Comparison of different agentic frameworks

•••

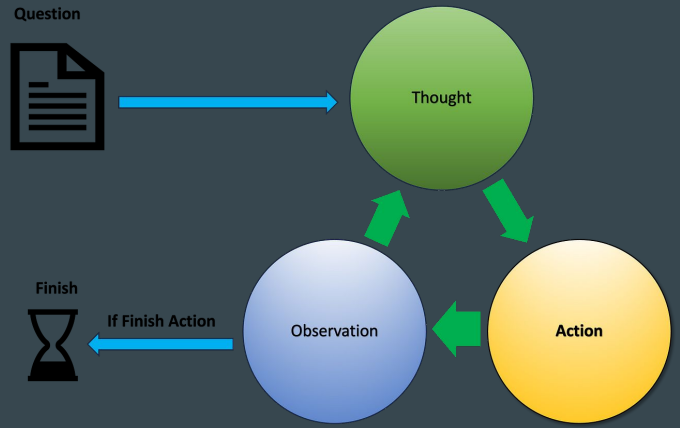
LlamaIndex vs CrewAI vs LangChain

Why an agentic framework is needed

- Automate and orchestrate the workflow of our AI component
- Tool calling
- Modularize (and extend) our components if necessary
- Self interpretation of tasks (rather than hard-code for different scenarios)

ReAct

- An architecture where the LLM Re: Reasons, and Act: Acts according to previous output
- An LLM is called repeatedly in a while-loop until satisfying result is obtained
- At each step the agent decides which tools to call and what the input should be

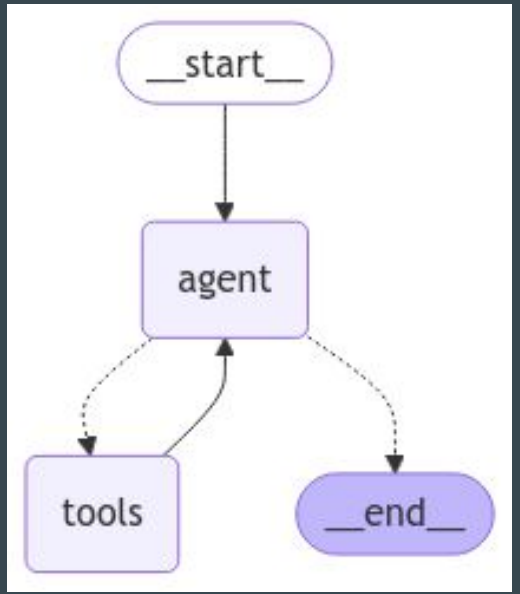


LlamaIndex

- Uses agents and workflows
- **Workflows:**
  - Multi-step processes
  - Combine one or more agents, data connectors, and other tools to complete a task
  - Event driven
- To have the capability of self managing workflow a ReAct agent needs to be implemented
- This is luckily pre-defined and easy to use
- Best overall performance, reliability and customizability

Langchain/Langgraph

- Similar to LlamaIndex.
- To use reasoning ReAct agent is needed
- Previously used AgentExecutor as runtime for agents
- Phasing out in favour of LangGraph to facilitate more sophisticated agent and tool calling
- Introduced breaking changes in new releases
- Needs a lot of steps to get base implementations
- From testing seems to be the least mature



Crew AI

- Agents take on different roles and personas
- Agents can make decisions autonomously based on the given context and available tools
- Agents can work together
- Tasks can be either “hardcoded” i.e in a sequential manner or an agent manager can be assigned to handle task delegation, no ReAct agent needed
- Seamless integration of langchain and Llamaindex tools
- While task delegation with a manager agent seems the most intuitive and easy, mixed results are obtained