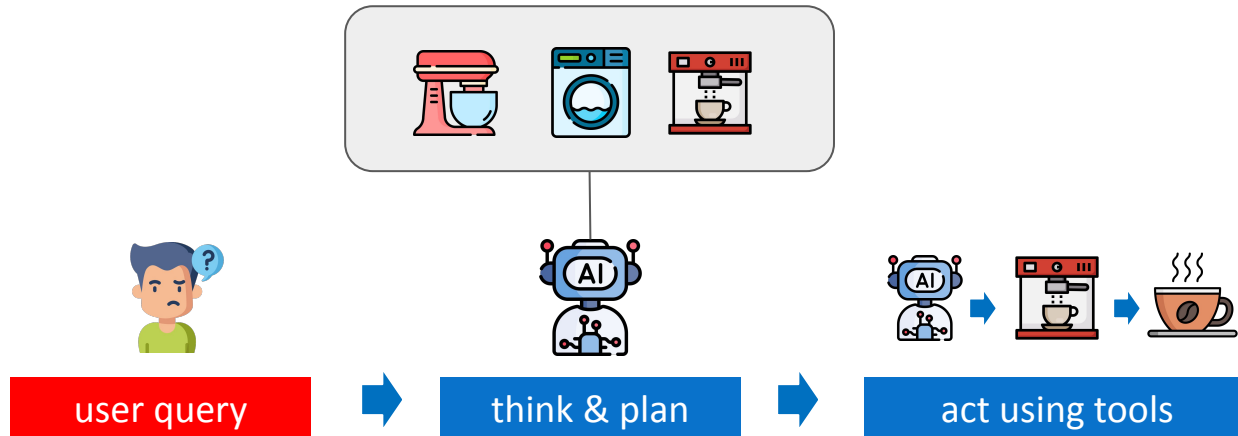


Agentes

Agentes

Un **agente de IA** es un programa que **razona** y **planifica acciones** basadas en las peticiones de un usuario, y luego **responde de manera coherente**, utilizando el lenguaje natural.

Lo llamamos Agente porque tiene la habilidad para **interactuar con el entorno**.



Tools

Las **Tools** son las interfaces para que un LLM pueda “interactuar con el mundo”

Ejemplo:



¿me puedes describir esta imagen?
<https://icdn.football-espana.net/wp-content/uploads/2021/08/1002614356.jpg>

¿tiene el LLM Tools?

no

si

Lo siento como IA de lenguaje no puedo acceder a ninguna página web externa ni entender imágenes

Se revisan una a una todas las descripciones de las Tools a ver si alguna cuadra con la pregunta del usuario

Tool 1



Tool 2



Tool Image Caption

¿me puedes describir esta imagen?
<https://icdn.football-espana.net/wp-content/uploads/2021/08/1002614356.jpg>

"use this tool when given the URL of an image that you'd like to be described. It will return a simple caption describing the image."



Si cuadran, **el propio LLM extrae los argumentos que necesita la tool para ser ejecutada** ☐ url


```
class ImageCaptionTool(BaseTool):  
    name = "Image captioner"  
    description = "use this tool when given the URL of an image that you'd like to be "  
        "described. It will return a simple caption describing the image."  
  
    def _run(self, url: str):  
        # download the image and convert to PIL object  
        image = Image.open(requests.get(img_url, stream=True).raw).convert('RGB')  
        # preprocess the image  
        inputs = processor(image, return_tensors="pt").to(device)  
        # generate the caption  
        out = model.generate(**inputs, max_new_tokens=20)  
        # get the caption  
        caption = processor.decode(out[0], skip_special_tokens=True)  
        return caption
```



Ejemplo

Y aquí un ejemplo de cómo ChatGPT4 acaba llamando a esas herramientas externas.


 ChatGPT 4 ▾


 **You**
dibujame un botón rojo en fondo verde

 **ChatGPT**
 Creating image



 ChatGPT 4 ▾

 **You**
mira en esta web si hay una tool para discord: <https://llamahub.ai/>

 **ChatGPT**
 Doing research with Bing



Hub de tools

<p>google_docs</p> <p>Verified Loader</p> <p>jerryjliu</p> <p>17 • 714 • 2 months ago</p>	<p>slack</p> <p>Verified Loader</p> <p>jerryjliu</p> <p>13 • 152 • 2 months ago</p>	<p>discord</p> <p>Verified Loader</p> <p>jerryjliu</p> <p>32 • 113 • 2 months ago</p>
<p>s3</p> <p>Verified Loader</p> <p>thejessezhang</p> <p>2 • 486 • 10 days ago</p>	<p>mongo</p> <p>Verified Loader</p> <p>jerryjliu</p> <p>11 • 412 • 1 month ago</p>	<p>file/pandas_excel</p> <p>Loader</p> <p>maccarini</p> <p>2 • 294 • 2 days ago</p>
<p>neo4j_query_engine</p> <p>LlamaPack</p> <p>wenqiglantz</p> <p>1 • 24 • 7 days ago</p>	<p>gmail</p> <p>Tool</p> <p>ajhofmann</p> <p>3 • 1 • 2 months ago</p>	<p>reddit</p> <p>Loader</p> <p>vanessahlyan</p> <p>1 • 5 • 5 months ago</p>

<https://llamahub.ai/>



Agentes

Un **agente** es un motor de razonamiento y toma de decisiones automatizado basado en lenguaje natural.

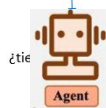
- En realidad, el LLM no es la pieza que llama a las Tools. Son los “agentes” o “routers” o “semantic functions”.

El agente utiliza un LLM como “cerebro” para exhibir capacidades más allá de la generación de texto, incluida la realización de conversaciones, la realización de tareas. Es decir, recibe una entrada o consulta del usuario y toma **decisiones internas** para ejecutar esa consulta con el fin de devolver el resultado correcto.

```
sys_msg = """Assistant is a large language model trained by OpenAI.  
  
Assistant is designed to be able to assist with a wide range of tasks, from answering simple ques  
  
Assistant is constantly learning and improving, and its capabilities are constantly evolving. It  
  
Overall, Assistant is a powerful system that can help with a wide range of tasks and provide valu  
"""  
  
new_prompt = agent.agent.create_prompt(  
    system_message=sys_msg,  
    tools=tools  
)
```



¿me puedes describir esta imagen?
<https://icdn.football-espana.net/wp-content/uploads/2021/08/1002614356.jpg>



Lo siento como IA de lenguaje no puedo acceder a ninguna página web externa ni entender imágenes

Se revisan una a una todas las descripciones de las Tools a ver si alguna cuadra con la pregunta del usuario

Nota, la “jerga” de langchain puede variar con otros orquestadores y parece estar cambiando. Pero el potencial de lo que se puede llegar a hacer se mantiene.

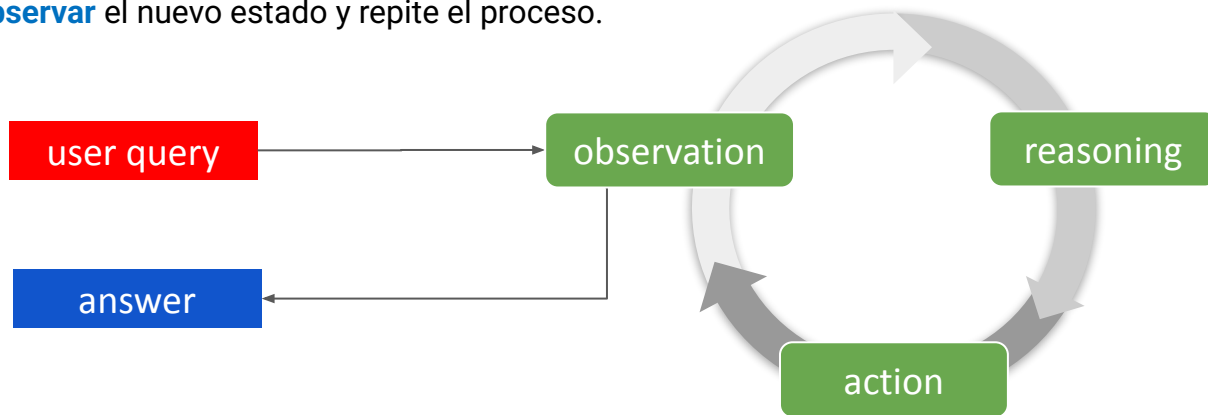


ReAct Agent

React viene de **Reasoning** and **Acting**

Funciona siguiendo un bucle de **estados** cuando un usuario realiza una consulta.

- **Observa** el estado actual (por ejemplo, una pregunta del usuario o el resultado de una acción previa).
- **Razonamiento**: decide qué hacer a continuación (usualmente mediante un modelo de lenguaje).
- **Ejecuta una acción**, como usar una herramienta, buscar algo o responder.
- **Vuelve a observar** el nuevo estado y repite el proceso.



Ejemplo ReAct



¿Cuál es la capital de Suecia y cuántos habitantes tiene?

observation

Necesito saber la capital de Suecia

reasoning

Usar una herramienta de búsqueda y darle como parámetro "Capital de Suecia"

action

Mi herramienta de búsqueda me dice que la capital de Suecia es "Estocolmo". Pero no puedo responder aún porque no sé cuántos habitantes tiene.

observation

Necesito saber la población de Estocolmo

reasoning

Usar una herramienta de búsqueda y darle como parámetro "Habitantes en Estocolmo"

action

Mi herramienta de búsqueda me dice que Estocolmo tiene "aproximadamente 1.5 millones de habitantes". Y puedo responder a la pregunta porque ya tengo toda la información necesaria.

observation

La capital de Suecia es Estocolmo, que tiene aproximadamente 1.5 millones de habitantes.



System prompt

Eres un agente inteligente que puede razonar y actuar paso a paso para resolver tareas complejas.
Tu comportamiento se basa en el patrón ReAct: pensar, actuar y observar antes de generar una respuesta final.

Sigue siempre el siguiente formato:

Pensamiento: Describe qué necesitas hacer a continuación.

Acción: Elige una acción de la lista de herramientas disponibles.

Entrada: El input necesario para esa acción.

Observación: El resultado de ejecutar la acción.

Repite este ciclo tantas veces como sea necesario.

Cuando tengas toda la información que necesitas, responde con:

Respuesta final: [tu respuesta al usuario]

Tools disponibles:

- RAG[query]: Para buscar información en tu base de conocimiento interna (documentos, base de datos, corpus).
- WEB[query]: Para buscar información actualizada en Internet.

Instrucciones:

- Usa ****RAG**** si crees que la respuesta puede estar en documentos internos o en tu base de conocimiento.
- Usa ****WEB**** solo si necesitas información actual, noticias o datos que probablemente no estén en RAG.
- No inventes observaciones.
- No ejecutes acciones sin explicar tu razonamiento primero.
- No proporciones una respuesta final hasta que estés seguro de tener toda la información necesaria.



El gran problema

A medida que cargamos al agente con más Tools y con más información:

- **Los prompts crecen**, tratando de cubrir todos los posibles comportamientos.
- Se incluyen **muchas instrucciones, herramientas y contextos** en una sola conversación.

Y esto puede causar:



Instrucciones contradictorias.



Latencia por contexto excesivo.



Pérdida de trazabilidad: difícil saber por qué el modelo tomó ciertas decisiones.



Dificultad para iterar o depurar.



Errores de razonamiento sobre múltiples objetivos al mismo tiempo.

Se pierde el control del Agente y no entendemos por qué lo hace



Sistema Multi-agente

Un **sistema multiagente** es un enfoque en el que múltiples agentes de interactúan entre sí para resolver tareas de manera colaborativa.

- **Cada agente tiene un rol o especialización**, y pueden comunicarse, cooperar o incluso competir entre ellos para lograr un objetivo común.
- Un sistema multiagente implica:
 - **Especialización de agentes**: Diferentes modelos o instancias de un modelo con configuraciones específicas para tareas particulares (ejemplo: uno para análisis, otro para generación de texto, otro para evaluación).
 - **Interacción y coordinación**: Los agentes pueden discutir, refutar, o confirmar información antes de dar una respuesta final.
 - **Toma de decisiones distribuida**: Cada agente contribuye con su propia perspectiva para mejorar la calidad de la respuesta global.





LangGraph = **LangChain** + **Graph-based control**

LangGraph permite construir **agentes con múltiples pasos**, pero en lugar de usar una cadena lineal (como en LangChain o LlamaIndex), organiza la lógica como un **grafo dirigido** con los siguientes **componentes**:

- **Nodes**: pasos en el flujo (pueden ser LLMs, funciones personalizadas, llamadas a API...).
- **Edges**: transiciones entre nodos según condiciones.
- **State**: el grafo puede mantener un estado compartido que se va actualizando.
- **Routing logic**: Condiciones definidas con lógica personalizada que deciden qué nodo visitar.

Caso de uso

Agente de Seguros

Objetivo:

Construir una IA que emule ser un Agente de seguros e interactúe con las consultas de los Clientes.

En el caso de que el cliente quiere hacer una pregunta que no tenga que ver con los seguros, habría que darle las gracias pero no proceder a responderle usando el agente de IA.

En el caso de que un cliente quiere reportar un siniestro, se pueda abrir una incidencia mandando un mensaje por Telegram (que bien podría ser un email).

