

LangChain, LangGraph y LangSmith

Guía General y Comparativa

Generado automáticamente por ChatGPT

Tabla de Contenidos

1. 1. Introducción
2. 2. ¿Qué es LangChain?
3. 3. Casos de Uso de LangChain
4. 4. ¿Qué es LangGraph?
5. 5. Ejemplo Avanzado con LangGraph
6. 6. ¿Qué es LangSmith?
7. 7. Comparación: LangChain, LangGraph y LangSmith
8. 8. Tabla Resumen
9. 9. Conclusiones
10. 10. Recursos Adicionales

1. Introducción

En los últimos años, los grandes modelos de lenguaje (LLM) han revolucionado el desarrollo de aplicaciones inteligentes. Herramientas como LangChain, LangGraph y LangSmith se han posicionado como piezas clave para crear, orquestar y monitorear flujos conversacionales, agentes autónomos y sistemas de IA integrados en aplicaciones empresariales.

Este documento brinda una visión general de estas herramientas, sus características principales y ejemplos de uso, ayudando a comprender cuándo y cómo aprovecharlas para proyectos de IA modernos.

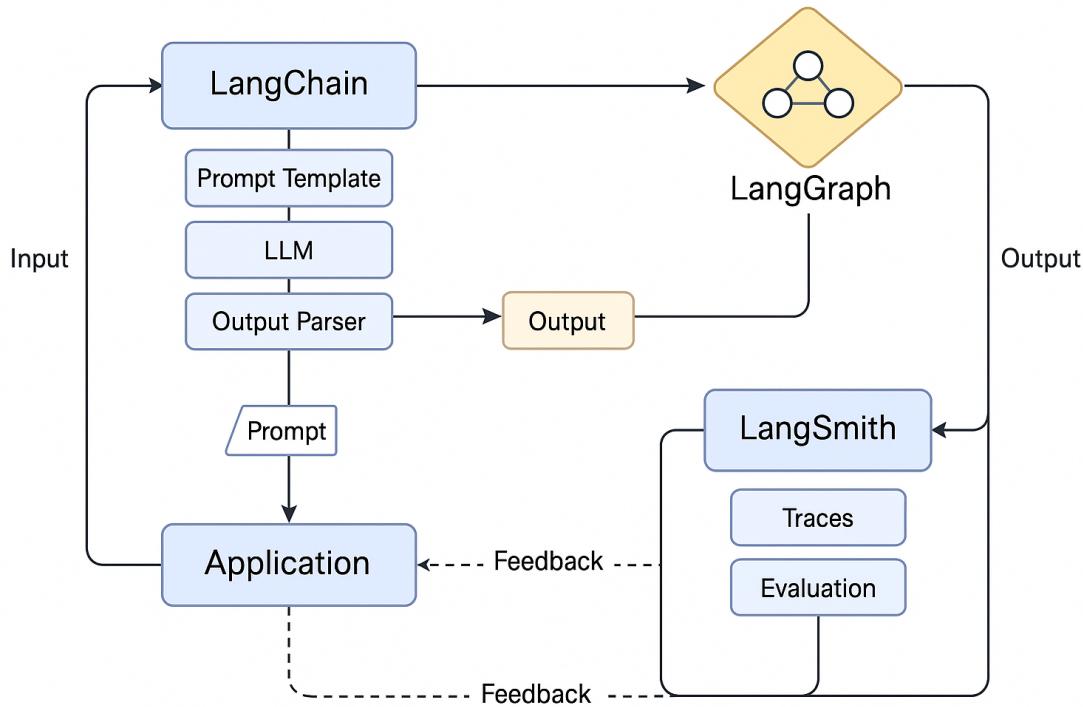


Figura 1. Representación esquemática de un pipeline de IA usando LangChain, LangGraph y LangSmith.

2. ¿Qué es LangChain?

LangChain es un framework de código abierto en Python y JavaScript/TypeScript, diseñado para facilitar la creación de aplicaciones que aprovechan modelos de lenguaje y flujos de datos complejos. LangChain permite integrar LLMs con herramientas, bases de datos, APIs externas y agentes, todo de forma modular y escalable.

Características principales:

- Composición modular de cadenas (chains) y agentes.
- Integración con LLMs de diferentes proveedores (OpenAI, Anthropic, Cohere, etc.).
- Soporte para Retrieval-Augmented Generation (RAG).
- Soporte para memoria conversacional, herramientas externas y workflows complejos.

Arquitectura básica de LangChain

LangChain se estructura en torno a "chains" (cadenas de pasos) y "agents" (agentes que deciden qué acción tomar). Un flujo típico puede incluir carga de documentos, extracción de chunks, embeddings, consultas a vectores y generación de respuestas.

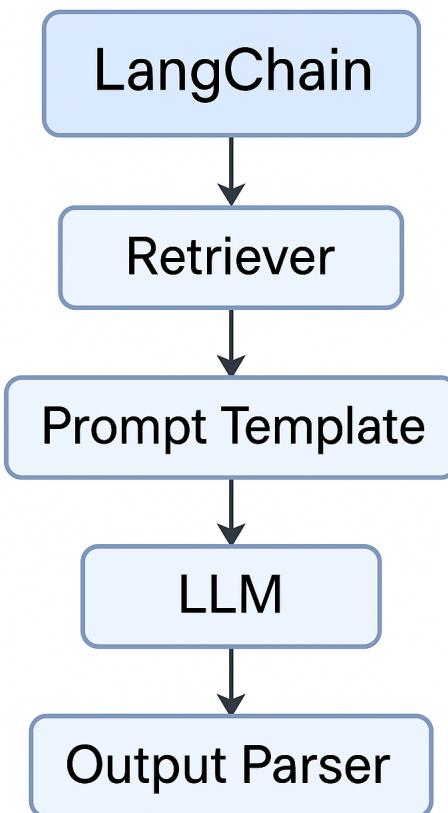


Figura 2. Diagrama conceptual de la arquitectura de LangChain.

Ejemplo básico de código:

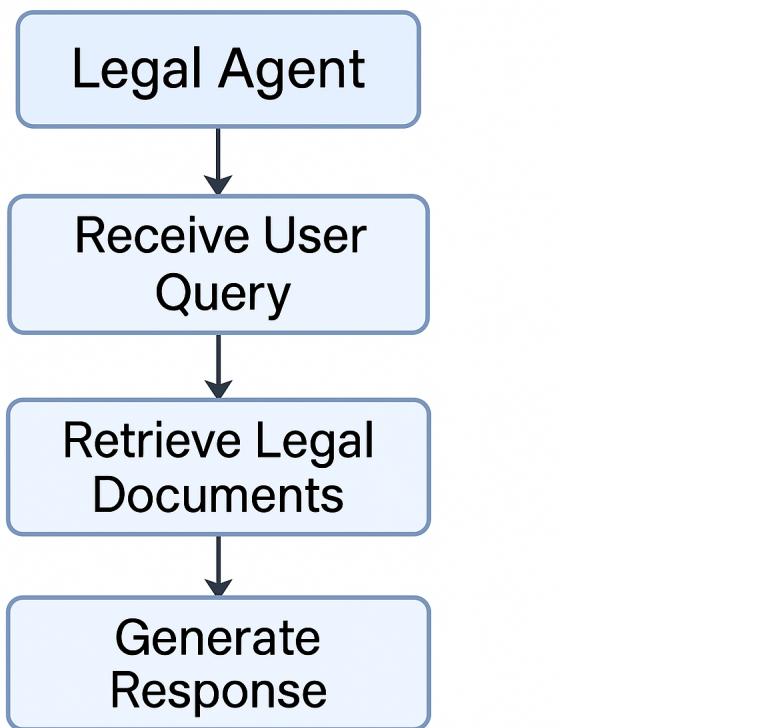
```
from langchain.llms import OpenAI
```

```
from langchain.chains import LLMChain
llm = OpenAI()
chain = LLMChain(llm=llm, prompt="Describe LangChain in simple terms")
result = chain.run("LangChain")
print(result)
```

3. Casos de Uso de LangChain

LangChain es utilizado en una amplia variedad de escenarios, incluyendo:

- Chatbots conversacionales y asistentes virtuales.
- Automatización de flujos legales y análisis de contratos.
- Sistemas de preguntas y respuestas empresariales.
- Extracción y resumen de información de documentos largos.
- Integración con herramientas como Google Search, bases de datos SQL, APIs y más.



Workflow of a Legal Agent
in LangChain

Figura 3. Flujo de trabajo de un agente legal usando LangChain.

Tabla: Ejemplos de Integración

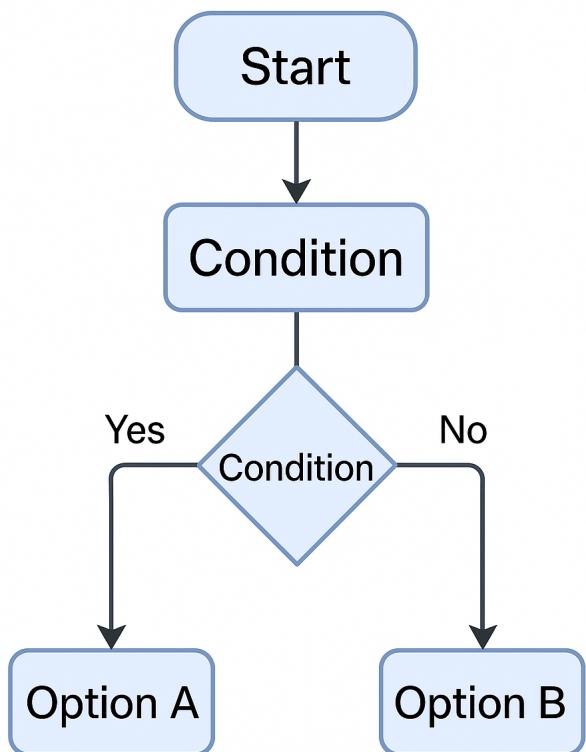
Caso de Uso	Herramientas Integradas
Chatbot Legal	OpenAI, Base de datos, API legal
Asistente de Investigación	Google Search, Arxiv, LLM
Resumen de Documentos	PDF loader, Embeddings, LLM

4. ¿Qué es LangGraph?

LangGraph es una librería creada para facilitar la construcción de agentes conversacionales y flujos de decisión complejos, modelados como grafos. Cada nodo del grafo representa un paso (tool, llamada a LLM, condición, etc.) y las aristas definen el flujo.

LangGraph destaca por:

- Orquestar procesos con múltiples caminos y condiciones.
- Crear agentes capaces de interactuar con múltiples herramientas.
- Visualizar y depurar flujos de trabajo conversacionales.



Decision Graph with
LangGraph

Figura 4. Ejemplo de grafo de decisión con LangGraph.

Sintaxis y ejemplo básico:

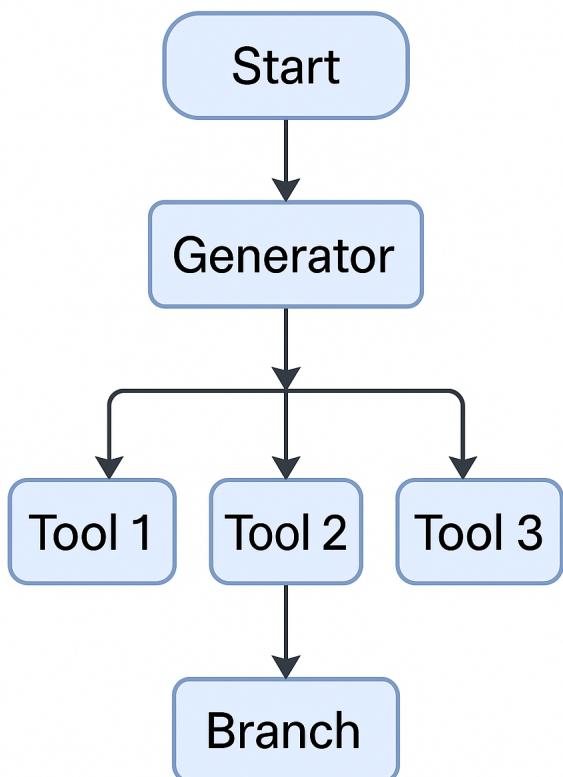
```
import langgraph
graph = langgraph.StateGraph()
graph.add_node("start", some_function)
graph.add_node("end", another_function)
graph.add_edge("start", "end")
graph.set_entry_node("start")
result = graph.run()
```

5. Ejemplo Avanzado con LangGraph

Supongamos que queremos construir un agente que primero analiza el contexto de una pregunta, decide si necesita buscar información externa y luego responde. Cada paso es un nodo, y las decisiones son las aristas.

A continuación, una tabla resumen de nodos y funciones en un flujo típico:

Nodo	Función
Analyze Question	Clasifica la pregunta
Search Tool	Busca información externa
Generate Answer	Genera la respuesta final
↓ el flujo	



Multitool Graph with
LangGraph

Figura 5. Grafo de flujo de agente multitool en LangGraph.

Código simplificado:

```
def analyze_question(input):
    if "legal" in input:
        return "Search Tool"
    return "Generate Answer"
# ... (definir nodos y edges en LangGraph)
```

6. ¿Qué es LangSmith?

LangSmith es una plataforma SaaS de la familia LangChain, orientada al monitoreo, depuración y evaluación de aplicaciones basadas en LLMs. Proporciona herramientas visuales y dashboards para rastrear interacciones, detectar errores, medir calidad y optimizar prompts.

Funcionalidades clave:

- Seguimiento en tiempo real de ejecuciones de chains y agents.
- Evaluación de la calidad de las respuestas generadas.
- Visualización de métricas y logs de uso.
- Detección de cuellos de botella y errores.



54

Total Calls

1.32s

Average Latency

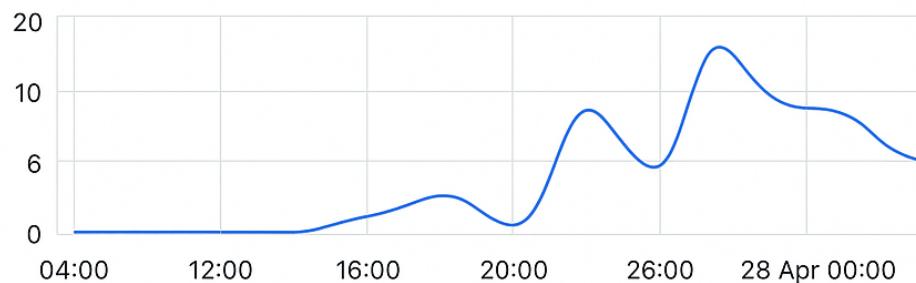
130

Total Tokens

0.83

Average Cost

Calls



Traces

ID	Start Time	Name	Latency
e8fef859	28 Apr 2024, 01:22 PM	agent-run	1.39s
0275222d	28 Apr 2024, 01:22 PM	legalQ...	201ms
e7c41149	28 Apr 2024, 01:22 PM	agent-run	1.56s
e48a676f	28 Apr 2024, 01:22 PM	1.36s	1.86s

Figura 6. Dashboard de LangSmith con métricas y trazas.

Ejemplo de integración:

```
from langsmith import LangSmithTracer  
tracer = LangSmithTracer(project_name="MiApp")  
tracer.start_run("test-chain")  
# Ejecutar chain...
```

```
tracer.end_run()
```

7. Comparación: LangChain, LangGraph y LangSmith

A continuación, se presenta una tabla comparativa entre estas tres herramientas:

Herramienta	Propósito Principal	Nivel
LangChain	Composición de chains/agents	Framework
LangGraph	Orquestación por grafos	Librería
Ventajas y limitaciones	Monitoreo y evaluación	Plataforma SaaS

- LangChain: Modularidad, gran comunidad, soporte para múltiples LLMs, ideal para prototipado rápido.
- LangGraph: Control fino de flujos, visualización de caminos, útil para agentes complejos.
- LangSmith: Visibilidad operacional, útil en producción y para mejorar la calidad.

8. Conclusiones

LangChain, LangGraph y LangSmith permiten desarrollar aplicaciones basadas en LLM de forma más estructurada, escalable y mantenible. Comprender sus diferencias y sinergias es clave para elegir la arquitectura adecuada según el proyecto.

Para agentes conversacionales, LangChain y LangGraph se complementan bien, y LangSmith ayuda a monitorear y mejorar la experiencia final.

9. Recursos Adicionales

- Documentación oficial de LangChain: <https://python.langchain.com>
- Ejemplos de LangGraph: <https://python.langchain.com/docs/langgraph>
- Plataforma LangSmith: <https://smith.langchain.com>
- Repositorio GitHub: <https://github.com/langchain-ai>