MASTERING AI AGENTS

A comprehensive guide to evaluating AI agents

Galileo

Preface

In our previous e-book, "Mastering RAG," our goal was clear: building enterprise-grade RAG systems, productionizing them, monitoring their performance, and improving them. At the core of it, we understood how RAG systems enhance an LLM's ability to work with specific knowledge by providing relevant context.

In this e-book, we're taking a step further and asking, "How do we use LLMs to accomplish end-to-end tasks?" This singular question opens up a door: AI agents. A RAG system helps an LLM provide accurate answers based on given context. An AI agent takes that answer and actually does something with it — makes decisions, executes tasks, or coordinates multiple steps to achieve a goal.

A RAG-enhanced LLM could help answer questions about policy details by pulling relevant information. But an AI agent could actually process the claim end-to-end by analyzing the documentation, checking policy compliance, calculating payments, and even coordinating with other systems or agents when needed.

The ideas behind agents has existed for years. It can be a software program or another computational entity that can accept input from its environment and take actions based on rules. With AI agents, you're getting what has never been there before: the ability to understand the context without predefined rules, the capacity to tune decisions based on context, and learning from every interaction. What you're getting is not just a bot working with a fixed set of rules but a system capable of making advanced decisions in real-time.

Companies have quickly adapted, adopted, and integrated AI agents into their workflows. Capgemini's research found that "10% of organizations already use AI agents, more than half plan to use them in 2025 and 82% plan to integrate them within the next three years."

This e-book aims to be your go-to guide for all things AI agents. If you're a leader looking to guide your company to build successful agentic applications, this e-book can serve as a great guide to get you started. We also explore approaches to measuring how well your AI agents perform, as well as common pitfalls you may encounter when designing, measuring, and improving them.

The book is divided into five chapters:

Chapter 1 introduces AI agents, their optimal applications, and scenarios where they might be excessive. It covers various agent types and includes three real-world use cases to illustrate their potential.

Chapter 2 details three frameworks—LangGraph, Autogen, and CrewAI—with evaluation criteria to help choose the best fit. It ends with case studies of companies using these frameworks for specific AI tasks.

Chapter 3 explores the evaluation of an AI agent through a step-by-step example of a finance research agent.

Chapter 4 explores how to measure agent performance across systems, task completion, quality control, and tool interaction, supported by five detailed use cases.

Chapter 5 addresses why many AI agents fail and offers practical solutions for successful AI deployment.

We hope this book will be a great stepping stone in your journey to build trustworthy agentic systems.

- Pratik Bhavsar

Contents

Chapter 1:
What are AI agents

7/27

Types of AI Agents 10

When to Use Agents? 21

When Not to Use Agents? 22

10 Questions to Ask Before You Consider an AI Agent 23

3 Interesting Real-World Use Cases of AI Agents 25

Chapter 2:
Frameworks for Building Agents

28/43

LangGraph vs. AutoGen vs. CrewAI 30

Practical Considerations 31

What Tools and Functionalities Do They Support? 31

How Well Do They Maintain the Context? 32

Are They Well-Organized and Easy to Interpret? 33

What's the Quality of Documentation? 34

Do They Provide Multi-Agent Support? 34

What About Caching? 35

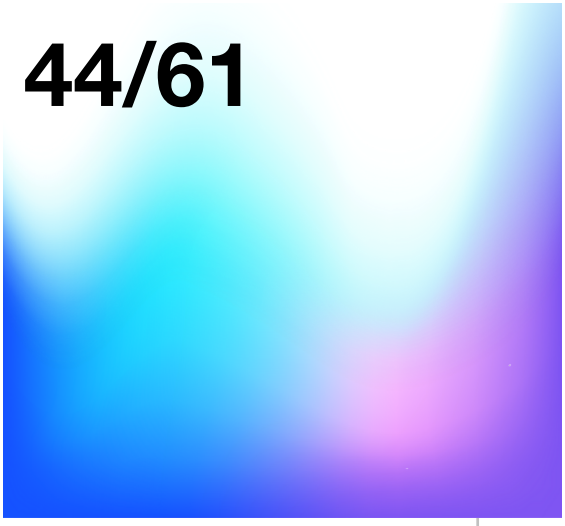
Looking at the Replay Functionality 35

What About Code Execution? 35

Human in the Loop Support? 37

Popular Use Cases Centered Around These Frameworks 40

Chapter 3:
How to Evaluate Agents



Requirements	44
Defining the Problem	44
Define the React Agent	45
State Management	46
Create the Graph	47
Create the LLM Judge	54
Use Galileo Callbacks	55

Chapter 4:
Metrics for Evaluating AI Agents

Case Study 1: Advancing the Claims Processing Agent	63
Case Study 2: Optimizing the Tax Audit Agent	66
Case Study 3: Elevating the Stock Analysis Agent	69
Case Study 4: Upgrading the Coding Agent	72
Case Study 5: Enhancing the Lead Scoring Agent	75

Chapter 5:
Why Most AI Agents Fail & How to Fix Them

Development Issues	81
LLM Issues	82
Production Issues	86

01
CHAPTER

WHAT ARE AI AGENTS?

What are AI agents?

Let's start by understanding what AI agents are and which tasks you should use them for to maximize their potential.

AI agents are software applications that use large language models (LLMs) to autonomously perform specific tasks, ranging from answering research questions to handling backend services. They're incredibly useful for tasks that demand complex decision-making, autonomy, and adaptability. You might find them especially helpful in dynamic environments where the workflow involves multiple steps or interactions that could benefit from automation.

Salesforce estimates that salespersons spend 71% of their time on non-selling tasks (like administrative tasks and manually entering data). Imagine the time that could have gone into directly engaging with customers, developing deeper relationships, and ultimately closing more sales. This is true across multiple domains and applications: finance, health care, tech, marketing, sales, and more.

Let's use an example to understand this better. Imagine you run an online retail business and receive hundreds of customer inquiries every day about order statuses, product details, and shipping information. Instead of answering each and every query yourself, you can integrate an AI agent into your solution to handle these queries.

Here's how it would typically work:

1. **Customer Interaction**
A customer messages your service asking, "When will my order ship?"

2. **Data Retrieval**
The AI agent accesses the order management system to find the specific order details.

3. **Response Generation**
Based on the data retrieved, the agent automatically provides an updates to the customer, such as sending "Your order will ship tomorrow and you'll receive a tracking link via email once it's on its way."

The return to having an AI agent is multifold here:

- Super quick response time that keeps your customers happy
- Frees up your human staff to handle more complex queries and issues
- Improves your overall productivity and efficiency

Fig 1.1 is an example of how agents are leveraged for code generation.

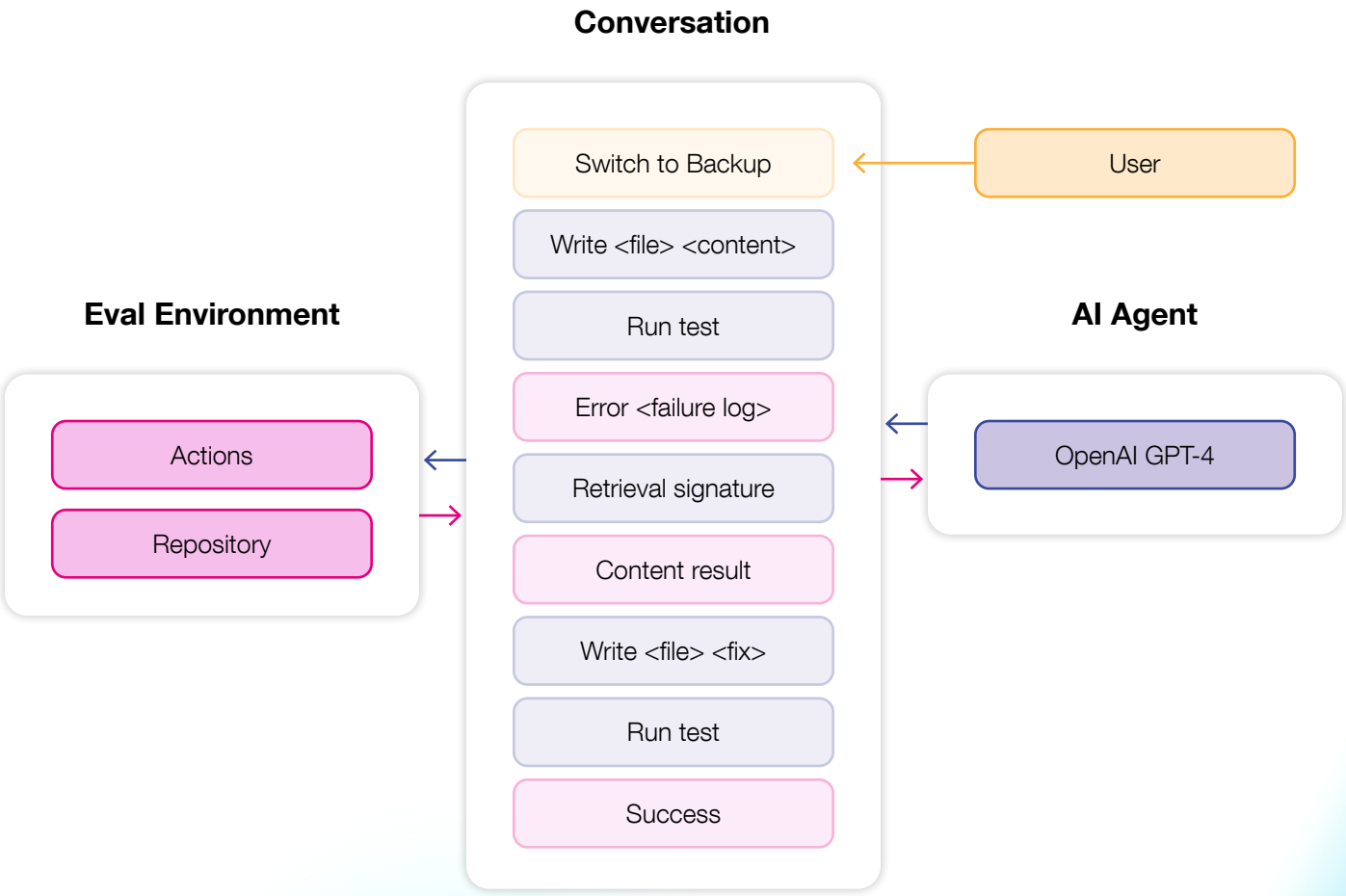


Fig 1.1: Automated AI-Driven Development using AI agents

Types of AI Agents

Now that we're familiar with what AI agents are, let's look at different types of AI agents along with their characteristics, examples, and when you can use them. See **Table 1.1** below to get a quick idea of the types of AI agents and where and when you can use them.

Name of the agent	Key Characteristics	Examples	Best For
Fixed Automation: The Digital Assembly Line	No intelligence, predictable behavior, limited scope	RPA, email autoresponders, basic scripts	Repetitive tasks, structured data, no need for adaptability
LLM-Enhanced: Smarter, but Not Einstein	Context-aware, rule-constrained, stateless	Email filters, content moderation, support ticket routing	Flexible tasks, high-volume/low-stakes, cost-sensitive scenarios
ReAct: Reasoning Meets Action	Multi-step workflows, dynamic planning, basic problem-solving	Travel planners, AI dungeon masters, project planning tools	Strategic planning, multi-stage queries, dynamic adjustments
ReAct + RAG: Grounded Intelligence	External knowledge access, low hallucinations, real-time data	Legal research tools, medical assistants, technical support	High-stakes decisions, domain-specific tasks, real-time knowledge needs
Tool-Enhanced: The Multi-Taskers	Multi-tool integration, dynamic execution, high automation	Code generation tools, data analysis bots	Complex workflows requiring multiple tools and APIs
Self-Reflecting: The Philosophers	Meta-cognition, explainability, self-improvement	Self-evaluating systems, QA agents	Tasks requiring accountability and improvement
Memory-Enhanced: The Personalized Powerhouses	Long-term memory, personalization, adaptive learning	Project management AI, personalized assistants	Individualized experiences, long-term interactions
Environment Controllers: The World Shapers	Active environment control, autonomous operation, feedback-driven	AutoGPT, adaptive robotics, smart cities	System control, IoT integration, autonomous operations
Self-Learning: The Evolutionaries	Autonomous learning, adaptive/scalable, evolutionary behavior	Neural networks, swarm AI, financial prediction models	Cutting-edge research, autonomous learning systems

Table 1.1: Types of agents and their characteristics

Fixed Automation – The Digital Assembly Line

This level of AI agents represents the simplest and most rigid form of automation. These agents don't adapt or think—they just execute pre-programmed instructions. They are like assembly-line workers in a digital factory: efficient but inflexible. Great for repetitive tasks, but throw them a curveball, and they'll freeze faster than Internet Explorer.
(See **Table 1.2** below)

Feature	Description
Intelligence	No learning, adaptation, or memory.
Behavior	Predictable and consistent, follows pre-defined rules.
Scope	Limited to repetitive, well-defined tasks. Struggles with unexpected scenarios.
Best Use Cases	Routine tasks, structured data, situations with minimal need for adaptability.
Examples	RPA for invoice processing, email autoresponders, basic scripting tools (Bash, PowerShell).

Table 1.2: Characteristics of a fixed automation agent

The fixed automation workflow (See **Fig 1.2**) follows a simple, linear path. It begins when a specific input (like a file or data) triggers the system, which consults its predefined rulebook to determine what to do. Based on these rules, it executes the required action and finally sends out the result or output. Think of it as a digital assembly line where each step must be completed in exact order, without deviation.

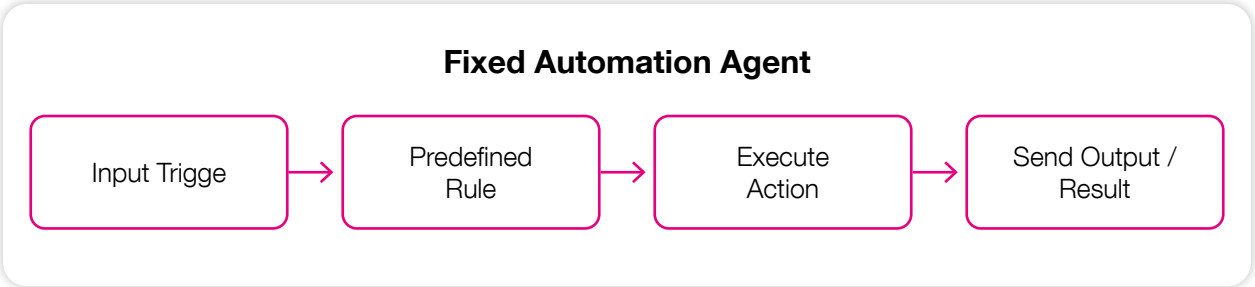


Fig 1.2: Workflow of a fixed automation agent

LLM-Enhanced – Smarter, but Not Exactly Einstein

These agents leverage LLMs to provide contextual understanding and handle ambiguous tasks while operating within strict boundaries. LLM-Enhanced Agents balance intelligence and simplicity, making them highly efficient for low-complexity, high-volume tasks. Take a look at their features below in **Table 1.3**.

Feature	Description
Intelligence	Context-aware; leverages LLMs to process ambiguous inputs with contextual reasoning.
Behavior	Rule-constrained; decisions are validated against predefined rules or thresholds.
Scope	Stateless; no long-term memory; each task is processed independently.
Best Use Cases	Tasks requiring flexibility with ambiguous inputs, high-volume/low-stakes scenarios, and cost-sensitive situations where "close enough" is sufficient.
Examples	Email filters, AI-enhanced content moderation, customer support classification.

Table 1.3: Characteristics of an LLM-enhanced agent

The workflow below **(Fig 1.3**) shows how these smarter agents process information: starting with the input, the agent uses LLM capabilities to analyze and understand the input context. This analysis then passes through rule-based constraints that keep the agent within defined boundaries, producing an appropriate output. It's like having a smart assistant who understands context but still follows company policy before making decisions.

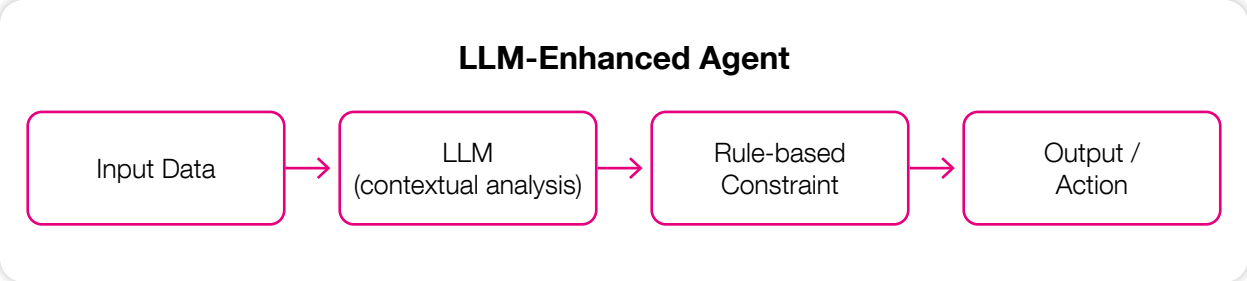


Fig 1.3: Workflow of a LLM-enhanced agent

ReAct – Reasoning Meets Action

ReAct agents combine Reasoning and Action to perform tasks that involve strategic thinking and multi-step decision-making. They break complex tasks into manageable steps, reasoning through problems dynamically and acting based on their analysis. These agents are like your type-A friend who plans their weekend down to the minute. **Table 1.4** lists their characteristics.

Feature	Description
Intelligence	Reasoning and action; mimics human problem-solving by thinking through a problem and executing the next step.
Behavior	Handles multi-step workflows, breaking them down into smaller, actionable parts. Dynamically adjusts strategy based on new data.
Scope	Assists with basic open-ended problem-solving, even without a direct solution path.
Best Use Cases	Strategic planning, multi-stage queries, tasks requiring dynamic adjustments, and re-strategizing.
Examples	Language agents solving multi-step queries, AI Dungeon Masters, project planning tools.

Table 1.4: Characteristics of a fixed ReAct agent

The ReAct workflow starts with an Input Query and then enters a dynamic cycle between the Reasoning and Action Phase, as you'll see in **Fig 1.4**. Unlike simpler agents, it can loop between thinking and acting repeatedly until the desired outcome is achieved before producing the final Output/Action. Think of it as a problem solver that keeps adjusting its approach - analyzing, trying something, checking if it worked, and trying again if needed.

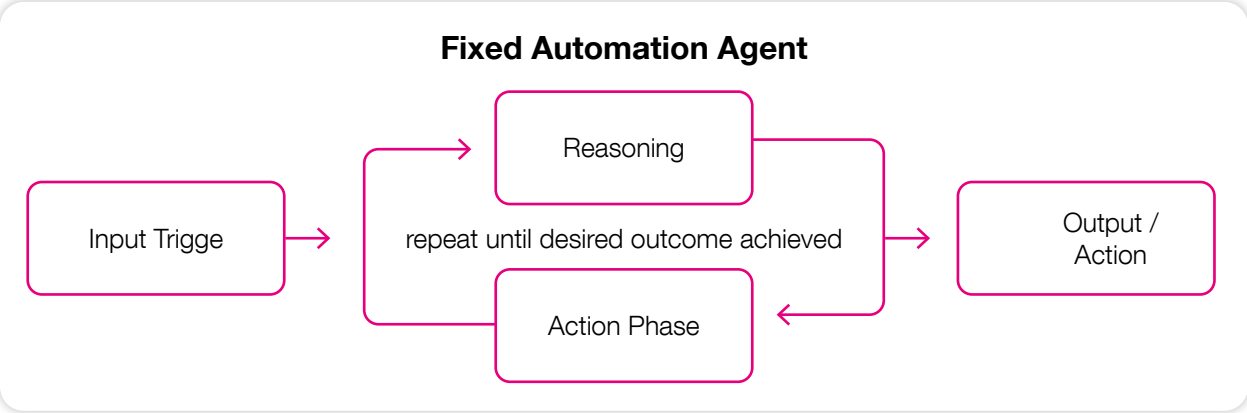


Fig 1.4: Workflow of a ReAct agent

ReAct + RAG – Grounded Intelligence

Now, moving on to agents who are much more intelligent, we come to ReAct + RAG agents that combine reasoning, action, and real-time access to external knowledge sources. This integration allows them to make informed decisions grounded in accurate, domain-specific data, making them ideal for high-stakes or precision-critical tasks (especially when you add evaluations). These agents are your ultimate trivia masters with Google on speed dial. See **Table 1.5** to learn how this agent works.

Feature	Description
Intelligence	Employs a RAG workflow, combining LLMs with external knowledge sources (databases, APIs, documentation) for enhanced context and accuracy.
Behavior	Uses ReAct-style reasoning to break down tasks, dynamically retrieving information as needed. Grounded in real-time or domain-specific knowledge.
Scope	Designed for scenarios requiring high accuracy and relevance, minimizing hallucinations.
Best Use Cases	High-stakes decision-making, domain-specific applications, tasks with dynamic knowledge needs (e.g., real-time updates).
Examples	Legal research tools, medical assistants referencing clinical studies, technical troubleshooting agents.

Table 1.5: Characteristics of a ReAct + RAG agent

Starting with an Input Query, this advanced workflow combines ReAct's reasoning-action loop with an additional Knowledge Retrieval step. The agent cycles between Reasoning, Action Phase, and Knowledge Retrieval (See **Fig 1.5**) — consulting external sources as needed — until it reaches the desired outcome and produces an Output/Action. It's like having a problem solver who not only thinks and acts but also fact-checks against reliable sources along the way.

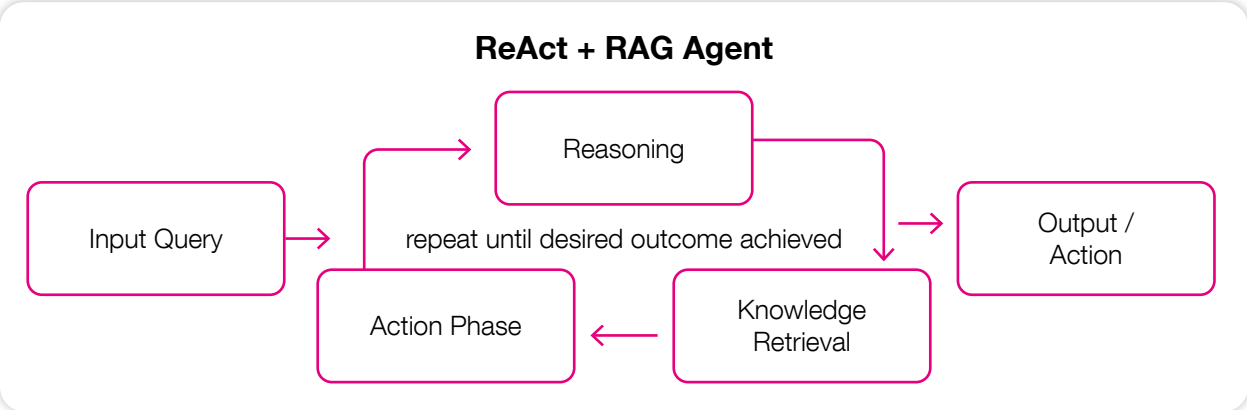


Fig 1.5: Workflow of a ReAct + RAG agent

Tool-Enhanced – The Multi-Taskers

Tool-enhanced agents are versatile problem solvers that integrate multiple tools, leveraging APIs, databases, and software to handle complex, multi-domain workflows. They combine reasoning, retrieval, and execution for seamless, dynamic task completion. Think of them as tech-savvy Swiss Army knives capable of combining reasoning, retrieval, and execution seamlessly! (See **Table 1.6**)

Feature	Description
Intelligence	Leverages APIs, databases, and software tools to perform tasks, acting as a multi-tool integrator.
Behavior	Handles multi-step workflows, dynamically switching between tools based on task requirements.
Scope	Automates repetitive or multi-stage processes by integrating and utilizing diverse tools.
Best Use Cases	Jobs requiring diverse tools and APIs in tandem for complex or multi-stage automation.
Examples	Code generation tools (GitHub CoPilot, Sourcegraph's Cody, Warp Terminal), data analysis bots combining multiple APIs.

Table 1.6: Characteristics of tool-enhanced agents

Starting with an Input Query, the agent combines reasoning with a specialized tool loop. After the initial reasoning phase, it selects the appropriate tool for the task (Tool Selection) and then executes it (Tool Execution). This cycle repeats until the desired outcome is achieved, leading to the final Output/Action. (See **Fig 1.6**)

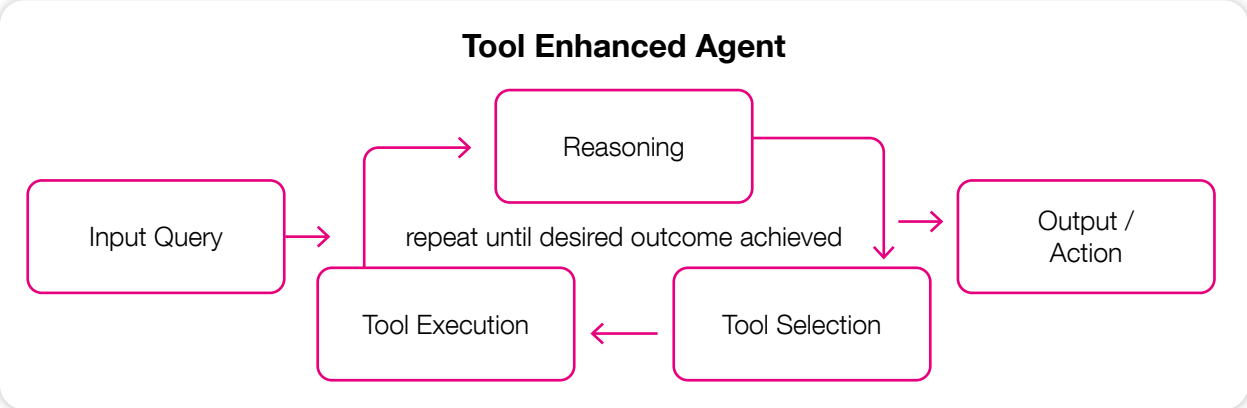


Fig 1.6: Workflow of tool-enhanced agents

Self-Reflecting – The Philosophers

These agents think about their thinking. Self-reflecting agents introduce meta-cognition—they analyze their reasoning, assess their decisions, and learn from mistakes. This enables them to solve tasks, explain their reasoning, and improve over time, ensuring greater reliability and accountability. (See **Table 1.7**)

Feature	Description
Intelligence	Exhibits meta-cognition, evaluating its own thought processes and decision outcomes.
Behavior	Provides explanations for actions, offering transparency into its reasoning. Learns from mistakes and improves performance over time.
Scope	Suited for tasks requiring accountability and continuous improvement.
Best Use Cases	Quality assurance, sensitive decision-making where explainability and self-improvement are crucial.
Examples	AI that explains its reasoning, self-evaluating learning systems, quality assurance (QA) agents.

Table 1.7: Characteristics of self-reflecting agents

Starting with an Input Query, the agent goes through a cycle of Reasoning and Execution, but with a crucial additional step: Reflection. After each execution, it reflects on its performance and feeds those insights back into its reasoning process. This continuous loop of thinking, doing, and learning continues until the desired outcome is achieved, producing the final Output/Action. This is evident in **Fig 1.7**.

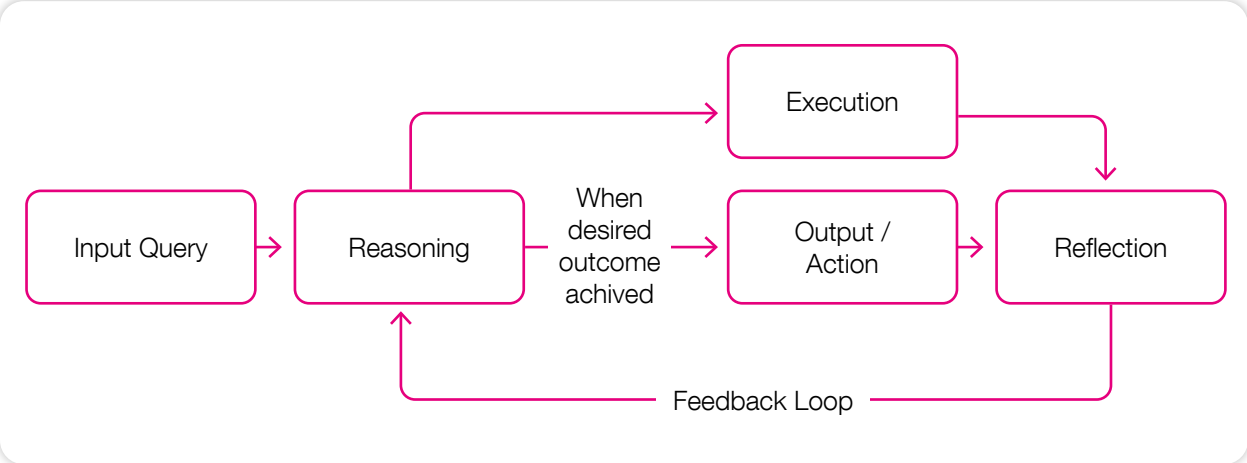
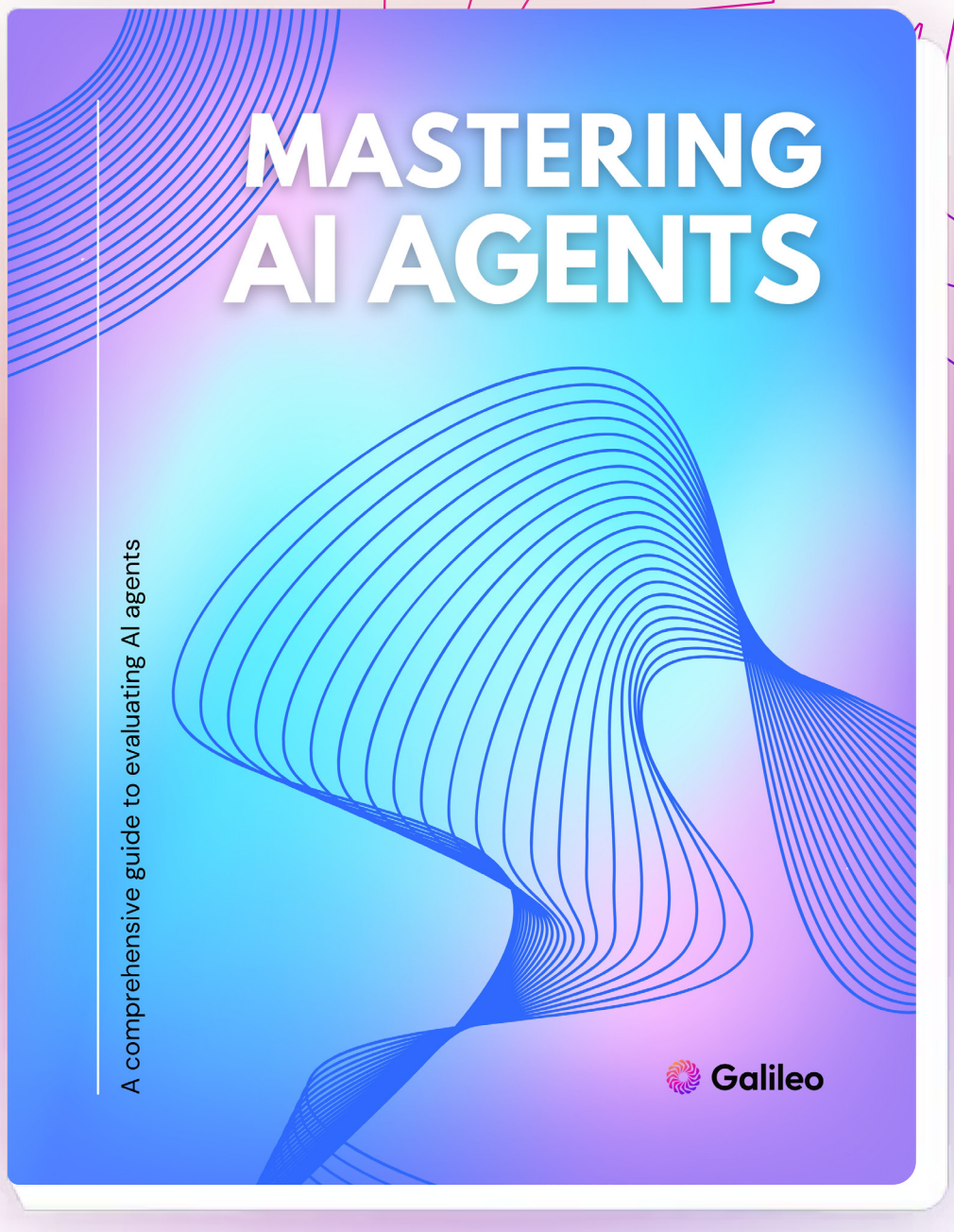


Fig 1.7: Workflow of self-reflecting agents

Download the Full ebook



Galileo