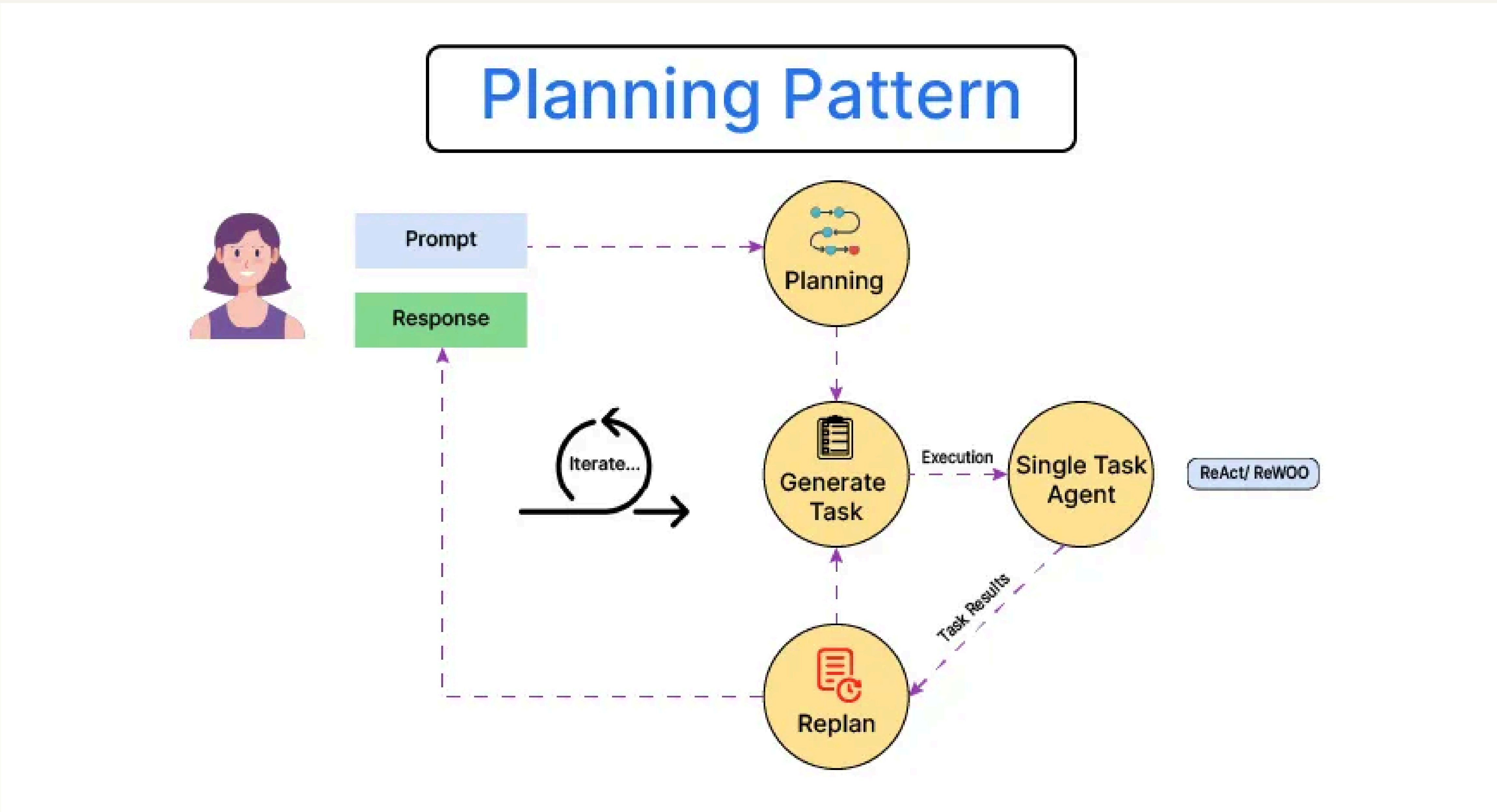
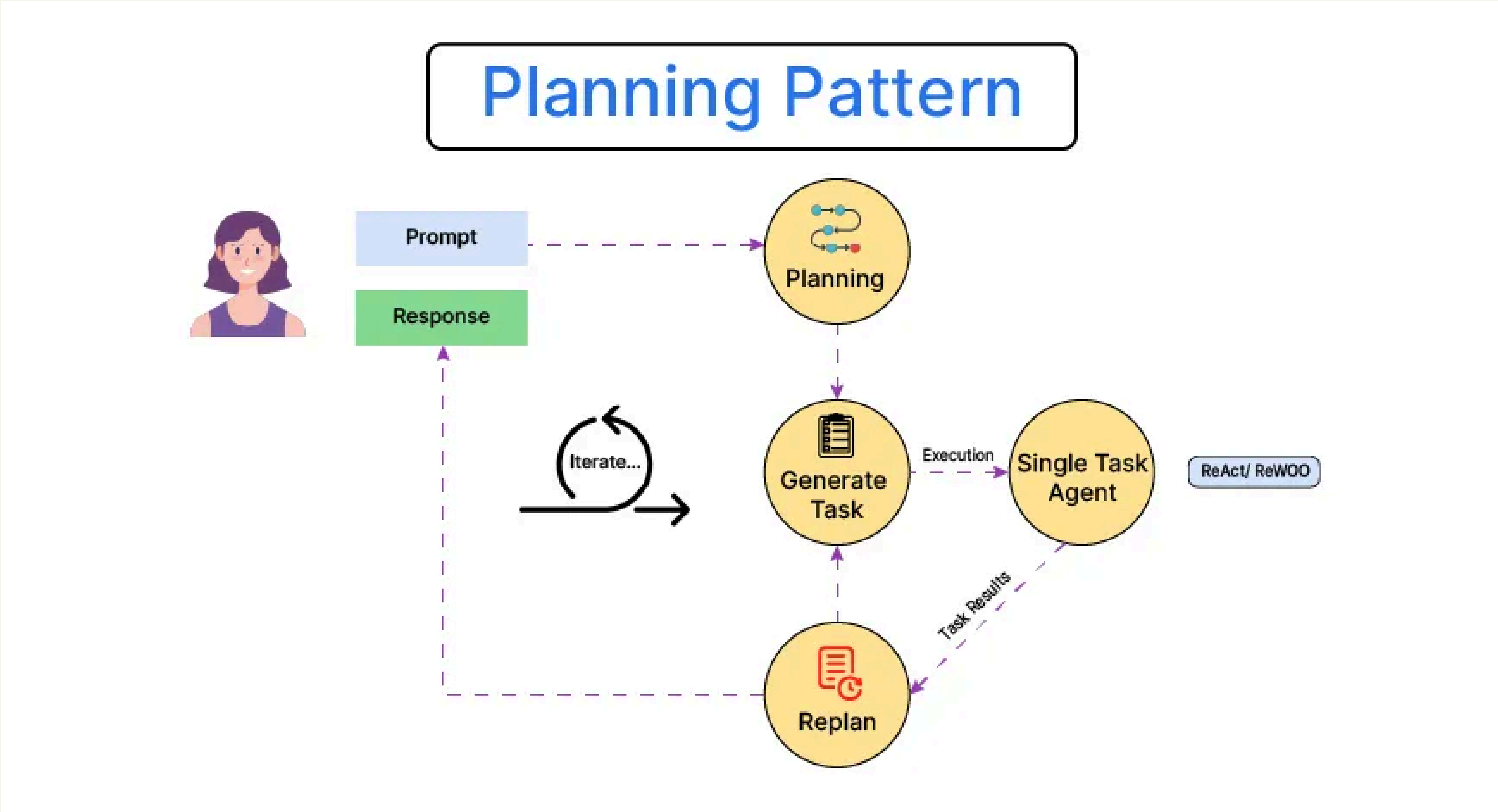
Guide to Agentic AI Planning Pattern



Dipanjan (DJ)

Planning Agents

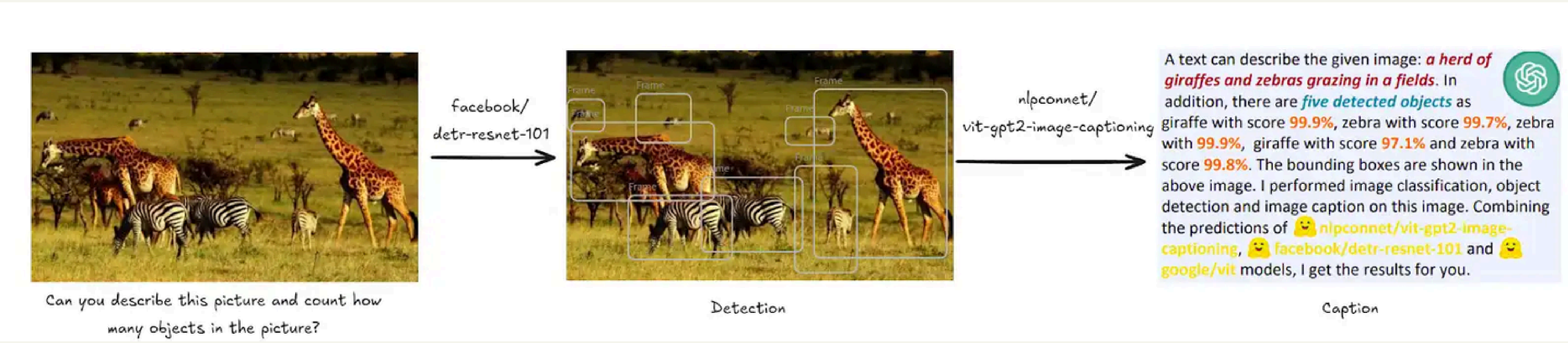


The Planning Pattern has the following main components:

1. **Planning**
   - In this initial stage, the AI agent interprets the prompt and devises an overall plan to tackle the problem, including high-level goals
2. **Generate Task**
   - From the plan, the AI system generates specific tasks that must be executed. Each task represents a smaller, manageable portion of the overarching goal
3. **Single Task Agent**
   - The Single Task Agent is responsible for completing each task generated in the previous step. This agent executes each task using predefined methods like ReAct (reflect-then-act) or ReWOo (rework-oriented operations).
   - Once a task is completed, the agent returns a Task Result, which is sent back to the planning loop.
4. **Replan**
   - The Replan stage evaluates the Task Result to determine if any adjustments are needed.
   - If the task execution does not fully meet the desired outcome, the system will replan and possibly modify the tasks or strategies.
5. **Iterate:**
   - This part of the pattern is a loop connecting Generate Task and Replan.

Source: What is Agentic AI Planning Pattern?

Planning Agent Example



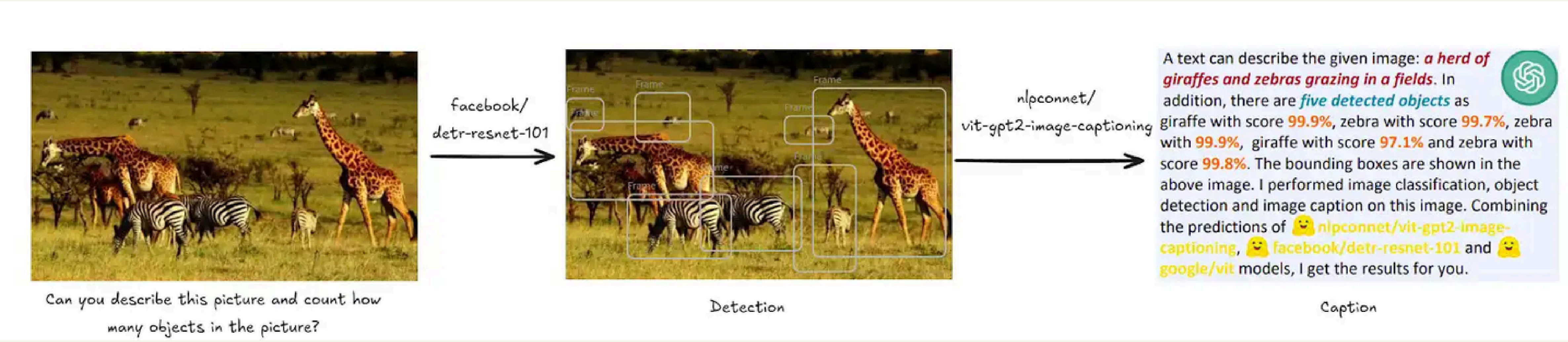
The above-given illustration depicts a sequential image understanding process, with steps that align with the agentic AI planning pattern

- **Evaluation and Iteration (Combining Results)**
  - **Processing and Aggregating Information:** The results from detection (bounding boxes and object types) and captioning (descriptive text) are combined. The agent evaluates its outputs, confirming both object detection confidence levels and the coherence of the description.
  - **Agentic AI Element:** The agent reviews its predictions (detection scores and bounding boxes) to ensure they align with the task's demands.

- **Goal Achievement (Answer Presentation)**
  - **Output Presentation:** The agent finally provides an answer that includes a count of detected objects, a list of identified objects with confidence scores, and a descriptive caption.
  - **Agentic AI Element:** The agent completes the goal by synthesising its perception and planning outcomes into a coherent response. In agentic AI, this step is about achieving the task's overarching goal and generating an output that addresses the user's initial question.

Source: What is Agentic AI Planning Pattern?

Planning Agent Example



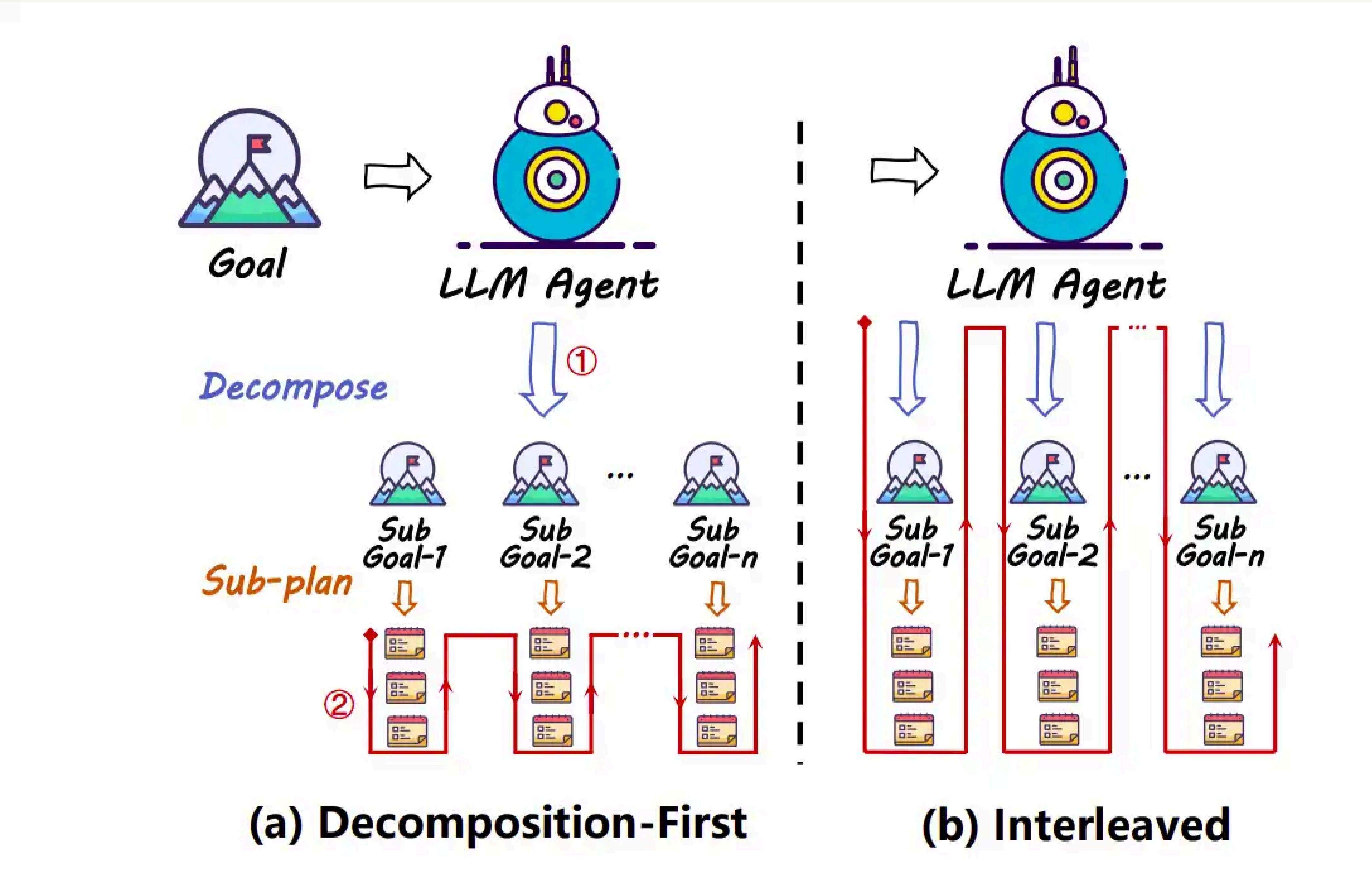
The above-given illustration depicts a sequential image understanding process, with steps that align with the agentic AI planning pattern

- **Evaluation and Iteration (Combining Results)**
  - **Processing and Aggregating Information:** The results from detection (bounding boxes and object types) and captioning (descriptive text) are combined. The agent evaluates its outputs, confirming both object detection confidence levels and the coherence of the description.
  - **Agentic AI Element:** The agent reviews its predictions (detection scores and bounding boxes) to ensure they align with the task's demands.

- **Goal Achievement (Answer Presentation)**
  - **Output Presentation:** The agent finally provides an answer that includes a count of detected objects, a list of identified objects with confidence scores, and a descriptive caption.
  - **Agentic AI Element:** The agent completes the goal by synthesising its perception and planning outcomes into a coherent response. In agentic AI, this step is about achieving the task's overarching goal and generating an output that addresses the user's initial question.

Source: What is Agentic AI Planning Pattern?

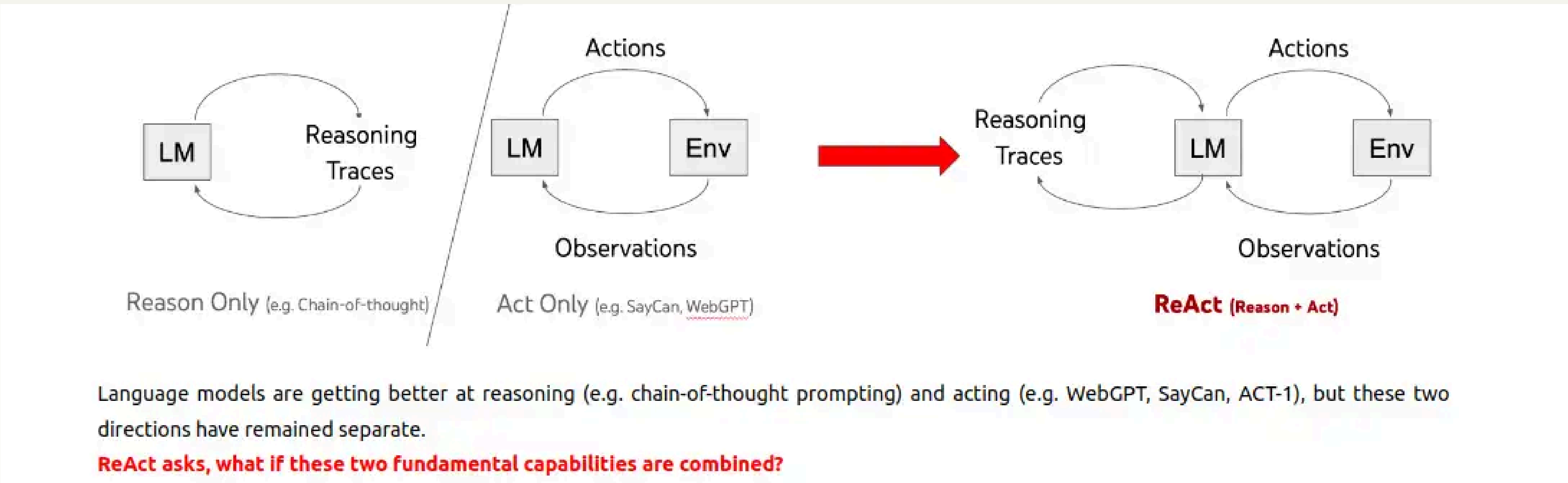
Task Decomposition for Agentic AI Planning



- **Decomposition-First:** A structured, step-by-step approach where all sub-goals are planned before any execution. Suitable for stable environments where the task is well-defined and unlikely to change during execution.
- **Interleaved:** A flexible, adaptive method where planning and execution happen concurrently. This approach is ideal for dynamic environments where real-time feedback and adjustments are essential.

Source: What is Agentic AI Planning Pattern?

ReAct Framework



- **The ReAct framework combines Reasoning and Acting within a single loop. Here, the language model alternates between Reasoning Traces and Actions in the environment.**
- **Process:**
  - The model first reasons about the task, creating a "thought" or hypothesis about what should be done next.
  - It then takes an action in the environment based on its reasoning.
  - After performing the action, the model observes the outcome in the environment, which it incorporates into its next reasoning step.
- **This cycle of reasoning, acting, and observing continues iteratively, allowing the model to learn and adapt based on real-time feedback from the environment.**
- **Significance:** By integrating reasoning and acting, ReAct allows the model to break down complex, multi-step tasks into manageable steps, adjust based on outcomes, and work towards solutions that require both planning and interaction.

Source: What is Agentic AI Planning Pattern?

Planning Pattern using ReAct with LangChain

Simple Web Search tool

```
from langchain_community.tools.tavily_search import TavilySearchResults
from langchain_core.tools import tool
import requests
import json
tv_search = TavilySearchResults(max_results=3, search_depth='advanced',
                               max_tokens=10000)

@tool
def search_web(query: str) -> list:
   """Search the web for a query."""
   tavily_tool = TavilySearchResults(max_results=2)
   results = tavily_tool.invoke(query)
   return results
```

Test Tool Calling with LLM

```
from langchain_openai import ChatOpenAI
chatgpt = ChatOpenAI(model="gpt-4o", temperature=0)
tools = [search_web]
chatgpt_with_tools = chatgpt.bind_tools(tools)
prompt = "What are the names of Ballon d'Or winners since its inception?"
response = chatgpt_with_tools.invoke(prompt)
response.tool_calls
```

Output

```
[{'name': 'search_web',
  'args': {'query': "list of Ballon d'Or winners"},
  'id': 'call_FW0h6OpObqVQAIJnOtGLJAXe',
  'type': 'tool_call'}]
```

Source: What is Agentic AI Planning Pattern?

Planning Pattern using ReAct with LangChain

Build and Test AI Agent

```
from langchain_core.prompts import ChatPromptTemplate, MessagesPlaceholder
SYS_PROMPT = """You run in a loop of Thought, Action, PAUSE, Observation.
               At the end of the loop, you output an Answer.
               Use Thought to describe your thoughts about the question you have been asked.
               Use Action to run one of the actions available to you - then return PAUSE.
               Observation will be the result of running those actions.
               wikipedia:
               e.g. wikipedia: Ballon d'Or
               Returns a summary from searching Wikipedia.
               Use the following format:
               Question: the input question you must answer
               Thought: you should always think about what to do
               Action: the action to take, should be one of [Wikipedia, duckduckgo_search, Calculator]
               Action Input: the input to the action
               Observation: the result of the action
               ... (this Thought/Action/Action Input/Observation can repeat N times)
               Thought: I now know the final answer
               Final Answer: the final answer to the original input question
             """
prompt_template = ChatPromptTemplate.from_messages(
   [
       ("system", SYS_PROMPT),
       MessagesPlaceholder(variable_name="history", optional=True),
       ("human", "{query}"),
       MessagesPlaceholder(variable_name="agent_scratchpad"),
   ]
)
```

Source: What is Agentic AI Planning Pattern?

Planning Pattern using ReAct with LangChain

Build and Test AI Agent

```
from langchain.agents import create_tool_calling_agent
agent = create_tool_calling_agent(chatgpt, tools, prompt_template)
```

```
from langchain.agents import AgentExecutor
agent_executor = AgentExecutor(agent=agent, tools=tools, verbose = True)
```

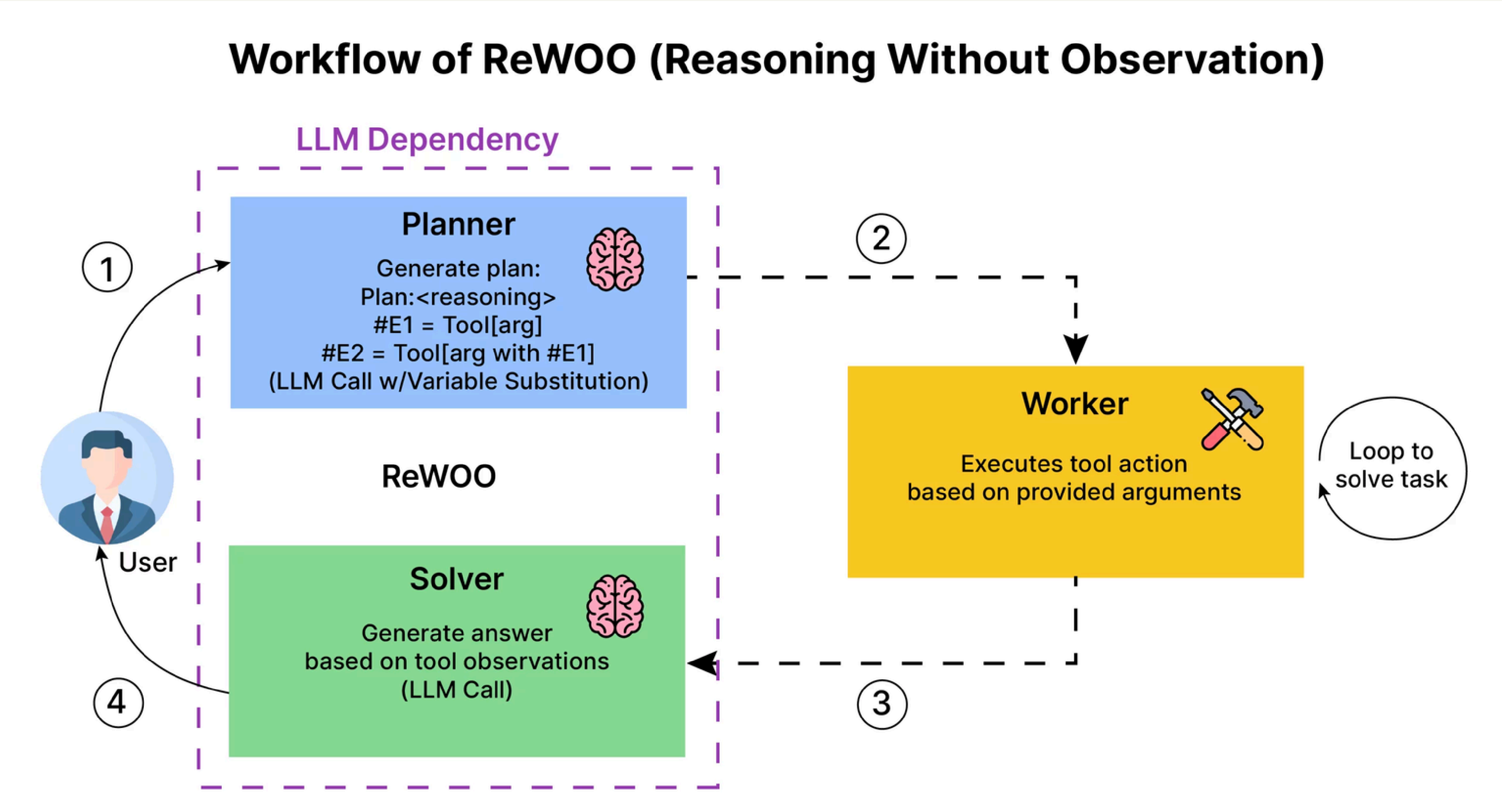
```
query = """Tell me the Ballon d'Or winners since it started?
       """
response = agent_executor.invoke({"query": query})

from IPython.display import display, Markdown

display(Markdown(response['output']))
```

The Ballon d'Or is an annual football award presented by France Football. It has been awarded since 1956, and here is a summary of the winners over the years:

- **1956**: Stanley Matthews (Blackpool, England)
- **1957**: Alfredo Di Stéfano (Real Madrid, Spain)
- **1958**: Raymond Kopa (Real Madrid, France)
- **1959**: Alfredo Di Stéfano (Real Madrid, Spain)
- **1960**: Luis Suárez (Barcelona, Spain)
- **1961**: Omar Sívori (Juventus, Italy)
- **1962**: Josef Masopust (Dukla Prague, Czechoslovakia)
- **1963**: Lev Yashin (Dynamo Moscow, Soviet Union)
- **1964**: Denis Law (Manchester United, Scotland)
- **1965**: Eusébio (Benfica, Portugal)
- **1966**: Bobby Charlton (Manchester United, England)
- **1967**: Florian Albert (Ferencváros, Hungary)
- **1968**: George Best (Manchester United, Northern Ireland)

Source: What is Agentic AI Planning Pattern?

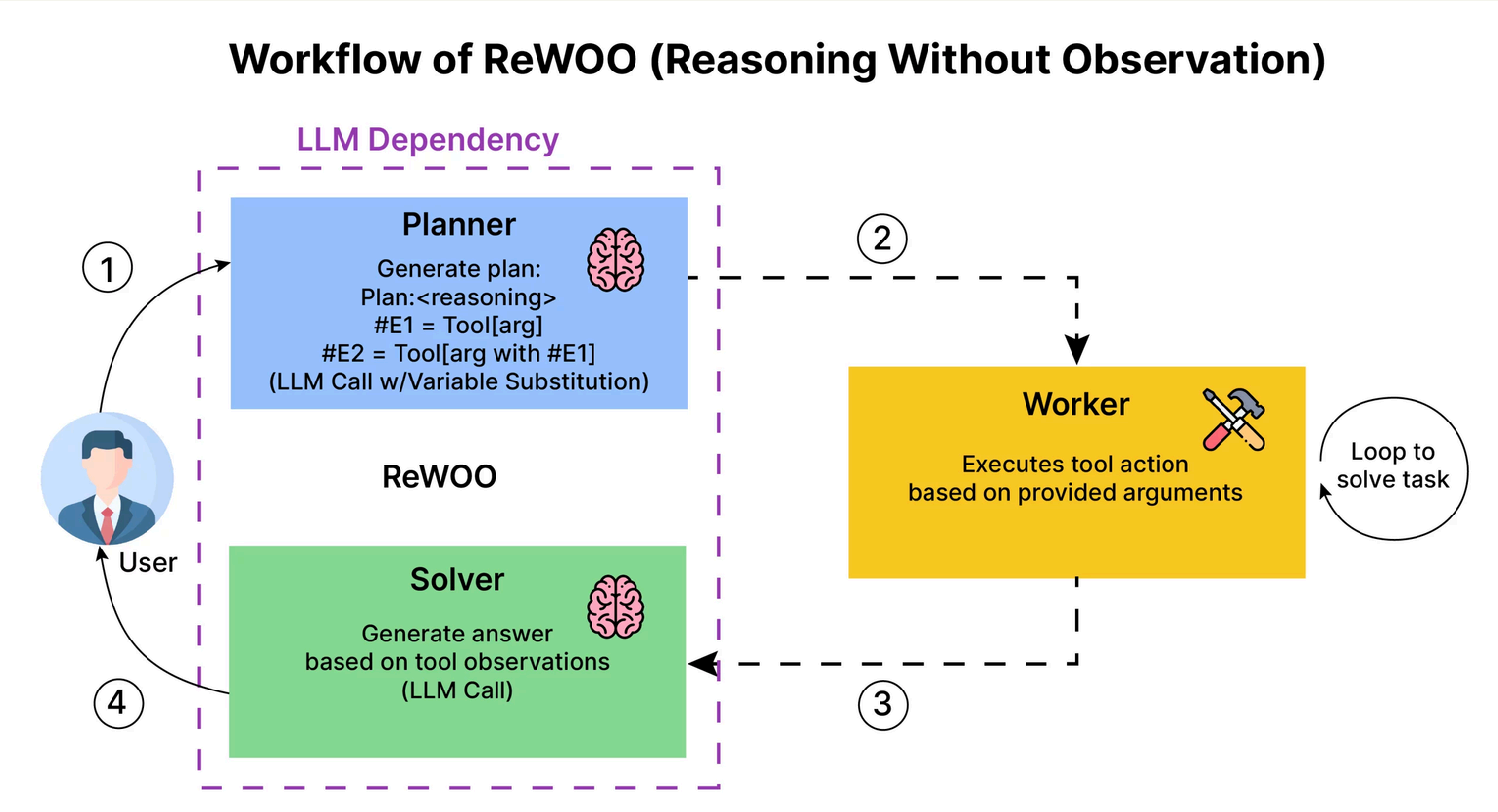
ReWOO Framework



- **Planner:**
  - The Planner is responsible for creating an entire plan at the beginning. It determines the sequence of actions or steps needed to solve the task.
  - **For each action step, the Planner specifies:**
    - **Tool:** The specific tool or function required for the step.
    - **Arguments (args):** The input values or variables needed for the tool.
  - The plan is defined using variable substitution, where the output of one tool (e.g., #E1) can be used as an argument in another tool (e.g., #E2), creating dependencies across steps.
  - Importantly, this planning process occurs in a single LLM call, making it more efficient by reducing token consumption than iterative, observation-based reasoning.
- **Worker:**
  - The Worker is responsible for executing the actions per the plan the Planner generated.
  - The Worker takes the arguments provided for each step, invokes the specified tool, and returns the result.
  - This execution can be looped until the task is solved, ensuring each tool action is completed in the correct order as outlined in the plan.
  - The Worker functions independently of the LLM, meaning it simply follows the Planner's instructions without additional calls to the LLM at each step.

Source: What is Agentic AI Planning Pattern?

ReWOO Framework



- **Solver:**
  - The Solver is the final component that interprets the results of the tools used by the Worker.
  - Based on the observations gathered from tool executions, the Solver generates the final answer to the user's query or task.
  - This part may involve a final LLM call to synthesize the information into a coherent response.

Source: What is Agentic AI Planning Pattern?

Detailed Article

What is Agentic AI Planning Pattern?

Pankaj Singh
Last Updated : 07 Nov, 2024

23 min read

This is the third article of the series, Agentic AI Design Patterns; here, we will talk about the Agentic AI Planning Pattern. Let's refresh what we have learned in the two articles – We have studied how agents can reflect and use tools to access information. In the Reflection pattern, we have seen the AI agents using the iterative process of generation and self-assessment to improve the output quality. Next, the Tool use pattern is a crucial mechanism that enables AI to interact with external systems, APIs, or resources beyond its internal capabilities.

You can find both the articles here:

- The Reflection Pattern
- The Tool Pattern

Also, here are the 4 Agentic AI Design Patterns: Top 4 Agentic AI Design Patterns for Architecting AI Systems.

Now, talking about the Planning Pattern. Let's take an example of a smart assistant who doesn't only reflect and pull in outside information when needed but also decides the sequence of steps to solve a bigger problem. Pretty cool, right? But here's where it gets really interesting: how does this assistant decide on the best sequence of steps to accomplish big, multi-layered goals? Effective **planning** is determining a structured sequence of actions to complete complex, multi-step objectives.

What does a planning pattern provide?

Planning Patterns provide strategies for language models to divide large tasks into manageable subgoals, enabling them to tackle intricate challenges step-by-step while keeping the overarching goal in focus. This article will discuss the Planning pattern in detail with the ReAct and ReWOO techniques.

CHECK OUT THE DETAILED ARTICLE HERE