What is AI agent development?

What is AI agent development?

AI agent development is the process of creating AI agents. It includes designing, building, training, testing and deploying agentic AI.

Enterprises can choose to craft AI agents from the ground up. This gives them full control over the agentic architecture and functionality. They can also tailor agent systems to their use cases and business needs and customize agentic AI for specific tasks. Building AI agents from scratch, on the other hand, requires significant expertise in artificial intelligence, machine learning and software development. Additionally, it can be expensive.

A swifter, more scalable approach, especially for beginners, involves using AI agent frameworks. As the foundational structure for AI-powered agents, these software platforms have built-in features that help streamline the agent development process, including predefined architectures and templates, task management systems and integration and monitoring tools.

A step-by-step guide to the AI agent development process

Implementing AI agents typically consists of these series of steps:

1. Goal setting and scoping
2. Design
3. Framework, model and tool selection
4. Build
5. Training
6. Evaluation
7. Deployment and monitoring

Goal setting and scoping

The first step is to outline clear objectives and a defined scope for an AI agent. Here are some questions that can help:

- What problem will the agent solve?
- What tasks will it need to accomplish?
- What data or user inputs will the AI agent require?
- What decisions will it need to make?
- Will decision-making be autonomous, or will a human-in-the-loop approach be necessary?
- Who are the users, and how will they use this AI system?

The answers to these questions can help steer the design step.

Design

An agent's blueprint is drafted during the design phase. This blueprint encompasses the architecture, workflows, integration and user experience.

For simple functions, such as customer support agents tracking orders in real time and providing customers with status updates, a single-agent architecture might suffice. But for complex tasks, a multi-agent system might be more suitable. In healthcare, for example, a multi-agent system can automate the complex workflows of drug discovery, with separate agents for exploring libraries of chemical compounds and summarizing medical research, and another generative AI (genAI) agent for generating new molecular designs.

The architecture helps determine the right type of AI agent and its components. It also aids in mapping out agentic workflows, including edge cases and error scenarios. For multi-agent ecosystems, communication protocols, orchestration and collaboration strategies must be taken into account.

If an agent will directly interact with users, enterprises can opt for an AI assistant interface similar to chatbots like OpenAI's ChatGPT. They'll also need a plan for integrating with other platforms and consider tool calling to access application programming interfaces (APIs), external plug-ins, customer data and other data sources for real-time information processing and dynamic decision-making.

Framework, model and tool selection

Once the design has been laid out, the next stage is to choose the right framework, AI model and other relevant AI tools or libraries.

Organizations can build agents on their own using programming languages such as Python or JavaScript. For those employing agentic frameworks, some common choices include open-source frameworks like BeeAI, CrewAI, LangChain, LangGraph and Microsoft's AutoGen and Semantic Kernel software development kit (SDK).

Model selection is crucial to align machine learning algorithms or large language models (LLMs) with an AI agent's functions and tasks. Companies might also look into specialized tools like retrieval augmented generation (RAG) systems or libraries such as PyTorch, scikit-learn and TensorFlow to further power their AI agents.

Build

The build phase is where the agent development action happens. To help avoid overwhelm, businesses can take a modular approach, crafting each component separately before combining them all together into a working AI agent. This modular strategy also allows for easier maintenance since alterations to each part will only have a minimal impact on the whole agent system.

Aside from building the AI agent itself, organizations must also consider these factors when developing agentic AI:

- **Efficiency:** AI agents must swiftly process data, make decisions, perform actions and produce responses.
- **Scalability:** Agents must be robust enough to handle growing volumes without their performance degrading.
- **Security:** Incorporating safety guardrails such as access control, authentication and encryption can help prevent adversarial attacks and unauthorized access and interactions.

Training

Model training entails an AI model learning from a training dataset of sample tasks relevant to an agent's functions and actions. It's an iterative process that involves preparing a dataset, running the model on this data, measuring its performance through a loss or reward signal, and adjusting the model's parameters to improve future predictions.

Training machine learning models from scratch can be lengthy, costly and resource-intensive. Companies might prefer using a pretrained model instead and fine-tune it on datasets specific to an AI agent's tasks.

Evaluation

AI agent evaluation is the process of testing and validating agentic AI to make sure it fulfills its goals and performs as expected. It requires a testing or validation dataset that's different from the training dataset and diverse enough to cover all possible test cases and reflect real-world scenarios.

Conducting tests in a sandbox or simulated environment can help pinpoint performance improvements early on and identify any security issues and ethical risks before deploying agents to actual users.

Like LLM benchmarks, AI agents also have a set of evaluation metrics. Common ones include functional metrics such as success rate or task completion, error rate and latency, and ethical metrics like bias and fairness score and prompt injection vulnerability. Agents and bots that interact with users are assessed according to their conversational flow, engagement rate and user satisfaction score.

After measuring metrics and analyzing test results, agent development teams can proceed with debugging algorithms, modifying agentic architectures, refining logic and optimizing performance.

Deployment and monitoring

This final phase entails deploying agentic systems to live production environments where customers can interact with and use AI agents. It also includes continuous monitoring, which is critical to tracking and enhancing agent performance and making sure it adapts to new situations and challenges.

Platforms like Amazon Bedrock AgentCore and IBM® watsonx.ai® help automate agent deployment and monitoring. With watsonx.ai, for instance, developers can take advantage of one-click deployment and tracing capabilities for observability.