𝗜𝗳 𝘆𝗼𝘂 𝗯𝘂𝗶𝗹𝗱 𝗔𝗜 𝗔𝗴𝗲𝗻𝘁 𝗮𝗽𝗽𝗹𝗶𝗰𝗮𝘁𝗶𝗼𝗻𝘀, 𝘆𝗼𝘂 𝗡𝗘𝗘𝗗 𝗧𝗢 𝗞𝗡𝗢𝗪 𝘁𝗵𝗶𝘀 𝘀𝗶𝘅 𝗱𝗲𝘀𝗶𝗴𝗻 𝗽𝗮𝘁𝘁𝗲𝗿𝗻𝘀! 🛠️  
  
AI Agents are evolving fast, opening up many possibilities for a new paradigm of applications. In today's world, we use LLMs mostly in zero-shot mode, in which a model generates final output token by token without revising its work. With an agent workflow, we can ask the LLM to iterate over certain output many times.   
  
There are many tools and possibilities for AI agents today, creating both exciting opportunities and a lot of noise. To cut through the confusion, here’s a framework of six key design patterns you can leverage to build powerful, scalable agentic applications:  
  
Let’s break it down: ⬇️  
  
1. 𝗥𝗲𝗔𝗰𝘁 𝗔𝗴𝗲𝗻𝘁   
Thinks, takes action, looks at the result, repeats.   
Classic loop: “Should I Google this?” → Does it → Adjusts.   
→ Used in most AI products today (like basic chat assistants).  
  
2. 𝗖𝗼𝗱𝗲𝗔𝗰𝘁 𝗔𝗴𝗲𝗻𝘁:  
Runs real code, not just JSON.   
So instead of saying “call API X,” it writes and runs a Python script.   
→ More powerful. Used in research agents and dev assistants.  
  
3. 𝗠𝗼𝗱𝗲𝗿𝗻 𝗧𝗼𝗼𝗹 𝗨𝘀𝗲:  
Sends tasks to smart APIs (search, cloud, data), and lets them do the heavy lifting.   
The agent mostly routes and formats info.   
→ Think: a smart middleman between you and powerful services.  
  
4. 𝗦𝗲𝗹𝗳-𝗥𝗲𝗳𝗹𝗲𝗰𝘁𝗶𝗼𝗻:  
Agent checks its own work.   
Did it make a mistake? It catches it, critiques it, and tries again.   
→ Most AI errors happen \*\*because this step is missing.\*\*  
  
5. 𝗠𝘂𝗹𝘁𝗶-𝗔𝗴𝗲𝗻𝘁 𝗪𝗼𝗿𝗸𝗳𝗹𝗼𝘄:   
One agent isn’t doing everything.   
You have a planner, a researcher, and a writer — all working together.   
→ Like a mini team of AIs. More accurate. Less chaos.  
  
6. 𝗔𝗴𝗲𝗻𝘁𝗶𝗰 𝗥𝗔𝗚:   
This is what powers apps like Perplexity.   
The agent looks stuff up (retrieval), thinks about it, uses tools, and gives you a smarter answer.   
→ Works with real-time data, not just model memory.  
  
𝗨𝗻𝗱𝗲𝗿𝘀𝘁𝗮𝗻𝗱𝗶𝗻𝗴 𝗔𝗜 𝗮𝗴𝗲𝗻𝘁 𝗱𝗲𝘀𝗶𝗴𝗻 𝗽𝗮𝘁𝘁𝗲𝗿𝗻𝘀 𝗰𝗮𝗻 𝗶𝗺𝗽𝗿𝗼𝘃𝗲 𝗽𝗿𝗼𝗯𝗹𝗲𝗺-𝘀𝗼𝗹𝘃𝗶𝗻𝗴, 𝗰𝗼𝗹𝗹𝗮𝗯𝗼𝗿𝗮𝘁𝗶𝗼𝗻, 𝗮𝗻𝗱 𝗽𝗲𝗿𝗳𝗼𝗿𝗺𝗮𝗻𝗰𝗲, 𝗹𝗲𝗮𝗱𝗶𝗻𝗴 𝘁𝗼 𝗺𝗼𝗿𝗲 𝘀𝗼𝗽𝗵𝗶𝘀𝘁𝗶𝗰𝗮𝘁𝗲𝗱 𝗮𝗻𝗱 𝗿𝗲𝗹𝗶𝗮𝗯𝗹𝗲 𝗔𝗜 𝗮𝗽𝗽𝗹𝗶𝗰𝗮𝘁𝗶𝗼𝗻𝘀. 𝗣𝗿𝗼𝗺𝗽𝘁𝘀 𝗮𝗹𝗼𝗻𝗲 𝘄𝗼𝗻’𝘁 𝗴𝗲𝘁 𝘆𝗼𝘂 𝘁𝗼 𝗽𝗿𝗼𝗱𝘂𝗰𝘁𝗶𝗼𝗻   
→ 𝗟𝗟𝗠𝘀 𝗮𝗿𝗲 𝘀𝗺𝗮𝗿𝘁 — 𝗯𝘂𝘁 𝘀𝘁𝗮𝘁𝗶𝗰   
→ 𝗦𝘆𝘀𝘁𝗲𝗺𝘀 𝘄𝗶𝗻. 𝗗𝗲𝘀𝗶𝗴𝗻 𝗺𝗮𝘁𝘁𝗲𝗿𝘀.  
  
[𝗡𝗼𝘁𝗲 𝘁𝗵𝗮𝘁 𝘁𝗵𝗲𝘀𝗲 𝗽𝗮𝘁𝘁𝗲𝗿𝗻𝘀 𝗰𝗮𝗻 𝗯𝗲 𝘂𝘀𝗲𝗱 𝘁𝗼𝗴𝗲𝘁𝗵𝗲𝗿, 𝗻𝗼𝘁 𝗷𝘂𝘀𝘁 𝗲𝘅𝗰𝗹𝘂𝘀𝗶𝘃𝗲𝗹𝘆.] 🛠️  
  
Kudos to [**Rakesh Gohel**](https://www.linkedin.com/in/ACoAAAGJmnYB5cHnVnBiVMGRCJ75ZsrVrhTqA9M) for this excellent visualization!  
  
𝗜 𝗲𝘅𝗽𝗹𝗼𝗿𝗲 𝘁𝗵𝗲𝘀𝗲 𝗱𝗲𝘃𝗲𝗹𝗼𝗽𝗺𝗲𝗻𝘁𝘀 — 𝗮𝗻𝗱 𝘄𝗵𝗮𝘁 𝘁𝗵𝗲𝘆 𝗺𝗲𝗮𝗻 𝗳𝗼𝗿 𝗿𝗲𝗮𝗹-𝘄𝗼𝗿𝗹𝗱 𝘂𝘀𝗲 𝗰𝗮𝘀𝗲𝘀 — 𝗶𝗻 𝗺𝘆 𝘄𝗲𝗲𝗸𝗹𝘆 𝗻𝗲𝘄𝘀𝗹𝗲𝘁𝘁𝗲𝗿. 𝗬𝗼𝘂 𝗰𝗮𝗻 𝘀𝘂𝗯𝘀𝗰𝗿𝗶𝗯𝗲 𝗵𝗲𝗿𝗲 𝗳𝗼𝗿 𝗳𝗿𝗲𝗲: [**https://lnkd.in/dbf74Y9E**](https://lnkd.in/dbf74Y9E)

Activate to view larger image,

