51ª Conferencia Latinoamericana de Informática (LI CLEI 2025)

CONICET ISISTAN

Introducción a LLMs y Agentes

Arquitecturas y Aplicaciones en Ingeniería de Software

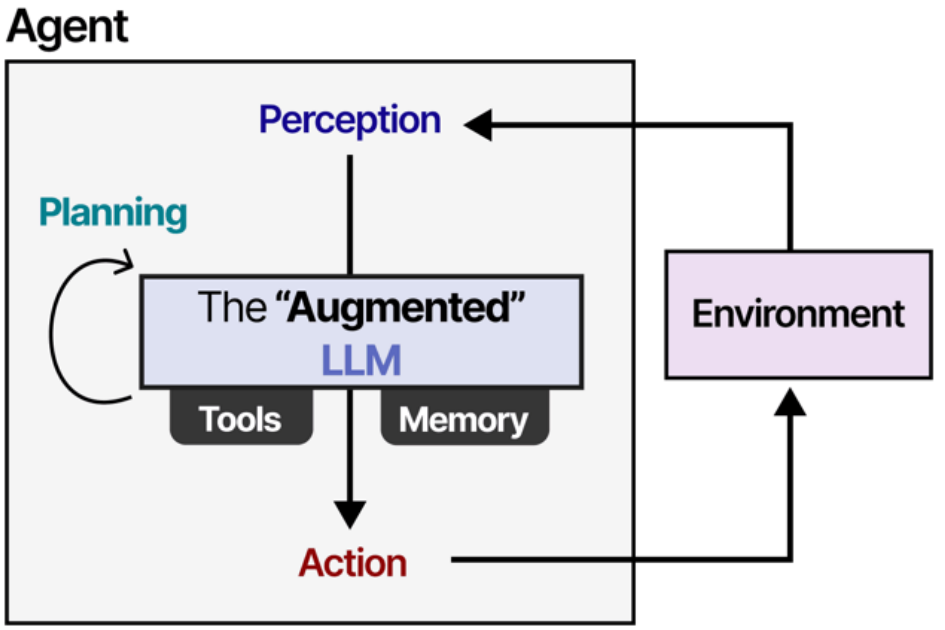
IV ESCUELA DE INFORMÁTICA

J. ANDRÉS DÍAZ PACE

2025

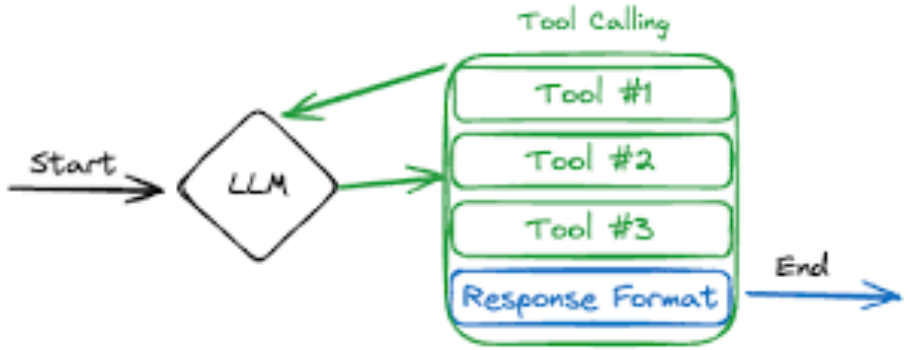
MCP
(Model Context Protocol)

Agentes “todopoderosos” con tools

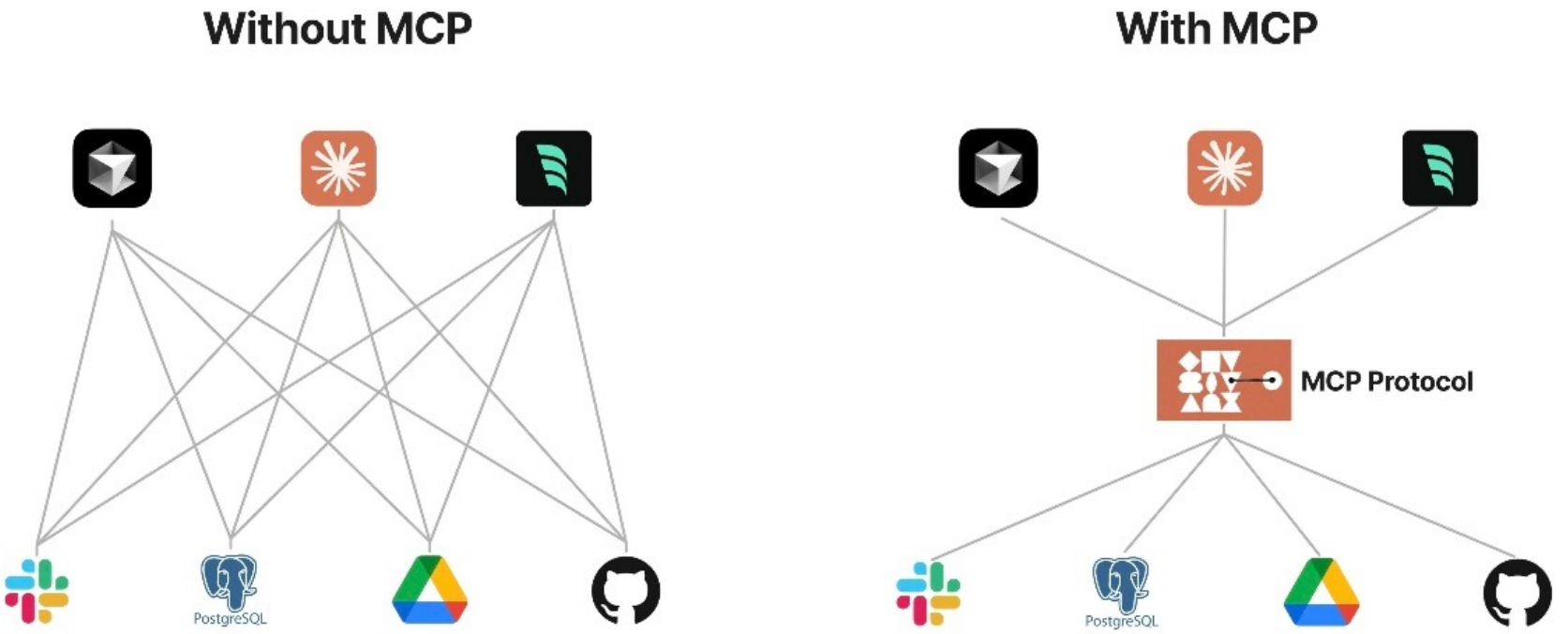


¿Que es MCP (Model Context Protocol)?

- Es un protocolo open-source, desarrollado por Anthropic, que facilita la **integración** entre:
  - aplicaciones basadas en IA / agentes / LLMs
  - tools y fuentes de datos

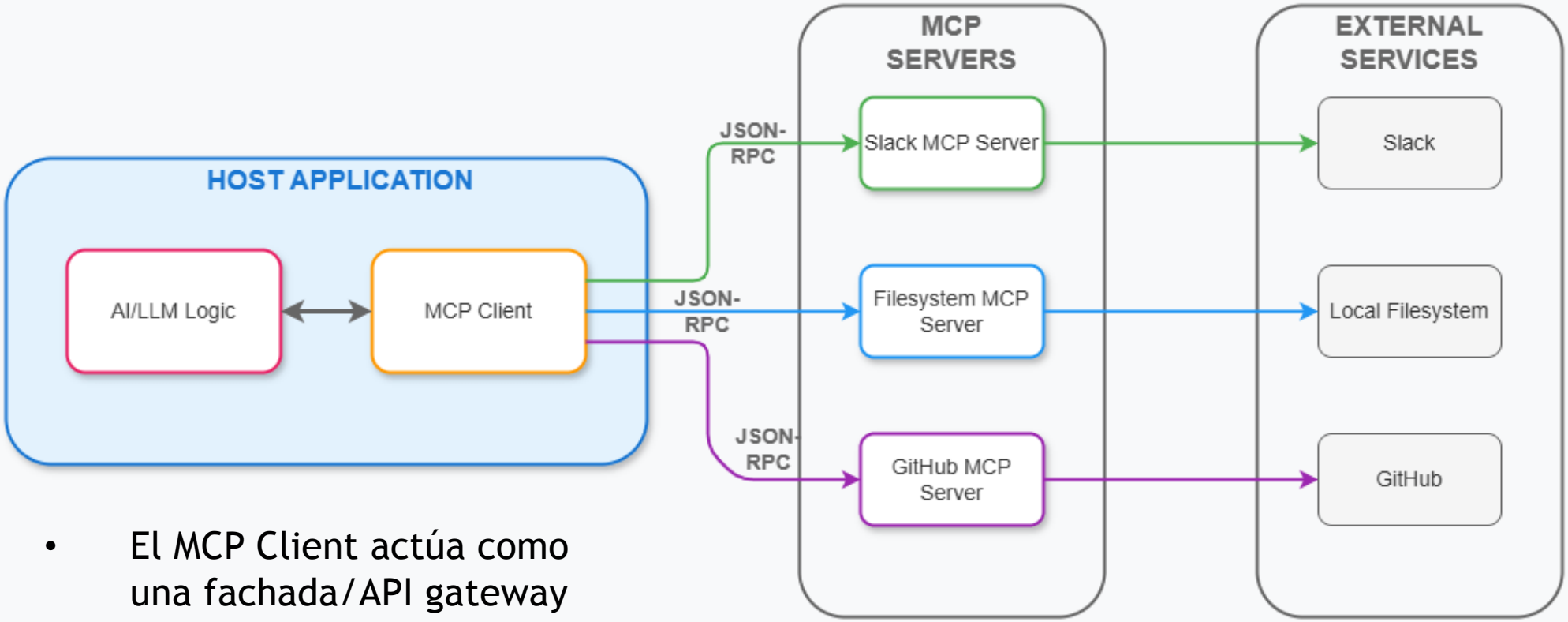


MCP como protocolo (estándard)



Arquitectura MCP

- El MCP Server actúa como un adapter



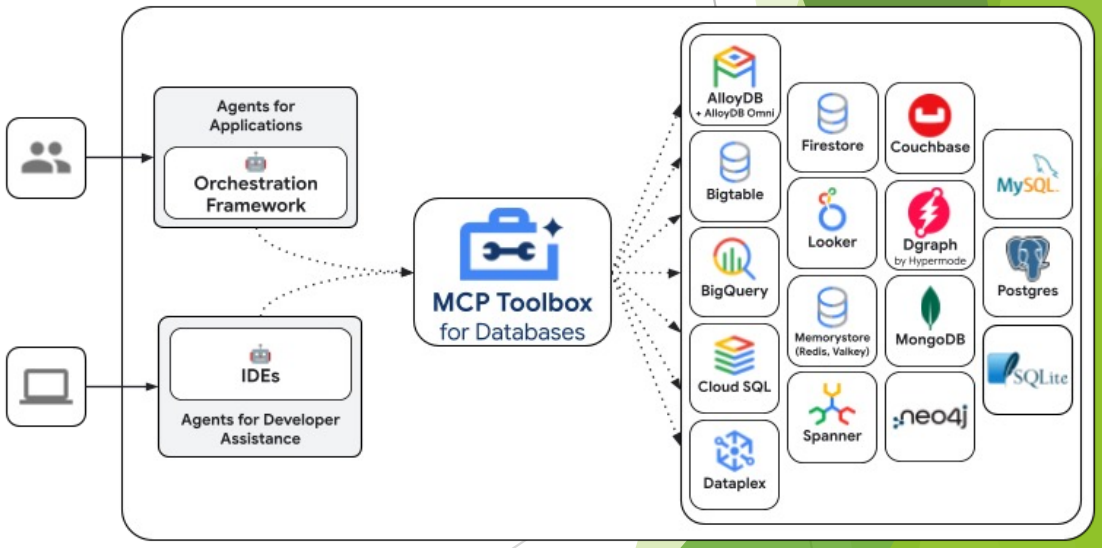
- El MCP Client actúa como una fachada/API gateway

No solo integración de tools …

- Además de conectar tools (y sus resultados), MCP permite que el cliente se comunique con:
  - **Recursos**: fuentes de datos (read-only) tales como archivos, esquemas de BDs, documentación de APIs
  - **Prompts**

- Tipos de transporte
  - **Stdio** (local)
  - **HTTP con SSE** (remoto)
  - **Streamable HTTP** (remoto)

- Desarrollo de servers propios
- Comunicación con servers de terceras partes



FASTMCP

Ejemplo: FastMCP

```
from fastmcp import FastMCP

mcp = FastMCP("My MCP Server")

@mcp.tool
def greet(name: str) -> str:
    return f"Hello, {name}!"

# stdio
if __name__ == "__main__":
    mcp.run()

# http
if __name__ == "__main__":
    mcp.run(transport="http", port=8000)
```

```
import asyncio
from fastmcp import Client

client =
Client("http://localhost:8000/mcp")

async def call_tool(name: str):
    async with client:
        result = await
client.call_tool("greet", {"name":
name})
        print(result)

asyncio.run(call_tool("Ford"))
```

https://github.com/jlowin/fastmcp

Challenge 1: ¿Un RAG sobre MCP?

FASTMCP

Pasos

- RAG básico sobre Langchain
  - Base vectorial para documentos
  - Ingesta (chunking)
  - Prompt
  - Cadena
  - Re-ranking de chunks (opcional)

- Definición de servidor MCP con FastMCP
- Definición de cliente
- Integración de cliente

Challenge 2: ¿Un agente con tools MCP?

Pasos

- *Combinación de un acceso a issues y un chequeo contra las fuentes de documentación*

- Encapsulamiento de tools
- Definición de agente en Langchain

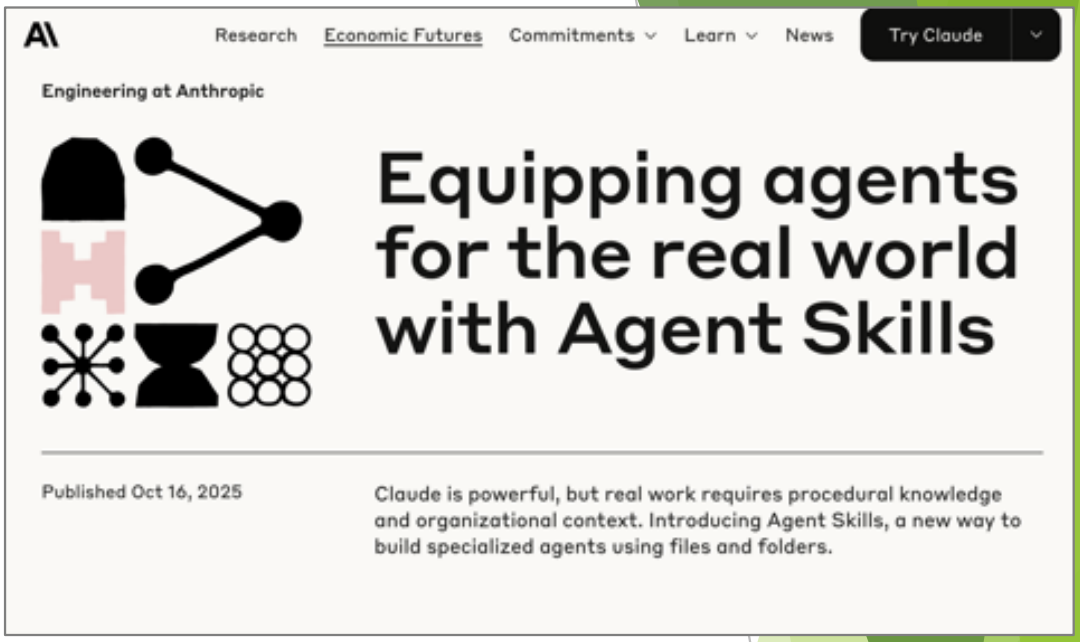
LangChain

Desafíos

- Seguridad
- Gestión de gran cantidad de tools por un agente (filtrado, indexado, RAG, etc.)
- Buenas prácticas de descripción de tools, manejo de errores, etc.
- Validación de tools
- Modelado y **orquestación** de tools (no simplemente API calls)

https://www.microsoft.com/en-us/research/blog/tool-space-interference-in-the-mcp-era-designing-for-agent-compatibility-at-scale/

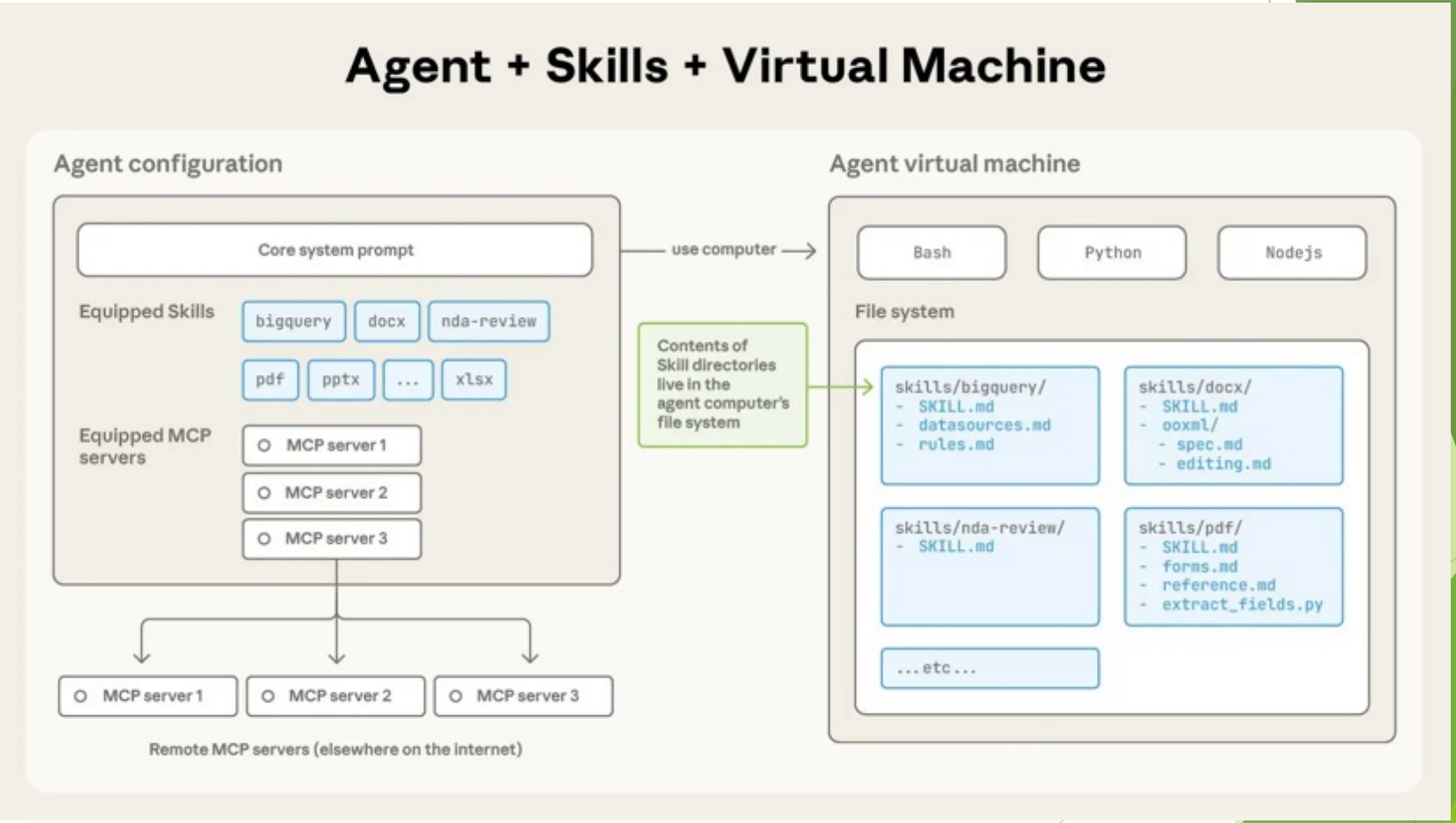
Claude Skills



- Mecanismo mas “liviano” que MCP
- Expertise específica de dominio, mediante recursos modulares y que pueden cargar “mas contexto” solo cuando es necesario (progressive disclosure)
  - Ej., metáfora de TOC, capítulos, apéndices …
- Aprovechan el mecanismo de acceso del agente a archivos (de instrucciones) en un file system

https://www.anthropic.com/news/skills
https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills

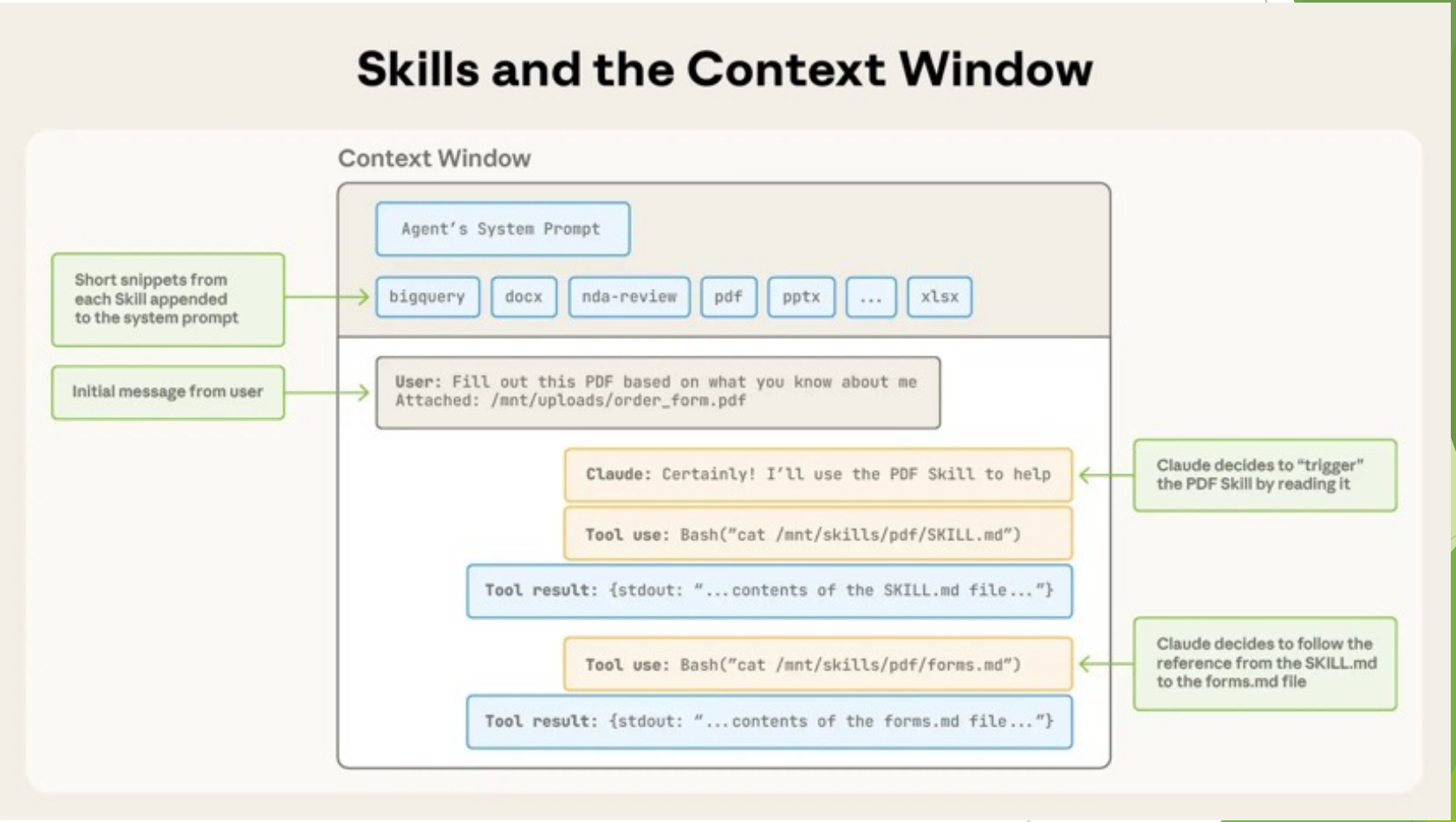
Claude Skills



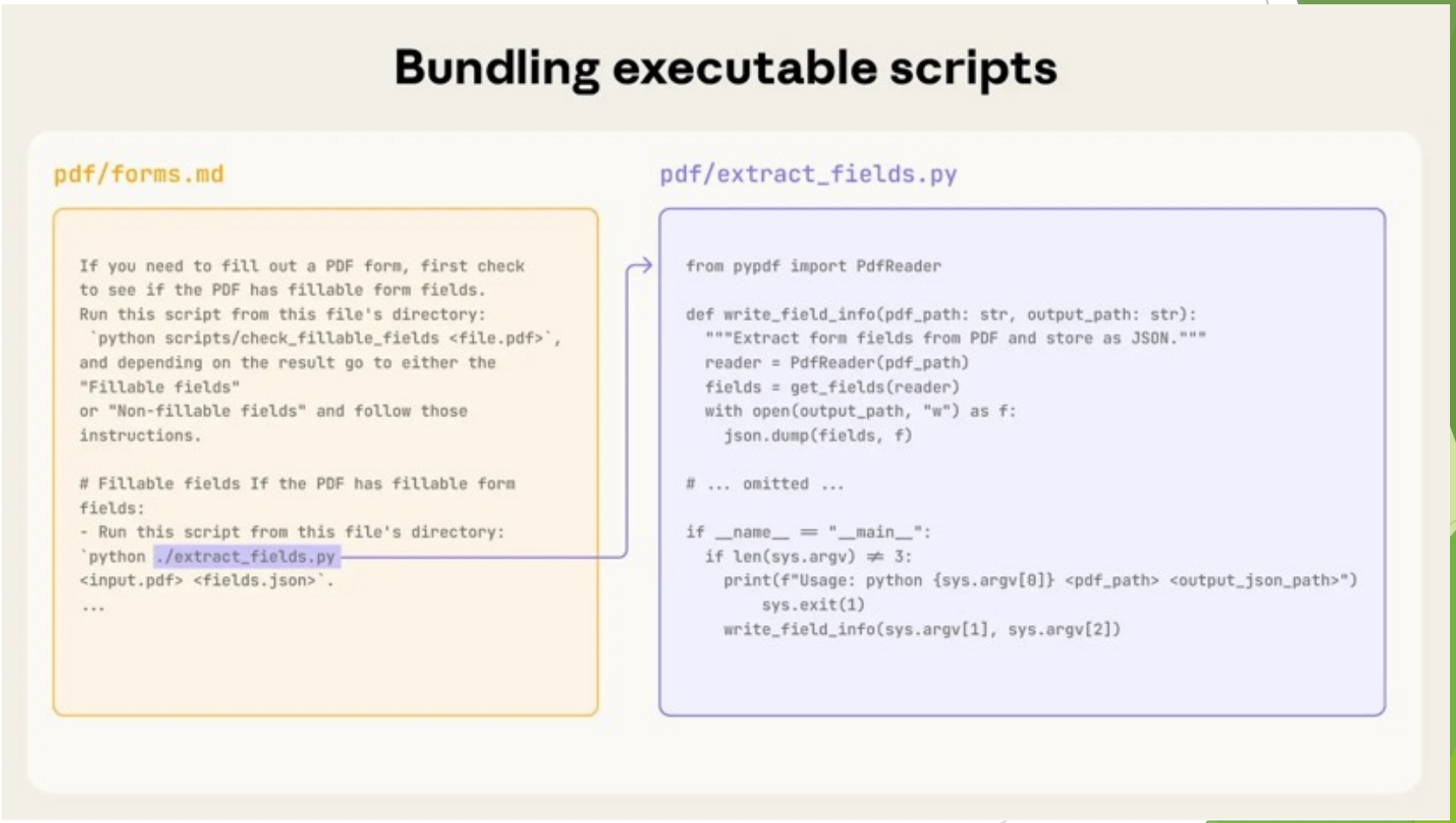
Claude Skills

https://github.com/anthropics/claude-cookbooks/tree/main/skills

Claude Skills



Claude Skills



Próximos pasos

- Realizar una propuesta/prueba de concepto
  - Identificar scope y requerimientos
  - Diseñar la solución (RAG, agentes, workflows, bases de datos, front?)
  - Tecnologías (LLMs, no necesariamente Python o Langchain)
  - Contar con datos (pueden ser semi-sintéticos) para probar
  - Implementar un prototipo que sea demonstrable (opcional)

Gracias!

Andres Diaz Pace

andres.diazpace@isistan.unicen.edu.ar