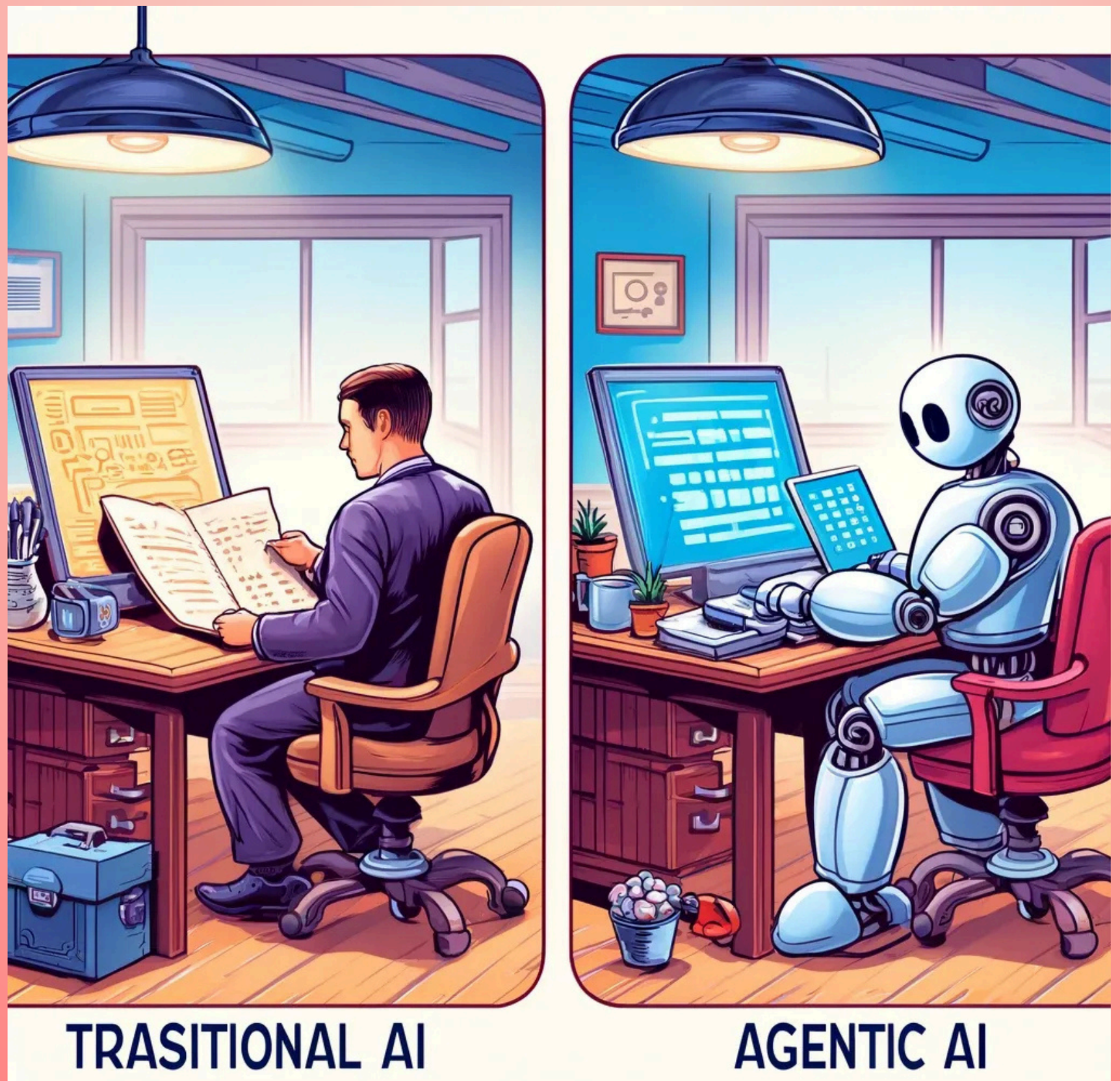
Building Effective

AI Agents

What are agents?

Agents are fully autonomous systems that operate independently over extended periods, using **various tools** to accomplish **complex tasks**.



Building Effective

AI Agents

Agent

1 The Brain → AI Model

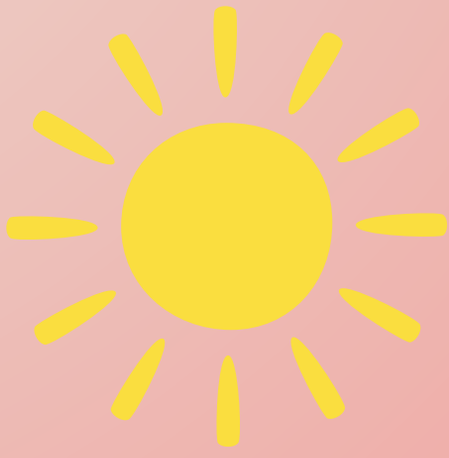
ex: GPT4, llama, Gemini

2 The Body → Capabilities & Tools

ex: websearch, image generation, API Interface

What’s the current weather in Chennai?

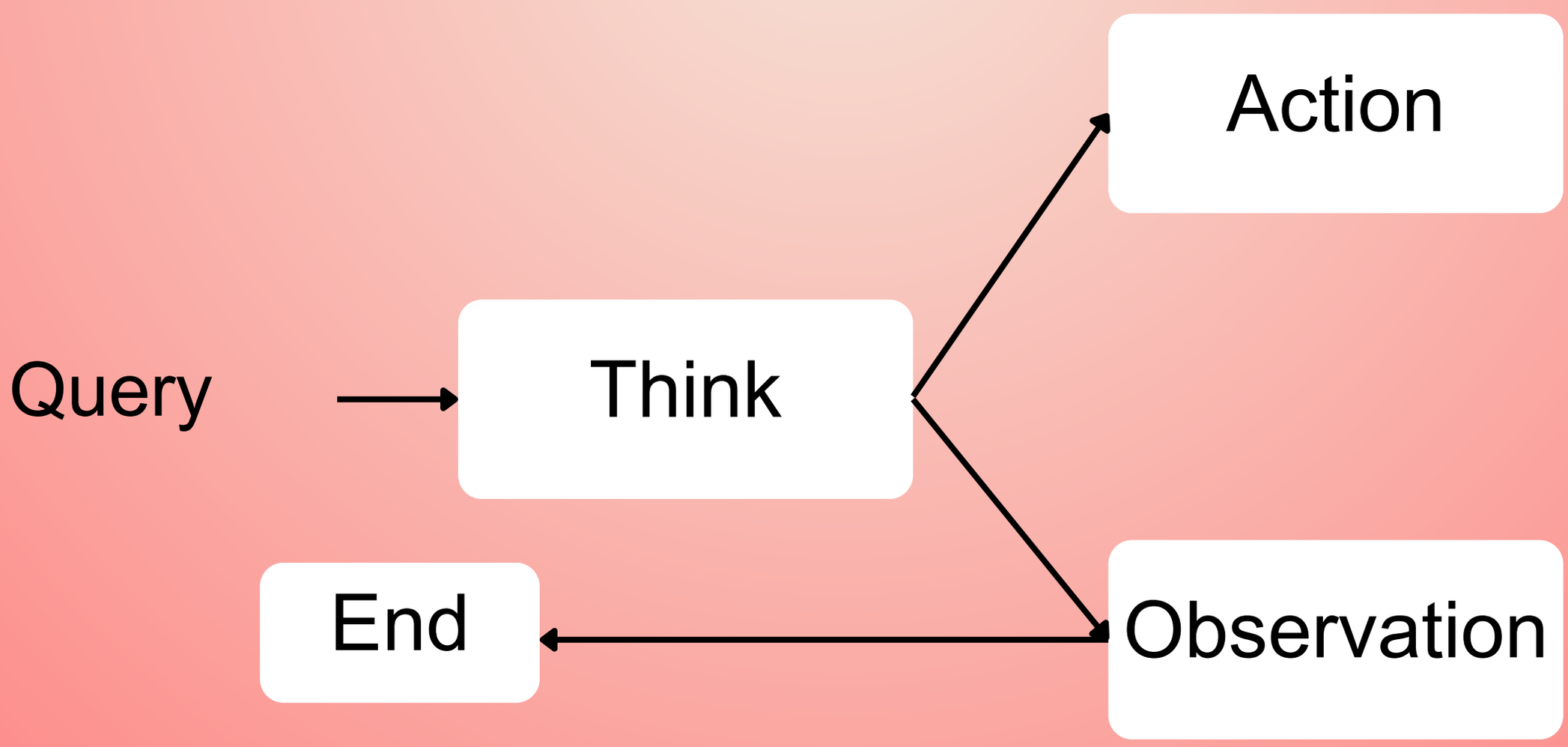
The LLM hallucinates



Agent that has Web Search tool - 35° C

The Core Components

1	2	3
Thinking (Thought) →	Acting (Act) →	Observing (Observe)
The LLM part of the Agent decides what the next step should be.	The agent takes an action, by calling the tools with the associated arguments.	The model reflects on the response from the tool.



Thought: Internal Reasoning

Thoughts → Agent's internal reasoning & planning process → To solve the task

Type of Thought	Example
Planning	1) Gather data 2) Analyse trends 3) Generate report
Analysis	Based on the error message, the issue appears to be with the database connection parameters.
Problem Solving	To optimize this code, I should first profile it to identify bottlenecks

Type of Thought	Example
Memory Integration	The user mentioned their preference for Python earlier, so I'll provide examples in Python
Self-Reflection	My last approach didn't work well, I should try a different strategy
Prioritization	The security vulnerability should be addressed before adding new features

Thought: The Re-Act Approach

ReAct approach → Reasoning (Think) + Acting (Act)

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Source: https://huggingface.co/learn/agents-course/en/unit1/thoughts

Code Agents

Instruction: Determine the most cost-effective country to purchase the smartphone model "CodeAct 1". The countries to consider are the USA, Japan, Germany, and India.

Available APIs
[1] lookup_rates(country: str) -> (float, float)
[2] convert_and_tax(price: float, exchange_rate: float, tax_rate: float) -> float
[3] estimate_final_price(converted_price: float, shipping_cost: float) -> float
[4] lookup_phone_price(model: str, country: str) -> float
[5] estimate_shipping_cost(destination_country: str) -> float

LLM Agent using [**Text/JSON**] as Action

Think I should calculate the phone price in USD for each country, then find the most cost-effective country.

Action
Text: lookup_rates, Germany
JSON: {"tool": "lookup_rates", "country": "Germany"}

Environment 1.1, 0.19

Action
Text: lookup_phone_price, CodeAct 1, Germany
JSON: {"tool": "lookup_phone_price", "model": "CodeAct 1", "country": "Germany"}

Environment 700

Action
Text: convert_and_tax, 700, 1.1, 0.19
JSON: {"tool": "convert_and_tax", "price": 700, "exchange_rate": 1.1, "tax_rate": 0.19}

Environment 916.3

[... interactions omitted (look up shipping cost and calculate final price) ...]

Action
Text: lookup_rates, Japan
JSON: {"tool": "lookup_rates", "country": "Japan"}

[... interactions omitted (calculate final price for all other countries)...]

Response The most cost-effective country to purchase the smartphone model is Japan with price 904.00 in USD.

Fewer Actio

Source: https://huggingface.co/learn/agents-course/en/unit1/thoughts

Code Agents

CodeAct: LLM Agent using [**Code**] as Action

Think I should calculate the phone price in USD for each country, then find the most cost-effective country.

Action

```
countries = ['USA', 'Japan', 'Germany', 'India']
final_prices = {}

for country in countries:
    exchange_rate, tax_rate = lookup_rates(country)
    local_price = lookup_phone_price("xAct 1", country)
    converted_price = convert_and_tax(
      local_price, exchange_rate, tax_rate
    )
    shipping_cost = estimate_shipping_cost(country)
    final_price = estimate_final_price(converted_price, shipping_cost)
    final_prices[country] = final_price

most_cost_effective_country = min(final_prices, key=final_prices.get)
most_cost_effective_price = final_prices[most_cost_effective_country]
print(most_cost_effective_country, most_cost_effective_price)
```

Control & Data Flow of Code Simplifies Complex Operations

>

ns Required!

Re-use `min` Function from Existing Software Infrastructures (Python library)

Environment 1.1, 0.19

Response The most cost-effective country to purchase the smartphone model is Japan with price 904.00 in USD.

Source: https://huggingface.co/learn/agents-course/en/unit1/thoughts

Code Agents

Steps

1. Get the free API from OpenWeatherMap.
2. Use the get weather function for a city

```python
# Code Agent Example: Retrieve Weather Information
def get_weather(city):
    import requests
    api_key = "196739a590f50b282f423aead9e0ff61"  # Your newly generated API key from OpenWeatherMap

    api_url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"

    response = requests.get(api_url)

    if response.status_code == 200:
        data = response.json()
        weather_description = data['weather'][0]['description']
        temperature = data['main']['temp']
        return f"{weather_description.capitalize()} with a temperature of {temperature}°C"
    else:
        print(f"Error: Unable to fetch weather data. Status code: {response.status_code}")
        print(f"Response content: {response.content}")
        return "Error: Unable to fetch weather data."

# Execute the function and prepare the final answer
result = get_weather("Chennai")
final_answer = f"The current weather in Chennai is: {result}"
print(final_answer)
```

```
The current weather in Chennai is: Few clouds with a temperature of 27.82°C
```

Source: https://huggingface.co/learn/agents-course/en/unit1/thoughts

smol Agents

```python
from smolagents import CodeAgent, DuckDuckGoSearchTool, HfApiModel

model = HfApiModel()
agent = CodeAgent(tools=[DuckDuckGoSearchTool()], model=model)

agent.run("How many seconds would it take for a cheetah at full speed to cross the Brooklyn Bridge?")
```

```
Step 4

─ Executing parsed code: ──────────────────────────────────────────────────────
 # Known values
 bridge_length_meters = 1825  # total length of the Brooklyn Bridge in meters
 cheetah_speed_mps = 31.29    # maximum speed of a cheetah in meters per second

 # Calculate the time it would take for the cheetah to cross the bridge
 time_seconds = bridge_length_meters / cheetah_speed_mps
 final_answer(time_seconds)
 ──────────────────────────────────────────────────────────────────────────────


Out - Final answer: 58.32534356024289


[Step 3: Duration 10.41 seconds| Input tokens: 9,933 | Output tokens: 575]


58.32534356024289
```

Source: https://github.com/huggingface/smolagents

CONGRATULATIONS

You have reached the end, now

If you want to help your network

REPOST THIS

Sarveshwaran R

https://github.com/DataSphereX/Agents