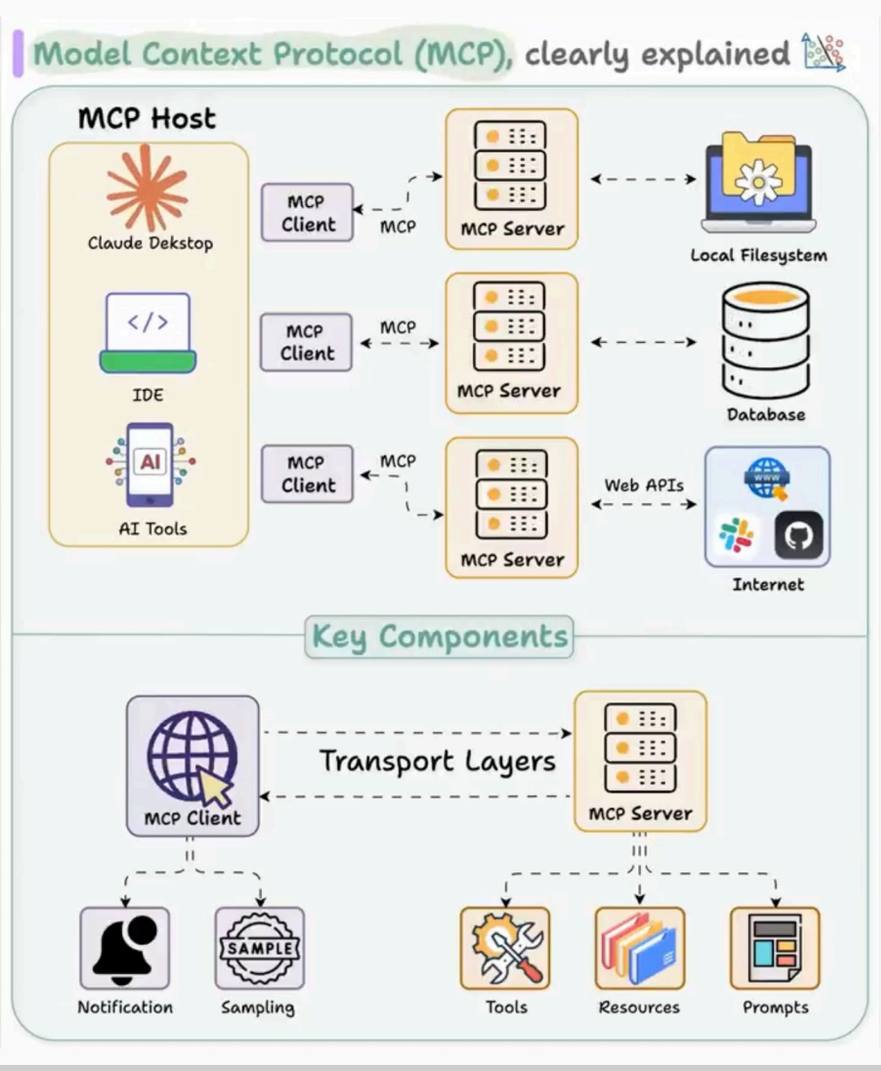
- 兴起
  - AI发展三阶段
    - Generative AI（生成式AI）、Agentic AI（智能体）、Physical AI（具身AI）"
    - Agentic概念的兴起
  - 2024年11月 anthropic 公布 mcp协议
    - "MCP 是一个新的开放标准，让 AI 助手能够连接到数据所在的系统"
  - manus的兴起
    - 通用型，但是缺乏各自垂直领域
    - 垂直领域
      - 做底层基础服务
        - 各种mcp server
  - 2025年3月 openai宣布支持mcp

- MCP相关概念
  - MCP Host
    - MCP主机通常指的是需要通过MCP协议访问数据或服务的程序，比如AI工具、IDE（如Visual Studio、PyCharm）或桌面应用（如Claude Desktop等），一个可交互的AI环境+MCP客户端
  - MCP Client
    - MCP客户端指通过标准化协议与服务器建立稳定连接并进行通信的程序。客户端与服务器之间通常是一对一连接。
  - MCP Server
    - MCP服务器是一些轻量级的软件程序，每个服务器暴露特定的功能或服务，供MCP客户端调用。这些功能和服务通过统一的"模型上下文协议"进行标准化管理和通信。
  - Local Data Source
    - 指本地计算机上的文件、数据库以及服务，这些资源可以通过MCP服务器安全地访问。
  - Remote Service
    - 通过互联网（如API接口）提供的外部系统，MCP服务器可连接并访问这些远程服务。

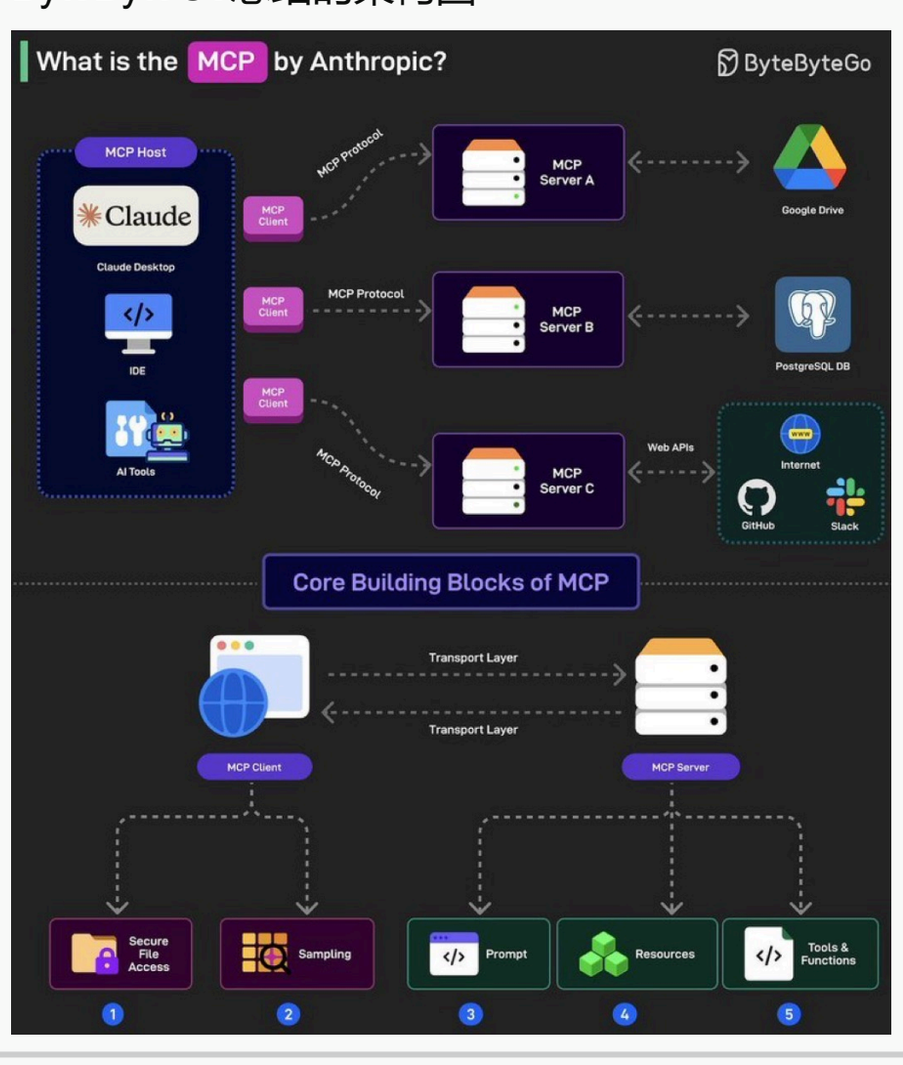
- MCP架构
  - anthropic 官方架构图



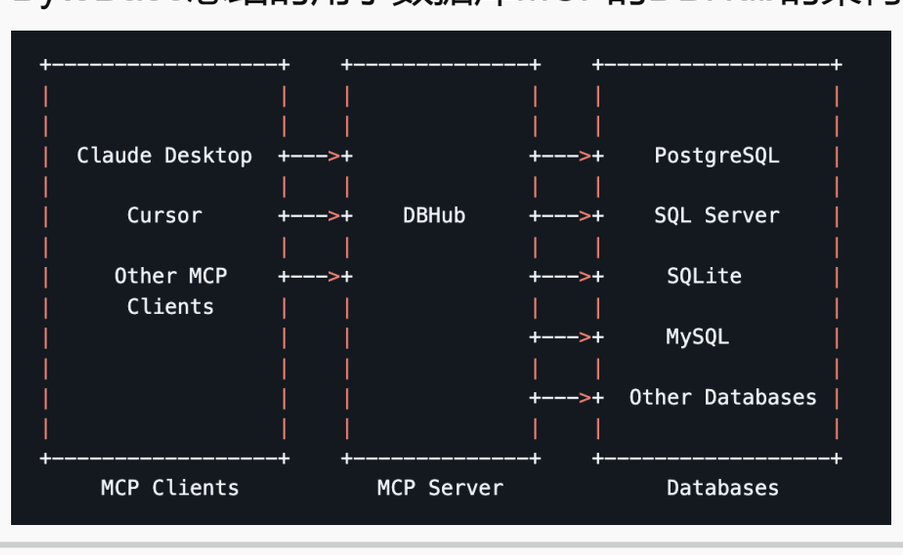
  - dailydoseofds总结的架构图



  - ByteByteGo总结的架构图



  - ByteBase总结的用于数据库MCP的DBHub的架构图



  - Client和Server之间
    - MCP协议
      - Server-关键能力
        - Resource
          - 把Data、文档等内容，暴露给LLMs大语言模型
        - Tools
          - 让LLM，在服务器上执行action
        - Prompt
          - 创建可复用的提示词模板和工作流
      - Server-Client沟通协议（Transport Layer）
        - STDIO
        - SSE

- 常用的MCP
  - MCP Client
    - 常用举例
      - Claude Desktop
      - Cursor
      - Cline
      - Windsurf
      - Cherry Studio
    - Client分类
      - IDE集成开发环境
      - 桌面类工具
      - AI ChatBot
  - MCP Server
    - 常用举例
      - 代码管理类
        - Github
        - Gitlab
      - 位置服务
        - 高德地图
        - 百度地图
        - Google map
      - 网络服务类
        - Cloudflare
      - 建模
        - Blender
      - 前端设计
        - Figma
      - 笔记类
        - obsidian
      - 文件管理
        - Google Drive
      - 资料查询
        - ArXiv（查论文）
      - 数据库类
        - 通用
          - DBHub
            - 通用数据库网关
        - 垂直
          - MySQL
          - Milvus
          - Polardb
          - OceanBase
          - PostgreSQL
          - Redis
          - MongoDB
          - SQLite
          - Supabase
    - MCP Server汇总
      - mcp.so
      - smithery.ai
      - guthub：awesome-mcp-servers
      - mcp-get.com
      - 导航不是关键，形成MCP Store平台是关键
        - 开源+商业化：MCP Server开发者，通过开发MCP Server盈利

- MCP和类似工具的区别

MCP vs LangChain's Tool Calling vs OpenAI's Function Calling

维度	MCP (Model Context Protocol)	LangChain Tool Calling	OpenAI Function Calling
核心功能/定义	- 以开放标准协议形式，让 LLM 在运行时可动态调用外部工具。- 类似 "USB-C 接口"，可跨平台、跨服务复用。- 提供统一的「服务器-客户端」MCP 协议。	- 通过一个 LangChain 框架，将工具 (Tool) 与多步推理 (Agent/Chain) 结合。- 强调在 Python/JS 等语言中内置各种工具定义并可灵活组合使用。- 适合多轮对话、复杂工作流。	- 在 OpenAI 模型里原生支持，通过预先定义函数及 JSON Schema，让模型直接返回调用参数。- 模型判断是否需要调用函数，再由开发者在代码中执行该函数。
主要优势	- **跨 LLM 和工具**：能与多种模型配合，且具备 "多对多" 的扩展能力。- **标准化**：协议统一，第三方服务可快速接入。- **动态扩展**：可在运行时新增或替换工具，避免重启系统。- **可自托管**：安全可控，适合企业内部应用。	- **抽象度高**：提供 Agents/Chains 等高层抽象，易于搭建复杂对话或工作流。- **工具定义灵活**：可在代码中自定义各种功能模块，社区已有丰富示例。- **多步推理**：天然支持多轮对话、推理、调用外部工具并返回结果。- **社区活跃**：文档与插件生态繁荣。	- **简单直接**：内置于 OpenAI 模型，无需额外框架即可用 JSON 格式调用函数。- **结构化输出**：减少解析文本的负担，更易获取确定的参数。- **集成度高**：对 OpenAI 模型 "原生" 支持，单步调用非常方便。
主要劣势	- **生态尚不成熟**：开发者社区相对较小，文档和插件支持仍在完善中。- **部署运维复杂**：需搭建 MCP 服务器并进行协议对接，配置成本高。- **学习曲线**：对使用者理解协议规范要求较高。	- **框架抽象层次深**：对新手有一定门槛，调试与性能优化需要较多经验。- **高并发时的额外开销**：多层封装可能带来一定性能损耗。- **灵活度大**：若项目仅需简单调用，反而可能过度设计。	- **适用范围有限**：目前只支持部分 OpenAI 模型，且对复杂多步场景不够灵活。- **稳定性问题**：可能会输出普通文本而非函数调用，需要额外的错误处理/重试。- **不可扩展**：无法在运行时动态增加函数，扩展新功能需重新定义。
适用场景	- 对外部工具和 LLM 数量均较多，且需在**运行时动态扩展**的企业级系统。- 对**安全和私有化部署**要求高，希望使用开放协议自托管。- 需要统一标准接口、跨团队/跨平台复用。	- 需要**多步推理**、复杂对话和丰富工具集成的应用 (如构建智能助手、数据问答 Agent 等)。- 追求**灵活组合**与快速迭代，对 Python/JS 等语言生态依赖度高。- 适合对 LLM 应用进行**深度定制**的场合。	- 场景相对**简单**，只需**单次或少量函数调用**即可完成任务 (如解析用户指令并返回结构化结果)。- **快速上线**：希望最小化集成成本、直接调用 OpenAI API 即可。- 对**多模型、多工具**的需求不强，单一场景足矣。

- MCP未来
  - 优势
    - 厂商中立带来的大生态
      - 协议，不依赖某个AI厂商，不是服务于某个AI的增值服务，不是function call
    - 创新门槛的降低
      - 大量开发者开发
        - 生态繁荣
        - 大量底层公共基础服务
    - 跨领域的融合
      - 非简单问答
      - 操作机器和设备
        - 不仅仅IT和互联网领域
        - 可能工业制造、医疗、教育等等
  - 风险
    - AI厂商推出替代协议
    - 安全和治理
    - LLMs大语言模型如何有效调用MCP服务
      - LLMs如何决策使用哪个或者哪些MCP
    - "公共基础服务"如何实现商业化价值
      - 如何持续投入