Technical Blogs

Building effective agents

2024年12月20日

Over the past year, we've worked with dozens of teams building large language model (LLM) agents across industries. Consistently, the most successful implementations weren't using complex frameworks or specialized libraries. Instead, they were building with simple, composable patterns.

In this post, we share what we've learned from working with our customers and building agents ourselves, and give practical advice for developers on building effective agents.

What are agents?

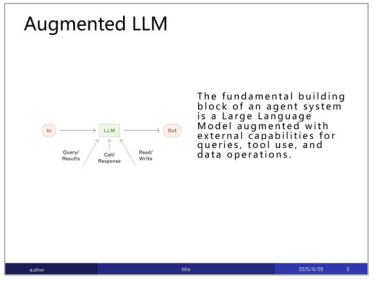
"Agent" can be defined in several ways. Some customers define agents as fully autonomous systems that operate independently over extended periods, using various tools to accomplish complex tasks. Others use the term to describe more prescriptive implementations that follow predefined workflows. At Anthropic, we categorize all these variations as **agentic systems**, but draw an important architectural distinction between **workflows** and **agents**:

- **Workflows** are systems where LLMs and tools are orchestrated through predefined code paths.
- **Agents**, on the other hand, are systems where LLMs dynamically direct their own processes and tool usage, maintaining control over how they accomplish tasks.

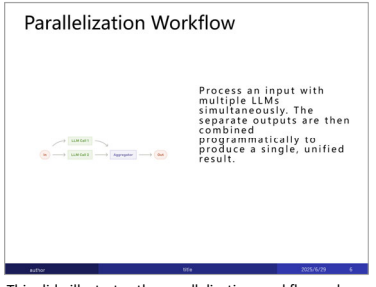
Below, we will explore both types of agentic systems in detail. In Appendix 1 ("Agents in Practice"), we describe two domains where customers have found particular value in using these kinds of systems.

Slides and captions

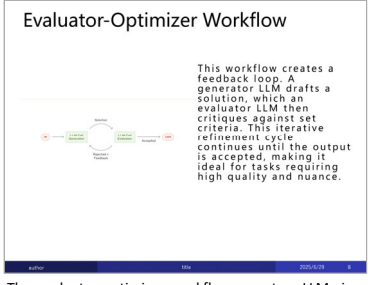
Hello everyone and welcome to the presentation. My name is [Your Name], and I'm delighted to be here with you today to discuss the topic you see on the screen. Let's dive right into our discussion.

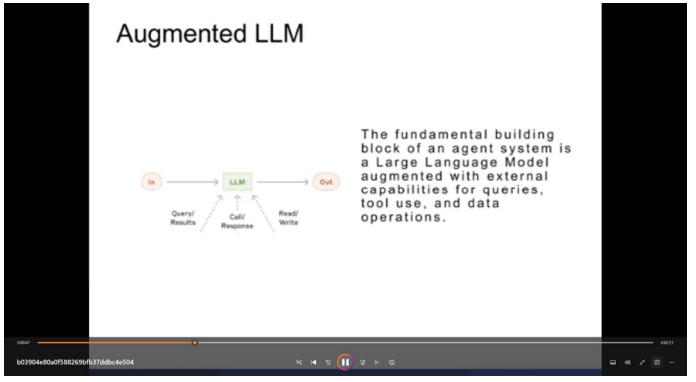
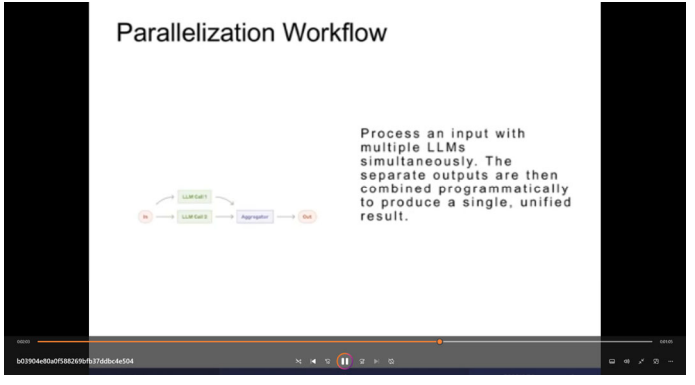


The fundamental building block of an agent system is a Large Language Model augmented with external capabilities. This allows the model to actively query for information, utilize tools, and manage memory, moving beyond simple text generation to dynamic problems ...



This slide illustrates the parallelization workflow, where multiple LLMs process an input simultaneously. We then aggregate their outputs, either by sectioning the task for speed or using a voting system to improve confidence in the final result.



The evaluator-optimizer workflow uses two LLMs in a feedback loop. One generates a response, while a second critiques it for iterative refinement. This process is highly effective for tasks with clear criteria, such as literary translation or complex search.

Videos