




Read in the Substack app

Open app

Agentic Workflows Explained - an in-depth but simple guide for Product Teams

 Google's AI Product Leader says this is one of the essential new skills for PMs that less than 75 people in the world have.

RICH HOLMES

SEP 15, 2025 · PAID




17



4

Share

 *The Knowledge Series* is available for paid subscribers. Get full ongoing access to **80 explainers** and **AI tutorials** to grow your technical knowledge at work. New guides add every month.

Google's Gemini product leader recently [announced](#) a list of what he believes are the essential skills for AI product people in 2025. Unlike traditional software, he argues that LLMs present infinite use cases and infinite failure modes. As a result, it takes a new set of skills to balance the versatility of LLMs with the focus required to build high-quality products.

And in his view, less than 75 people in the world currently possess these skills.

Nobody's quite sure how he came up with the arbitrary figure of 75 people but one of the core skills he mentions is “**an understanding of agentic flows**”. And that's exactly what this Knowledge Series is all about.

We'll take a deeper look at the core steps involved in an agentic workflow, the different frameworks and orchestrators like Langchain that engineers might use along with some real world examples of how companies are using these agentic workflows.

If you're curious about the essential pieces involved in AI agent workflows then this Knowledge Series should help.

Coming up:

- The basic anatomy of an Agentic Workflow
- A step by step look at each part of the important pieces involved in the process
- What is Langchain and what is the role of orchestrators like LangGraph?
- How reliable are agentic workflows?
- A real world example to bring the core concepts to life
- How Replit's new AI coding Agent uses these concepts to help product teams build software
- Tools you can use for measuring the performance of Agentic Workflows



The Knowledge Series

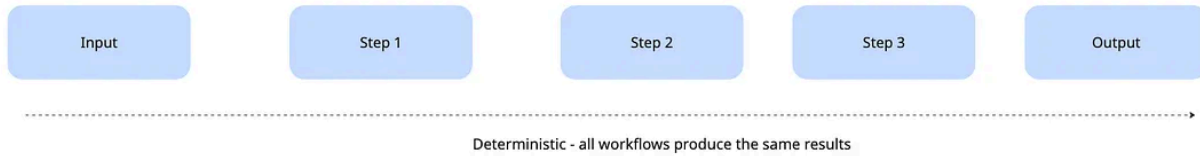
The basic anatomy of an Agentic Workflow

We'll structure this Knowledge Series around the basic anatomy of an agentic workflow. Here's a diagram which shows the difference between a standard, automated workflow and an agentic workflow:

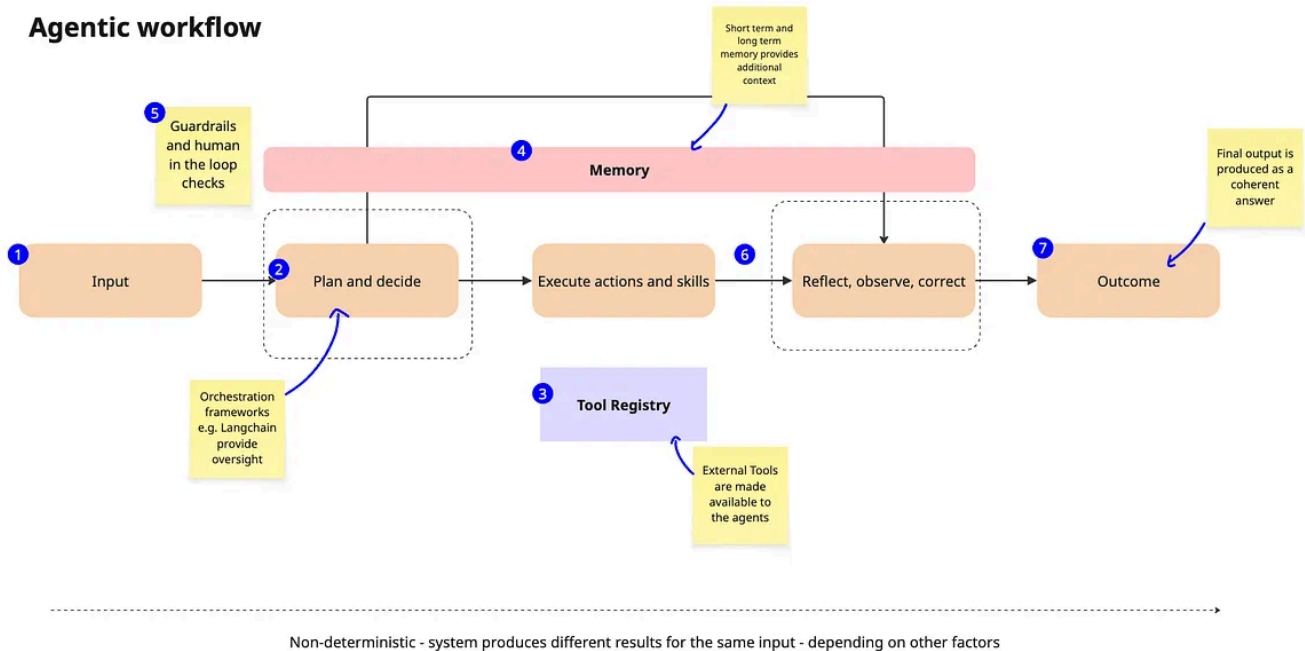
Agentic Workflows Explained

The core pieces in an agentic system explained for product teams

Automated workflow



Agentic workflow



An automated workflow is deterministic; unless a major variable is changed somewhere along the way, the outcome is the same each time the workflow runs. It's predictable, stable but not dynamic or intelligent in any measurable sense.

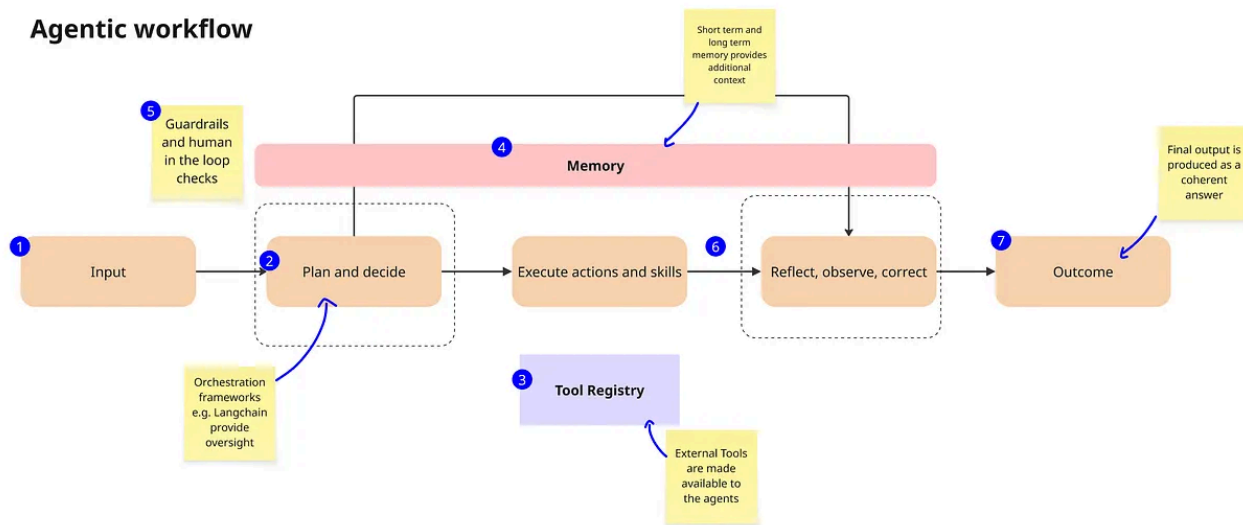
An agentic workflow, on the other hand, is non-deterministic; the system produces different results each time it runs depending on each of the variable factors including the input it receives, the tools it has access to, the decisions it makes and the errors it fixes or doesn't fix.

This contrast between deterministic and non-deterministic is the best way to think about agentic workflows for folks completely new to the concept.

A closer look at each part of the workflow

With a basic understanding under our belt, let's now take a look at each of the different pieces involved in more detail. We'll look at a full, end to end real world example later, but before we do that we'll make sure we fully understand each of the core pieces involved.

The diagram is annotated with numbers 1 through 7 - and we'll follow these numbers throughout to ensure you know what we're referring to.



1. Input

In agentic systems, the input layer is the agent's interface with its environment that triggers the system and the start of the agentic workflow.

For most product teams, this would typically either be the user typing a query command in e.g. "find me the cheapest flight", "where's my order" or an API call from somewhere else.

In simple terms, the input layer is where the system gets its incoming message and the starting point of all other downstream actions, kicking off the workflow. Without an input, the system wouldn't start - and wouldn't know what to do.

2. Planning and orchestration

This is where the system decomposes a complex goal into smaller, actionable tasks. This differentiates agentic systems not just from the basic deterministic flow we referenced at the start but also standard interactions with an LLM; most conversations with LLMs don't require agentic workflows unless you're asking to perform a more complex task.

Tools like Grok and Perplexity will make it clear to a user when a prompt is using an agentic workflow since it will clearly show the thought process in a transparent and traceable format.

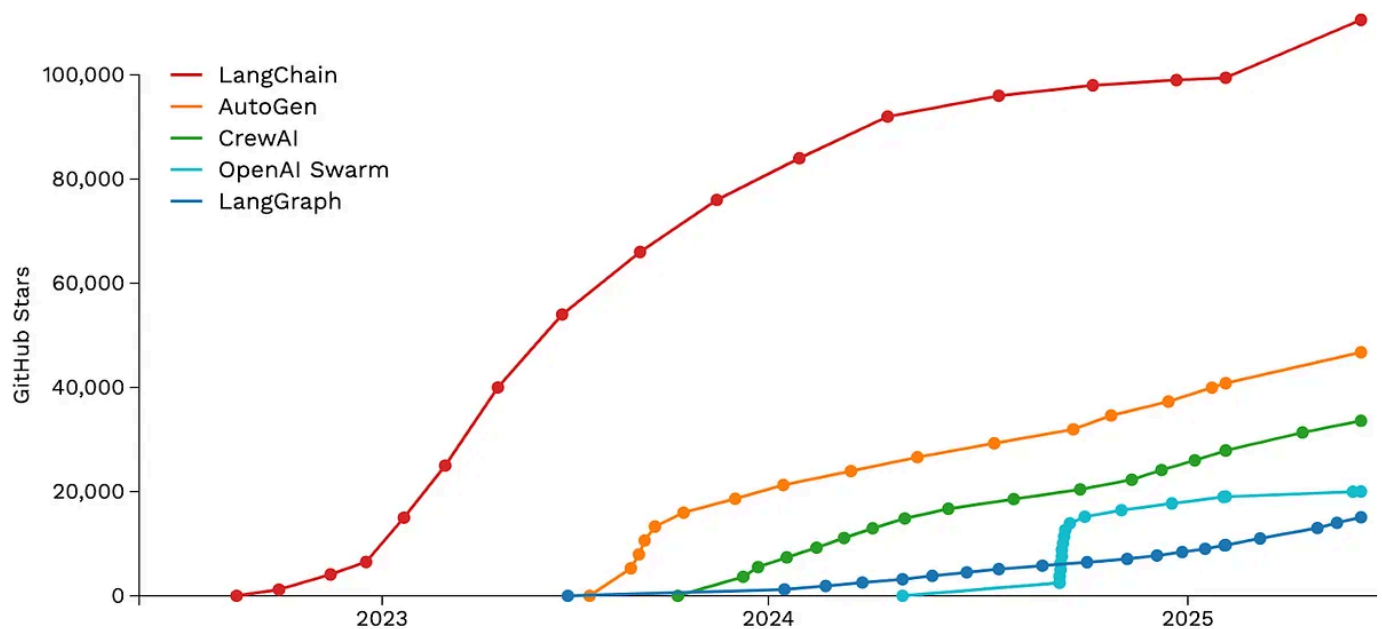
During this phase, the agent analyzes the objective, breaks it down into sub-tasks and creates a logical roadmap for executing. Each subtask is sequenced appropriately during this stage and

Orchestration is essentially the control layer that manages, monitors, and sequences workflow execution, integrating multiple agents, APIs, databases, or human participants if required. More on that later.

Many product teams now rely on frameworks to support this orchestration.

Orchestration frameworks explained

One of the most popular tools for orchestration is Langchain (alongside its product LangGraph). Here's a snapshot of the most popular orchestration tools used by engineers ranked based on the number of stars in GitHub:



As you can see, Langchain is far ahead of the rest. But what exactly is it?

What is Langchain?

LangChain is an open-source orchestration framework designed for developing applications that use large language models (LLMs). It supports both Python and JavaScript and provides a generic interface for nearly any LLM, allowing developers to choose and combine different models.

It is founded by an ex-Airbnb engineer and is now valued at over \$1 billion.

Here's a one-pager on it that covers a lot of the essentials that are worth knowing for product teams:

Hi leguntan@gmail.com

This post is for paid subscribers

[Upgrade to paid](#)

Already a paid subscriber? [Switch accounts](#)

[← Previous](#)

[Next →](#)