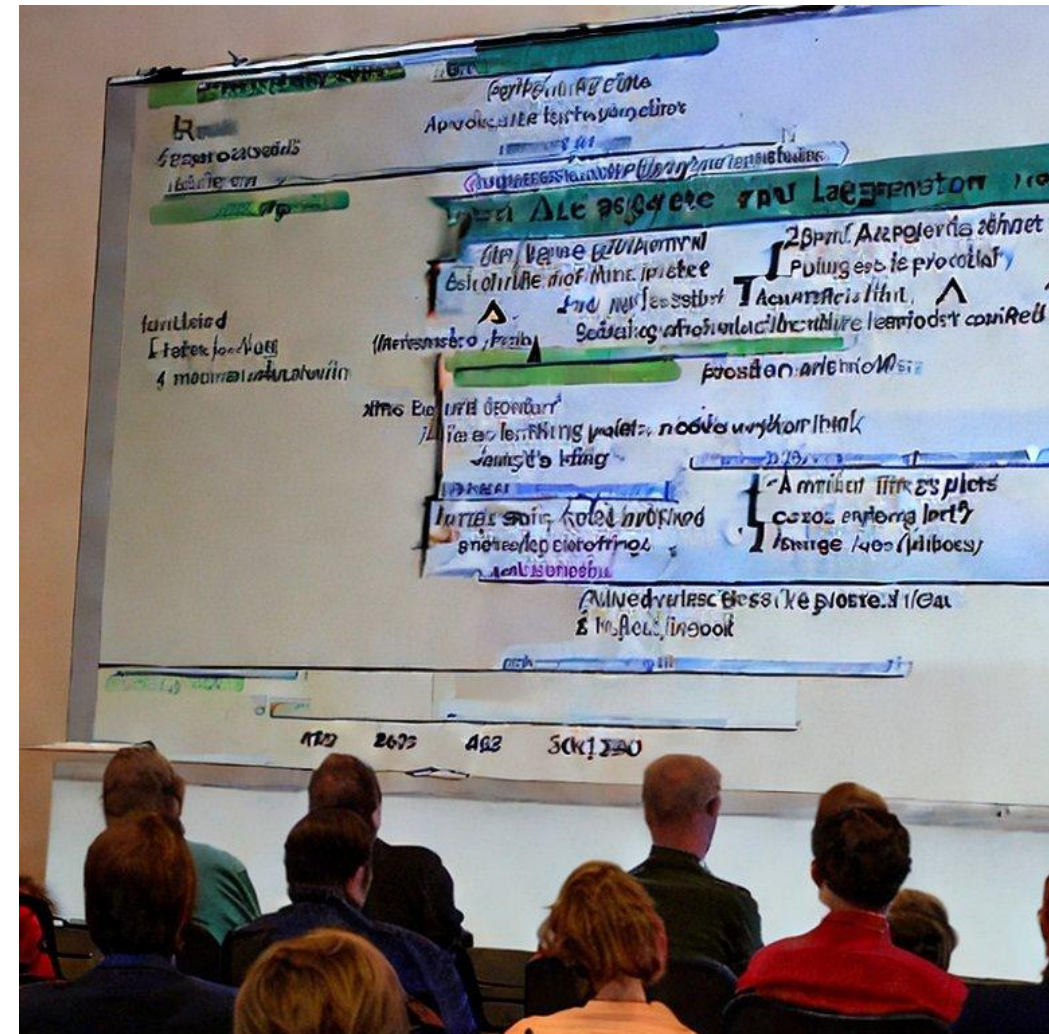


## LLMs e IA Generativa Clase 8

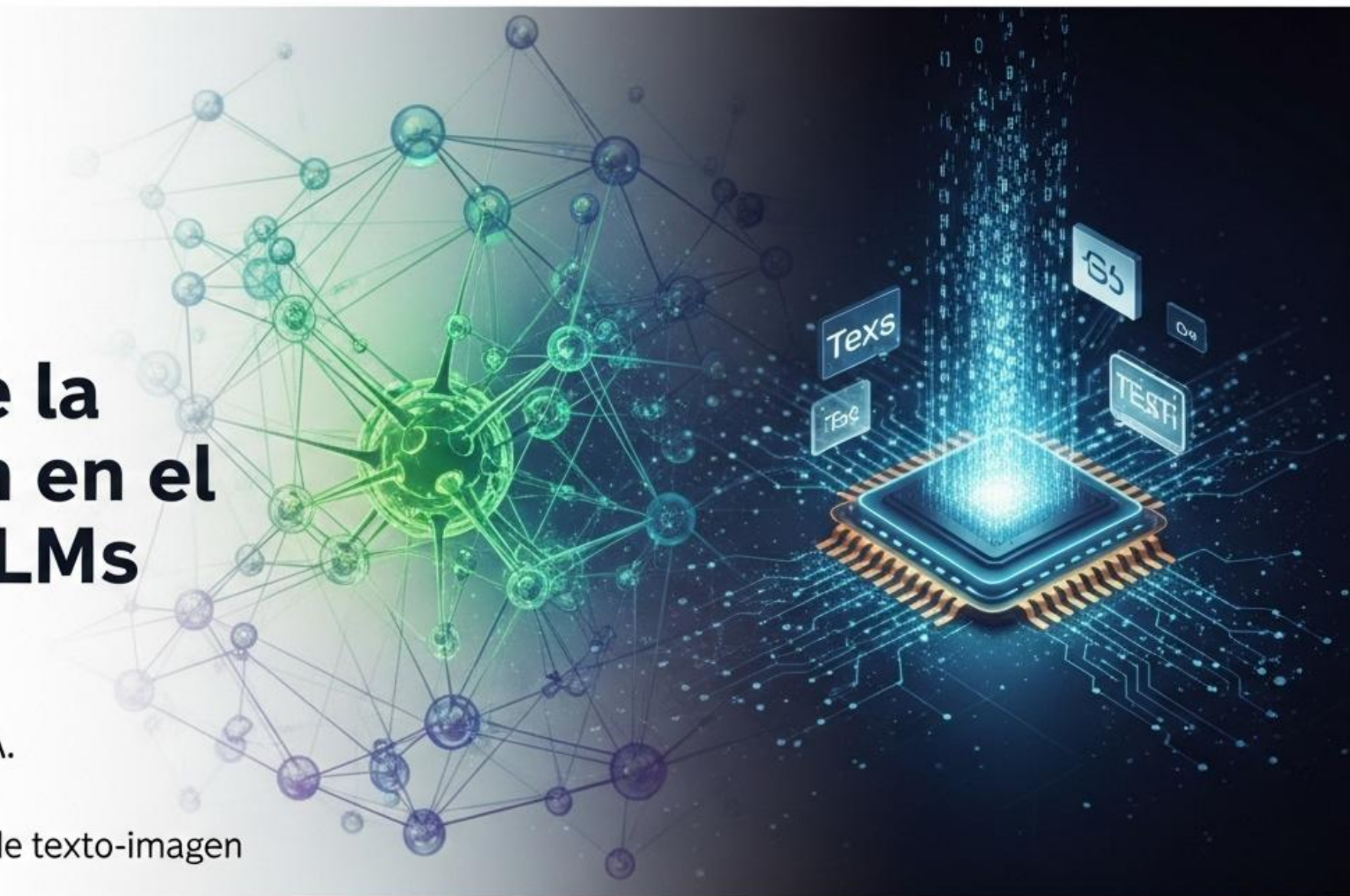
Optimización de LLMs  
LLMs de Razonamiento  
Model Context Protocol  
GenIA multimodal

Mg. Ing. Ezequiel Guinsburg  
ezequiel.guinsburg@gmail.com



# El camino de la optimización en el uso de los LLMs

- LLMs de razonamiento
- Model Context Protocol I.A. generativa multimodal
- Ejemplo teórico de modelo de texto-imagen
- Ejemplo práctico











# PLN2 – Clase 8

Universidad de Buenos Aires

## Temario:

-  El camino de la optimización en el uso de los LLMs
-  LLMs de razonamiento
-  Model Context Protocol
-  I.A. generativa multimodal
-  Ejemplo teórico de modelo de texto-imagen
-  Ejemplo práctico



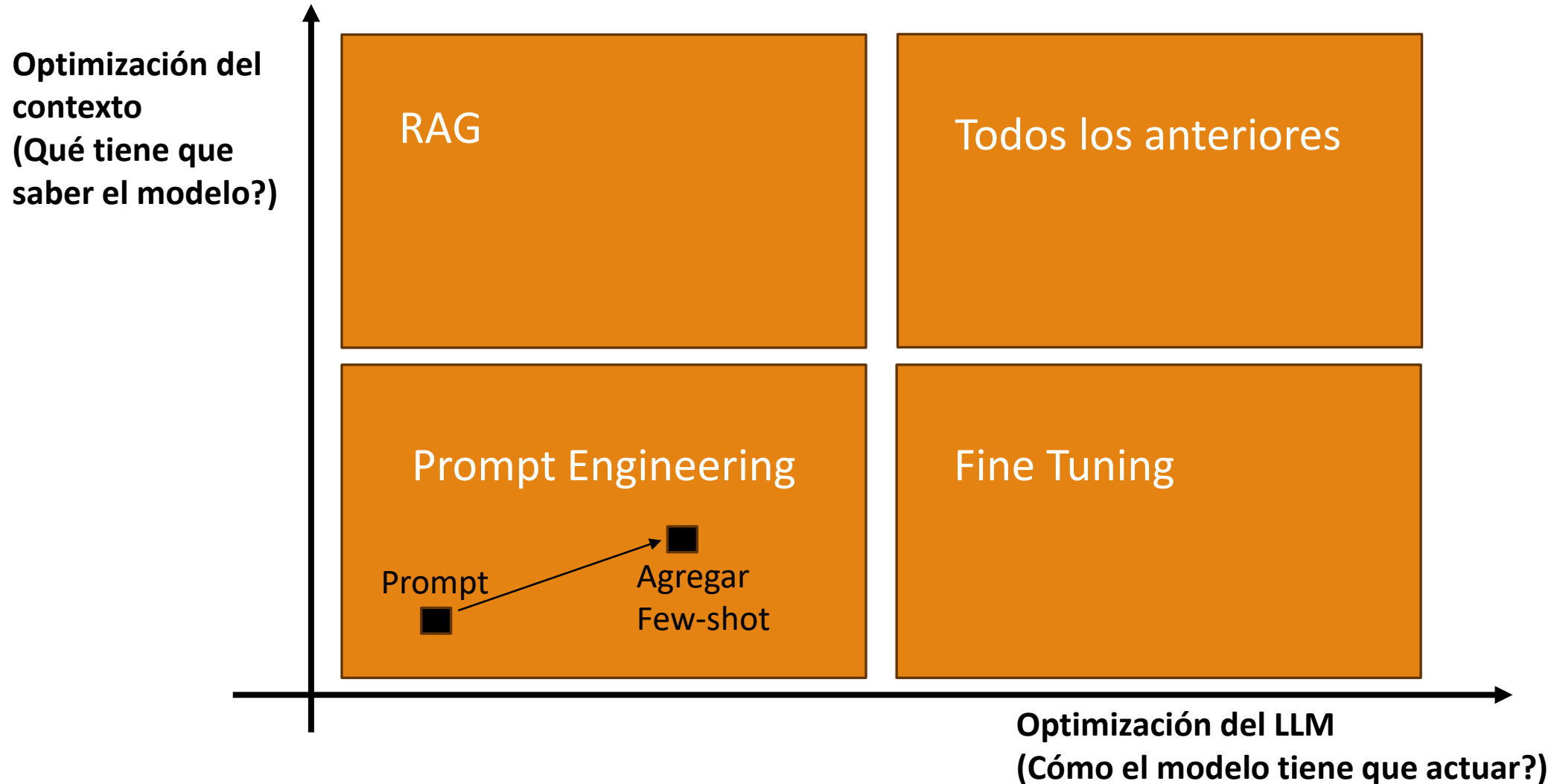
# Temas:

- El camino de la optimización en el uso de los LLMs.
- LLMs de razonamiento.
- Model Context Protocol (MCP)
- I.A. generativa multimodal.
  - Ejemplo teórico de modelo de texto-imagen.
  - Ejemplo práctico.

## Referencias:

- [Paper Reasoning With LLMs, a Survey](#)
- Model Context Protocol: Landscape, Security Threats and Future Research Directions.- <https://arxiv.org/pdf/2503.23278>
- Paper Visual Instruction Tuning - Haotian Liu
- Paper Hicherical Text-Conditional Image Generation with CLIP Latents - <https://arxiv.org/pdf/2204.06125>
- Denoising Diffusion Probabilistic Models - <https://arxiv.org/pdf/2006.11239>

# Prompting Vs. RAG Vs. Fine tuning



# Prompting Vs. RAG Vs. Fine tuning

## Prompt Engineering

### Bueno para:

- Testear y aprender rápidamente.
- Cuando se utiliza y se evalúa nos da una idea de cómo optimizar

### Malo para:

- Introducir nueva información.
- Replicar consistentemente un estilo o método (i.e. aprender un nuevo lenguaje de programación).
- Minimizar el uso de tokens.

# Prompting Vs. RAG Vs. Fine tuning

## Prompt Engineering

### Claves:

- Instrucciones claras.
- Dar tiempo para pensar (step-by-step).
- Dividir un problema complejo en instrucciones simples.
- Incluir ejemplos y evidencia -> Few-shot examples.



# Prompting Vs. RAG Vs. Fine tuning

## RAG

### Bueno para:

- Introducir nueva información para actualizar la del modelo.
- Reducir alucinaciones controlando el contenido

### Malo para:

- Entendimiento de temas muy abarcativos y complejos.
- Enseñar al modelo a aprender nuevos lenguajes, formatos o estilos.
- Reducir el uso de tokens.

# Prompting Vs. RAG Vs. Fine tuning

## Fine tuning

Objetivo: Continuar el proceso de entrenamiento en un dataset de dominio menor para optimizar el modelo en una tarea específica.

### Bueno para:

- Mejorar la performance del modelo en un dominio específico.
- Enfatizar el conocimiento que ya existe en un modelo.
- Mejorar la eficiencia (reducción de la cantidad de tokens).

### Malo para:

- Agregar nueva información al modelo.
- Iteración rápida en una optimización.

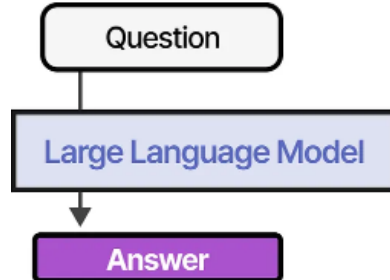
# LLMs de Razonamiento

2024

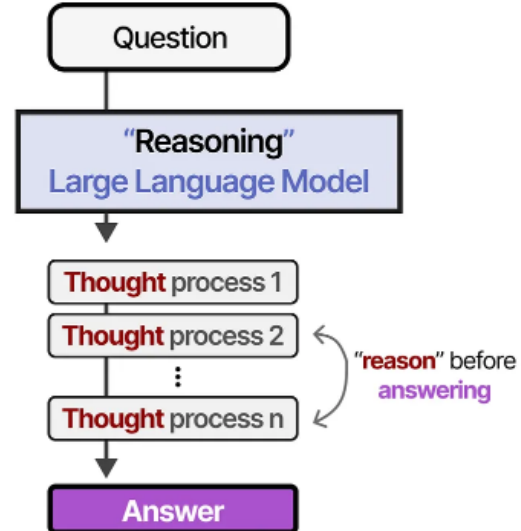
2025

DeepSeek-V3 	o1 	GLM zero-preview 	DeepSeek-R1 	Grok-3 
Gemini 2.0-Flash 	QwQ 32B-preview 		Kimi-k1.5 	o3-mini 
			Claude 3.7 Sonnet 	
			QwQ 32B 	

“Regular” LLMs

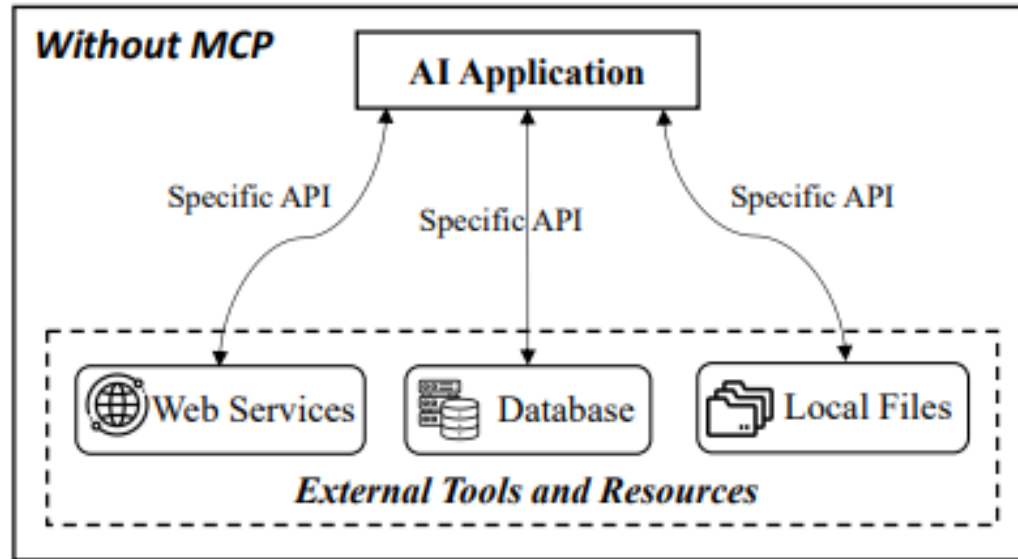


“Reasoning” LLMs

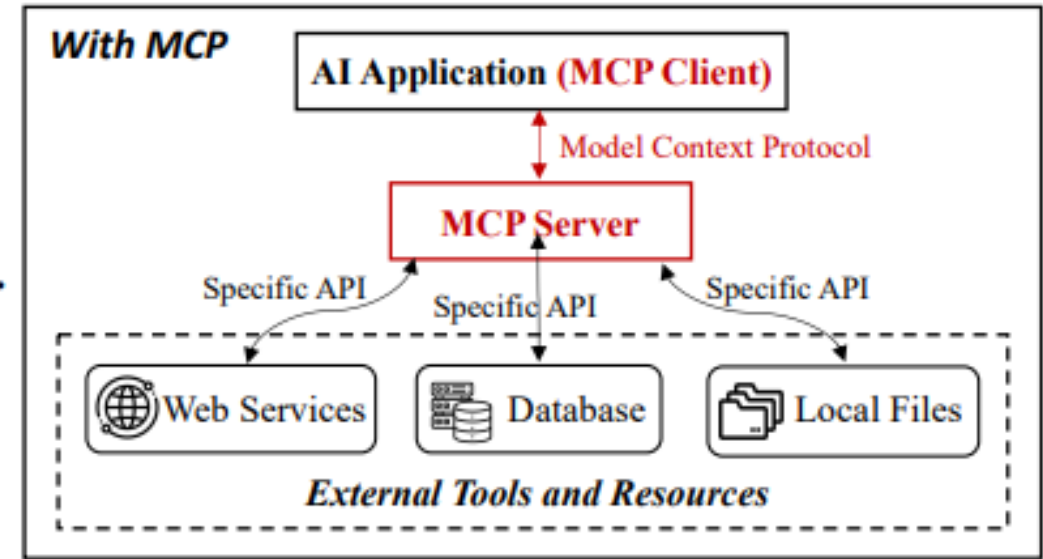


# MCP – Que es y para que sirve (origen)

Model Context Protocol: estándar para que los modelos de IA interactúen de forma segura y uniforme con herramientas, datos y sistemas externos.



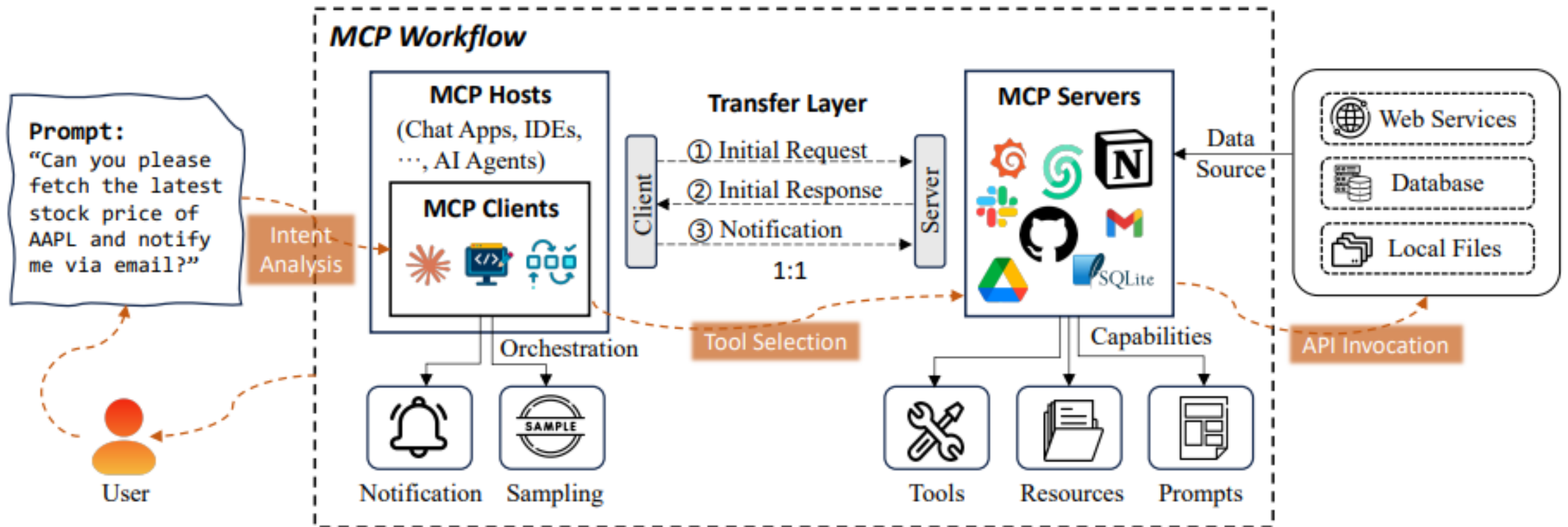
VS.





# MCP – Arquitectura y flujo de trabajo

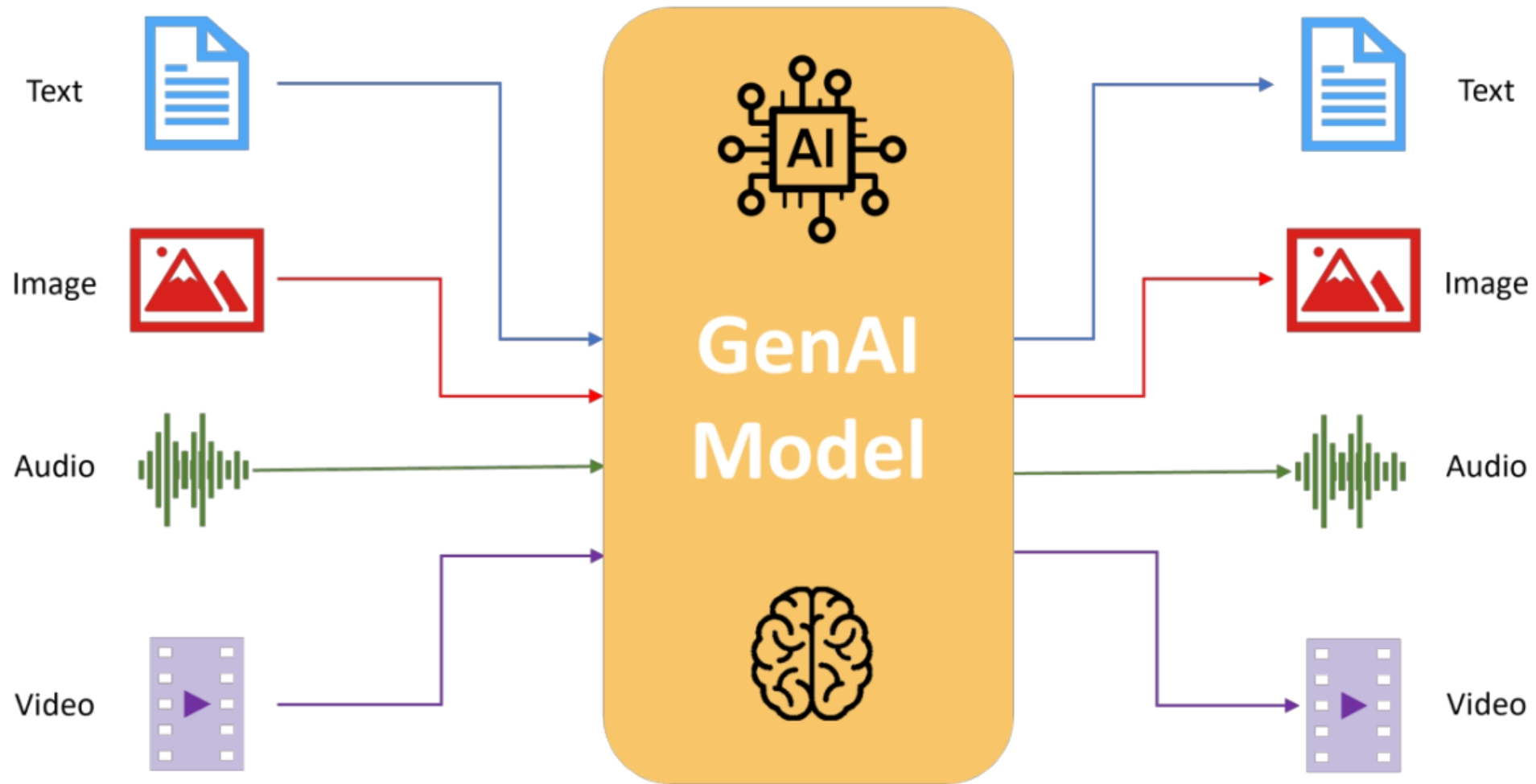
Roles: Host (app que corre la experiencia), Cliente (orquesta y conversa) y Servidor (exposición de capacidades).



# MCP – Adopción del protocolo

Categoría	Compañía/Producto	Funciones / Casos de uso
Modelos y Frameworks de IA	Anthropic (Claude), OpenAI, Baidu Maps, Blender MCP	Soporte MCP en versión desktop y SDK, integración API (ej. geolocalización), generación 3D con comandos en lenguaje natural.
Herramientas de Desarrollo	Replit, MS Copilot Studio, Sourcegraph Cody, Codeium, Cursor, Cline	Entornos con integración MCP, soporte para asistentes de código, ejecución en VS Code, gestión de servidores/herramientas MCP.
IDEs/ Editores	Zed, JetBrains, Windsurf Editor, Theia/IDE, Emacs MCP, OpenSumi	Comandos y tooling MCP en IDEs, soporte AI, integración transparente de herramientas.
Plataformas y Servicios Cloud	Cloudflare, Block (Square), Stripe	Hosting de servidores MCP, integración OAuth, APIs expuestas para plataformas financieras.
Automatización Web y Datos	Apify MCP Tester, LibreChat	Conexión a servidores MCP vía SSE, ecosistema de herramientas ampliado mediante MCP.

# I.A. generativa multimodal



# I.A. generativa multimodal



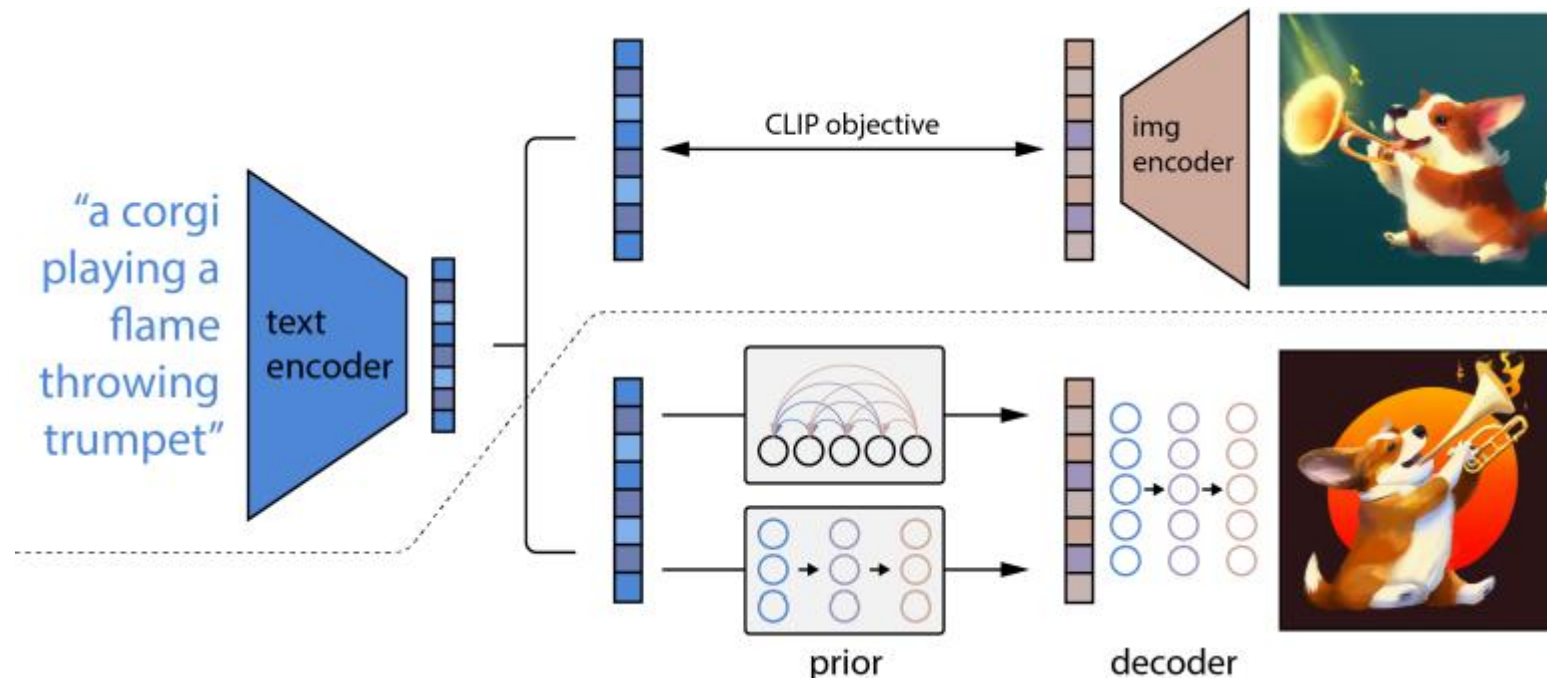


# I.A. generativa multimodal – Ejemplo de modelo

Generación de imágenes condicionada por embeddings de texto.

Encoder (CLIP) – Decoder (Difussion model). **[unCLIP]**

Se utiliza el estado latente del modelo CLIP para manipular los embeddings y sacar distintos tipos de imagen



## I.A. generativa multimodal – Ejemplo de modelo

Dataset de entrenamiento  $(x, y)$

$x$  = imágenes |  $y$  = descripciones

$z_i$  = embeddings de imagen |  $z_t$  = embeddings de texto

Se utilizan dos componentes:

prior  $P(z_i | y)$  = produce los embeddings de imagenes CLIP

decoder  $P(x | z_i, y)$  = produce las imágenes  $x$  condicionadas a  $z_i$  e  $y$

Combinando los dos componentes tenemos el modelo generativo de imágenes  $x$  dado las descripciones  $y$ :

$$P(x|y) = P(x, z_i|y) = P(x|z_i, y)P(z_i|y)$$

# I.A. generativa multimodal – Ejemplo de modelo

Reconstrucción de imágenes a partir de espacios latentes reducidos



Trabajo comparativo de modelos:

<https://distill.pub/2021/multimodal-neurons/>

Repositorio del código para utilizar:

<https://github.com/Stability-AI/stablediffusion>

# I.A. generativa multimodal

## Ejemplos

Ejemplo 1: (text2img)

<https://huggingface.co/CompVis/stable-diffusion-v1-4>

<https://colab.research.google.com/drive/1SXnX2zlx5oE3sltz65stMYvz0nYyfEvl?usp=sharing>

Ejemplo 2 (Entendimiento Multimodal)

<https://huggingface.co/deepseek-ai/Janus-1.3B>

Ejemplo 3 (Imagen a texto estructurado)

Ver “Codigo->Ejemplo\_IMG2TXT.ipynb”



# Ejercicio en clase:

- **Consigna:** Implementar una aplicación que funcione como un LLM con razonamiento, el cual recibe una pregunta compleja y utiliza diferentes agentes para resolver parcialmente y luego se compaginan todas las respuestas para ofrecer la solución.
- Además de la respuesta se debe imprimir la cantidad de tokens de entrada, salida y razonamiento (estos últimos son los que tokens utilizados en las etapas intermedias).
- **Entregables:** Link a al repositorio y video demostrativo (ídem clases anteriores).
- **Fecha límite de entrega:** Domingo 27 de Abril. No hay excepciones ya que estamos pasados de la fecha de entrega de las notas.
- Además de valorar la calidad técnica de la solución propuesta, la **prolijidad general** será un **criterio fundamental en la calificación**. Esto incluye:
  - La **claridad y redacción del texto entregado** (explicaciones, documentación, etc.).
  - La **organización del repositorio**: estructura de carpetas, nombres de archivos, uso de README, etc.
  - La **presentación general** del trabajo: que sea fácil de entender, limpio y bien presentado.Un buen trabajo técnico también debe ser claro, ordenado y profesional. Tener una solución funcional pero mal presentada puede afectar negativamente la nota final.