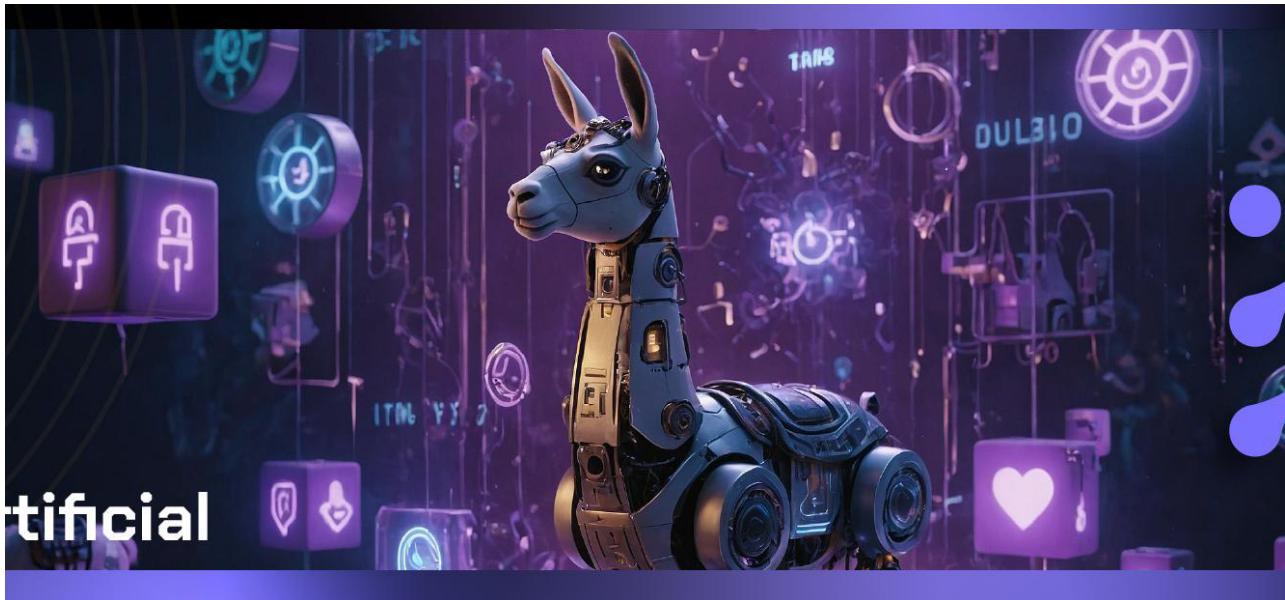


[VER PLANOS](#)[PROGRAMAÇÃO](#)[FRONT-END](#)[DATA SCIENCE](#)[INTELIGÊNCIA ARTIFICIAL](#)[DEVOPS](#)[UX & DESIGN](#)[MOBILE](#)[INOVAÇÃO & GESTÃO](#)[Artigos > Inteligência Artificial](#)

# Como agentes potencializam a performance das LLMs

**Larissa-dubiella**

01/12/2024

**COMPARTILHE**

Oi! Posso indicar os melhores artigos para tirar suas dúvidas!

Quem estuda tecnologia já conhece muito bem o poder de geração de texto das LLMs e que seu uso não se limita a apenas [chatbots](#) cheios de certezas.

Desde meados de 2022, quem está próximo dos meios digitais no seu dia a dia de trabalho já tem sentido a chamada *Quarta Revolução Industrial* na pele.

## Confira neste artigo:

- [O que são agentes?](#)
- [Quais são as capacidades dos agentes?](#)
- [E quais as principais características dos agentes?](#)
- [Como os agentes tomam decisões?](#)
- [Como seria um sistema multi-agentes?](#)
- [Coordenação entre agentes simples e complexos](#)
- [Como aplicar um agente com LlamalIndex](#)
- [Conclusão](#)

Com a chegada da [IA](#) da Meta no WhatsApp em outubro de 2024, a tendência é que essa tecnologia se popularize ainda mais e passe a fazer parte da vida de pessoas que ainda não a conheciam.

Na onda da indústria 4.0, pelas vivências com Internet das Coisas e até por inspiração de filmes de Ficção Científica, imaginamos um futuro em que a Inteligência Artificial se integre ao cotidiano tendo conhecimento sobre informações importantes em tempo real e sendo responsável pela realização de tarefas repetitivas. Deixa eu te contar: estamos no caminho!

Os **agentes de IA** podem ser a peça que transforme radicalmente nossa interação com a tecnologia.

Neste artigo, vamos explorar o conceito de agentes de IA, suas características e aplicações práticas, além de refletir sobre os impactos que essa ferramenta pode ter na indústria.

# O que são agentes?

Agentes atuam como **controladores da lógica e das ações** de um sistema modular de IA. Com uma LLM servindo como “cérebro”, um agente toma decisões e executa tarefas de forma autônoma.

Se você é um cozinheiro experiente e precisa preparar um jantar, não precisa seguir uma receita estritamente do começo ao fim.

Enquanto você se dedica ao prato principal, pode pedir para alguém arrumar a mesa, outra pessoa picar a salsinha e uma terceira higienizar a salada.

Agora, se você nunca cozinhou antes, precisará seguir as instruções com cuidado e gastar muito mais tempo para concluir a refeição.

Quando construímos um sistema de IA baseado em agentes, o agente funciona como esse chef de cozinha experiente, com a capacidade de planejar, agir e delegar tarefas.

**A lógica do sistema está sob o seu controle** – ele decide o que fazer com base nos dados que tem em mãos e em como melhor organizá-los para atingir o objetivo.

Isso se diferencia de sistemas que, mesmo usando LLMs para realizar tarefas, seguem condições exatas impostas pela pessoa que programou o sistema.

Em resumo, o que diferencia o uso de uma LLM em agentes de um sistema tradicional programado por humanos é que, em vez de decisões pré-programadas ou regras fixas, o modelo de linguagem usa sua capacidade de entendimento para decidir dinamicamente qual caminho seguir, o que dá maior flexibilidade e adaptabilidade ao sistema.

The advertisement features a woman smiling and pointing upwards, with icons representing AI and learning in the background. The text reads: "Matricule-se na escola de INTELIGÊNCIA ARTIFICIAL. Junte-se a uma comunidade de +500 mil estudantes. → Acesso a TODOS os cursos em uma única assinatura → Novos lançamentos a cada semana → Desafios práticos. SAIBA MAIS".

## Quais são as capacidades dos agentes?

De forma geral, as capacidades dos agentes são:

- **Recuperação de dados** de diferentes tipos (não estruturados, semi-estruturados, estruturados) e realização de **buscas automáticas**.
- **Interação com APIs** de serviços externos, processando ou armazenando dados para uso futuro.
- Manter e analisar **históricos de conversas**.

- Executar tarefas de dados complexas.

# E quais as principais características dos agentes?

Agora que entendemos as capacidades essenciais dos agentes, vamos nos aprofundar nas características que os tornam tão adaptáveis e eficazes para diversas aplicações.

- **Autonomia:** Os agentes são capazes de tomar decisões e executar tarefas por conta própria, sem a necessidade de intervenção humana constante.
- **Capacidade de Planejamento:** Um agente não apenas reage a inputs, mas também antecipa etapas futuras, organizando uma sequência lógica de ações para alcançar o objetivo. Ele calcula o que precisa ser feito e em que ordem, considerando a situação atual.
- **Delegação e Coordenação:** Em um sistema modular, um agente pode distribuir tarefas entre outros agentes ou componentes do sistema, coordenando o trabalho de maneira eficiente.
- **Adaptação e Flexibilidade:** Diferente de sistemas tradicionais que seguem regras pré-programadas, os agentes podem ajustar suas ações com base nas condições e no contexto. Isso permite que o sistema lide com situações imprevistas ou variáveis.
- **Uso de LLMs para Tomada de Decisão:** A lógica do sistema é controlada por um modelo de linguagem, que atua como "cérebro" do agente. Esse LLM processa informações, entende o contexto e escolhe a melhor resposta ou ação.
- **Orquestração de Tarefas:** Um agente pode servir como orquestrador em um sistema complexo, garantindo que todas as partes do sistema cooperem de maneira eficaz. Ele pode gerenciar múltiplos agentes menores ou tarefas paralelas, assegurando que o trabalho flua.

## Como os agentes tomam decisões?

A tomada de decisão autônoma é uma das maiores vantagens dos agentes de IA em comparação com sistemas tradicionais.

Diferente de simples autômatos que seguem uma programação rígida, os agentes de IA tomam decisões de forma adaptativa e dinâmica.

Este processo, chamado de **ciclo de decisão autônomo**, é uma sequência que envolve:

1. **Análise do estado atual:** O agente coleta dados ou interpreta um contexto específico, seja ele uma mensagem recebida, uma nova entrada de sensor ou um dado externo.
2. **Planejamento de ações:** Com base em seu objetivo, o agente considera o conjunto de ferramentas e dados à sua disposição, planejando uma sequência de ações.
3. **Execução e Avaliação:** O agente realiza a ação planejada e, caso necessário, ajusta o plano conforme o feedback, criando um sistema que se aprimora com a prática.

Uma abordagem utilizada nesse ciclo é o **ReAct** (*Reason + Act: Pensamento + Ação*). Em vez de simplesmente seguir comandos, o ReAct permite que o agente alterne entre raciocinar sobre um problema e realizar ações para resolvê-lo.

Assim, o agente consegue refletir sobre cada etapa antes de agir, o que aumenta a precisão de sua resposta e a adaptabilidade do sistema.



# Como seria um sistema multi-agentes?

Em um sistema multi-agentes, cada agente é especializado em uma tarefa ou área, e a colaboração entre eles potencializa o funcionamento do sistema.

Considere um exemplo onde a empresa utiliza um sistema multi-agentes para monitorar e gerenciar operações de uma fábrica inteligente.

- **Agente de Monitoramento:** Coleta dados de sensores espalhados pela fábrica e detecta problemas, como aquecimento excessivo de uma máquina.
- **Agente de Diagnóstico:** Recebe o alerta do Agente de Monitoramento e realiza uma análise para determinar a causa provável, podendo consultar histórico e dados técnicos.
- **Agente de Manutenção:** Planeja uma intervenção, caso necessário, e organiza os passos que um operador humano ou outro agente mecânico devem seguir para resolver o problema.
- **Agente de Logística:** Coordena o reabastecimento de peças ou componentes que possam ser necessários para a intervenção, organizando o fluxo de materiais na fábrica.

Assim, o sistema multi-agentes opera em conjunto, com cada unidade focada em uma tarefa, mas colaborando de forma coordenada.

Este tipo de sistema já encontra aplicações em indústrias avançadas, tornando a automação mais inteligente e adaptável às mudanças do ambiente de produção.

Porém, nem todos os agentes precisam ser complexos ou responsáveis por uma ampla gama de decisões. **Agentes simples** também têm sua importância e podem compor um sistema ao serem responsáveis pela execução de uma única tarefa ou conjunto muito restrito de ações.

Por exemplo, em um sistema de atendimento automatizado de uma loja virtual, podemos ter:

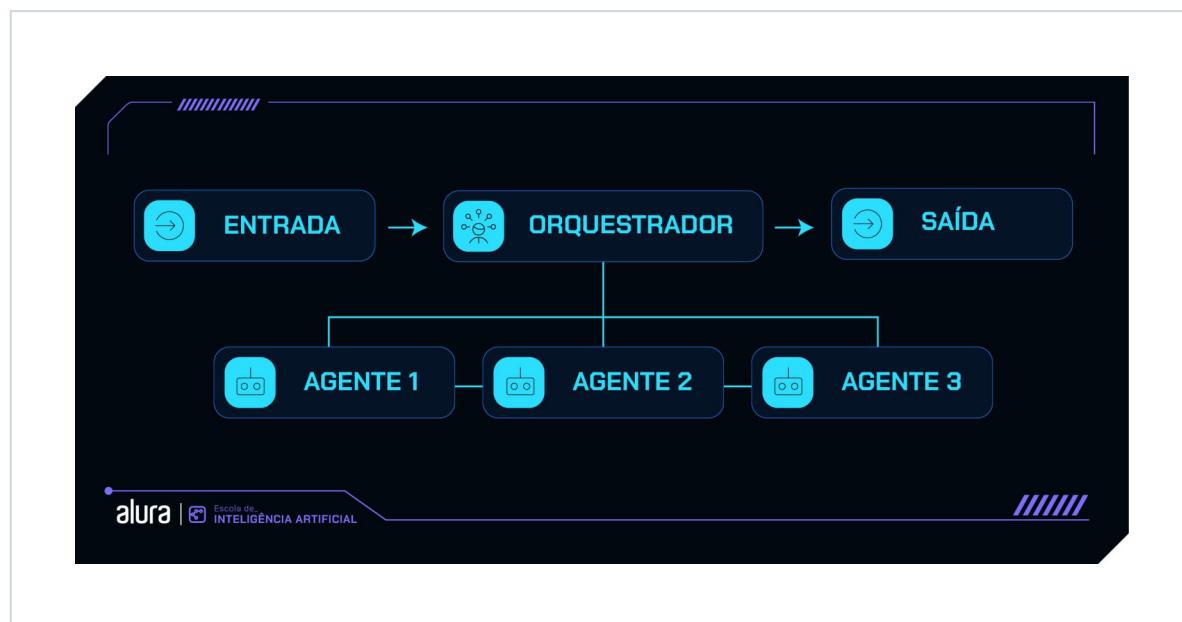
- **Agente de Verificação de Estoque:** Ao receber uma consulta sobre um produto, este agente verifica a quantidade disponível no estoque e responde sobre a disponibilidade.
- **Agente de Cotação de Frete:** Responsável por calcular o valor do frete de acordo com o endereço do cliente e as opções de entrega disponíveis. Esse agente retorna o valor calculado e possíveis datas de entrega.
- **Agente de Emissão de Nota Fiscal:** Após uma compra ser finalizada, este agente emite a nota fiscal e a envia ao cliente.

# Coordenação entre agentes simples e complexos

Em sistemas multi-agentes mais sofisticados, agentes simples são frequentemente coordenados por um **agente orquestrador** ou **agente principal**, que tem uma visão mais ampla e pode direcionar os agentes especializados conforme necessário.

Esse orquestrador pode ser responsável por:

- Delegar tarefas a diferentes agentes simples, garantindo que todos os passos do processo sejam executados na sequência correta.
- Gerenciar falhas ou atrasos, redistribuindo tarefas para que o sistema continue a operar sem interrupções.
- Supervisionar o desempenho dos agentes e ajustar a alocação de tarefas conforme a demanda.



Sistemas multi-agentes não estão limitados a fábricas e operações físicas; no varejo online, na logística e até na saúde, esses sistemas coordenados são aplicados para otimizar processos, reduzir erros e aumentar a autonomia das operações.

# Como aplicar um agente com LlamalIndex

Vamos experimentar a aplicação de um agente de matemática? Utilizei o [Google Colab](#) para esse exemplo.

Primeiro, vamos instalar as bibliotecas necessárias para criar e configurar nosso agente de IA com LlamalIndex.

```
!pip install llama-index  
!pip install llama-index-llms-groq  
!pip install llama_index.core.agent  
!pip llama_index.core.tools
```

Agora que instalamos as bibliotecas, o próximo passo é importar os módulos e classes principais que vão nos ajudar a construir o agente.

Aqui, chamamos o [ReActAgent](#) e [FunctionCallingAgentWorker](#) da biblioteca LlamalIndex, além de ferramentas específicas para lidar com funções e o modelo [Groq](#), que será responsável por trazer a LLM para o nosso ambiente. A importação de `userdata` do Google nos permitirá acessar uma chave de API.

```
from llama_index.core.agent import ReActAgent, FunctionCallingAger  
from llama_index.core.tools import FunctionTool  
from llama_index.llms.groq import Groq  
from google.colab import userdata
```

Agora, precisamos de uma chave de API que permita utilizar o Llama. Você pode criar uma chave através do [site do Groq](#). Crie sua conta, crie a API Key e copie.

No Colab, para adicionar uma chave de API, clique no ícone de chave que fica no menu lateral esquerdo. Então, dê um nome (aqui utilizei `GROQ_API`) e, no valor, cole sua chave. É necessário permitir acesso ao notebook.

Para acessar a chave, podemos utilizar o seguinte código:

```
api_key= userdata.get('GROQ_API')
```

Podemos escolher um modelo disponibilizado pelo Groq. Aqui, escolhi o Llama3. Na plataforma Groq, [você pode consultar uma lista de outros modelos disponíveis](#) e, dependendo da sua necessidade, optar por outro modelo ajustando o parâmetro `model`.

```
llama3 = Groq(model="llama3-70b-8192", api_key=api_key)  
llama3
```

Legal! Agora é o momento de **criar as ferramentas** que o agente poderá utilizar. Com LlamaIndex, essas ferramentas são criadas como simples *funções Python* e então cada função é registrada como uma `FunctionTool`, permitindo que o agente acesse e execute essas operações quando necessário.

Criamos uma função `multiply` que retorna a multiplicação de dois números inteiros e uma função `add` que retorna a soma de dois números inteiros.

```
from llama_index.core.tools import FunctionTool  
  
def multiply(a: int, b: int) -> int:  
    """Multiple two integers and returns the result integer"""  
    return a * b  
  
multiply_tool = FunctionTool.from_defaults(fn=multiply)  
  
def add(a: int, b: int) -> int:  
    """Add two integers and returns the result integer"""  
    return a + b  
  
add_tool = FunctionTool.from_defaults(fn=add)
```

Com as ferramentas criadas, vamos configurar o agente usando o modelo ReAct. Passamos a ele as ferramentas criadas anteriormente e a LLM que

também já configuramos.

O parâmetro `verbose=True` permite que o agente exiba os passos que toma durante o processamento das instruções - assim, teremos uma visualização bem gráfica de como o processo de raciocínio e ação do agente funciona.

```
react_agent = ReActAgent.from_tools(tools=[multiply_tool, add_tool])
react_agent
```

Eba, agente definido! Simples assim. Tenha em mente que esse é um exemplo de aplicação muito básica, onde definimos apenas duas operações matemáticas como ferramentas.

Porém, a escalabilidade do projeto pode acontecer de acordo com as necessidades a qual o sistema se propõe, com múltiplos agentes e ferramentas mais complexas.

Neste teste, pedimos ao agente para calcular "Quanto é a soma de 154 e 90 vezes cinco?". Essa pergunta exige que ele interprete corretamente a ordem das operações (multiplicação e adição) e utilize as funções que criamos anteriormente para somar e multiplicar.

O agente deve realizar a multiplicação primeiro (90 vezes 5), passando os valores em formato correto para a ferramenta e, em seguida, somar o resultado a 154, garantindo que a resposta esteja correta.

```
response = react_agent.chat("Quanto é a soma de 154 e 90 vezes cinco?")
print(str(response))
```

Executando, temos a seguinte resposta:

```
Running step 3ca66650-4add-4593-b036-8d6fea670a5. Step input: Qu
Thought: The current language of the user is: Portuguese. I need to
To calculate 90 times 5, I'll use the multiply tool.
```

```
Action: multiply  
Action Input: {'a': 90, 'b': 5}  
Observation: 450  
> Running step 31ae365c-db7d-47ce-b5e4-43de579064ad. Step input: N  
Thought: Now I have the result of 90 times 5, which is 450. I need  
To add 154 and 450, I'll use the add tool.  
Action: add  
Action Input: {'a': 154, 'b': 450}  
Observation: 604  
> Running step d8d71fe6-b6e6-4917-8573-8ae137bac149. Step input: N  
Thought: I can answer without using any more tools. I'll use the u  
Answer: A soma de 154 e 90 vezes 5 é 604.  
A soma de 154 e 90 vezes 5 é 604.
```

Perceba que bacana! A cada passo, o agente “pensa” no que precisa ser feito. Primeiro, ele detecta que a pergunta foi feita em português e que deve manter a resposta nesse idioma.

Em seguida, ele interpreta que precisa resolver 90 vezes 5 antes de somar, seguindo a ordem de precedência das operações matemáticas.

Com o resultado da multiplicação em mãos, o agente segue para a próxima etapa e utiliza a ferramenta de adição. Com a resposta da ferramenta, o agente responde respeitando o idioma da entrada.

No LlamalIndex, é possível construir agentes do zero ou utilizar os que já estão construídos no [LlamaHub](#). O *workflow* para a construção de agentes no LlamalIndex é assíncrono e *event-driven*. Você pode aprender mais sobre isso [na documentação](#).

## Conclusão

Por sua capacidade de adaptação e flexibilidade, agentes são ideais para uma ampla variedade de setores.

Ao permitir a automação inteligente de processos complexos, os sistemas baseados em agentes podem contribuir para a eficiência operacional e a otimização de recursos, aliviando os profissionais de tarefas repetitivas e criando espaço para iniciativas mais estratégicas.

Nesse artigo, conhecemos como os agentes funcionam e como poder levar a aplicação de LLMs em sistemas a outro patamar!

Certamente, ainda teremos muitas novidades nesse campo de estudos. A comunidade está dedicada a sempre construir soluções melhores para o uso de IA. Acompanhe as notificações da Alura para ficar por dentro dos nossos lançamentos em dados e IA!

E falando em novidade, a Alura acabou de lançar um curso rápido sobre agentes de IA para Devs:

## Domine Agentes de IA em menos de 2h com o Flash Skills!

Aprenda, na prática, a programar um **agente de IA com LangGraph do zero**, aplicando IA Generativa em um projeto real. Tudo de forma prática, para que você saia com um **projeto prático para seu portfólio**. Além de acelerar seu aprendizado, você ainda garante um **certificado da Alura** para fortalecer seu perfil profissional.

O Flash Skills **Domine Agentes de IA: crie seu primeiro projeto com LangGraph e transforme sua Carreira** foi feito para devs que tem pouco tempo, quer um aprendizado direto ao ponto e está buscando um curso acessível e prático para aplicar no dia-a-dia.

Acesse agora a [página do Flash Skills](#) e matricule-se!

Abraço e até breve.



## Leia também



O que é Inteligência Artificial?

Como funciona uma IA, quais

os tipos e exemplos

DeepSeek: desvendando a IA

que pensa antes de responder

As I

IA q



Veja outros artigos sobre  
[Inteligência Artificial](#)

## Quer mergulhar em tecnologia e aprendizagem?

Receba conteúdos, dicas, notícias, inovações e tendências sobre o mercado tech diretamente na sua caixa de entrada.

[bins.br@gmail.com](mailto:bins.br@gmail.com)

**ENVIAR**

Nossas redes e apps



## Institucional

Sobre nós

Trabalhe na Alura

Para Empresas

Para Sua Escola

Política de Privacidade

Compromisso de Integridade

Termos de Uso

Documentos Institucionais

Status

Uma empresa do grupo Alun

## A Alura

Como Funciona

Formações

Plataforma

Depoimentos

Instrutores(as)

Dev em <T>

Luri, a inteligência artificial da Alura

IA Conference 2025

Cursos imersivos

Certificações

## Conteúdos

Alura Cases

Imersões

Artigos

Podcasts

Artigos de educação  
corporativa

Imersão Dev Agentes de IA Google

## Fale Conosco

Email e telefone

Perguntas frequentes

## Novidades e Lançamentos

bins.br@gmail.com

## CURSOS

### Cursos de Programação

Lógica | Python | PHP | Java | .NET | Node JS | C | Computação | Jogos | IoT

### Cursos de Front-end

HTML, CSS | React | Angular | JavaScript | jQuery

### Cursos de Data Science

Ciência de dados | BI | SQL e Banco de Dados | Excel | Machine Learning | NoSQL | Estatística

### Cursos de Inteligência Artificial

IA para Programação | IA para Dados

### Cursos de DevOps

AWS | Azure | Docker | Segurança | IaC | Linux

### Cursos de UX & Design

Usabilidade e UX | Vídeo e Motion | 3D

### Cursos de Mobile

Flutter | iOS e Swift | Android, Kotlin | Jogos

### Cursos de Inovação & Gestão

Métodos Ágeis | Softskills | Liderança e Gestão | Startups | Vendas

## CURSOS UNIVERSITÁRIOS FIAP

Graduação | Pós-graduação | MBA