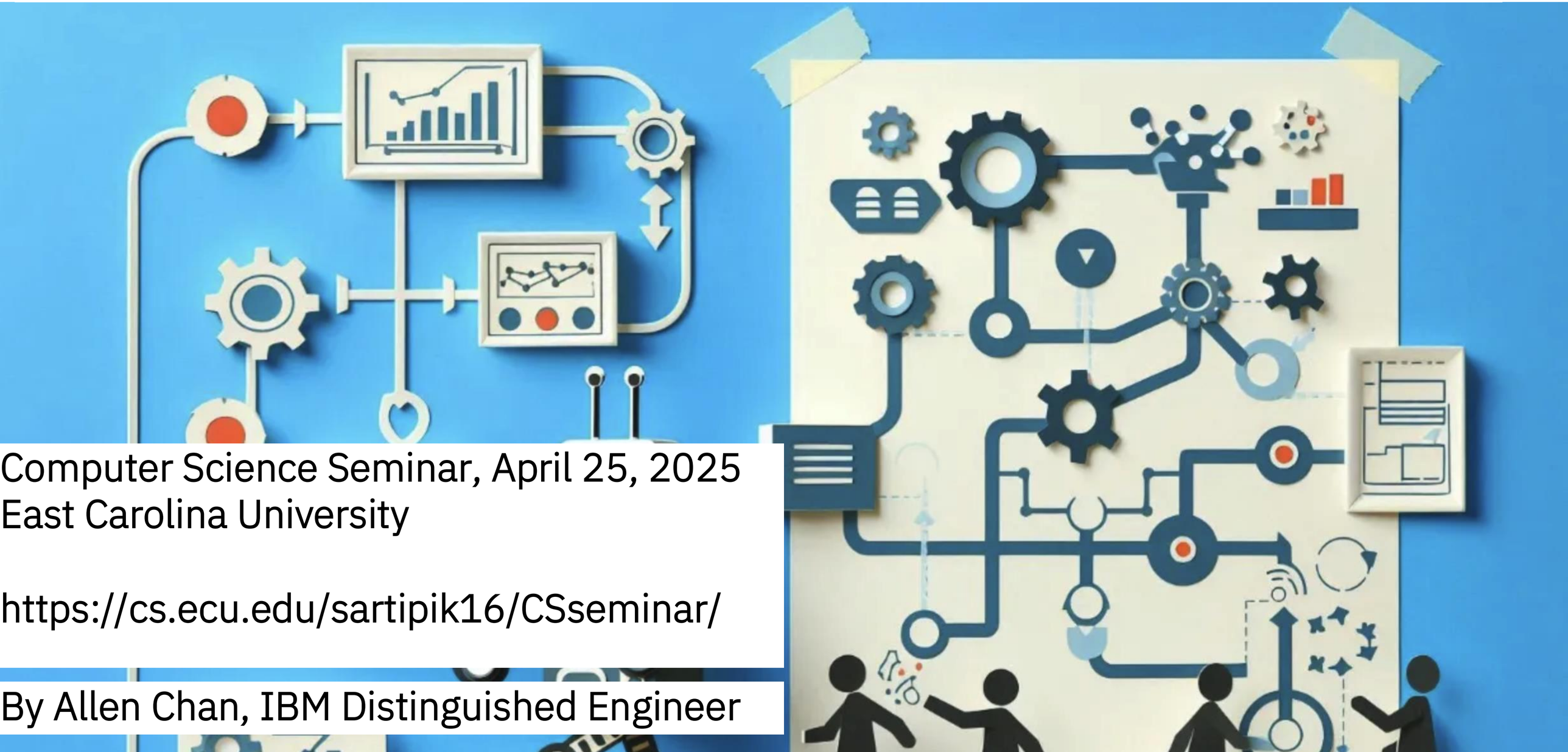
Strategic Design of AI Agents and Agentic Patterns



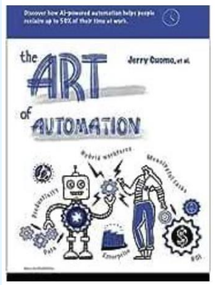
Computer Science Seminar, April 25, 2025
East Carolina University

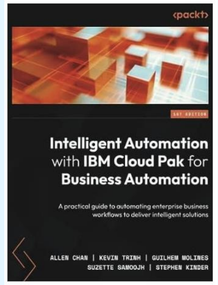
https://cs.ecu.edu/sartipik16/CSseminar/

By Allen Chan, IBM Distinguished Engineer

Allen Chan

Co-Author of 2 books on Automation and AI





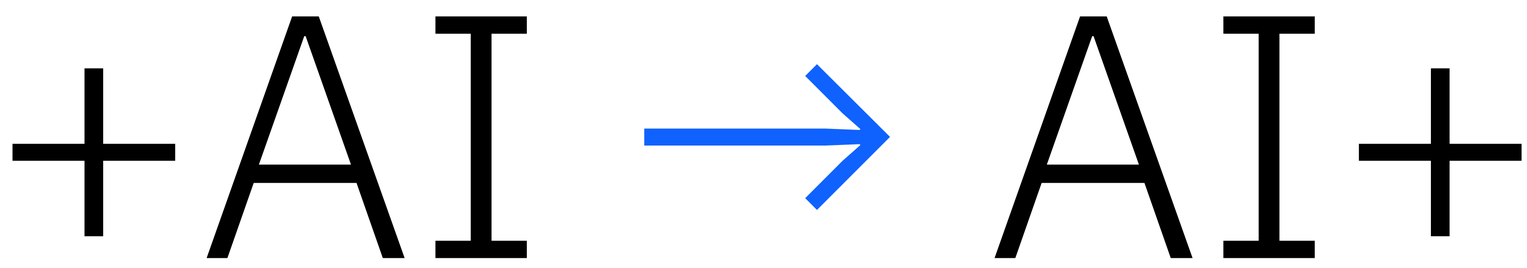
Recent publications on medium.com about AI and AI Agents

1. AI Agents together with Fixed Processes: A new frontier. April 2025.
2. Choosing the right AI Agent Strategy. March 2025.
3. How Tool Complexity Impacts AI Agents Selection Accuracy. Feb 2025.
4. How many tools/functions can an AI Agent has? Feb 2025

For the entire list, go to my GenAI and AI Agent series.

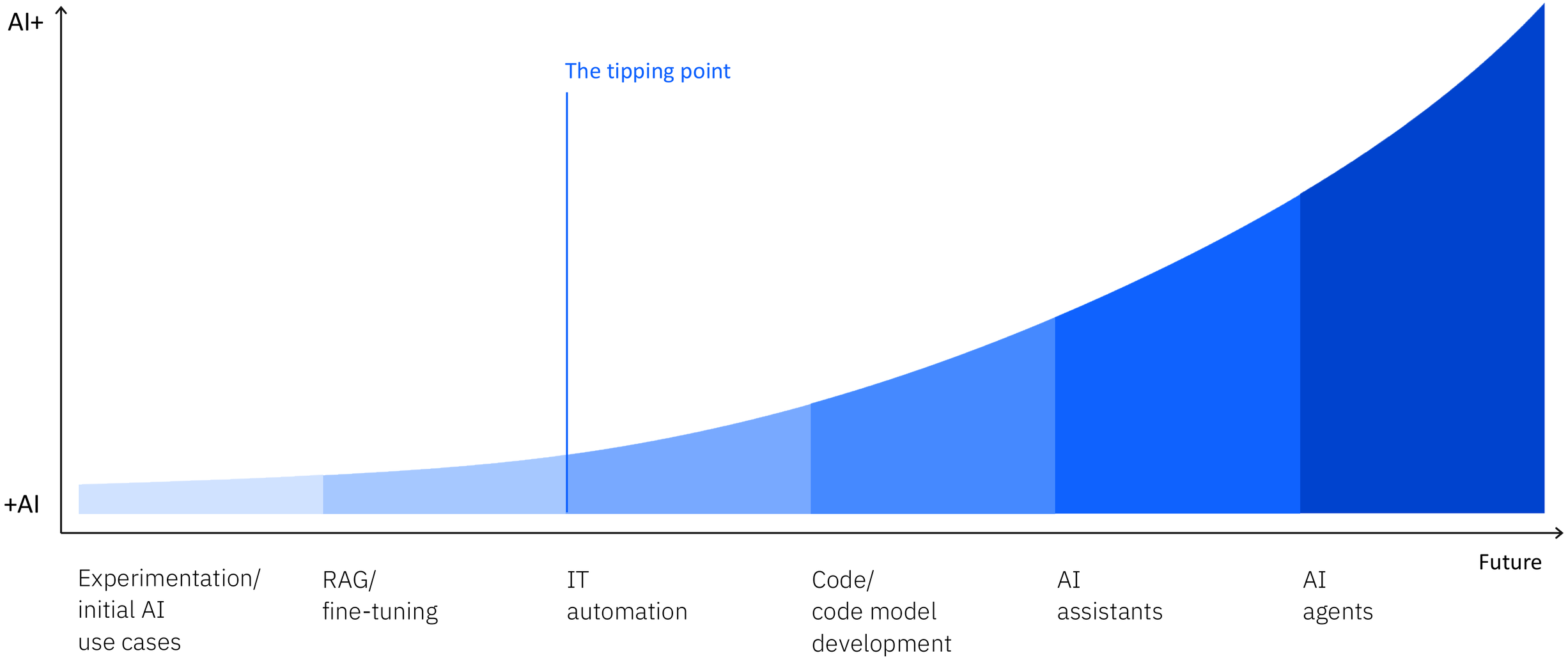
Recent collaboration with University of Toronto and York University on AI Research

1. Transformer Models for Activity Mining in Knowledge-Intensive Processes. Feb 2023. Book: Business Process Management Workshops
2. Managing and Simplifying Cognitive Business Operations Using Process Architecture Models. May 2019. Book: Advanced Information Systems Engineering
3. Solution Patterns for Machine Learning. May 2019. Book: Advanced Information System Engineering.
4. Modeling and Analyzing Process Architecture for Context-Driven Adaptation: Designing Cognitively-Enhanced Business Processes for Enterprises. Oct 2018. 2018 IEEE 22nd International Enterprise Distributed Object Computing Conference (EDOC)
5. Designing Process Architectures for User Engagement with Enterprise Cognitive Systems. Oct 2017. IFIP Working Conference on The Practice of Enterprise Modeling



Reinventing how work gets done across business domains and industries

AI value creation curve

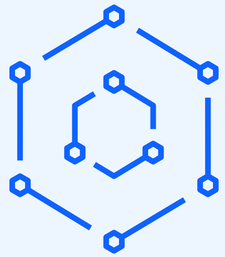


Models

- Problem solving
- Logical thinking
- Reasoning

Assistants

- Information retrieval
- Prescriptive tasks
- Single-step processes



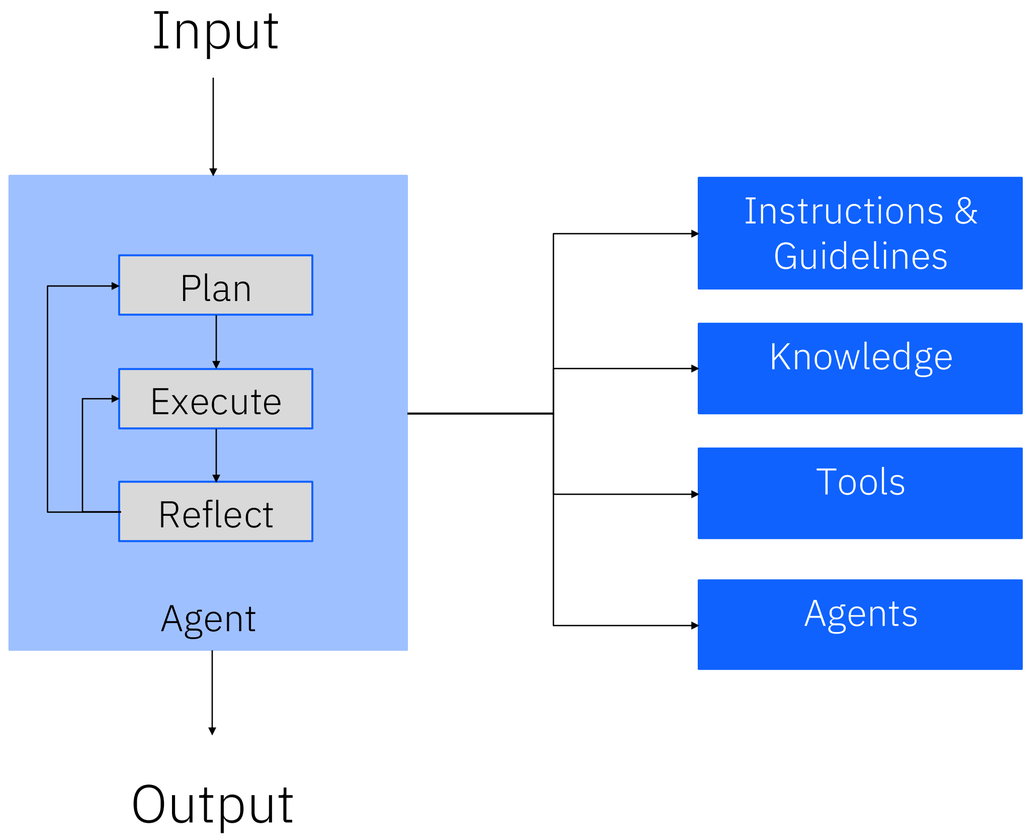
Agents

- Multistep processes
- Autonomous action taking
- Self-correcting

What is an AI Agent?

There is no formal definition of an AI Agent but here is the one we're using (at the moment …)

An AI agent is an application that acts autonomously to understand, plan, and execute a request (from a human or another agent). AI agents use LLMs to reason and reflect, and can interface with tools, other models, and other IT systems to fulfill user goals.

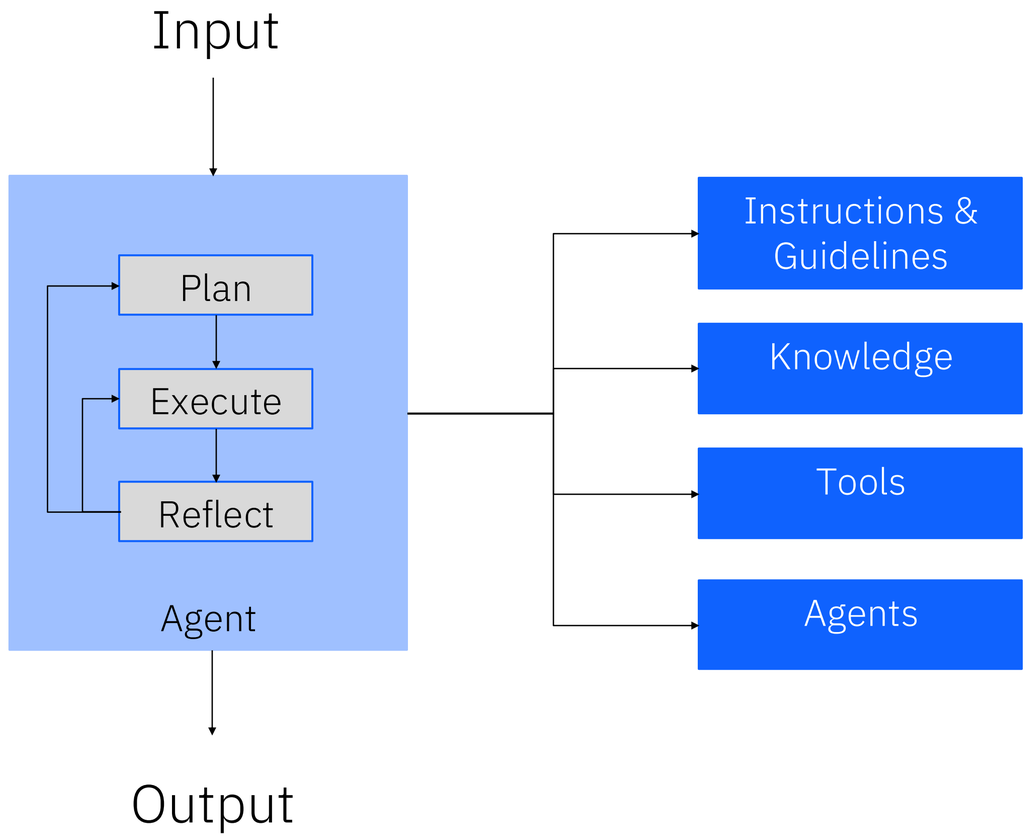


How to build an Agent?

Basically, we need to:

1. Specify Instructions & Guidelines
2. Identify Knowledge Source (optional)
3. Identify Available Tools (optional)
4. Implement Reflection Loop (optional)
5. Test and Repeat 1 – 4 until the desired behavior is achieved

Depending on the use case, there are different “strategies” in how we can specify the instructions and how we can implement the reflection mechanism.



Major Strategies

Chain of Thought (CoT)

For step-by-step reasoning

Reasoning + Acting (ReAct)

For real-world interaction and tool use

Plan and Execute

For planning long horizon tasks

Reasoning WithOut Observation (ReWOO)

For iterative self-improvement

Chain Of Thoughts

How It Works

- Chain-of-Thought (CoT) **forces the model to break down its thought process** into explicit reasoning steps before giving a final answer.
- This method improves **logical reasoning** and **complex problem-solving**.

For the full example prompt, please go to my article on medium.com.

See https://achan2013.medium.com/b72625bf49f3?sk=7e3da38958e6700881ed950b68b3368a

System Instructions:
1. You are an ****AI-powered travel planner**** that helps users organize trips by providing detailed itineraries, budget-friendly recommendations, and travel tips.
2. You ****always**** follow a structured step-by-step Chain-of-Thought (CoT) reasoning process. You will **always** prefix the reasoning with "<reasoning>" phrase, and end with the phrase "</reasoning>".
3. Finally, you will create a a daily itinerary with the details for each day, followed by a budget breakdown and a list of key travel tips. You will **always** prefix the final response with "<response>" phrase, and end with the phrase "</response>".

Example Interaction:
****User Query:****
"I want to plan a 5-day trip to Tokyo on a budget. Can you help?"
<reasoning>
Step 1: Understand User Preferences
- Destination: Tokyo
- Duration: 5 days

. . . See article for full prompt . . .

</reasoning>
<response>
"Here's a detailed 5-day ****budget-friendly** Tokyo itinerary with flight, stay, activities, and cost estimates! Let me know if you'd like adjustments."**
Please specify the final itinerary here.
</response>

Chain Of Thoughts

Pros and Cons

- ✅ Best for math and logic problems
- ✅ Improves performance in structured reasoning tasks
- ✅ Can be implemented based on prompt logic only
- ❌ Lacks adaptability beyond predefined reasoning paths
- ❌ Can still produce hallucinated reasoning

Chain Of Thoughts is good for problems that require logical reasoning, but examples should be provided to guide the agent.

ReAct

How It Works

- **ReAct (Reasoning + Acting)** enables the model to **think, take action, observe results, and refine its approach dynamically**.

- It can **interact with tools, retrieve external data, and adjust based on outcomes**.

(LLM by itself does not call tools, but rather, just compute the tool calling signature)

For the full example prompt, please go to my article on medium.com.

See https://achan2013.medium.com/b72625bf49f3?sk=7e3da38958e6700881ed950b68b3368a

```
### System Instructions:
1. You are a **customer service AI assistant** that helps users track their orders.
2. You have access to an API function called `get_order_status` that retrieves live order details.
3. Follow the ReAct **Think → Act → Observe → Respond** process. You will *always* prefix the reasoning with "<reasoning>" phrase, and end with the phrase "</reasoning>".
4. If there is a matching function, write down the specification for the function call. If there is no matching function, just say so. You will *always* prefix the final response with "<response>" phrase, and end with the phrase "</response>".

### Available Function:
You can call the following function when needed:
tools = [{
 "name": "get_order_status",
 "description": "Retrieves the status of a customer's order, including   current status and estimated delivery date.",
 "parameters": {
  "order_id": {
   "type": "string",
   "description": "The unique identifier for the order."
  }
 }
}]
---
### Example Interaction:
#### **User Query:**
> "Where is my order #12345? I placed it last week."
<reasoning>
. . .
</reasoning>
<response>
. . .
</response>
```

ReAct

Pros and Cons

✅ Allows interaction with **external tools/APIs**

✅ More adaptive than CoT in real-world applications

✅ A proper ReAct agent will require a Reflect loop

❌ Higher latency due to multiple reasoning cycles

❌ Requires external integrations for full effectiveness

ReAct-based agent is the defacto standard when tool calling is involved.

Plan-and-Execute

How It Works

- **Plan-and-Execute** first **creates a structured plan**, then **executes each step independently**.

- For more sophisticated cases, we might choose to implement this a Plan-and-Execute loop, i.e. 1. Plan, 2. Execute 3. Replan based on previous result, 4. Execute remaining steps.

For the full example prompt, please go to my article on medium.com.

See https://achan2013.medium.com/b72625bf49f3?sk=7e3da38958e6700881ed950b68b3368a

```
1. You are a **HR onboarding assistant** that helps new employees integrate smoothly into their company.
2. You use a **Plan-and-Execute** to work through the problem, there will be 2 phrase: Planning and Execution.
3. In the Planning phase. In here, you will **create a step-by-step onboarding plan. You will *always* enclose planning steps with "<planning>" tag, and end with the tag "</planning>".
4. In the Execution phase. In here, you will specific execution steps in sequence. You will *always* enclose execution steps with "<execution>" tag, and end with the tag "</execution>".

Please response in the following format in Markdown:
<planning>
 <step>Step 1. specify step 1 details</step>
 <step>Step 2. specify step 2 details</step>
</planning>
<execution>
 <step>Step 1. identify work needed for planning step 1.</step>
 <step>Step 2. identify work needed for planning step 2.</step>
</execution>
```

Plan-and-Execute

Pros and Cons

✅ Works well for **multi-step, long-horizon tasks**

✅ **More scalable** than CoT for complex problem-solving

❌ **Planning quality depends on LLM reliability.** The quality of the single prompt plan-and-execute varies a lot across different LLM models.

❌ **Rigid separation** between planning and execution can be inefficient

Incorporating a RAG pipeline to a relevant knowledge base as part of the planning phase will improve the planning outcome.

ReWOO

How It Works

- **ReWOO** improves upon **ReAct** by allowing an agent to plan its entire tool-use strategy in a **single pass** rather than iterating step by step.

- **There are 3 steps in ReWOO:**
  - **Step 1: Planner** — Generate a Plan including Tool Calling signatures
  - **Step 2: Worker** — Execute the tools based on the plan, and store the results
  - **Step 3: Solver** — Combine the plan and result of the tool calling to formulate the final answer.

Instructions for Planner

```
### **System Instructions:**
You are an **AI-powered RFP drafting assistant** that helps organizations create well-structured Requests for Proposals (RFPs).
You follow the **ReWOO framework**, which consists of **three steps**:

1. **Planner** - Generate a structured plan for the RFP, including tool-use signatures.
2. **Worker** - Execute all required tool calls based on the plan and store results.
3. **Solver** - Combine the tool results with the plan to generate a finalized RFP.
Your goal is to **maximize efficiency** by generating all tool calls in a single pass before execution.

---
### **Tool Available:**
...
```

Instructions for Solver

```
### **System Instructions:**
You are an **AI-powered RFP Solver** responsible for assembling structured RFP documents.
You receive pre-generated **RFP section content** and your task is to:
1. **Combine all sections into a single, logically structured RFP document.**
2. **Ensure clarity, consistency, and professional formatting.**
3. **Apply appropriate headings, structure, and formatting conventions for a standard RFP.**
...
```

ReWOO

Pros and Cons

✅ Produces **higher-quality outputs** through iterative refinement
✅ Reduces **hallucinations** compared to one-shot generations
❌ **Computationally expensive** (requires multiple passes)
❌ Slower than simpler one-shot techniques

ReWOO increases output accuracy and reduces token consumption but at the expense of increasing execution time making it more efficient for structured multi-step tasks.

Balance Speed and Accuracy

1. Some strategy will take longer than other but they also produce generally more accurate results.

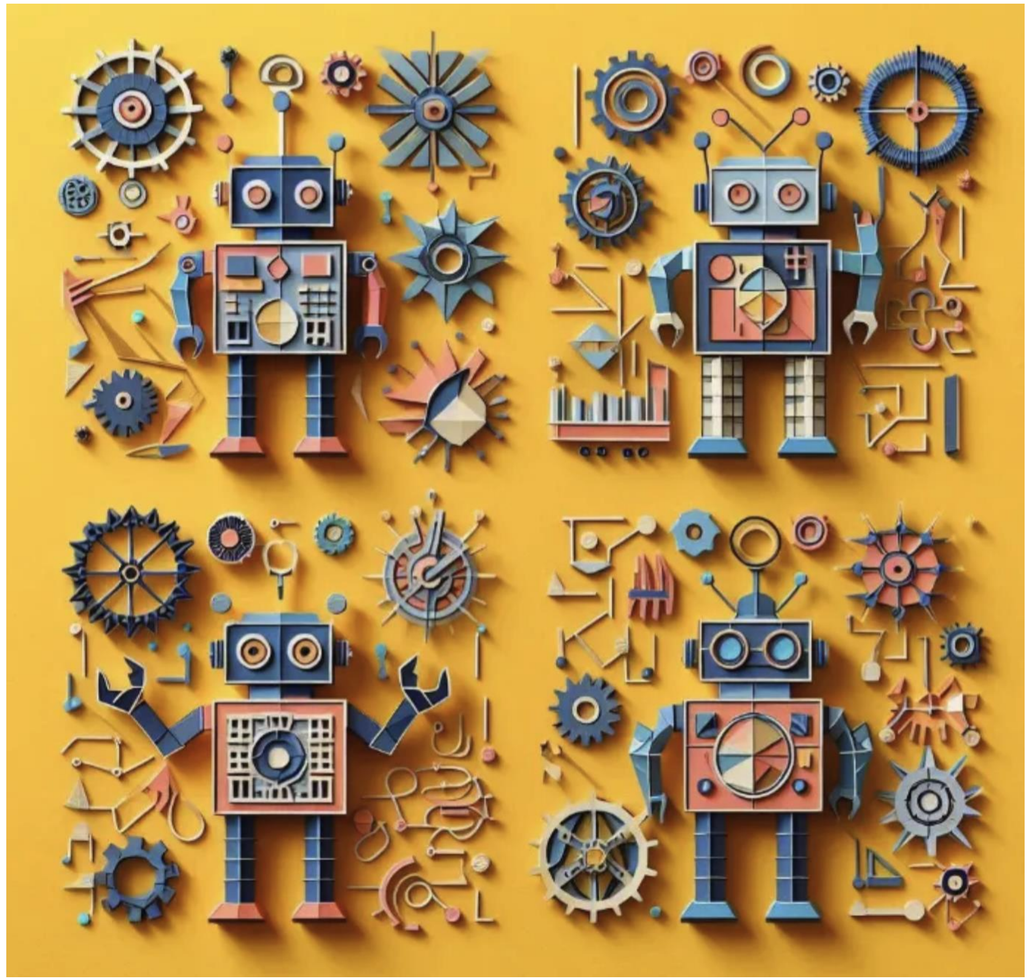
2. Asking the AI to write down the <reasoning> will help with the inference process.

3. Providing examples usually help. However, if the task is “creative” in nature, providing examples will limit the freedom of the AI.

4. Even if the AI writes down the reason, it does not been it follow through on all the reasoning points.

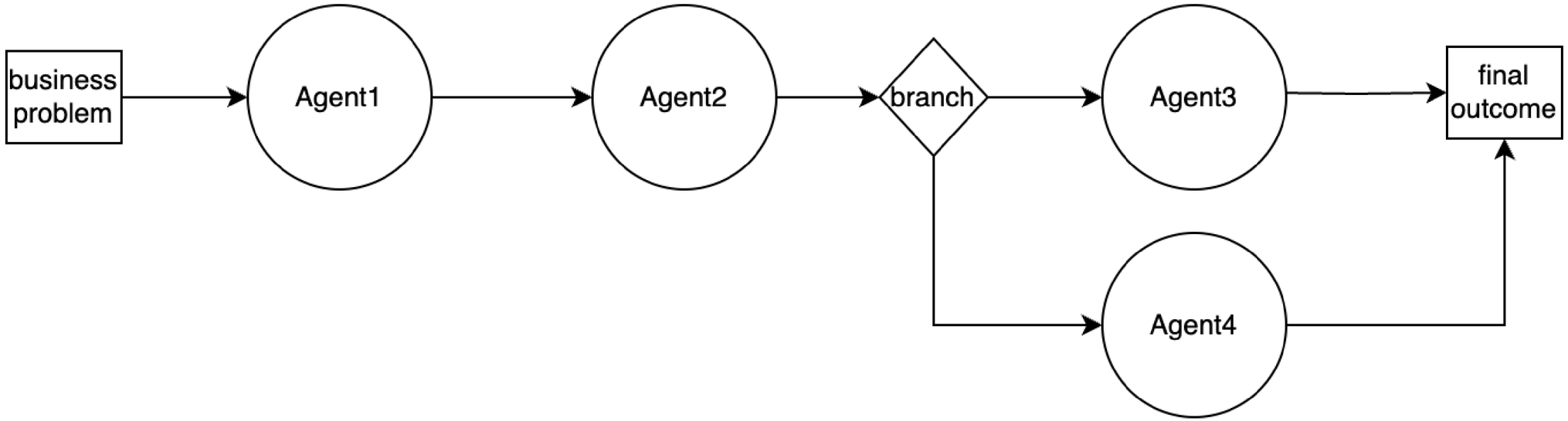
5. It is still important to make sure the answer is correct.

Agentic Patterns



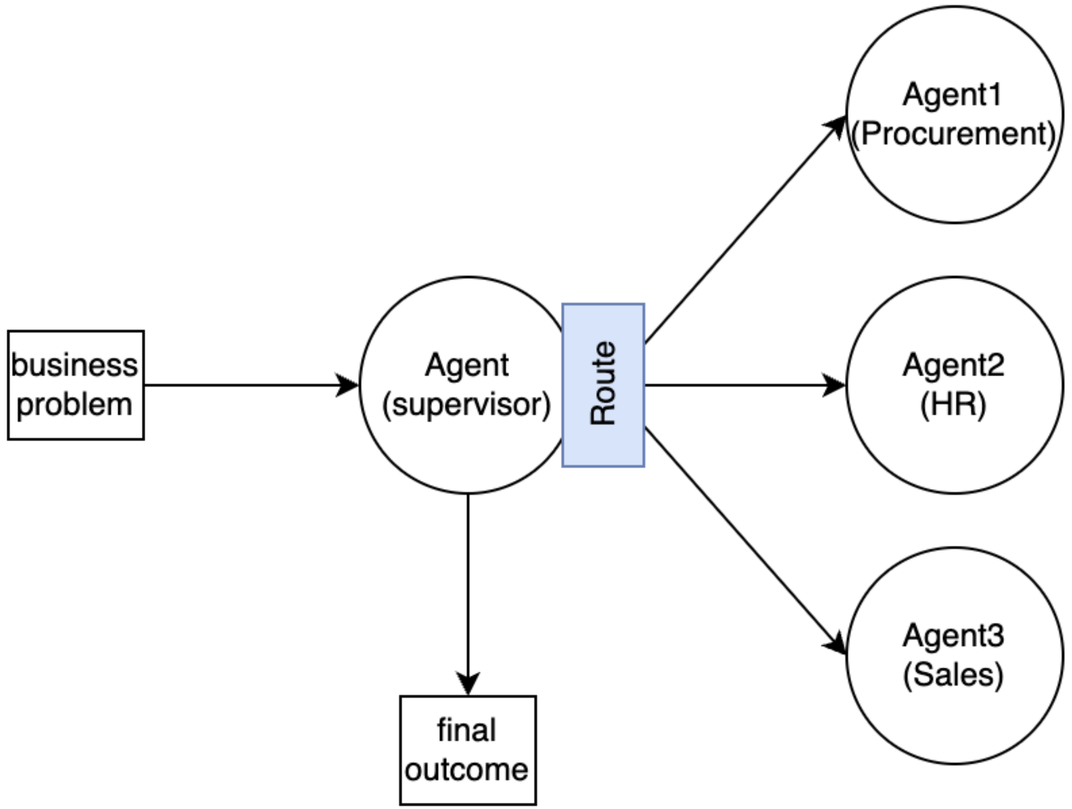
Linear Sequence (Pipeline)

1. We can have multiple agents – each trying to solve the problem at different level.
2. First agent will break down the initial problem and propose the solution to subsequent agents.
3. It is also possible each Agent is solving a different aspect of the same problem.



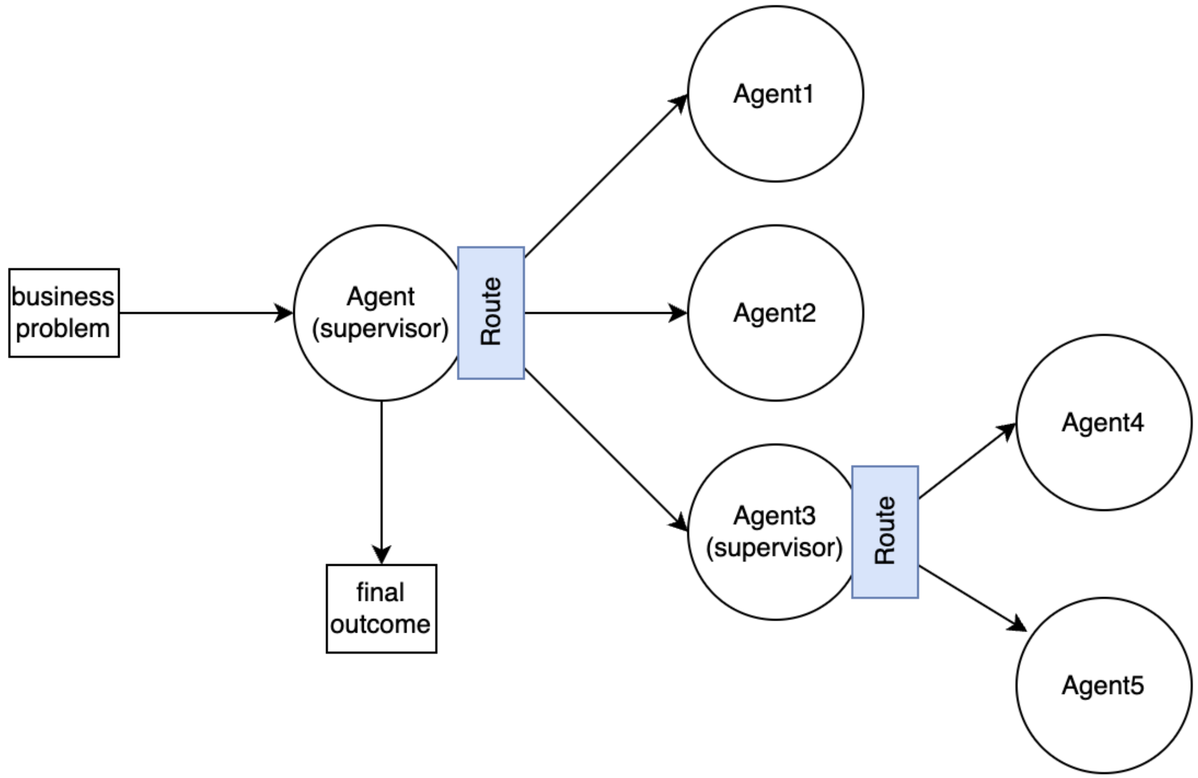
Supervisor

1. One of the first agentic patterns in practice in the industry
2. First agent (supervisor) will analyze the initial business problem and then routes the request to a sub-agent.



Multi-Level Supervisor

1. Just like we can have first-line manager and 2nd line manager, we can have multi-line agent as well.
2. First agent (supervisor) will analyze the initial business problem and then routes the request to another supervisor.

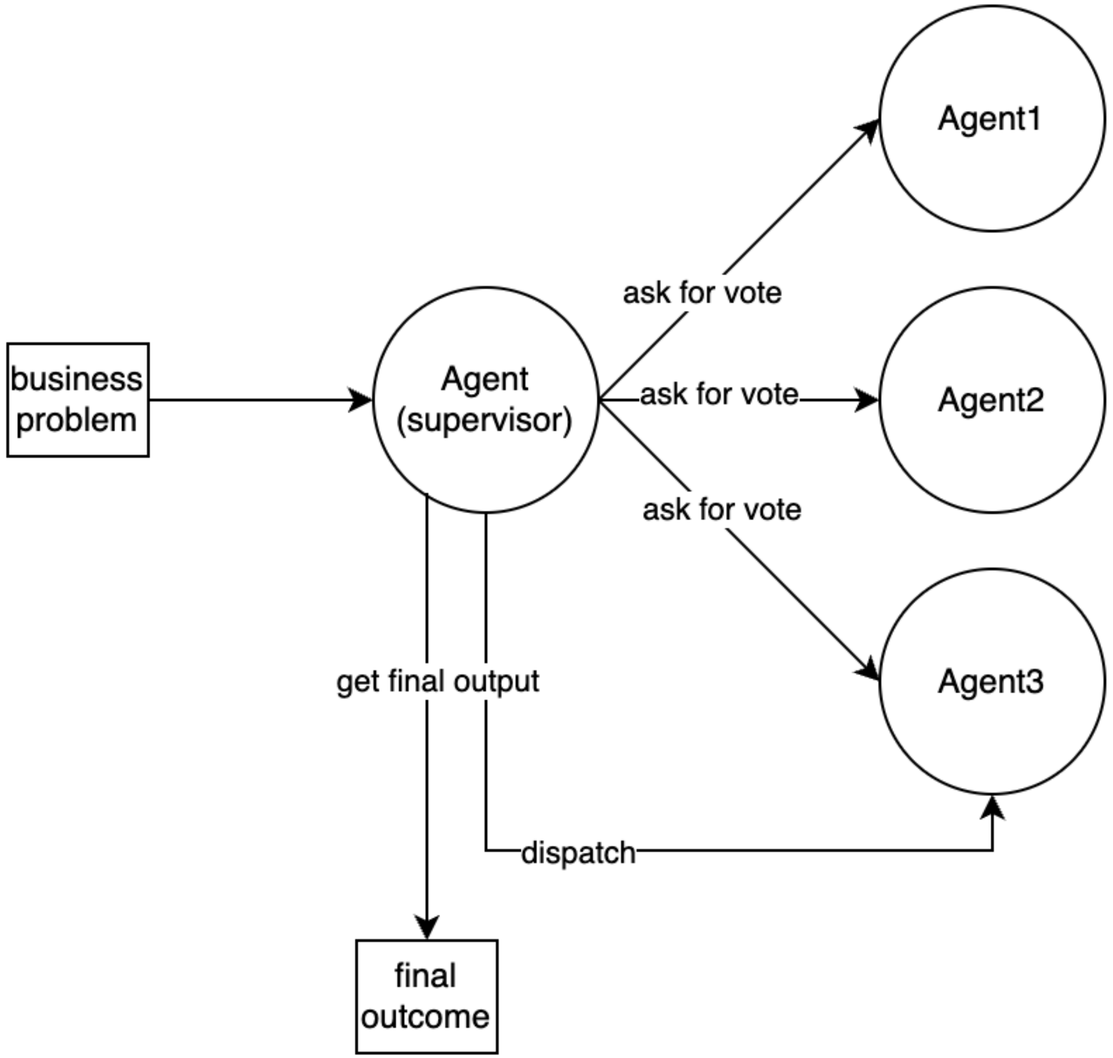


Vote and Dispatch

1. A variant of the Supervisor pattern, but in this case, the Supervisor is not routing the work directly but instead will post the problem to each sub-agent and ask their confidence level in solving the problem.

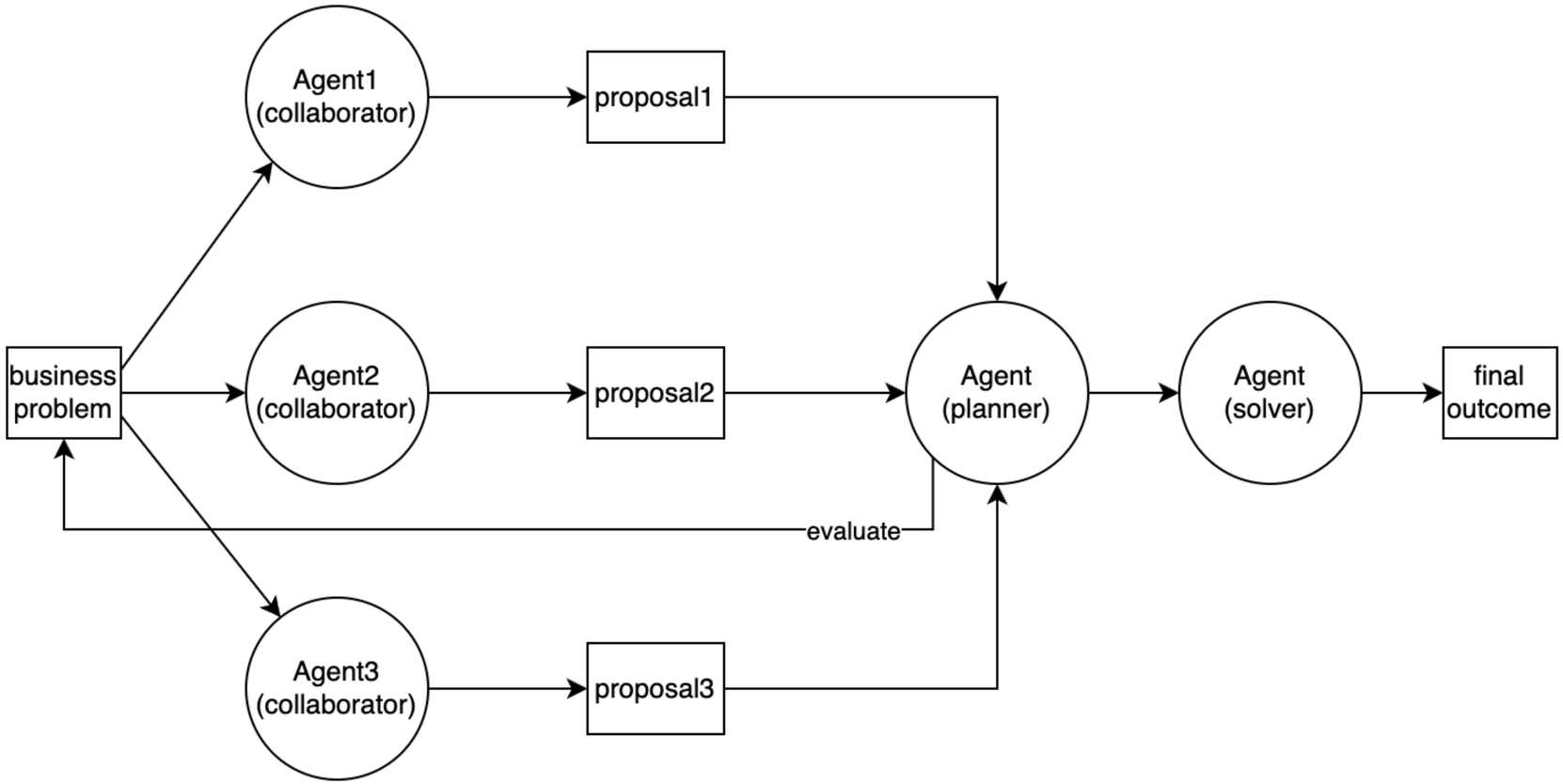
2. Agent can nominate others as well.

3. Once a majority is reached, the supervisor will dispatch the work the nominated agent.



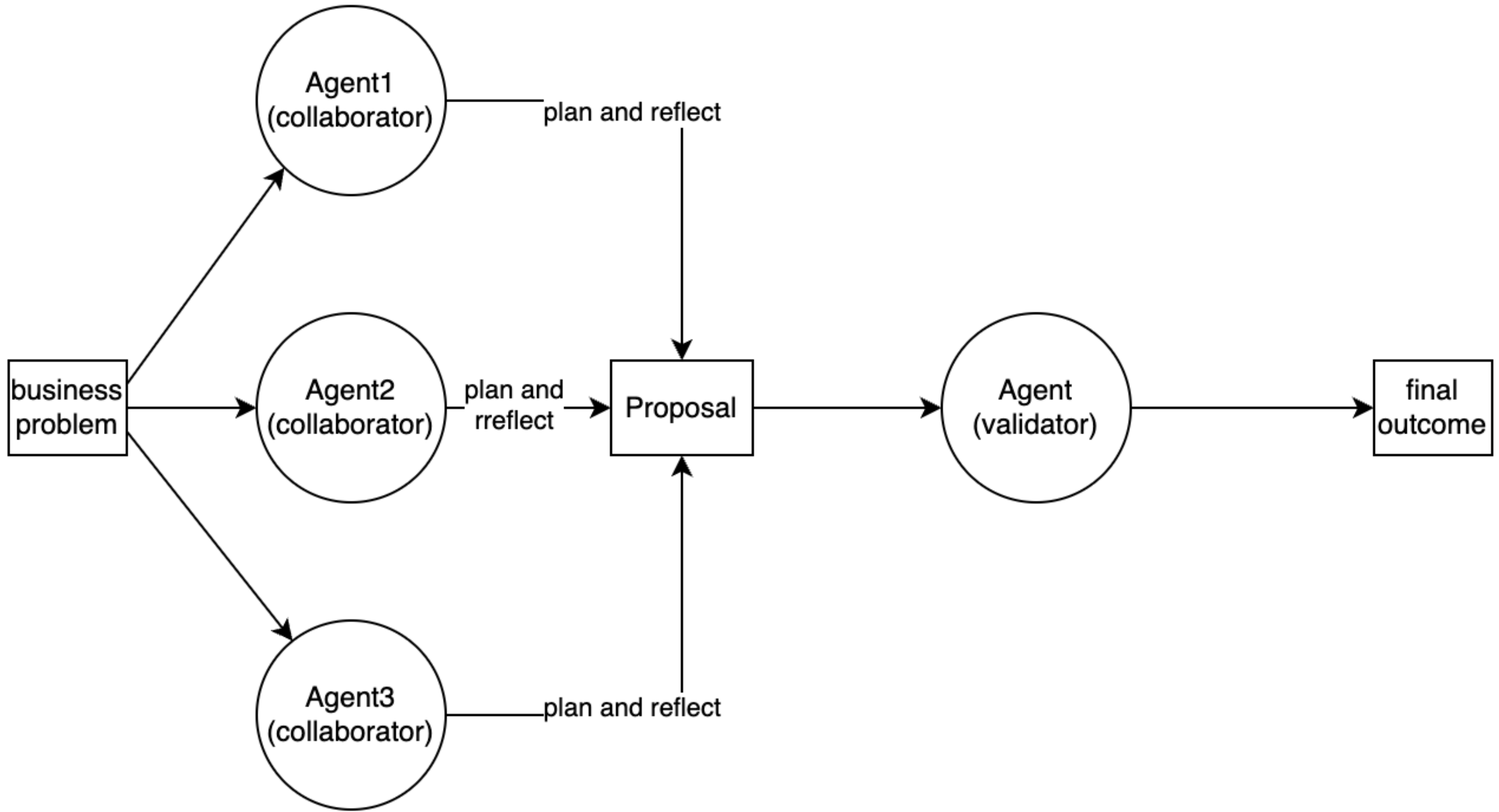
Collaborators

1. Sometimes a problem is very broad and will require expertise from multiple AI Agents.
2. Each collaborator will solve what they can, and Planner will be used to aggregate the results and propose the final solution.



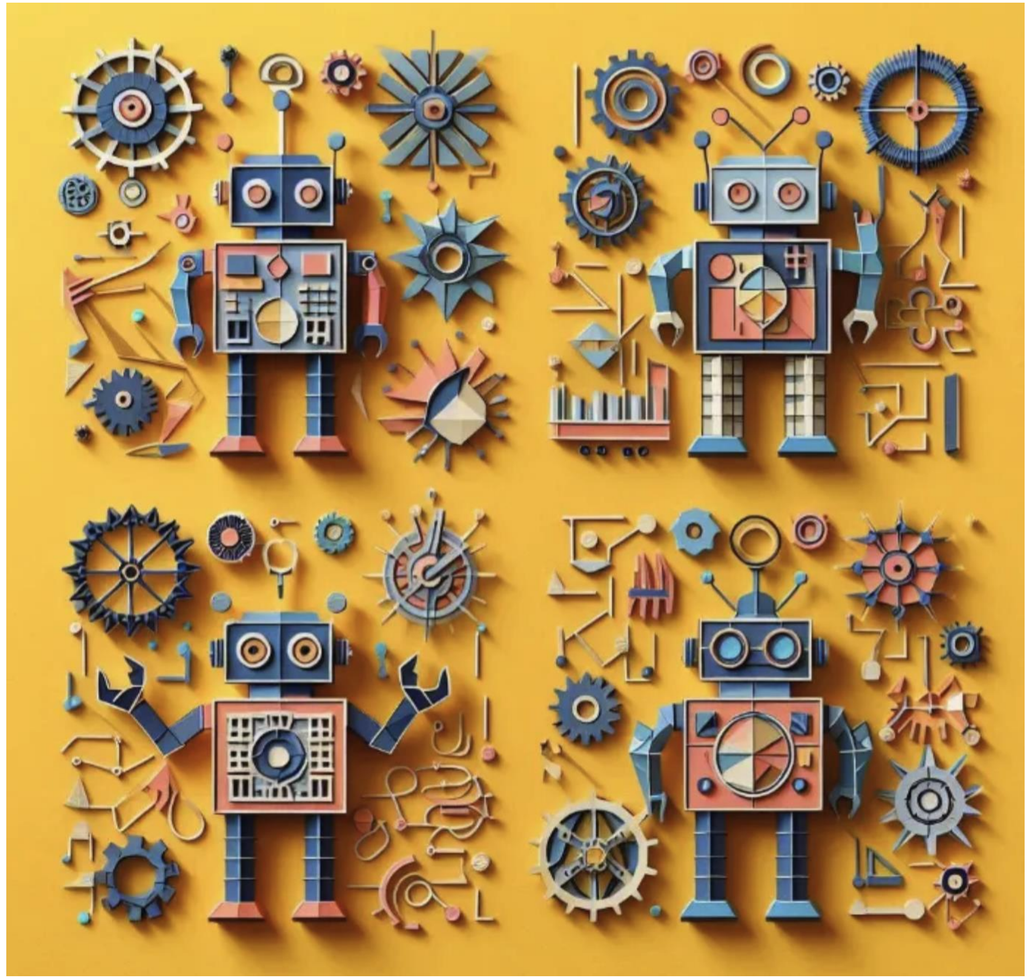
Crowd Sourcing

1. A variation of the Collaborators pattern. In this case, all collaborators will write their opinions on the same document and might use others' insights to further "reflect" on their opinions.
2. Some validation and fact checking will likely be needed to make sure agents are not building upon each others' hallucinations.



Agent Strategies and Agentic Patterns

1. There is no one-size-fits-all approach.
2. This field is rapidly evolving.
3. Hallucinations and Biases are real
4. AI is not always right
5. Human expertise and guidance will be required.



Thank You

https://www.linkedin.com/in/allen-chan