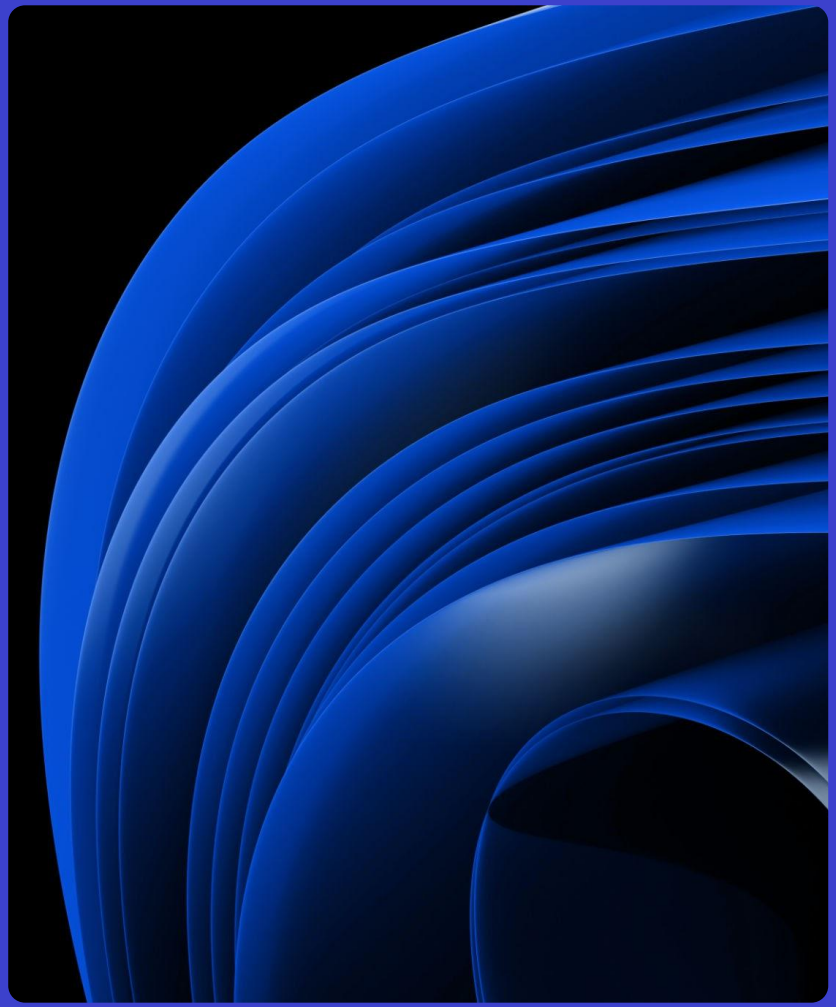
QCON SF 2024

10 Reasons Your Multi-Agent Workflows Fail

And what you can do about it

Victor Dibia, PhD | @vykthur
Nov 18, 2024

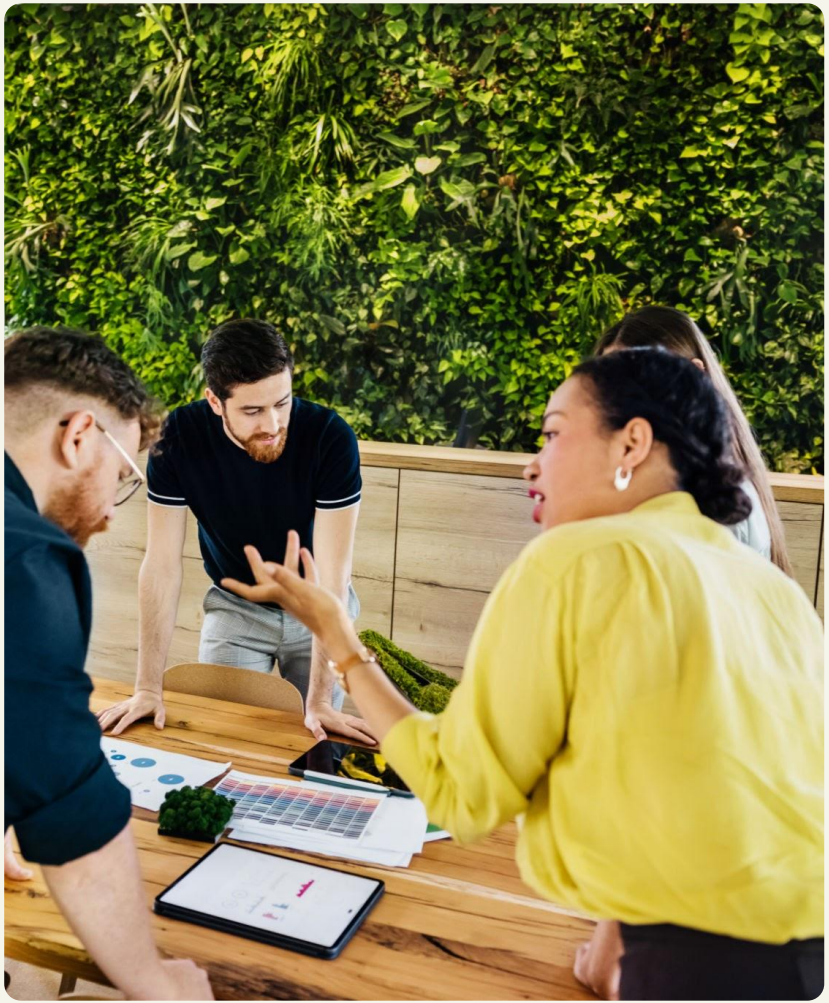


Why are Multi-Agent Systems? Interesting?

Imagine a scenario where computers could handle increasingly complex tasks on your behalf ..

“Download email attachments from clients, load them into Excel ..”

Back office data entry across multiple systems



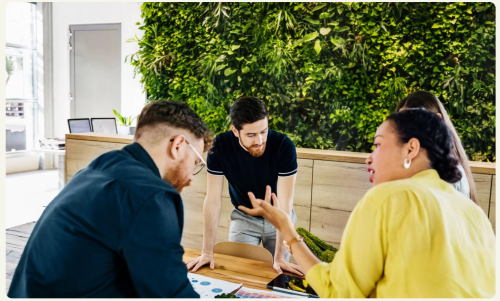
“Build an Android app that can help users view and purchase stocks”

Software engineering

“File my taxes”

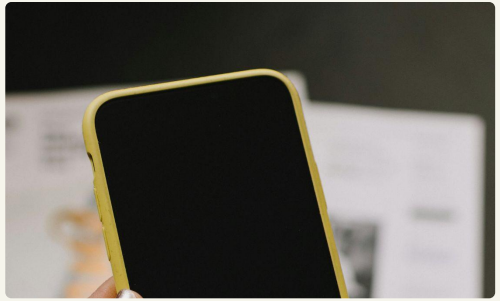
Finance

Back Office



Software Engineering

Finance



- Tedious and repetitive
- Important
- Involves many, sometimes *proactive* steps

3 Key Insights

On why agents/multi-agent systems are so interesting right now!

- **Save time** (autonomous task completion)
- A **new digital interface**
- **Disrupt** current approaches to solving tasks

The industry is reacting ..

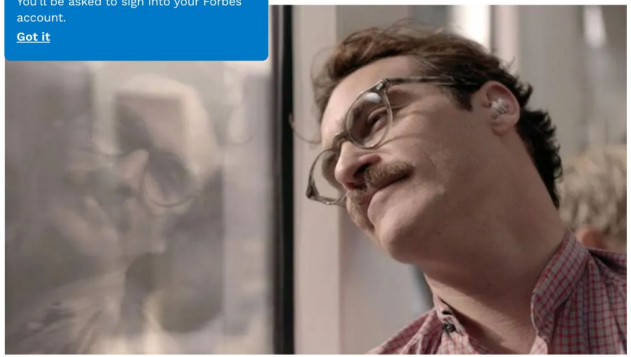
Agents Are The Future Of AI. Where Are The Startup Opportunities?

Rob Toews Contributor
I write about the big picture of artificial intelligence.

Follow

1

Jul 9, 2024, 10:00am EDT



AI agents, popularized in science fiction works like the 2013 film "Her", are fast becoming a ...
[+] SOURCE: HER

If you are wondering what the next great chapter in artificial intelligence will be, here is your answer.

"This seems like as good of a time as any to talk about how we view the future," wrote OpenAI leaders Sam Altman and Greg Brockman recently. "Users will increasingly interact with systems - composed of many multimodal models plus tools - which can take actions on their behalf, rather than talking to a single model."

This is as clear a description as any of the concept of "agents," which has

Agents Are T
Where Are T
Opportunitie

Rob Toews Contributor
I write about the big picture of artificial

AI agents, popularized in science fiction wo
[+] SOURCE: HER

If you are wondering what the n
be, here is your answer.

"This seems like as good of a tim
future," wrote OpenAI leaders S
"Users will increasingly interact
multimodal models plus tools -
than talking to a single model."

This is as clear a description as a

THE FUTURE OF AGENTS

AI is about to completely change how you use computers

And upend the software industry.

By **Bill Gates** | November 09, 2023 • 12 minute read

I still love software as much today as I did when Paul Allen and I started Microsoft. But—even though it has improved a lot in the decades since then—in many ways, software is still pretty dumb.

To do any task on a computer, you have to tell your device which app to use. You can use Microsoft Word and Google Docs to draft a business proposal, but they can't help you send an email, share a selfie, analyze data, schedule a party, or buy movie tickets. And even the best sites have an incomplete understanding of your work, personal life, interests, and relationships and a limited ability to use this information to do things for you. That's the kind of thing that is only possible today with another human being, like a close friend or personal assistant.

In the next five years, this will change completely. You won't have to use different apps for

FORBES > INNOVATION > AI

Agents Are T
Where Are T
Opportunitie

Rob Toews Contributor

I write about the big picture of artificial

1

Click to save this article.
You'll be asked to sign into your Forbes account.
Got it

AI agents, popularized in science fiction wo
[+] SOURCE: HER

If you are wondering what the n
be, here is your answer.

"This seems like as good of a tim
future," wrote OpenAI leaders S
"Users will increasingly interact
multimodal models plus tools -
than talking to a single model."

Forbes

This is as clear a description as a

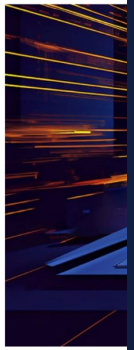
GatesNotes THE BLOG OF BILL GATES

THE FUTURE OF AGE

AI is
chan
comp

And upend the so

By Bill Gates | Novemb



I still love so
though it ha
dumb.

To do any ta
Microsoft V
an email, sh
sites have ar
relationship
of thing that
assistant.

In the next f

MIT Technology Review

SIGN IN SUBSCRIBE

ARTIFICIAL INTELLIGENCE

Sam Altman says helpful agents are poised to become AI's killer function

Open AI's CEO says we won't need new hardware or lots more training data to get there.

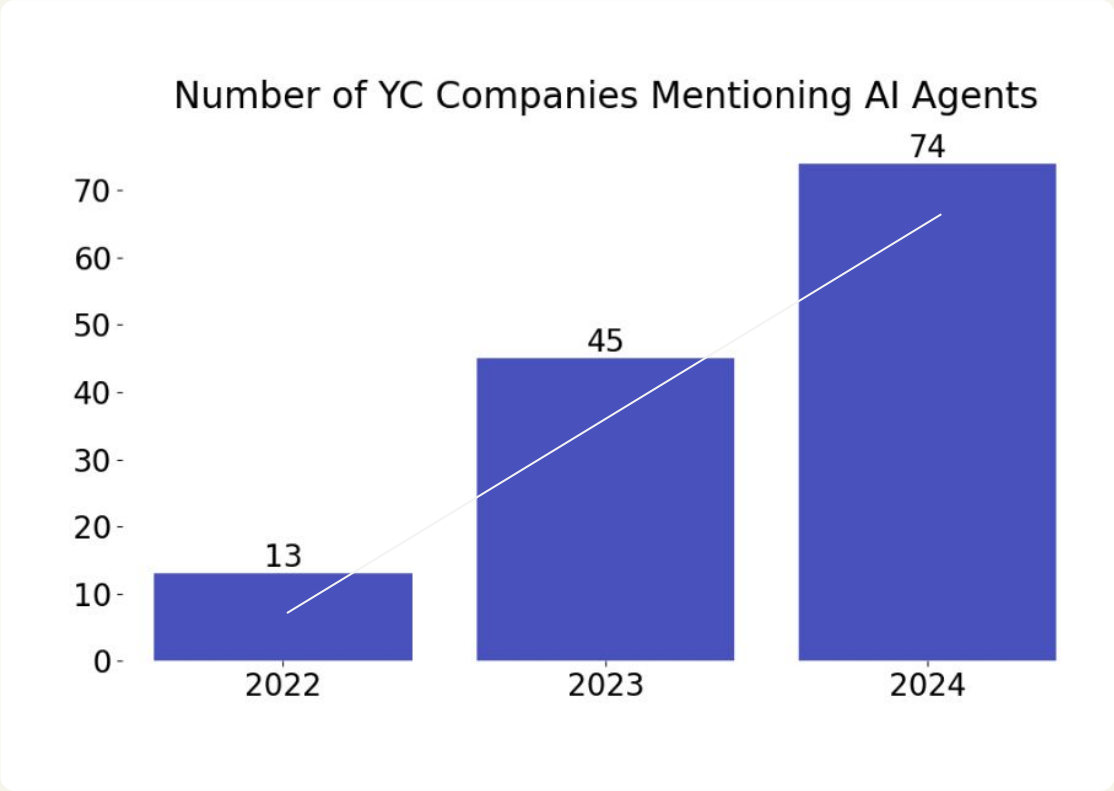
By James O'Donnell

May 1, 2024

469%

increase in # of agent startups (YC) over the last 2 years

Count of YC companies that explicitly **mention AI Agents** in their company description



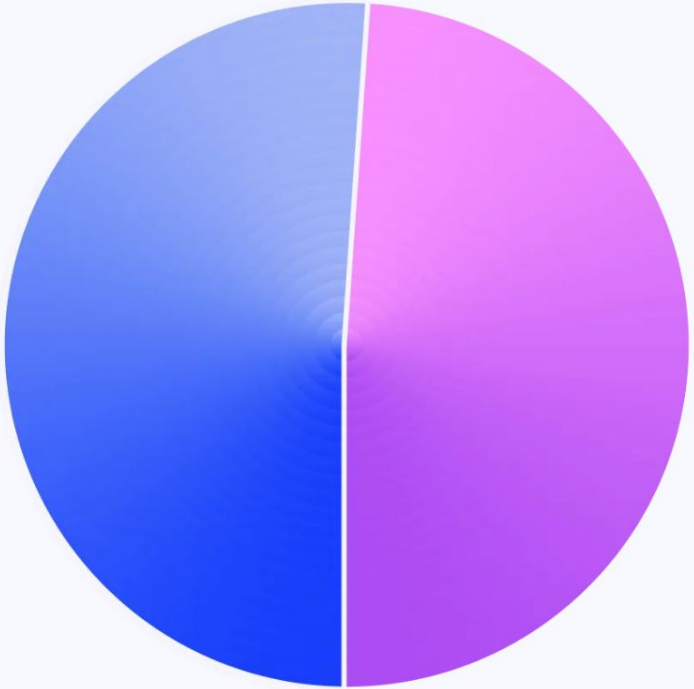
Source : YC data

There seems to be universal agreement that ..

The future is <u>Agentic</u>

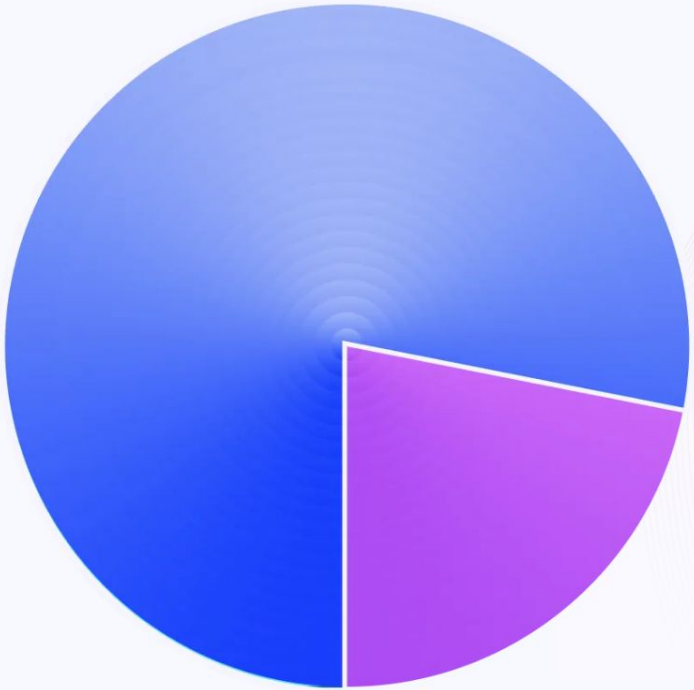
But
.. there are a few issues

Does your company currently have agents in production?



Yes - 51.1% No - 48.9%

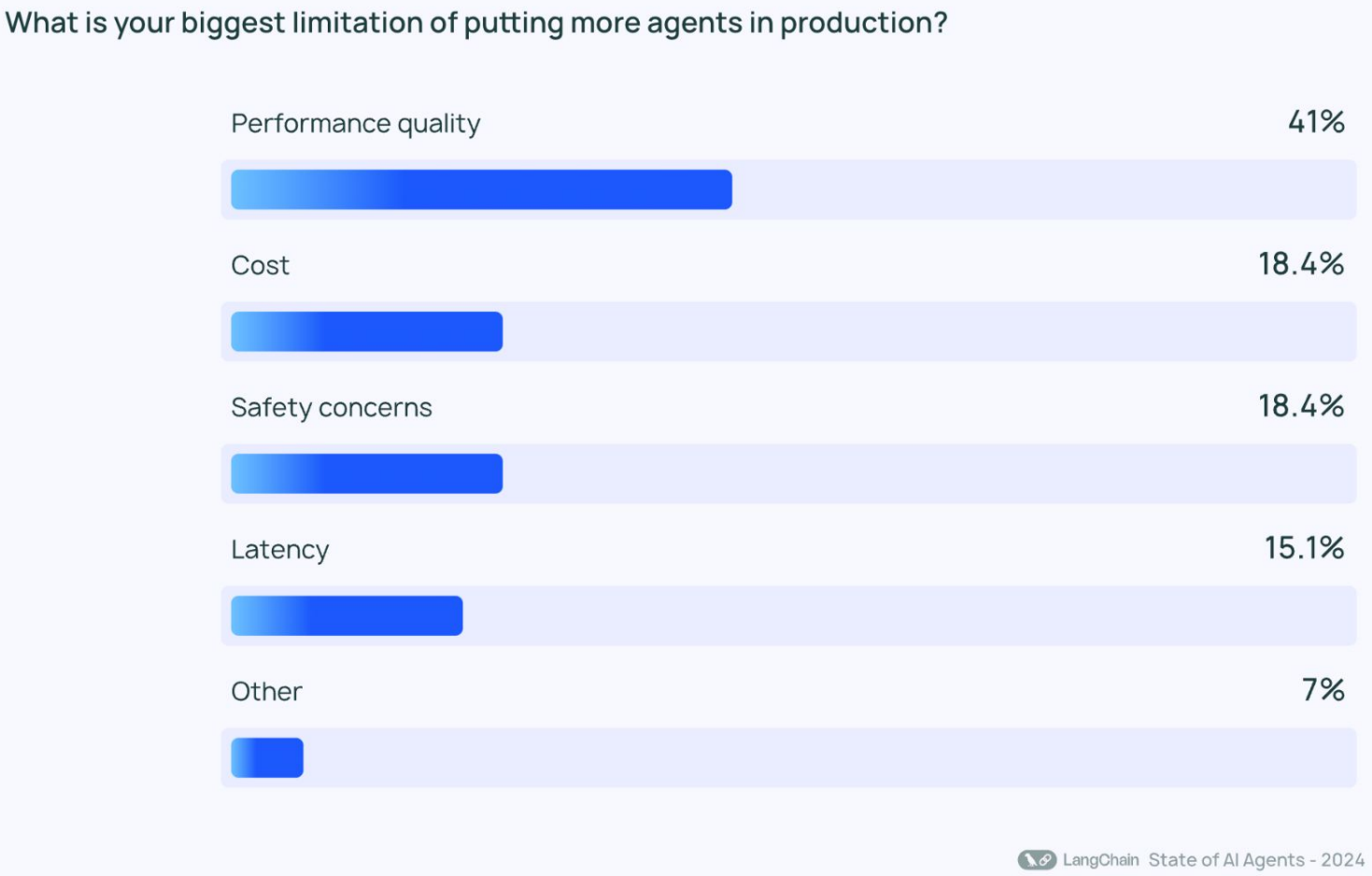
Are you currently developing an agent with plans to put it into production?



Yes - 78.1% No - 21.9%

Source : langchain

LangChain State of AI Agents - 2024



Source : langchain

Autonomous Agents have the *last mile* problem

Richard Socher
@RichardSocher

If each step of an ai agent is 95% accurate. None of the 30 step work flows will work.

Going from 95-> 99.9 is a similar last mile problem as with self driving cars.

Easy to hack up a prototype. Hard to make it work reliably at scale.

Source: Richard Socher (CEO, You.com)

A paradox and questions ..

- What are multi-agent systems and how to build them?
- What factors drive reliability issues?
- Should I invest in an autonomous multi-agent system?

Talk Agenda

Part 1

Introduction

- What are **autonomous multi-agent** systems?
- How can you build them?

Part 2

Failure Modes

- 10 reasons current multi-agent workflows fail

Part 3

What you can do

- Key takeaways
- Next steps

Victor Dibia

x.com | linkedIn | newsletter

victordibia.com

Principal RSDE, Microsoft Research

Focused on Human AI Experiences and Agents

Previously Worked @

- **Cloudera** - ML Engineer
- IBM Research - Research Staff Member

Core Contributor AutoGen, AutoGen Studio

Leading OSS framework for building multi-agent applications.

AutoGen v0.4 (preview api!)

https://github.com/microsoft/autogen
MIT License | 35k Stars | 5k forks

- Event driven
- Asynchronous by design
- Supports conversational programming
- Low Level expressive API and high level API with presets.

README | Code of conduct | CC-BY-4.0 license | MIT license | Security

Follow @pyautogen

AutoGen

> Important
>
> - (10/13/24) Interested in the standard AutoGen as a prior user? Find it at the actively-maintained *AutoGen 0.2 branch* and `autogen-agentchat~=0.2` PyPi package.
> - (10/02/24) AutoGen 0.4 is a from-the-ground-up rewrite of AutoGen. Learn more about the history, goals and future at this blog post. We're excited to work with the community to gather feedback, refine, and improve the project before we officially release 0.4. This is a big change, so AutoGen 0.2 is still available, maintained, and developed in the 0.2 branch.

AutoGen is an open-source framework for building AI agent systems. It simplifies the creation of event-driven, distributed, scalable, and resilient agentic applications. It allows you to quickly build systems where AI agents collaborate and perform tasks autonomously or with human oversight.

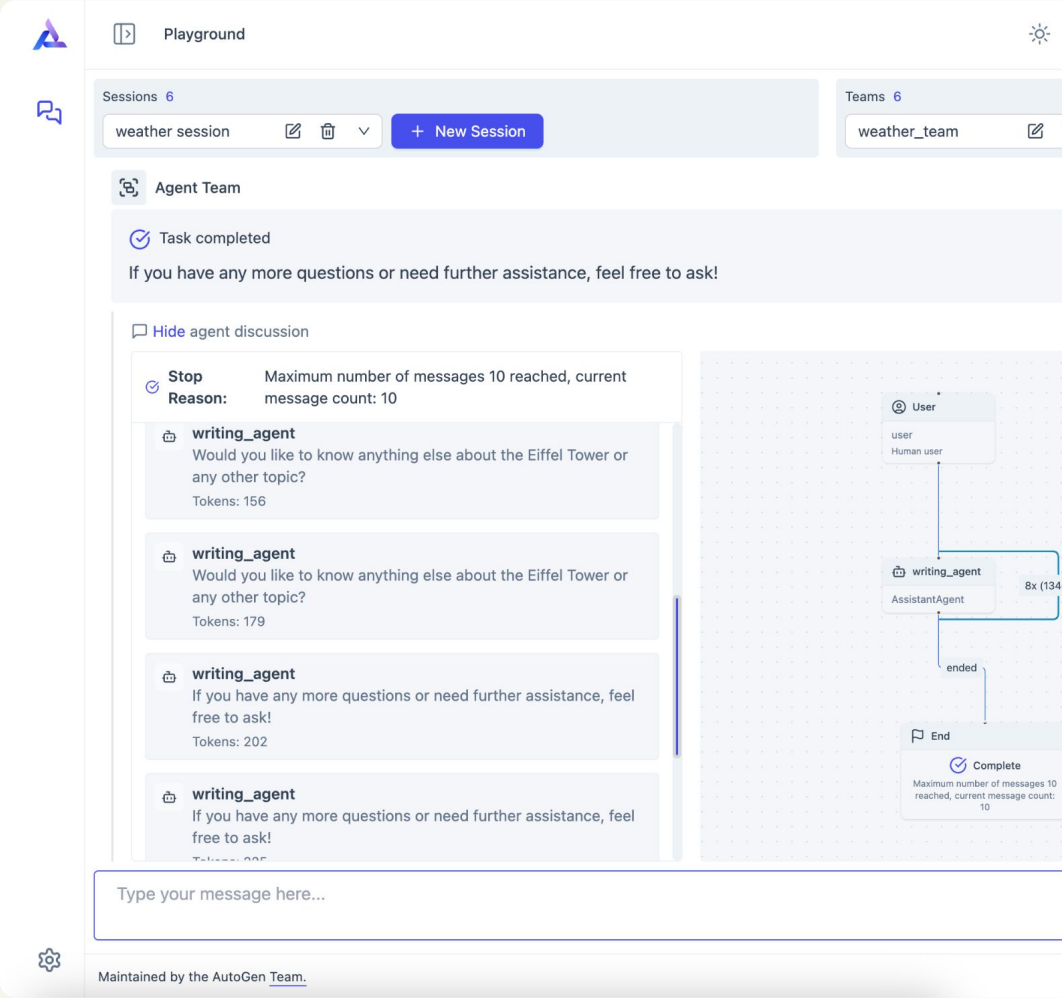
- Key Features
- API Layering
- Quickstart
- Roadmap
- FAQs

AutoGen streamlines AI development and research, enabling the use of multiple large language models (LLMs), integrated tools, and advanced multi-agent design patterns. You can develop and test your agent systems locally, then deploy to a distributed cloud environment as your needs grow.

AutoGen Studio

https://github.com/microsoft/autogen
MIT License

Low-code developer tool for prototyping and debugging multi-agent apps built with autogen.

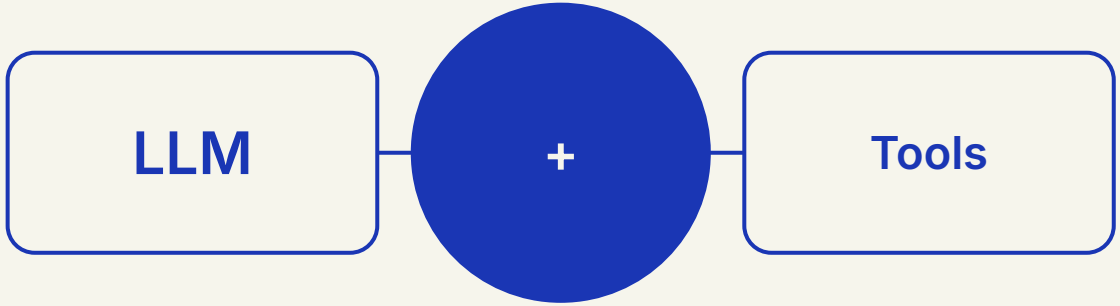


What are Multi-Agent Systems?

Part 1

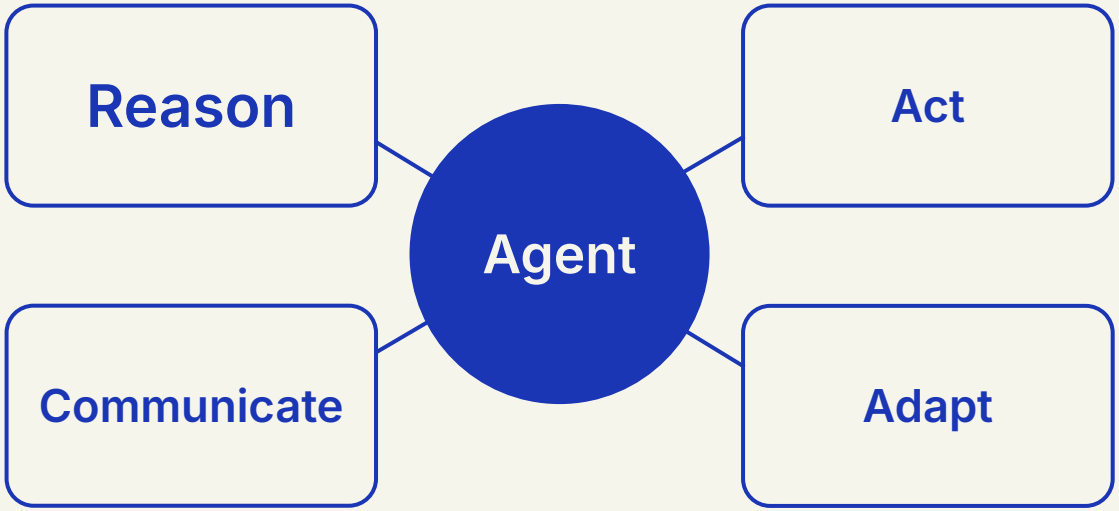
What is an Agent?

Agent = LLM + Tools



Agent

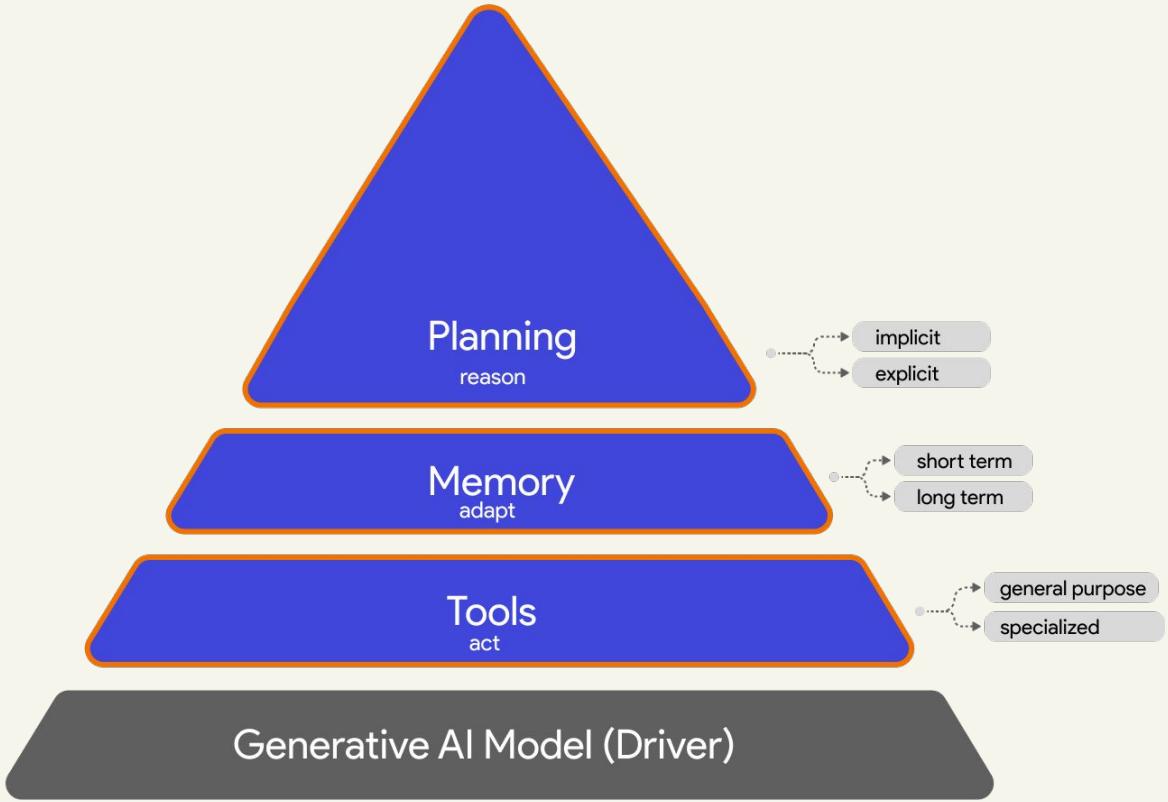
An entity that can **reason**, **act**, **communicate** and **and adapt** to solve tasks



https://multiagentbook.com

Agent

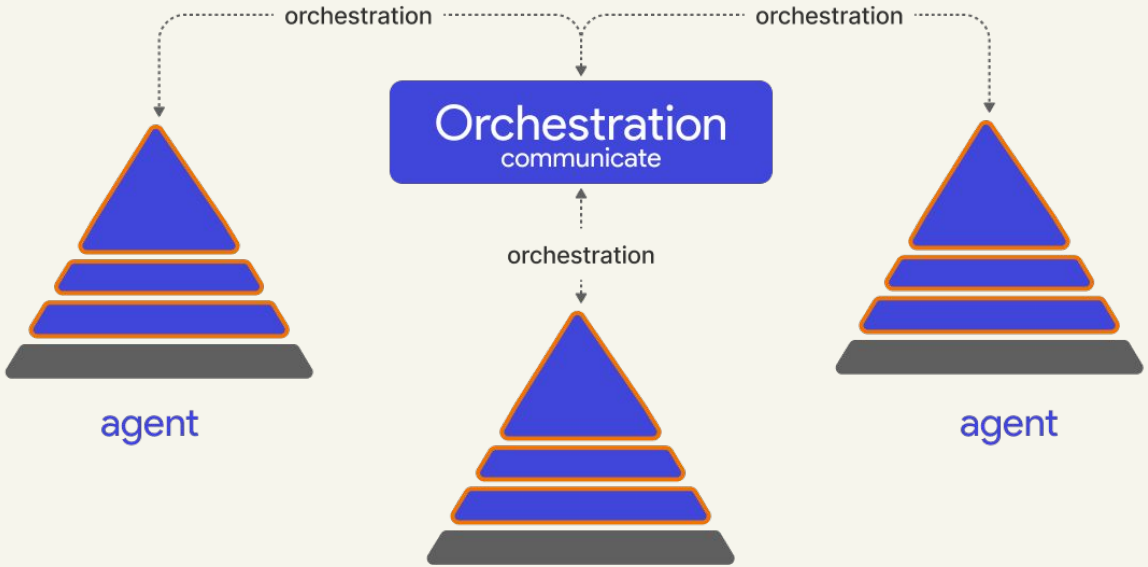
An entity that can **reason**, **act**, **communicate** and **and adapt** to solve tasks



https://multiagentbook.com

Multi-Agent System

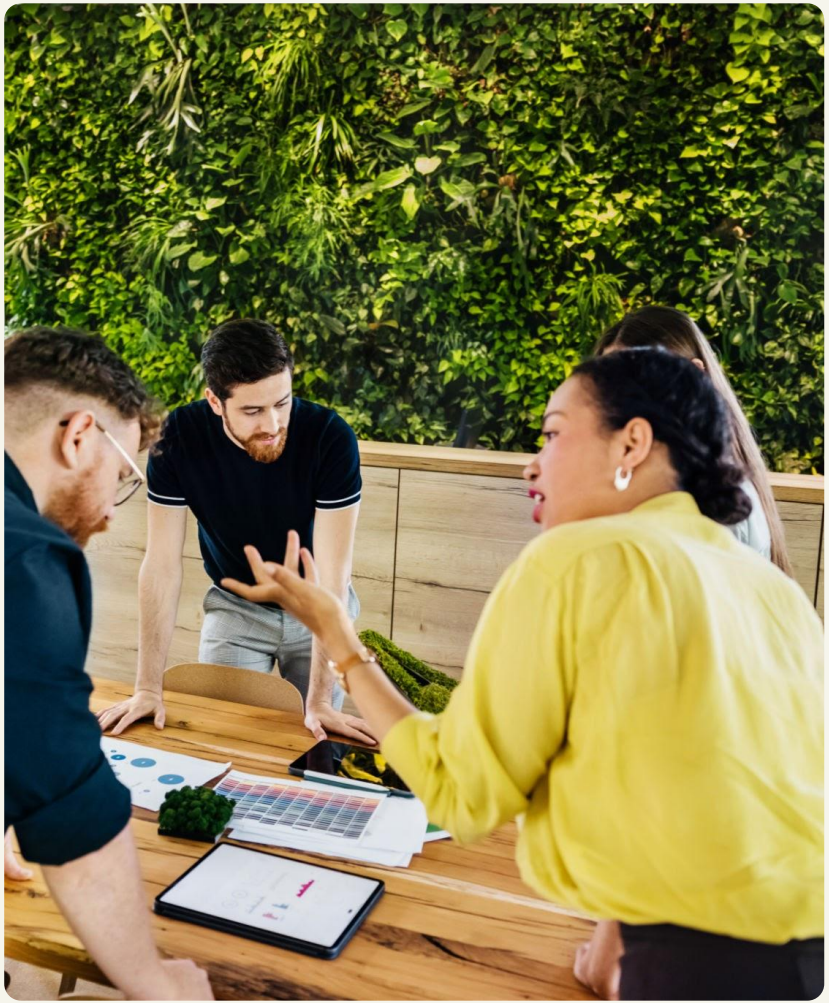
Group of agents that follow some **communication**/ collaboration pattern (**orchestration**) to solve tasks.



For this talk, groups of agents = workflow = team

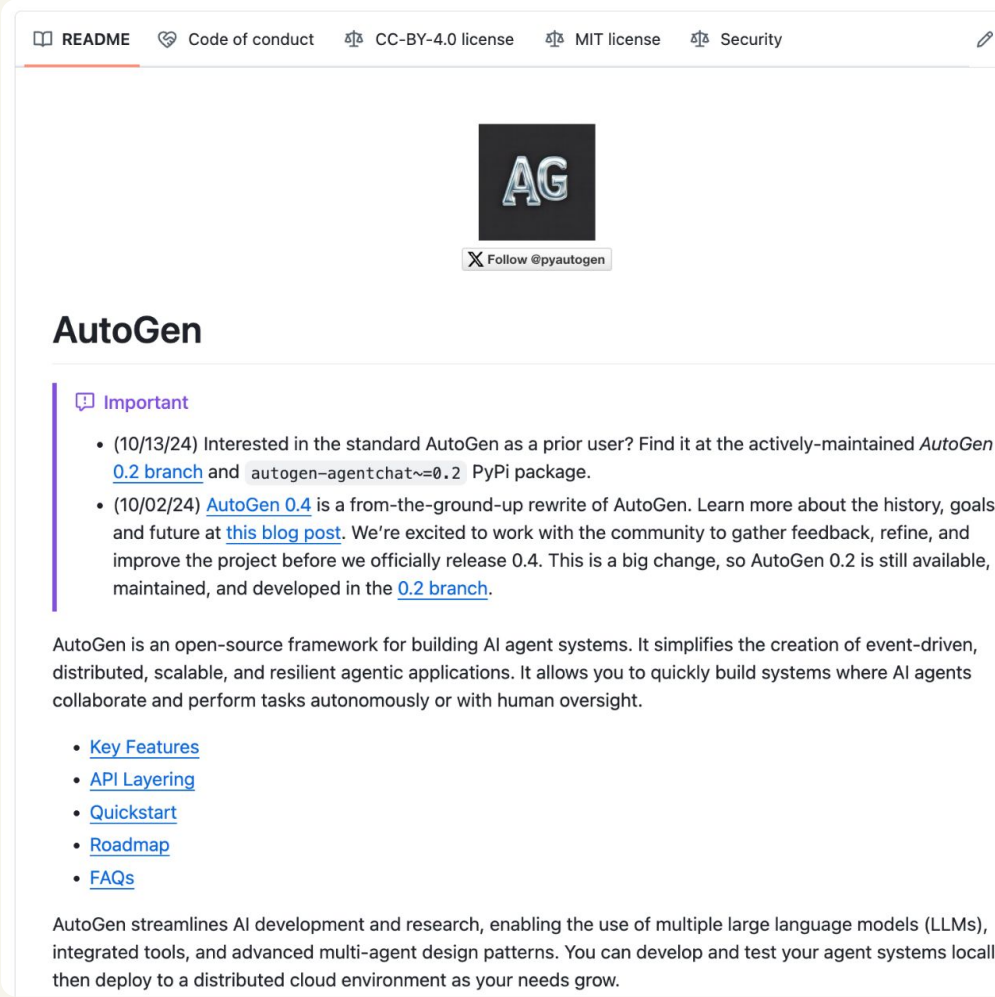
https://multiagentbook.com

Expressing Multi-Agent Systems as Code *with* AutoGen



AutoGen v0.4

- Event driven
- Asynchronous by design
- Supports conversational programming
- Runtime that handles message delivery and agent lifecycle
- Low-level expressive API (**Core**) and high-level API (**AgentChat**) with presets.



README | Code of conduct | CC-BY-4.0 license | MIT license | Security

Follow @pyautogen

AutoGen

Important

- (10/13/24) Interested in the standard AutoGen as a prior user? Find it at the actively-maintained *AutoGen 0.2 branch* and `autogen-agentchat~=0.2` PyPi package.
- (10/02/24) AutoGen 0.4 is a from-the-ground-up rewrite of AutoGen. Learn more about the history, goals and future at this blog post. We're excited to work with the community to gather feedback, refine, and improve the project before we officially release 0.4. This is a big change, so AutoGen 0.2 is still available, maintained, and developed in the 0.2 branch.

AutoGen is an open-source framework for building AI agent systems. It simplifies the creation of event-driven, distributed, scalable, and resilient agentic applications. It allows you to quickly build systems where AI agents collaborate and perform tasks autonomously or with human oversight.

- Key Features
- API Layering
- Quickstart
- Roadmap
- FAQs

AutoGen streamlines AI development and research, enabling the use of multiple large language models (LLMs), integrated tools, and advanced multi-agent design patterns. You can develop and test your agent systems locall then deploy to a distributed cloud environment as your needs grow.

AutoGen Core v0.4

- Unopinionated low-level expressive API
- An Agent simply responds to a **message** event.

README | Code of conduct | CC-BY-4.0 license | MIT license | Security

Follow @pyautogen

AutoGen

> Important
>
> - (10/13/24) Interested in the standard AutoGen as a prior user? Find it at the actively-maintained *AutoGen 0.2 branch* and `autogen-agentchat~=0.2` PyPi package.
> - (10/02/24) AutoGen 0.4 is a from-the-ground-up rewrite of AutoGen. Learn more about the history, goals and future at this blog post. We're excited to work with the community to gather feedback, refine, and improve the project before we officially release 0.4. This is a big change, so AutoGen 0.2 is still available, maintained, and developed in the 0.2 branch.

AutoGen is an open-source framework for building AI agent systems. It simplifies the creation of event-driven, distributed, scalable, and resilient agentic applications. It allows you to quickly build systems where AI agents collaborate and perform tasks autonomously or with human oversight.

- Key Features
- API Layering
- Quickstart
- Roadmap
- FAQs

AutoGen streamlines AI development and research, enabling the use of multiple large language models (LLMs), integrated tools, and advanced multi-agent design patterns. You can develop and test your agent systems locally, then deploy to a distributed cloud environment as your needs grow.

AutoGen AgentChat v0.4

- High Level API
- **Presets** for
  - Agents
  - Teams
  - Termination Conditions

README | Code of conduct | CC-BY-4.0 license | MIT license | Security

Follow @pyautogen

AutoGen

> Important
>
> - (10/13/24) Interested in the standard AutoGen as a prior user? Find it at the actively-maintained *AutoG*
> 0.2 branch and `autogen-agentchat~=0.2` PyPi package.
> - (10/02/24) AutoGen 0.4 is a from-the-ground-up rewrite of AutoGen. Learn more about the history, go
> and future at this blog post. We're excited to work with the community to gather feedback, refine, and
> improve the project before we officially release 0.4. This is a big change, so AutoGen 0.2 is still availab
> maintained, and developed in the 0.2 branch.

AutoGen is an open-source framework for building AI agent systems. It simplifies the creation of event-driven
distributed, scalable, and resilient agentic applications. It allows you to quickly build systems where AI agents
collaborate and perform tasks autonomously or with human oversight.

- Key Features
- API Layering
- Quickstart
- Roadmap
- FAQs

AutoGen streamlines AI development and research, enabling the use of multiple large language models (LLM
integrated tools, and advanced multi-agent design patterns. You can develop and test your agent systems lo
then deploy to a distributed cloud environment as your needs grow.

Key Features

AutoGen offers the following key features:

- **Asynchronous Messaging**: Agents communicate via asynchronous messages, supporting both event-d

Define an agent

```
agent = AssistantAgent(
name="single_agent",
model_client=OpenAIChatCompletionClient(
model="gpt-4o-mini"))
```

AssistantAgent preset.

Define an agent

AssistantAgent preset.

```
agent = AssistantAgent(
name="single_agent",
model_client=OpenAIChatCompletionClient(
model="gpt-4o-mini"))
```

```
result = await agent.run(task="What is the
height of the eiffel tower?")
```

We can test the agent by calling `.run()`

The Eiffel Tower stands at a height of approximately 1,083 feet (330 meters) including its antennas. The structure itself is about 1,063 feet (324 meters) tall without antennas.

TERMINATE

Define an agent

AssistantAgent preset.

```
agent = AssistantAgent( name="single_agent",
model_client=OpenAIChatCompletionClient(
model="gpt-4o-mini"))
```

```
result = await agent.run(task="What is the
Weather in San Francisco?")
```

> I'm **unable to provide real-time data including current weather updates**. You can check a reliable weather website or app for the latest information on the weather in San Francisco.
>
> TERMINATE

Define a tool

Define a tool

```python
def get_weather(city: str) -> str:
    return f"The weather in {city} is 73 degrees and Sunny."
```

A tool may be a python function or LangChain Tool

Giving agents access to tools

A tool may be a python function or LangChain Tool

```
def get_weather(city: str) -> str:
    return f"The weather in {city} is 73 degrees
    and Sunny."

agent = AssistantAgent(
    name="basic_agent",
    model_client=OpenAIChatCompletionClient
    (model="gpt-4o-mini"),
    tools=[get_weather])
result = await agent.run(task="What is the
Weather in San Francisco?")
```

> The weather in San Francisco is currently 73 degrees and sunny.

Defining a team

Defining a team

Uses the RoundRobinGroupChat preset.

A team may contain one or many agents.

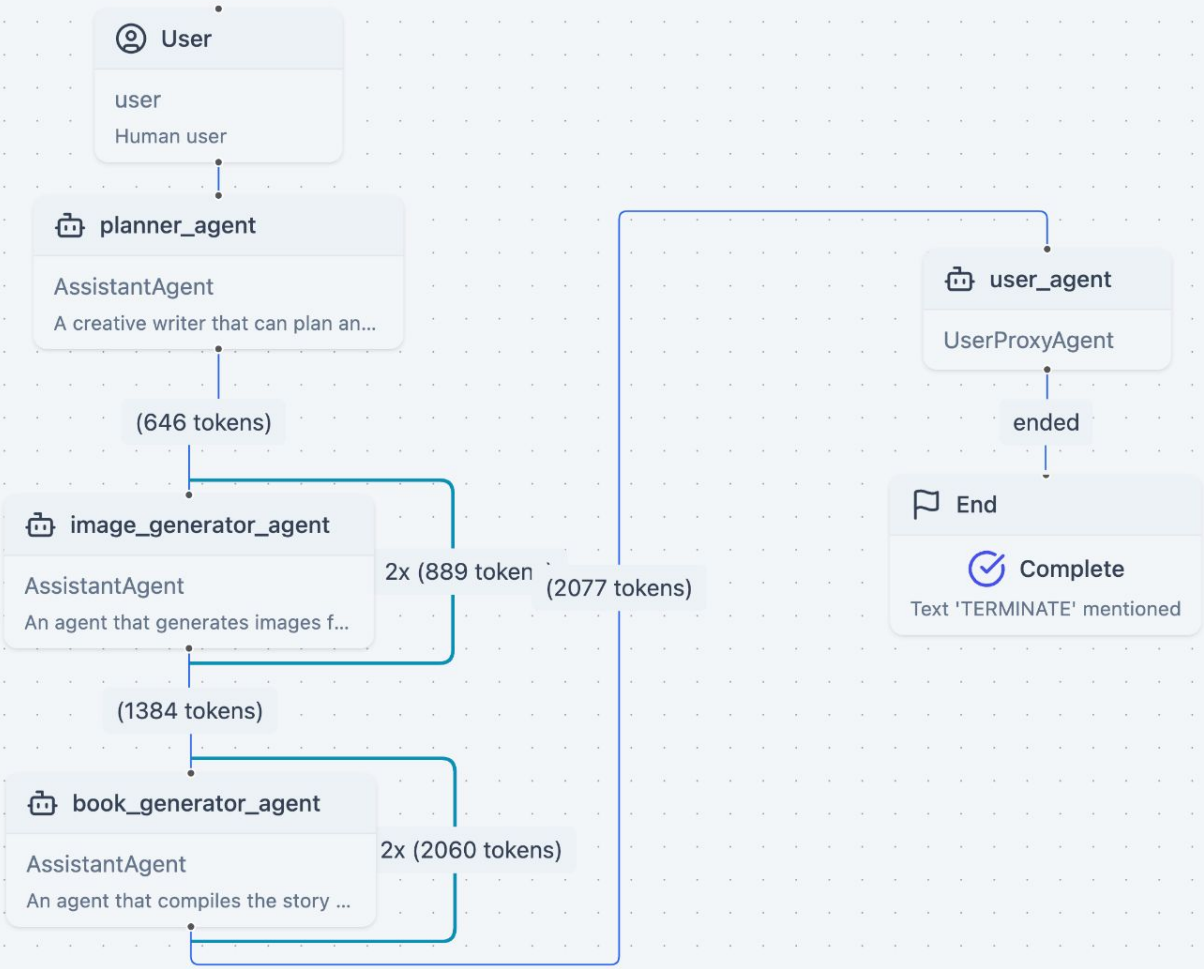
```
def get_weather(city: str) -> str:
    return f"The weather in {city} is 73
    degrees and Sunny."

agent = AssistantAgent(
    name="basic_agent",
    model_client=OpenAIChatCompletionClient
    (model="gpt-4o-mini"),
    tools=[get_weather])

team = RoundRobinGroupChat(
    participants=[agent],
    termination_condition=TextMentionTerminatio
    n("TERMINATE"))

team_result = await team.run(task="What is the
weather in San Francisco?")
```

Defining a team

Uses the SelectorGroupChat preset.

Multiple agents

```
book_team = SelectorGroupChat(
    participants=[planner_agent, image_generator_agent,
    book_generator_agent],
    termination_condition=TextMentionTermination("TERMINATE")
    model_client=OpenAIChatCompletionClient(model="gpt-4o-min
    selector_prompt="You are  book generation cordinator. The
    following agent roles are helping you create a book: {rol
    Your goal is to read the progress so far and then select
    next role from {participants} to take a turn. Only return
    role. . {history}. You must call the agents in the right
    order  \nRead the above conversation.  Then select the ne
    role from {participants} to play. Only return the role."
)
book_result = book_team.run_stream(task="Create a 1 page
children's story with 2 images and text about the wonders of
amazon rainforest.")
```


User
user
Human user
planner_agent
AssistantAgent
A creative writer that can plan an...
(646 tokens)
image_generator_agent
AssistantAgent
An agent that generates images f...
2x (889 token
(1384 tokens)
book_generator_agent
AssistantAgent
An agent that compiles the story ...
2x (2060 tokens)
(2077 tokens)
user_agent
UserProxyAgent
ended
End
Complete
Text 'TERMINATE' mentioned

amazon_adventure.pdf

Page 1 of 4

The Magical Amazon Adventure

Once upon a time, in a vibrant town nestled near the Amazon rainforest, lived a curious little girl named Mia. Mia loved nature and was fascinated by the stories her grandmother told her about the wonders of the Amazon. One sunny morning, Mia decided to venture into the jungle to see it for herself. Wearing her favourite green hat and a backpack filled with snacks, she set off on her adventure.

As Mia entered the rainforest, she was greeted by a symphony of sounds - chirping birds, buzzing insects, and rustling leaves. Brightly colored butterflies fluttered around her, dancing like little jewels in the air. ?Wow!? Mia exclaimed. ?This place is like a treasure chest!?

Mia marveled at the towering trees, their trunks thick and twisted. She spotted various creatures peeking out: a toucan with its big beak, a sloth hanging lazily from a branch, and tiny red poison dart frogs hopping along the ground. Each step was an exploration, a discovery waiting to happen!

The configuration space for multi-agent systems is exponential

With configurations on a spectrum of predefined to autonomous behaviors

Exponential Configuration Space

Planning / Orchestration

- Centralized / explicit
- Implicit (after each step)

Agent Definition

- Developer defined
- Automated/task based

Tool Definition

- Developer defined
- Automated/task based

Memory

- What to learn/index
- When to learn/index

Termination

- LLM/task based
- Resource budget based (time, tokens, cost, rounds)
- External monitor/tool

Human Delegation

- After each turn/action
- Intelligent delegation

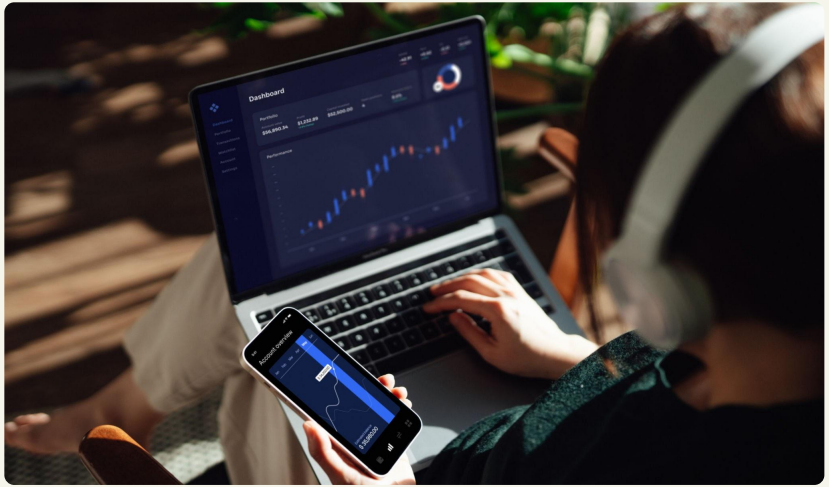
Improper configuration can lead to mistakes and can drive poor performance

Frameworks can help/simplify the process

10 Reasons Agents Fail

Part 2

1. Your agent lacks detailed instructions



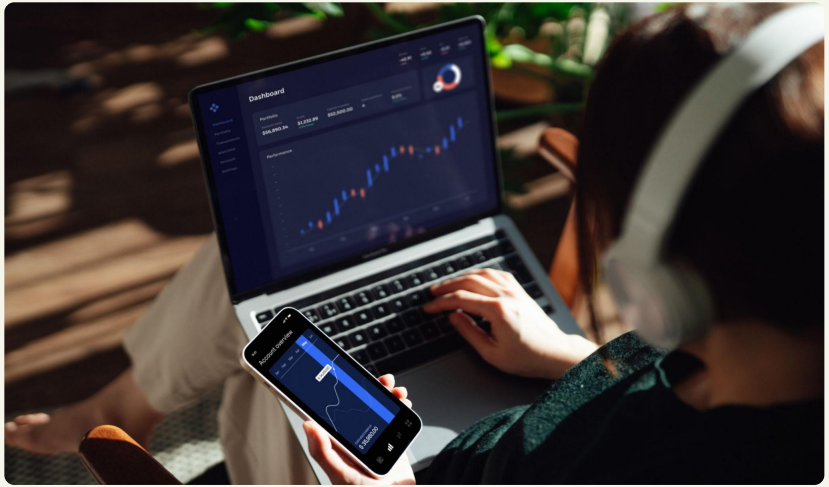
- Agents are driven by LLM's which in turn require careful prompting
- Good agents have lengthy, detailed instructions - from how to respond, tools to use and behaviors to avoid

```python
# Book compiler agent
book_generator_agent = AssistantAgent(
    "book_generator_agent",
    model_client=OpenAIChatCompletionClient(model="gpt-4"),
    description="An agent that compiles the story and images into a
    PDF book.",
    system_message="""You are a book compilation specialist.
    Your role is to collect story sections and images, format them
    for PDF generation, and create the final book.
    IMPORTANT: Use the actual image file paths returned by the
    image generator, not placeholder names. For example if the
    image generator return '71e6aba5-1a7e-488c-9388-e3bc1eeb88c7.
    png', then use this exactly as the image path in the book
    generation without any prefix or suffix.
    Respond with 'TERMINATE' when the book is successfully
    generated.""",
    tools=[generate_and_save_pdf_report]
)
```

2. Stop using small models

- Smaller models show significantly reduced instruction-following capabilities.
- Your LLAMA -7B etc models will not work well for agents out of the box without specific optimizations.

3. Your agent instructions do not match your LLM



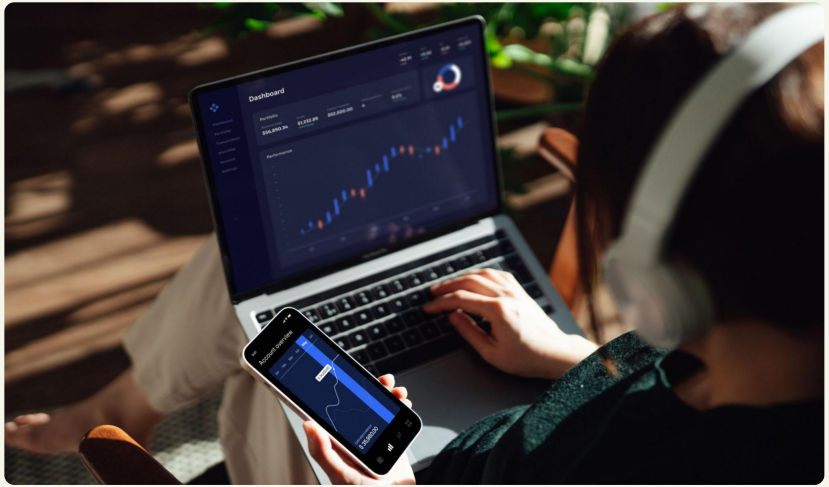
- System messages are **not** portable across versions of the same model and especially across model providers.
- Simply changing the model and expecting similar behaviours is often a mistake!

4. Your agents lack *good* tools

- Tools dictate the action space of agents
- Your agents actions are as good as the tools available to them - with implications for reliability.
  (General purpose vs task-specific tools)

```python
from fpdf import FPDF
import requests
import os
from tempfile import gettempdir
from PIL import Image
from io import BytesIO

def generate_and_save_pdf_report(sections: list, output_file: str = "book.pdf",
report_title: str = "Book") -> str:
    """
    Generate a PDF report with text and images from provided sections.

    Args:
        sections (list): List of dictionaries containing section data. Each section
        should have:
            - title (str): Section title
            - content (str): Section content text
            - image (str): Path or URL to image file
            - level (str): Heading level (e.g., 'h1', 'h2')
        output_file (str, optional): Name of output PDF file. Defaults to "book.pdf".
        report_title (str, optional): Title of the report. Defaults to "Book".
```

5. Your agents do not know when to stop



- Defining the right termination condition is critical especially to manage resource and latency
- Termination conditions depend on the agent and team configuration.

Playground

Guest User

Sessions 6

weather session

+ New Session

Teams 6

weather_team

+ New Team

Send a message to begin!

what is the height of the eiffel tower

6. You have the wrong multi-agent pattern

- Should you use a predefined chain, a chain with autonomous steps, or a fully autonomous team workflow?

Note: finding the right pattern is an evolving area of research and community practice. See https://multiagentbook.com/labs/multi-agent-patterns

7. Your agents are not learning (memory)

- Most agents today are like the classic forgetful goldfish
- A good memory implementation addresses
  - Ability to learn from explicit feedback
  - Learning from implicit feedback
  - Intelligent recall (when and what to retrieve given the task)

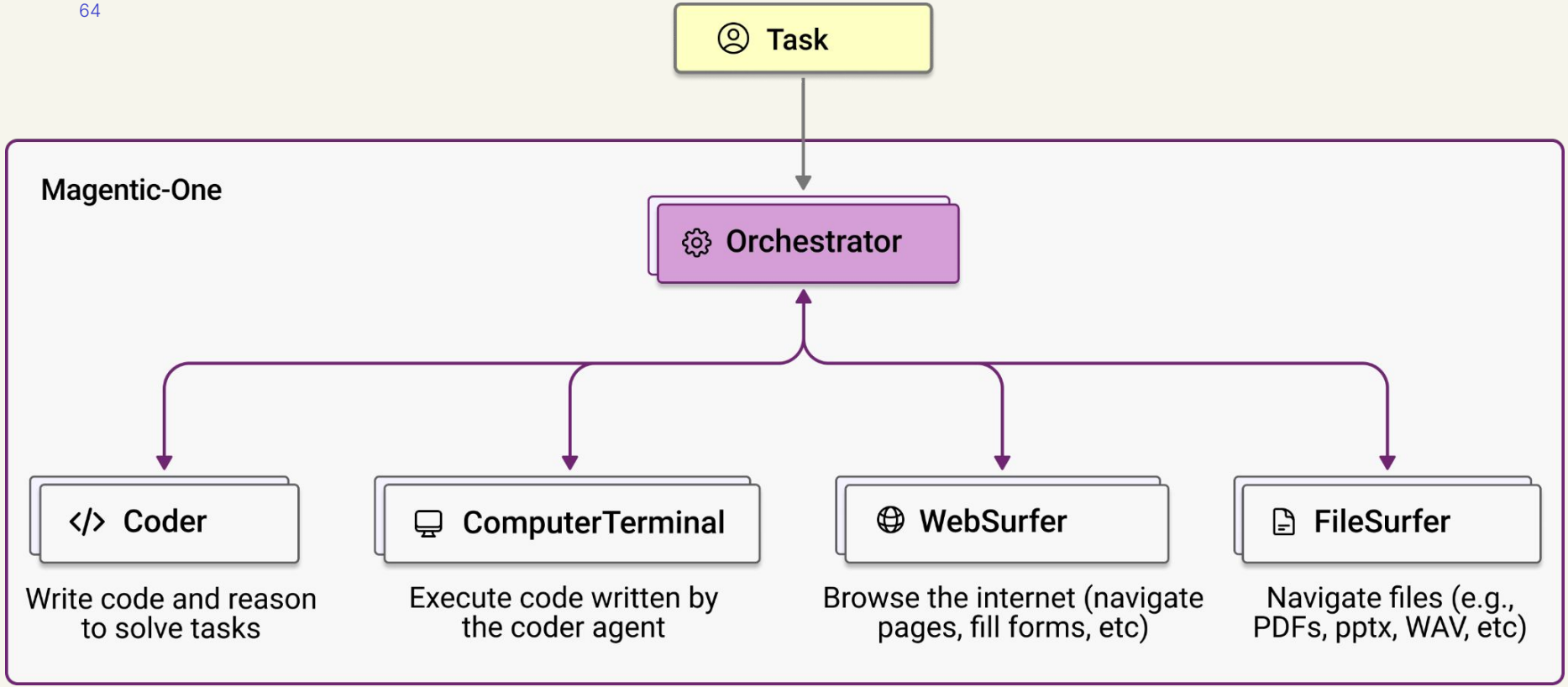
8. Your agents lack Metacogniti on

- Long-running complex tasks often benefit from the ability to plan, review, monitor progress.
  - Assess task state
  - Abandon compromised trajectories
  - Reset state

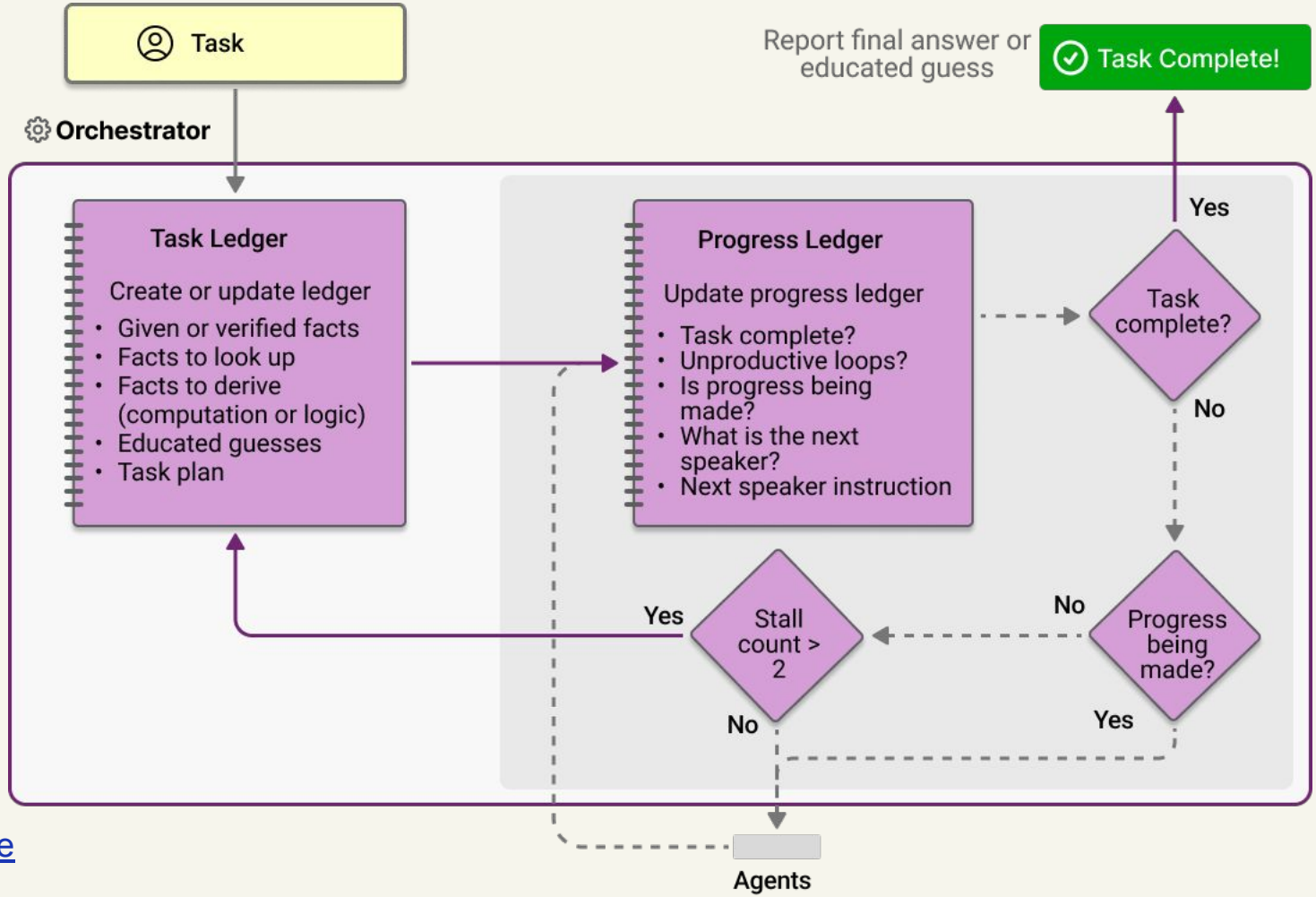
Magnetic-One - A Generalist Multi-Agent System for Solving Complex Tasks

> Of the cities within the United States where U.S. presidents were born, which two are the farthest apart from the westernmost to the easternmost going east, giving the city names only? Give them to me in alphabetical order, in a comma-separated list

An example task from the GAIA benchmark solved by Magentic-One
Task requires **web searching** for presidents' birthplaces, retrieving city coordinates, finding westernmost/easternmost points
coding/computation for distances



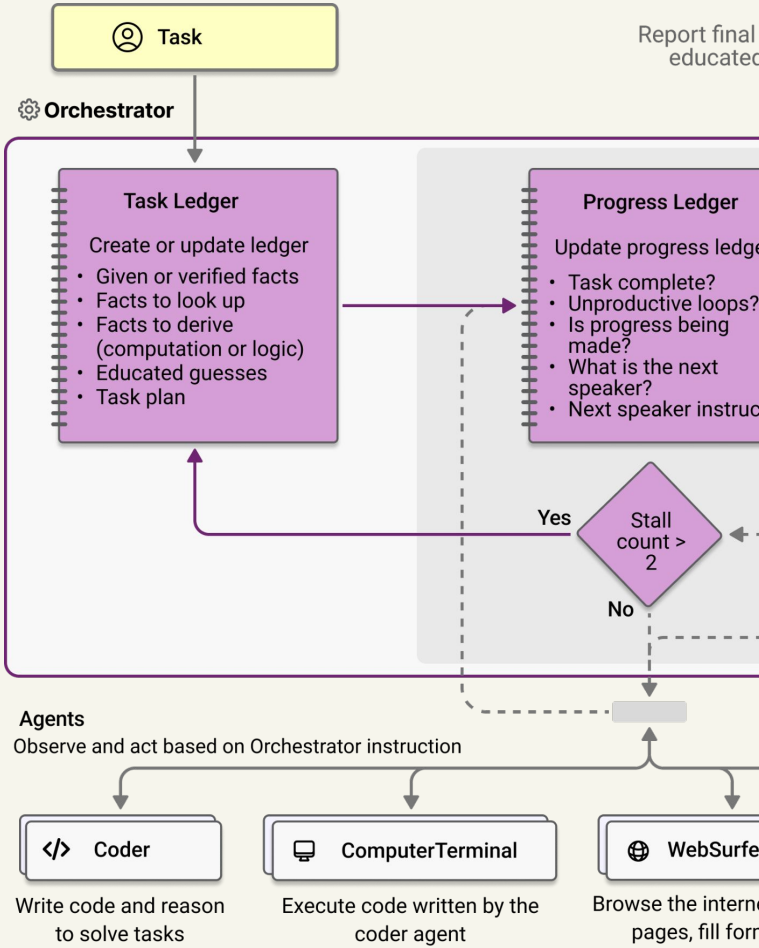
Source [Magnetic-One]



Source [Magnetic-One]

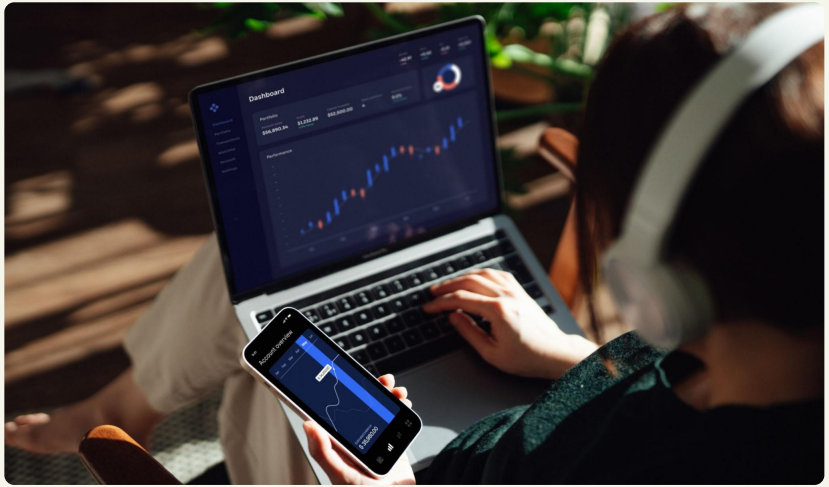
Insights

- Competitive generalization results across multiple task types (web search, file handling, coding, interactive web tasks).
- Use of a ledger shows up to **~31%** increase in task performance



9. You do not have evals for your tasks!

- Evals are critical to understanding the state of your application and how updates to the vast configuration stack impacts **your** task.



10. Your agents do not know *when* to delegate to humans

- Agents that can act should model the **cost/risk** of actions before they are performed
- Intelligently delegate high-risk/irreversible actions to users.

To agents, actions are ~~equal?~~

Action 1

Low

" I made a call to **fetch the weather**. It is Cloudy in San Francisco today "

Action 2

Medium

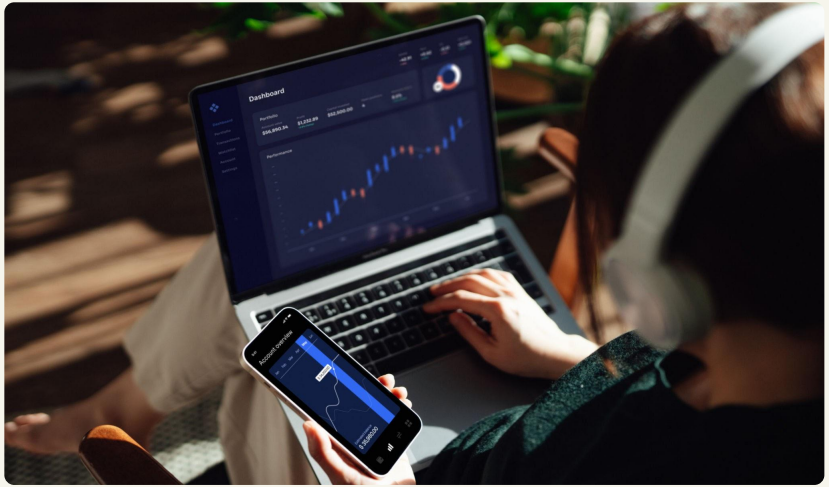
"To free up space, I **deleted 2 video** files"

Action 3

High

"I **transferred $xxxx** to Victor. He asked for it."

Bonus. You probably DO NOT need a multi-agent system.

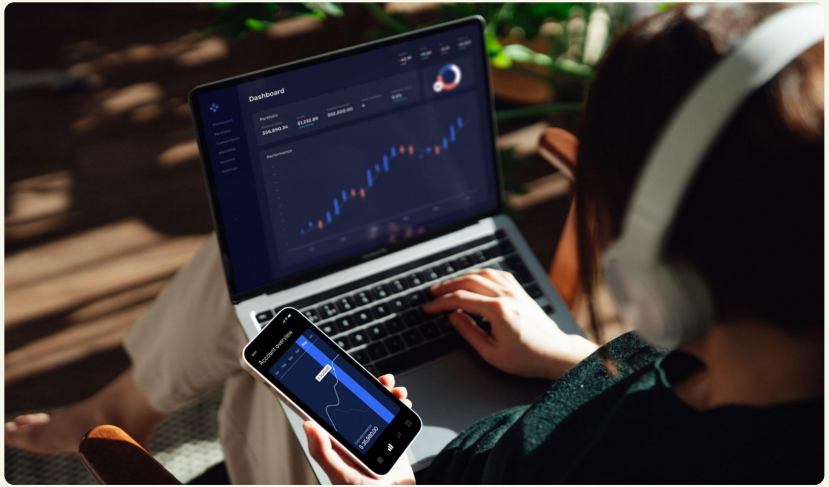


Really .. you very likely do not.

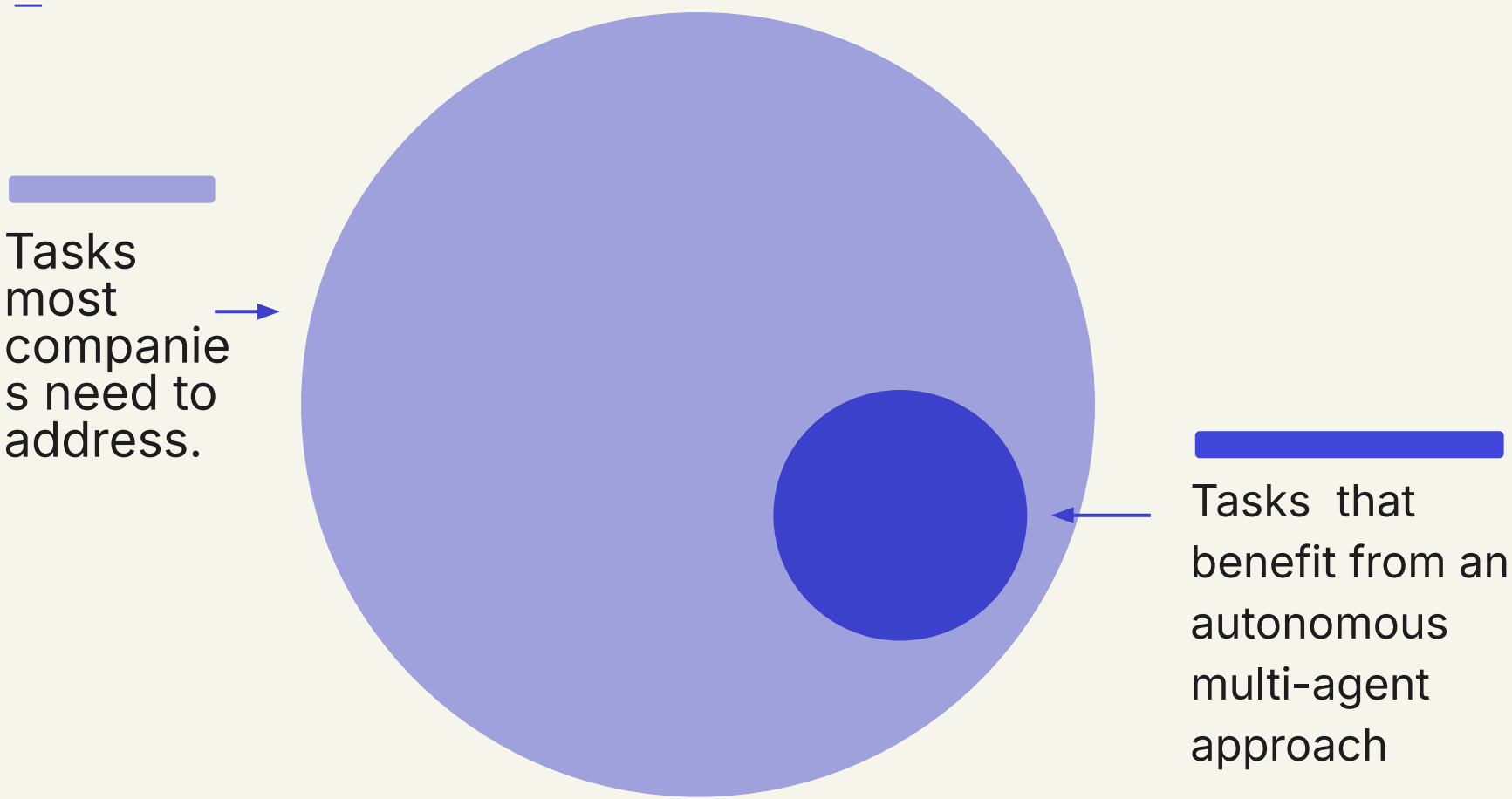
What you can do!

Part 3

0. Know when to use a multi-agent approach!



- Multiple agents collaborating, with autonomy increases the surface for errors and reliability issues
- Like any other tool, they should be selected when they are the right tool for the job.


Tasks most companies need to address.
Tasks that benefit from an autonomous multi-agent approach

How do I know if my task benefits from a multi-agent approach?

A Complex Task Perspective / Checklist

Planning

- Task can be decomposed into a set of steps that lead to a goal state

Adaptive Solution

- Task exists in a dynamic environment, solution is unknown until actions taken

Diverse Perspectives

- Steps in the solution can be mapped into distinct **domains/expertise**

Extensive Context

- Task involves processing extensive context per step

1. Eval driven design

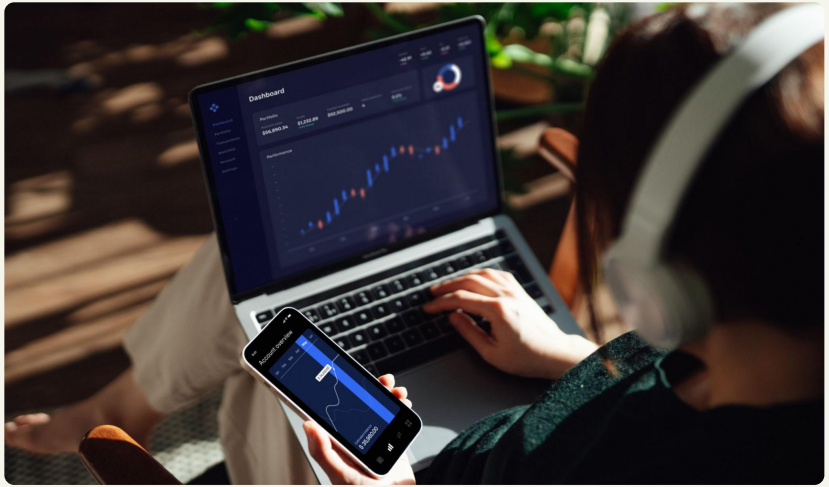
- Define your task
- Define evaluation metrics and test harness
- Build a non-agent baseline
- **Build and improve your agents** and monitor progress on metrics
- Academic benchmarks, while helpful are NOT your task.

2. Constrained, tool-focused implementation

- Invest in building and testing a catalog of high quality tools or functions.
- Attach tools to agents, leverage highly reliably too-calling capabilities in modern LLMs

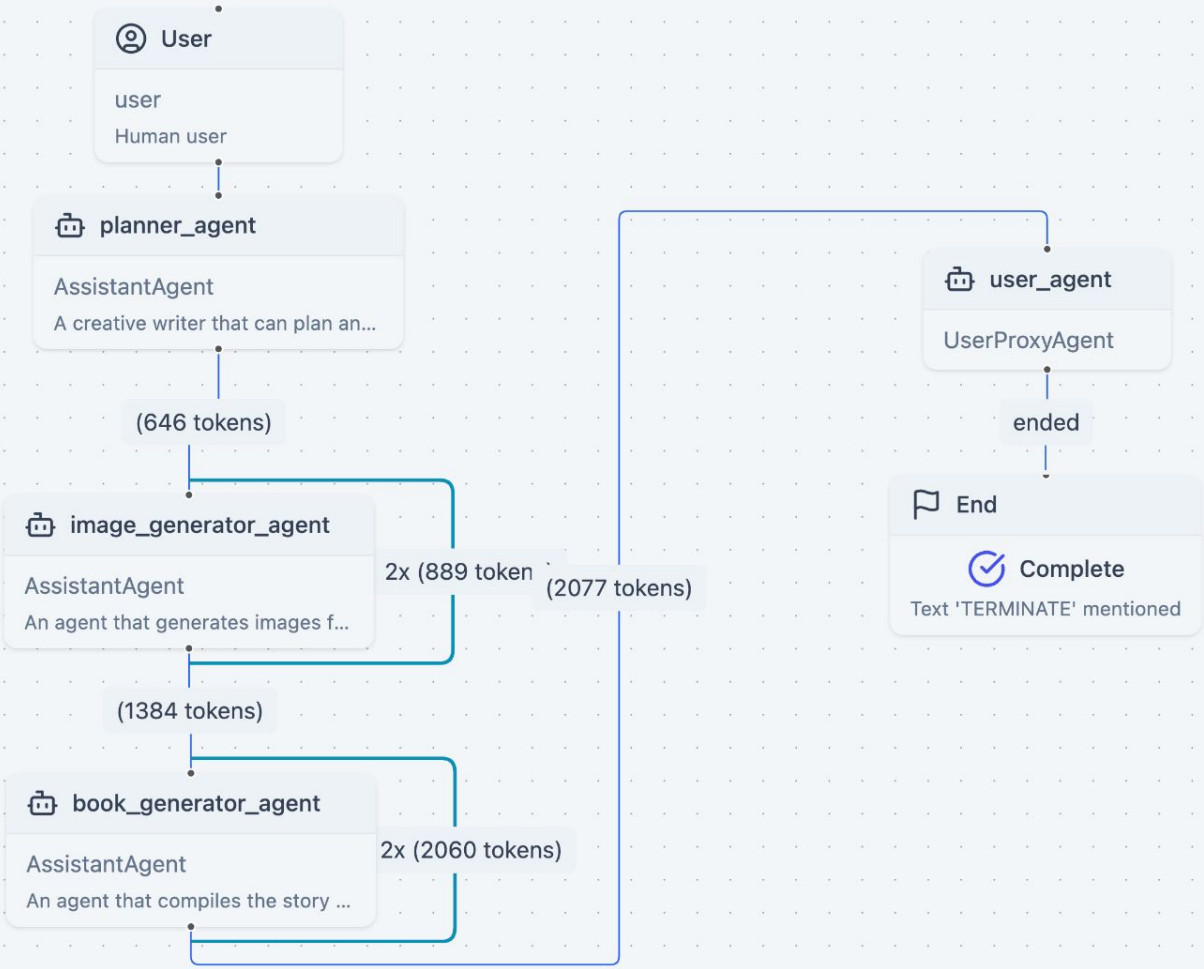
Most production agents today use the LLM to **encapsulate battle-tested reliable business tools**.

3. Observability and Debugging Tools

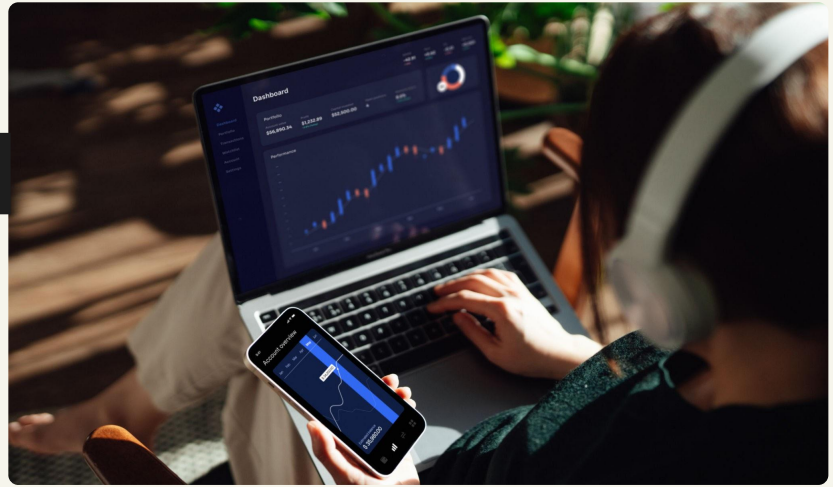


- Invest in observability and provenance tools to help make sense of agent behaviour (e.g. AutoGenStudio)

- Visualize control flows, loops, cost etc

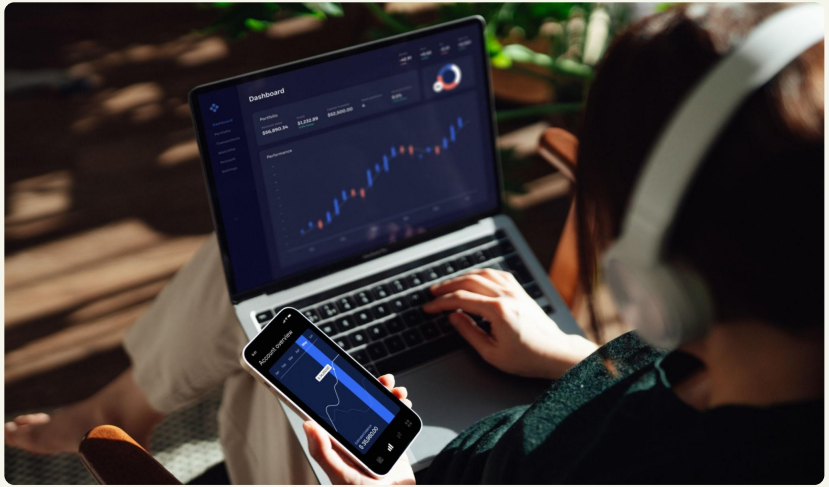

User
user
Human user
planner_agent
AssistantAgent
A creative writer that can plan an...
(646 tokens)
image_generator_agent
AssistantAgent
An agent that generates images f...
2x (889 token
(1384 tokens)
book_generator_agent
AssistantAgent
An agent that compiles the story ...
2x (2060 tokens)
(2077 tokens)
user_agent
UserProxyAgent
ended
End
Complete
Text 'TERMINATE' mentioned

4. Combinations of soft (LLM) and hard Logic (programs)



Future looking:
Consider neuro-symbolic approaches that apply the reasoning capabilities of LLMs but enforce known business logic using clear rules.

Should You Invest in Multi-Agent Systems?



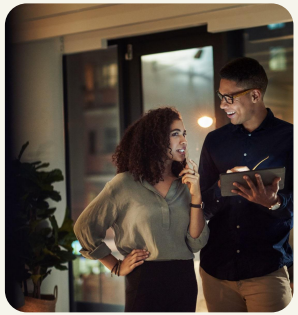
- Let your benchmarks and metrics help you decide
- Models are getting better and many issues are/will be addressed at the model level
- Does your business have disruption exposure?

Note: The right patterns are still emerging

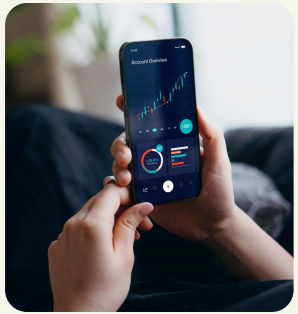
Next Steps

Part 3

What we covered today



What are Multi-agent Systems



How to build them with AutoGen

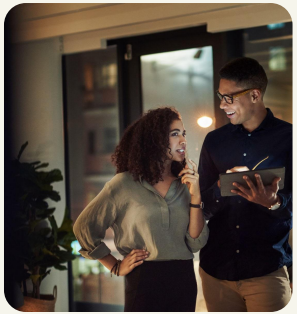
10 common reasons multiagent workflows fail

Insights on steps to take

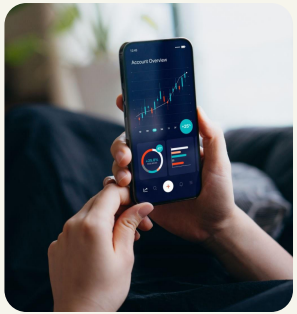
What we did NOT cover today

If interested in any of these topics, I am writing a book.

multiagentbook.com



UX for Multi-Agent Systems



Interface Agents

Optimizing Multi-Agent Systems

Responsible AI Considerations

Multi-Agent Patterns

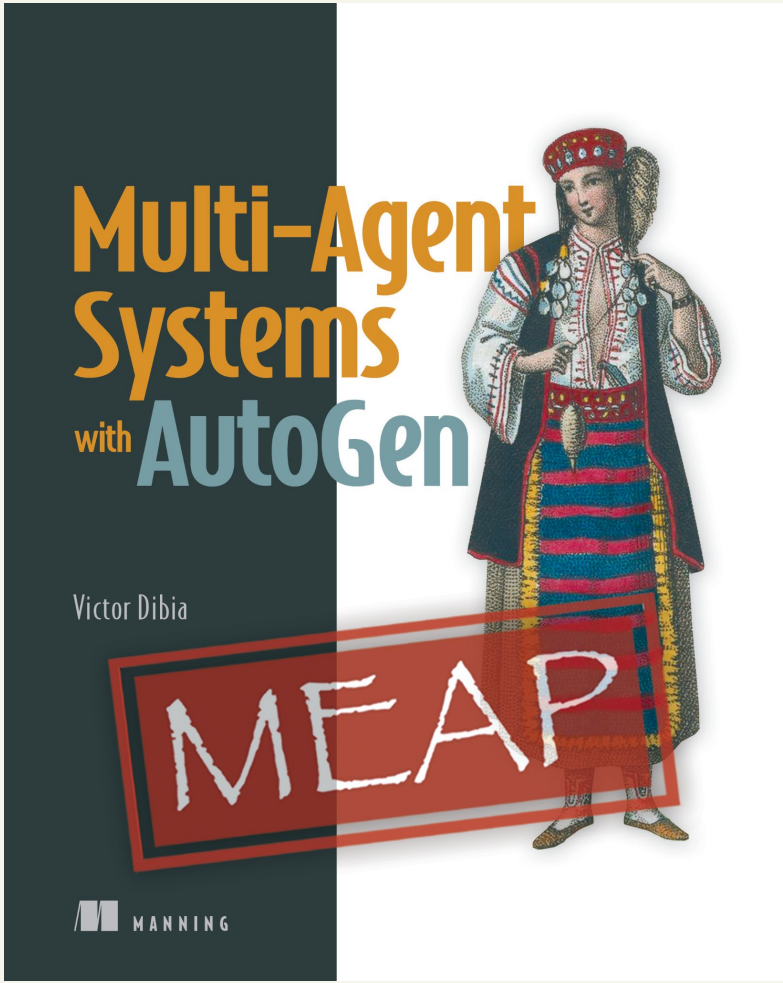
Use Cases

And more ...

Published by Manning

Expected Spring 2025
First 3 Chapters available
on Manning.com

multiagentbook.com



Thank You!

- Multi-Agent Systems with AutoGen
  multiagentbook.com
- Code notebook shown today
- Contribute - **AutoGen on GitHub**