# What are AI agents?

## What are AI agents?

An artificial intelligence (AI) agent is a system that autonomously performs tasks by designing workflows with available tools.

AI agents can encompass a wide range of functions beyond natural language processing including decision-making, problem-solving, interacting with external environments and performing actions.

AI agents solve complex tasks across enterprise applications, including software design, IT automation, code generation and conversational assistance. They use the advanced natural language processing techniques of large language models (LLMs) to comprehend and respond to user inputs step-by-step and determine when to call on external tools.

## How AI agents work

At the core of AI agents are large language models (LLMs). For this reason, AI agents are often referred to as LLM agents. Traditional LLMs, such as IBM® Granite® models, produce their responses based on the data used to train them and are bounded by knowledge and reasoning limitations. In contrast, agentic technology uses tool calling on the backend to obtain up-to-date information, optimize workflows and create subtasks autonomously to achieve complex goals.

In this process, the autonomous agent learns to adapt to user expectations over time. The agent's ability to store past interactions in memory and plan future actions encourages a personalized experience and comprehensive responses.[1] This tool calling can be achieved without human intervention and broadens the possibilities for real-world applications of these AI systems. These three stages or agentic components define how agents operate:

### Goal initialization and planning

Although AI agents are autonomous in their decision-making processes, they require goals and predefined rules defined by humans.[2] There are three main influences on autonomous agent behavior:

- The team of developers that design and train the agentic AI system.
- The team that deploys the agent and provides the user with access to it.
- The user that provides the AI agent with specific goals to accomplish and establishes available tools to use.

Given the user's goals and the agent’s available tools, the AI agent then performs task decomposition to improve performance.[3] Essentially, the agent creates a plan of specific tasks and subtasks to accomplish the complex goal.

For simple tasks, planning is not a necessary step. Instead, an agent can iteratively reflect on its responses and improve them without planning its next steps.

## Reasoning with available tools

AI agents base their actions on the information that they perceive. However, they often lack the full knowledge required to tackle every subtask within a complex goal. To bridge this gap, they turn to available tools such as external datasets, web searches, APIs and even other agents.

Once the missing information is gathered, the agent updates its knowledge base and engages in agentic reasoning. This process involves continuously reassessing its plan of action and making self-corrections, which enables more informed and adaptive decision-making.

To help illustrate this process, imagine a user planning their vacation. The user tasks an AI agent with predicting which week in the next year would likely have the best weather for their surfing trip in Greece.

Because the LLM model at the core of the agent does not specialize in weather patterns, it cannot rely solely on its internal knowledge. Therefore, the agent gathers information from an external database containing daily weather reports for Greece over the past several years.

Despite acquiring this new information, the agent still cannot determine the optimal weather conditions for surfing and so, the next subtask is created. For this subtask, the agent communicates with an external agent that specializes in surfing. Let's say that in doing so, the agent learns that high tides and sunny weather with little to no rain provide the best surfing conditions.

The agent can now combine the information it has learned from its tools to identify patterns. It can predict which week next year in Greece will likely have high tides, sunny weather and a low chance of rain. These findings are then presented to the user. This sharing of information between tools is what allows AI agents to be more general purpose than traditional AI models.[3]

## Learning and reflection

AI agents use feedback mechanisms, such as other AI agents and human-in-the-loop (HITL) to improve the accuracy of their responses. Let's return to our previous surfing example to highlight this process. After the agent forms its response to the user, it stores the learned information along with the user's feedback to improve performance and adjust to user preferences for future goals.

If other agents were used to reach the goal, their feedback might also be used. Multiagent feedback can be especially useful in minimizing the time that human users spend providing direction. However, users can also provide feedback throughout the agent's actions and internal reasoning to better align the results with the intended goal.[2]

Feedback mechanisms improve the AI agent's reasoning and accuracy, which is commonly referred to as iterative refinement.[3] To avoid repeating the same mistakes, AI agents can also store data about solutions to previous obstacles in a knowledge base.

Your subscription will be delivered in English. You will find an unsubscribe link in every newsletter. You can manage your subscriptions or unsubscribe here. Refer to our IBM Privacy Statement for more information.
https://www.ibm.com/us-en/privacy

## Agentic versus nonagentic AI chatbots

AI chatbots use conversational AI techniques such as natural language processing (NLP) to understand user questions and automate responses to them. These chatbots are a modality whereas agency is a technological framework.

Nonagentic AI chatbots are ones without available tools, memory or reasoning. They can only reach short-term goals and cannot plan ahead. As we know them, nonagentic chatbots require continuous user input to respond.

They can produce responses to common prompts that most likely align with user expectations but perform poorly on questions unique to the user and their data. Because these chatbots do not hold memory, they cannot learn from their mistakes if their responses are unsatisfactory.

In contrast, agentic AI chatbots learn to adapt to user expectations over time, providing a more personalized experience and comprehensive responses. They can complete complex tasks by creating subtasks without human intervention and considering different plans. These plans can also be self-corrected and updated as needed. Agentic AI chatbots, unlike nonagentic ones, assess their tools and use their available resources to complete information gaps.

## Reasoning paradigms

There is not one standard architecture for building AI agents. Several paradigms exist for solving multistep problems.

### ReAct (reasoning and action)

With the ReAct paradigm, we can instruct agents to "think" and plan after each action taken and with each tool response to decide which tool to use next. These Think-Act-Observe loops are used to solve problems step by step and iteratively improve upon responses.

Through the prompt structure, agents can be instructed to reason slowly and to display each "thought".[4] The agent's verbal reasoning gives insight into how responses are formulated. In this framework, agents continuously update their context with new reasoning. This approach can be interpreted as a form of Chain-of-Thought prompting.

### ReWOO (reasoning without observation)

The ReWOO method, unlike ReAct, eliminates the dependence on tool outputs for action planning. Instead, agents plan upfront. Redundant tool usage is avoided by anticipating which tools to use upon receiving the initial prompt from the user. This approach is desirable from a human-centered perspective because the user can confirm the plan before it is executed.

The ReWOO workflow is made up of three modules. In the planning module, the agent anticipates its next steps given a user's prompt. The next stage entails collecting the outputs produced by calling these tools. Lastly, the agent pairs the initial plan with the tool outputs to formulate a response. This planning ahead can greatly reduce token usage and computational complexity and the repercussions of intermediate tool failure.[5]

## Types of AI agents

AI agents can be developed to have varying levels of capabilities. A simple agent might be preferred for straightforward goals to limit unnecessary computational complexity. In order of simplest to most advanced, there are 5 main agent types:

### 1. Simple reflex agents

Simple reflex agents are the simplest agent form that grounds actions on perception. This agent does not hold any memory, nor does it interact with other agents if it is missing information. These agents function on a set of so-called reflexes or rules. This behavior means that the agent is preprogrammed to perform actions that correspond to certain conditions being met.

If the agent encounters a situation that it is not prepared for, it cannot respond appropriately. The agents are effective in environments that are fully observable granting access to all necessary information.[6]

**Example**: If it is 8 PM, then the heating is activated—for instance, a thermostat that turns on the heating system at a set time every night.

### 2. Model-based reflex agents

Model-based reflex agents use both their current perception and memory to maintain an internal model of the world. As the agent continues to receive new information, the model is updated. The agent’s actions depend on its model, reflexes, previous precepts and current state.

These agents, unlike simple reflex agents, can store information in memory and can operate in environments that are partially observable and changing. However, they are still limited by their set of rules.[6]

**Example**: A robot vacuum cleaner. As it cleans a dirty room, it senses obstacles such as furniture and adjusts around them. The robot also stores a model of the areas that it has already cleaned to not get stuck in a loop of repeated cleaning.

### 3. Goal-based agents

Goal-based agents have an internal model of the world and also a goal or set of goals. These agents search for action sequences that reach their goal and plan these actions before acting on them. This search and planning improve their effectiveness when compared to simple and model-based reflex agents.[7]

**Example**: A navigation system that recommends the fastest route to your destination. The model considers various routes that reach your destination, or in other words, your goal. In this example, the agent's condition-action rule states that if a quicker route is found, the agent recommends that one instead.

## 4. Utility-based agents

Utility-based agents select the sequence of actions that reach the goal and also maximize utility or reward. Utility is calculated through a utility function. This function assigns a utility value, a metric measuring the usefulness of an action or how "happy" makes the agent, to each scenario based on a set of fixed criteria.

The criteria can include factors such as progression toward the goal, time requirements or computational complexity. The agent then selects the actions that maximize the expected utility. Hence, these agents are useful in cases where multiple scenarios achieve a wanted goal and an optimal one must be selected.[7]

**Example**: A navigation system that recommends the route to your destination that optimizes fuel efficiency and minimizes the time spent in traffic and the cost of tolls. This agent measures utility through this set of criteria to select the most favorable route.

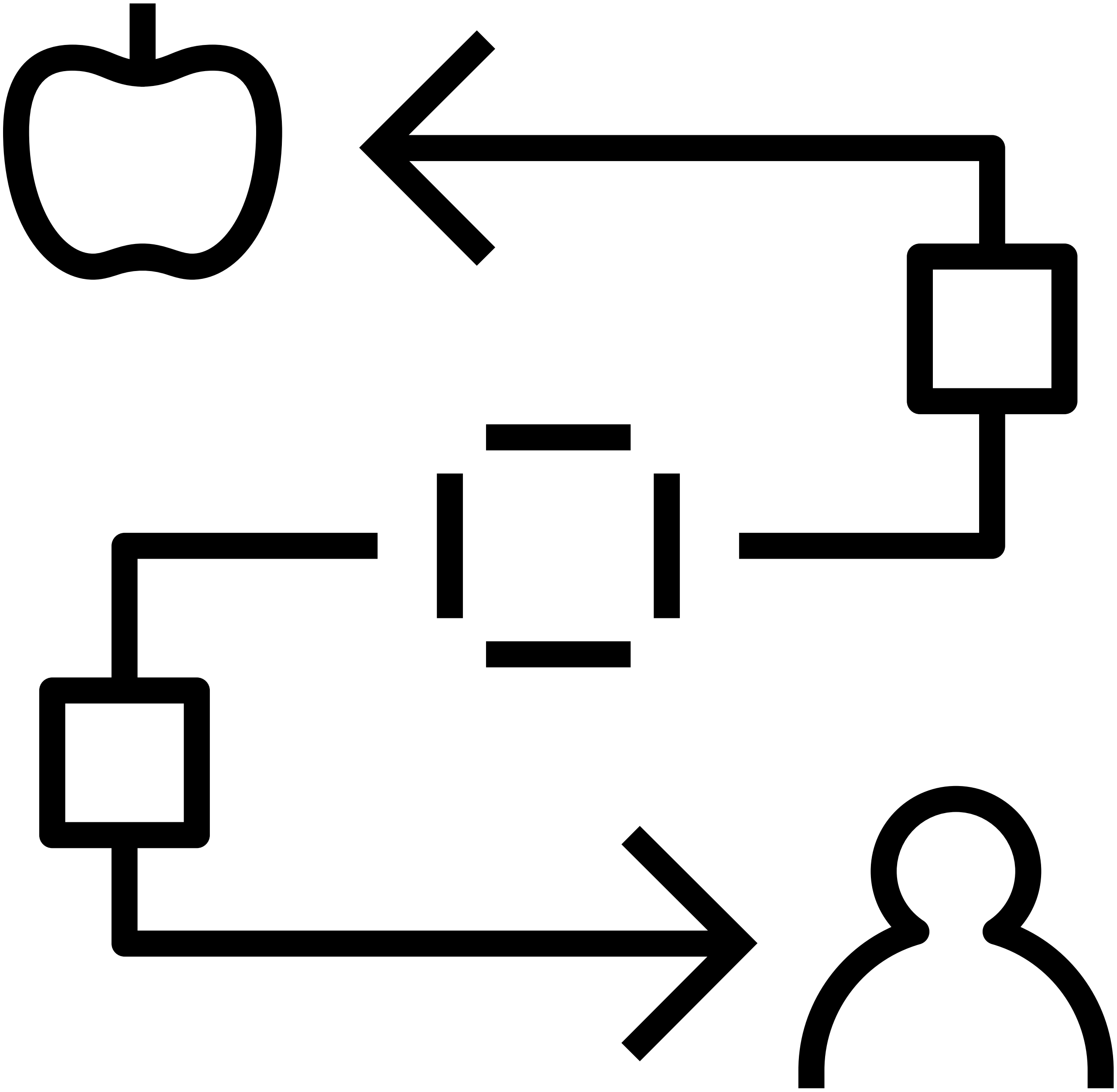
## 5. Learning agents

Learning agents hold the same capabilities as the other agent types but are unique in their ability to learn. New experiences are added to their initial knowledge base, which occurs autonomously. This learning enhances the agent's ability to operate in unfamiliar environments. Learning agents might be utility or goal-based in their reasoning and are composed of four main elements:[7]

- **Learning:** This process improves the agent's knowledge by learning from the environment through its precepts and sensors.
- **Critic:** This component provides feedback to the agent on whether the quality of its responses meets the performance standard.
- **Performance:** This element is responsible for selecting actions upon learning.
- **Problem generator:** This module creates various proposals for actions to be taken.

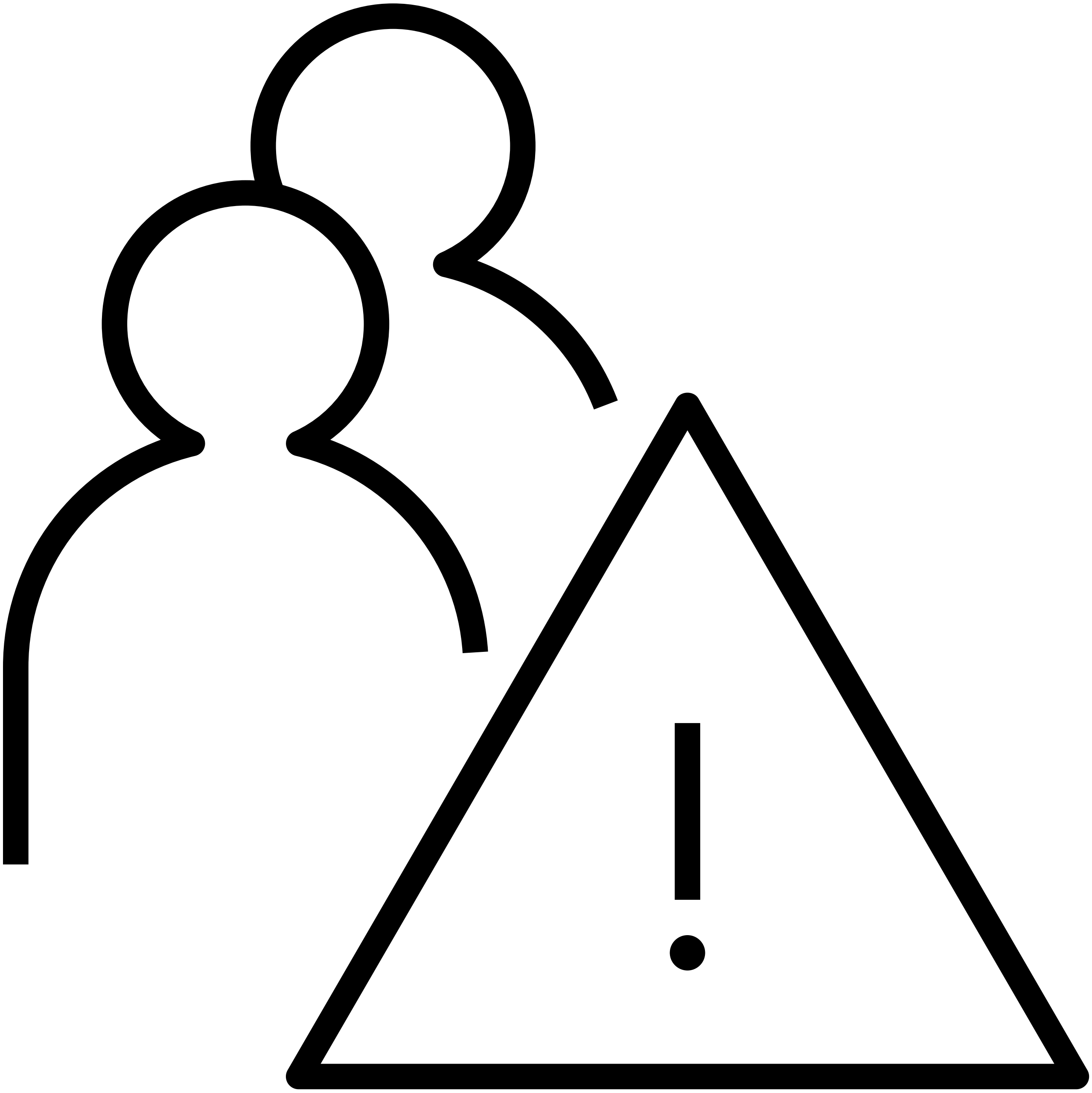
**Example**: Personalized recommendations on e-commerce sites. These agents track user activity and preferences in their memory. This information is used to recommend certain products and services to the user. The cycle repeats each time new recommendations are made. The user's activity is continuously stored for learning purposes. In doing so, the agent improves its accuracy over time.

## Use cases of AI agents



### Customer experience

AI agents can be integrated into websites and apps to enhance the customer experience by serving as a virtual assistant, providing mental health support, simulating interviews and other related tasks.[8] There are many no-code templates for user implementation, making the process of creating these AI agents even easier.



Emergency response

If there is a natural disaster, AI agents can use deep learning algorithms to retrieve the information of users on social media sites that need rescue. The locations of these users can be mapped to assist rescue services in saving more people in less time. Therefore, AI agents can greatly benefit human life in both mundane, repetitive tasks and life-saving situations.[10]

## Benefits of AI agents

### Task automation

With the ongoing advancements in generative AI and machine learning, there is a growing interest in workflow optimization through AI, or intelligent automation. AI agents are AI tools that can automate complex tasks that would otherwise require human resources. This shift translates to goals being reached inexpensively, rapidly and at scale. In turn, these advancements mean human agents do not need to provide direction to the AI assistant for creating and navigating its tasks.

### Greater performance

Multiagent frameworks tend to outperform singular agents.[11] This is because the more plans of action are available to an agent, the more learning and reflection occur.

An AI agent incorporating knowledge and feedback from other AI agents specializing in related areas can be useful for information synthesis. This backend collaboration of AI agents and the ability to fill information gaps are unique to agentic frameworks, making them a powerful tool and a meaningful advancement in artificial intelligence.

### Quality of responses

AI agents provide responses that are more comprehensive, accurate and personalized to the user than traditional AI models. This adaptability is important to us as users because higher-quality responses typically yield a better customer experience. As previously described, this capability is made possible through exchanging information with other agents, through tools and updating their memory stream. These behaviors emerge on their own and are not preprogrammed.[12]

## Risks and limitations

### Multiagent dependencies

Certain complex tasks require the knowledge of multiple AI agents. Orchestration of these multiagent frameworks has a risk of malfunction. Multiagent systems built on the same foundation models might experience shared pitfalls. Such weaknesses can cause a system-wide failure of all involved agents or expose vulnerability to adverse attacks.[13] This highlights the importance of data governance in building foundation models and thorough training and testing processes.

### Infinite feedback loops

The convenience of the hands-off reasoning for human users enabled by AI agents also comes with its risks. Agents that are unable to create a comprehensive plan or reflect on their findings, might find themselves repeatedly calling the same tools, causing infinite feedback loops. To avoid these redundancies, some level of real-time human monitoring might be used.[13]

### Computational complexity

Building AI agents from scratch is both time-consuming and can also be computationally expensive. The resources required for training a high-performance agent can be extensive. In addition, depending on the complexity of the task, agents can take several days to complete tasks.[12]

### Data privacy

If mismanaged, the integration of AI agents with business processes and customer management systems can raise some serious security concerns. For example, imagine AI agents leading the software development process, taking coding copilots to the next level, or determining pricing for clients—without any human oversight or guardrails. The results of such scenarios might be detrimental due to the experimental and often unpredictable behavior of agentic AI.

Therefore, it is essential for AI providers such as IBM, Microsoft and OpenAI to remain proactive. They must implement extensive security protocols to ensure that sensitive employee and customer data are securely stored. Responsible deployment practices are key to minimizing risk and maintaining trust in these rapidly evolving technologies.

## Best practices

### Activity logs

To address the concerns of multiagent dependencies, developers can provide users with access to a log of agent actions.[14] The actions can include the use of external tools and describe the external agents used to reach the goal. This transparency grants users insight into the iterative decision-making process, provides the opportunity to discover errors and builds trust.

### Interruption

Preventing autonomous AI agents from running for overly long periods of time is recommended. Particularly, in cases of unintended infinite feedback loops, changes in access to certain tools, or malfunctioning due to design flaws. One way to accomplish this goal is by implementing interruptibility.

Maintaining control of this decision involves allowing human users the option to gracefully interrupt a sequence of actions or the entire operation. Choosing if and when to interrupt an AI agent requires some thoughtfulness as some terminations can cause more harm than good. For instance, it might be safer to allow a faulty agent to continue assisting in a life-threatening emergency than to completely shut it down.[5]

### Unique agent identifiers

To mitigate the risk of agentic systems being used for malicious purposes, unique identifiers can be implemented. If these identifiers were required for agents to access external systems, tracing the origin of the agent’s developers, deployers and user would become easier.

This approach adds an essential layer of accountability. Traceability helps identify responsible parties when an agent causes malicious use or unintended harm. Ultimately, this kind of safeguard would foster a safer operational environment for AI agents.

### Human supervision

To assist in the learning process for AI agents, especially in their early stages in a new environment, it can be helpful to provide some level of human oversight. So, based on this guidance, the AI agent can compare its performance to the expected standard and make adjustments. This form of feedback is helpful in improving the agent's adaptability to user preferences.[5]

Apart from this safeguard, it is best practice to require human approval before an AI agent takes highly impactful actions. For instance, actions ranging from sending mass emails to financial trading should require human confirmation.[7] Some level of human monitoring is recommended for such high-risk domains.