When (and When Not) to Use AI Agents

Estimated time: **5** minutes

Why evaluating whether to use an agent matters

AI agents offer powerful capabilities, but they aren't always the best solution. In this reading, you'll learn how to evaluate when agents make sense—and when simpler tools are more effective. You'll explore real-world decision frameworks and understand how to balance innovation with practicality.

Learning Objectives

By the end of this reading, you will be able to:

- Identify where agents fit on the AI system spectrum.
- Apply a four-step decision framework to determine when to use agents.
- Recognize scenarios where agents are a poor fit.
- Describe current limitations of agent technology.
- Apply guidelines for managing risk when deploying agents.
- Understand the key elements of effective agent architecture.

The AI system spectrum

Not all AI systems operate at the same level of complexity. The following table describes the spectrum of differences.

Type	Description	Best Use Cases
Simple AI Features	Perform tasks like classification or text summarization	Fast, repeatable tasks with clear outputs
Orchestrated Workflows	Predefined multi-step logic combining different AI features	Structured processes like document review or routing
Autonomous Agents	Make decisions independently and adapt to new information	Complex reasoning, exploration, or strategy tasks

Next, learn about the four criteria framework you can use to evaluate when to use agents.

The four-criteria framework for using agents

Before you build or deploy an AI agent, ask yourself the following questions:

1. Is the task ambiguous or predictable?

Use *agents* when the task is ambiguous:

- The decision path is unclear or cannot be mapped in advance
- Tasks involve exploration, troubleshooting, or creativity

Use *workflows* when the task is predictable:

- You can define all rules and outcomes
- The process follows a clear, repeatable structure

2. Is the value of the task worth the cost?

AI agents are more expensive to operate due to exploration overhead. They can consume **10 to 100× more tokens** than a workflow. Let's look at some scenarios:

Scenario	Recommendation
Strategic planning with high ROI	Use an agent
Basic customer support task	Use a workflow instead

3. Does the agent meet minimum capabilities?

Before launch, test the agent on three to five key skills.

Here are some examples:

- A research agent must identify, filter, and summarize credible sources
- A coding agent must write, fix, and validate code snippets
- A customer support agent must classify issues, resolve common queries, and escalate complex cases appropriately
- A data analysis agent must clean datasets, detect anomalies, and summarize key trends

If the agent fails these tests, **scale back or redesign the agent**.

4. What happens if the agent makes a mistake?

Evaluate the answers to these questions

- Can you catch and correct errors quickly? If so, then using an agent might be appropriate.
- What's the risk if something is missed? Does the consequence of missing the answer affect the customer's or organization's well-being or safety?
- Does the agent include built-in correction or validation tools?
- Use agents when risk is manageable or reversible.

Current AI agent challenges

Even powerful agents have challenges.

Challenge	Why It Matters
Reasoning inconsistency	Agents may succeed once but fail on similar tasks
Unpredictable costs	Resource use can spike depending on complexity
Tool integration issues	Agents need well-integrated tools and stable APIs

When *not* to use agents

There are some situations when agents are not a good solution. Avoid agents for:

- High-volume, low-margin tasks, such as basic chat support
- Real-time applications, such as instant fraud detection
- Zero-error systems, including medical or security decisions
- Heavily regulated industries need deterministic outcomes

Guidelines for managing risk when deploying agents

Let's review some helpful tips for effective agent architecture.

Prioritize key elements of effective agent architecture

Keep the agent architecture simple. Every agent relies on three key architectural components:

- **Environment**: The digital space where the agent operates
- **Tools**: The interfaces the agent uses to act or observe
- **System Prompts**: The rules, goals, and behaviors that guide the agent's operation

Key takeaway:
Start simple with your agent's actions. Add task complexity only after confirming the agent's reliable performance.

Assessing your deployment plan

You'll want to assess your deployment plan to assess and mitigate the risks associated with agent implementations

If the risk Level is . . .	Implement the following response strategy . . .
High-stakes and difficult to notice	Use human review and multiple validation layers
High-stakes and visible	Add automated checks and oversight mechanisms
Low-stakes	Monitor the agent's actions user feedback and lightweight validation

Best practices:

When starting to use agents, begin with these best practices:

- Start with **read-only access** to tools and systems
- Add **human approvals** for critical steps
- Use **staged deployments** with monitoring
- Enable **comprehensive logging**

Next, learn about implementing agents responsibly.

Implementing agents responsibly

When implementing agents, consider a phased deployment plan that consists of the following steps:

1. **Validate the Proof of Concept**: Try low-risk, reversible tasks
2. **Implement a Pilot Program**: Test the agent using moderate-risk tasks under supervision
3. **Production Scaling**: Expand use of the agent only after demonstrating its safety and performance

Looking ahead: What will improve?

Agents continue to evolve. Expect future improvements in:

- More consistent reasoning
- Smarter, leaner architectures
- Advanced monitoring and error detection tools

But remember: even with improvements, **thoughtful deployment and risk analysis remain essential.**

Now, let's review the key points of what you've learned.

Recap

You now know that:

- **AI agents** sit at the highest end of the AI complexity spectrum, excelling at tasks that require autonomous decision-making, adaptation, and strategy.
  Use the four-step decision framework—task ambiguity, step flexibility, tool variety, and failure impact—to decide if an agent is the right fit for the task.
- **Avoid using agents** for simple, repeatable, or high-risk tasks where errors are costly or predictable; tools can perform better.
- **Today's agents** struggle with reliability, high compute costs, and often need human oversight to avoid hallucinations or missteps.
- **Manage risk** by setting boundaries, using logs, monitoring outcomes, and keeping a human-in-the-loop for oversight.
- **Effective agent architecture** includes modular components like memory, tool use, planning strategies, and clear reasoning paths.

Author

[Tenzin Migmar]

Contributors

[Faranak Heidari]