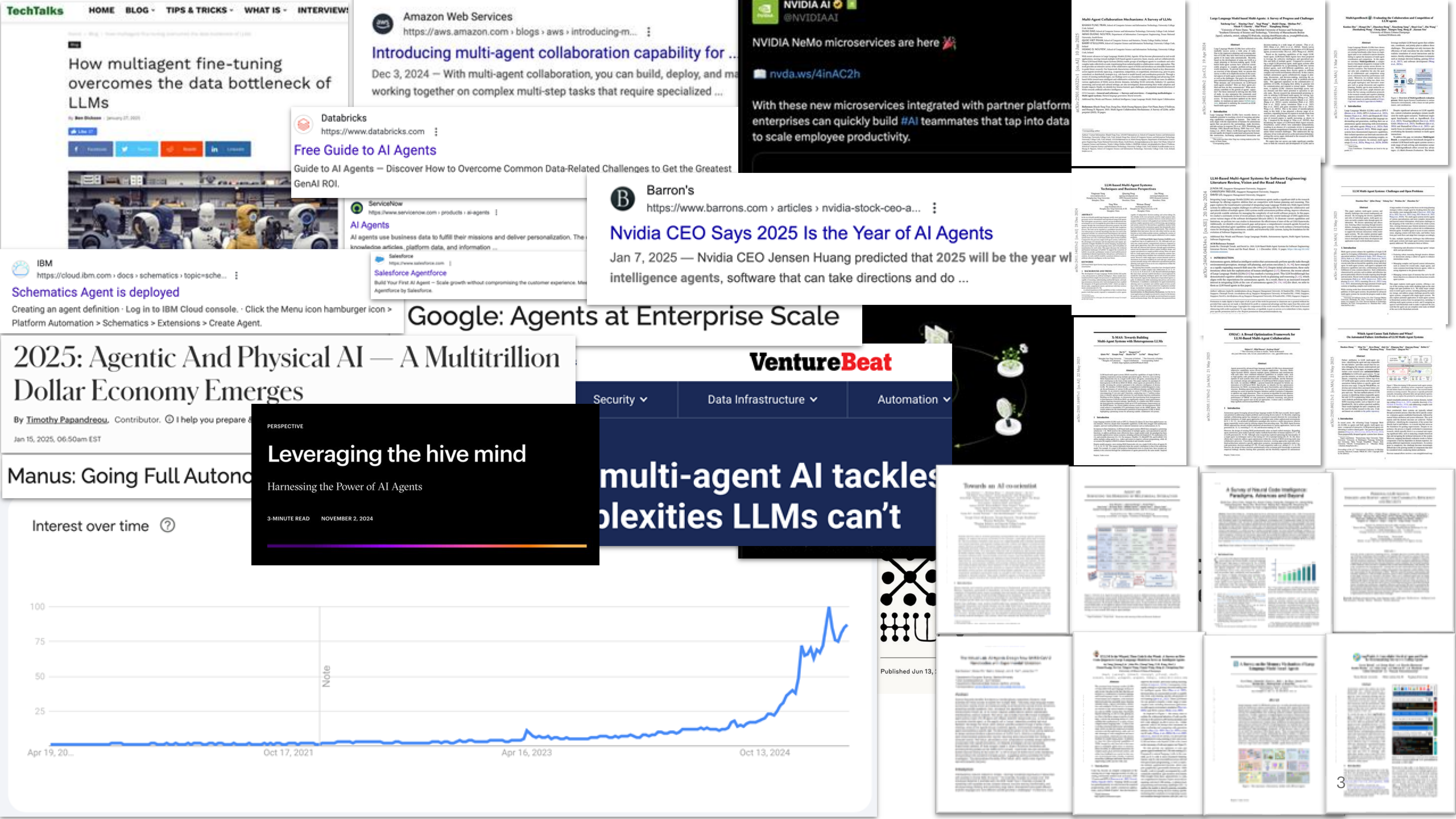
# Why Do Multi-Agent LLM Systems Fail?

Mert Cemri*, **Melissa Pan***, Shuyi Yang*, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, Ion Stoica

Slide by: Melissa Pan

# 2025 AGENTS



TechTalks HOME BLOG TIPS & TRICKS WHAT IS INTERVIEWS

How multiagent fine-tuning overcomes the data bottleneck of LLMs

Amazon Web Services
https://aws.amazon.com › blogs › aws › introducing-m...

Introducing multi-agent collaboration capability for ...

Dec 3, 2024 — With multi-agent collaboration, you can build, deploy, and ma working together on complex multi-step tasks that require specialized ...

Databricks
https://www.databricks.com

Free Guide to AI Agents

Guide to AI Agents — Discover How to Overcome Common Data-Related Challenges to Get the Greatest GenAI ROI.

NVIDIA AI
@NVIDIAAI

NVIDIA NeMo microservices are here 🎉

nvda.ws/4inAC2h

With these new microservices integrated with partner platform enterprises can quickly build #AI teammates that tap into data to scale employee productivity.

ServiceNow
https://www.servicenow.com › products › ai-agents

AI Agents

AI agents use business data to fulfill their missions and deliver personalization. The knowledge articles, platform data, and information ...

Barron's
https://www.barrons.com › articles › nvidia-stock-ceo-a...

Nvidia CEO Says 2025 Is the Year of AI Agents

Jan 7, 2025 — Nvidia CEO Jensen Huang predicted that 2025 will be the year wh intelligence agents, software that can take directions and do ...

IBM
https://cloud.ibm.com › docs › schematics › topic=sche...

Schematics Agent is deployed

Creating an agent definition · Log in to IBM Cloud console. · Click the Menu icon hamburger icon > Platform Automation > Schematics > Extensions > Create Agent.

Salesforce
https://www.salesforce.com

Salesforce Agentforce

Build Your First AI Agent – Scale growth without compromising the customer experience. Me Agentforce by Salesforce.

Google: Agents at Enterprise Scale

2025: Agentic And Physical AI — A Multitrillion Dollar Economy Emerges

By Timothy Papandreou, Contributor. ⓘ I help you prepare

Jan 15, 2025, 06:50am EST

Manus: Going Full Autono

PERSPECTIVE

Leveraging the hive mind

Harnessing the Power of AI Agents

3-MINUTE READ NOVEMBER 2, 2024

VentureBeat

Security ⌄ Data Infrastructure ⌄ Automation ⌄

multi-agent AI tackles plexities LLMs can't

Published Jun 13,

Interest over time ⓘ

Multi-Agent Collaboration Mechanisms: A Survey of LLMs

Large Language Model based Multi-Agents: A Survey of Progress and Challenges

MultiAgentBench: Evaluating the Collaboration and Competition of LLM agents

LLM-based Multi-Agent Systems: Techniques and Business Perspectives

LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision and the Road Ahead

LLM Multi-Agent Systems: Challenges and Open Problems

X-MAS: Towards Building Multi-Agent Systems with Heterogeneous LLMs

OMAC: A Broad Optimization Framework for LLM-Based Multi-Agent Collaboration

Which Agent Causes Task Failures and When? On Automated Failure Attribution of LLM Multi-Agent Systems

Towards an AI co-scientist

A Survey of Neural Code Intelligence: Paradigms, Advances and Beyond

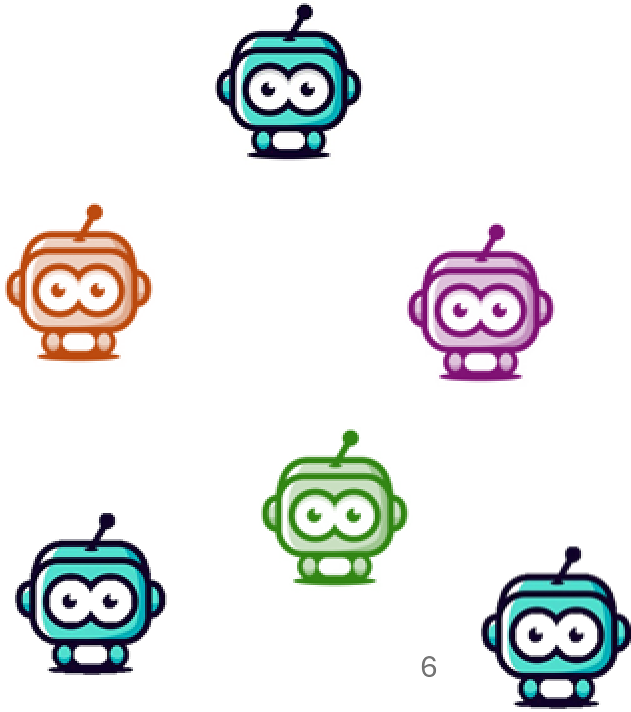
# What is a Multi-Agent Systems?

# What is a Multi-Agent Systems?

A multi-agent system (MAS) exhibits **collective intelligence** from agent-to-agent interactions.

Broadly, each agent is defined by:

- **Specific skills, traits, and actions** towards a goal
- **Autonomy** to make decisions
- Ability to **use external tools** and resources
- **Memory** of its actions, plans, and internal state
- Ability to conduct **inter-agent communications**



# Why Multi-Agent Systems?

① Task Decomposition

② Parallelism and Performance

③ Context Management and Diversity

④ Simulation

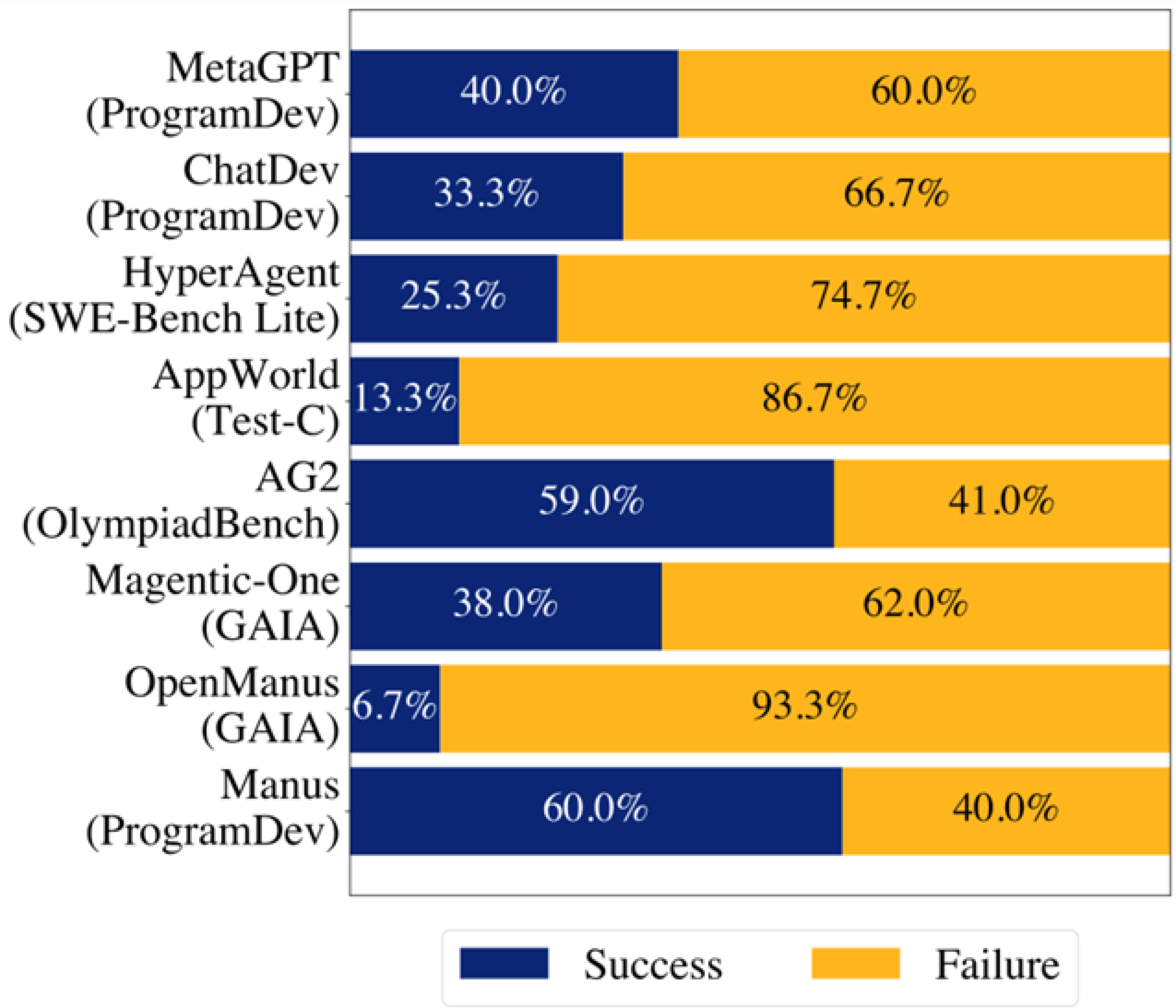
⑤ Ensemble Specialized Models

⑥ Security Isolation

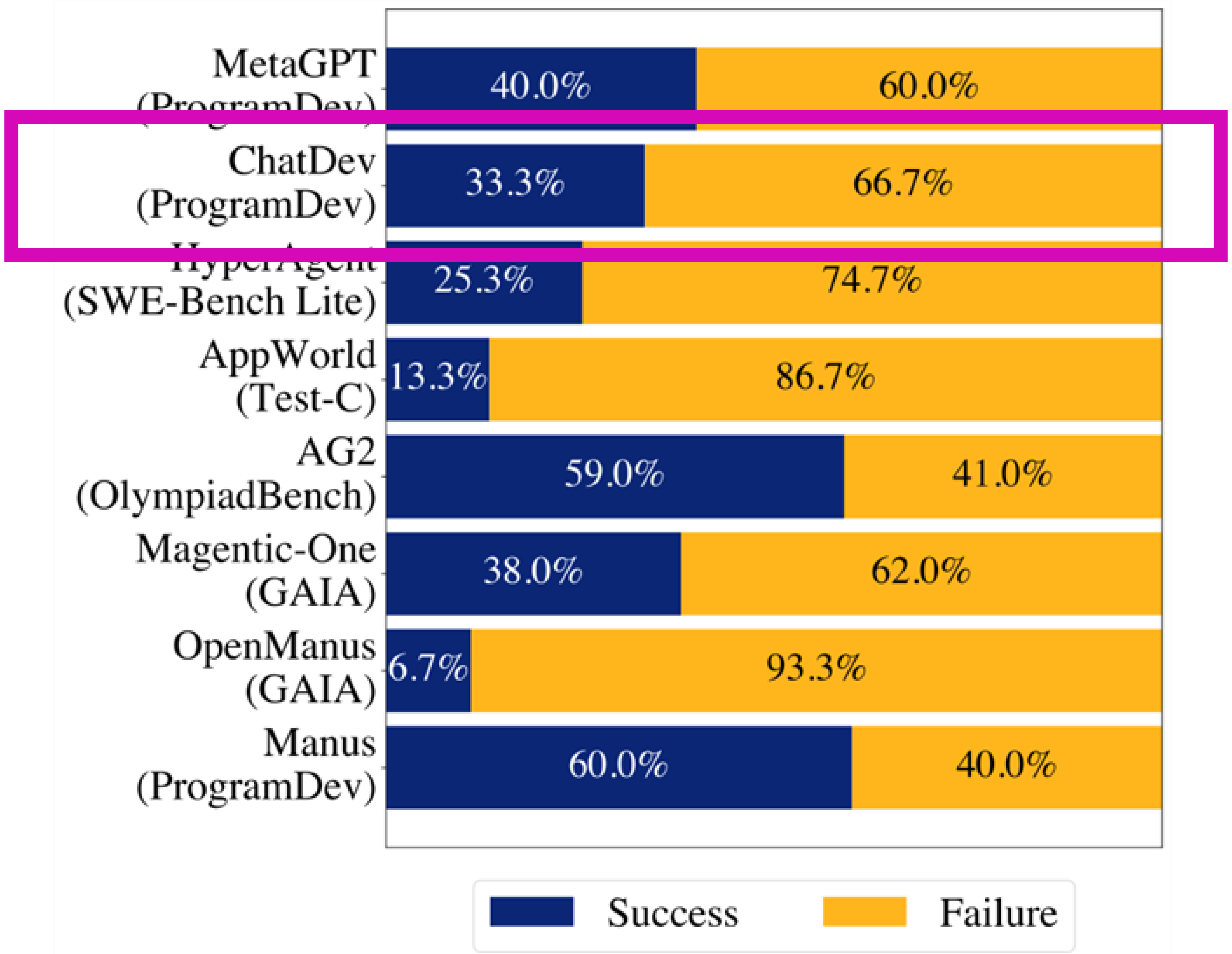
• • •

# BUT

# BUT MAS FAIL ~66% on average



# MAS FAILURES are highly diverse



CEO

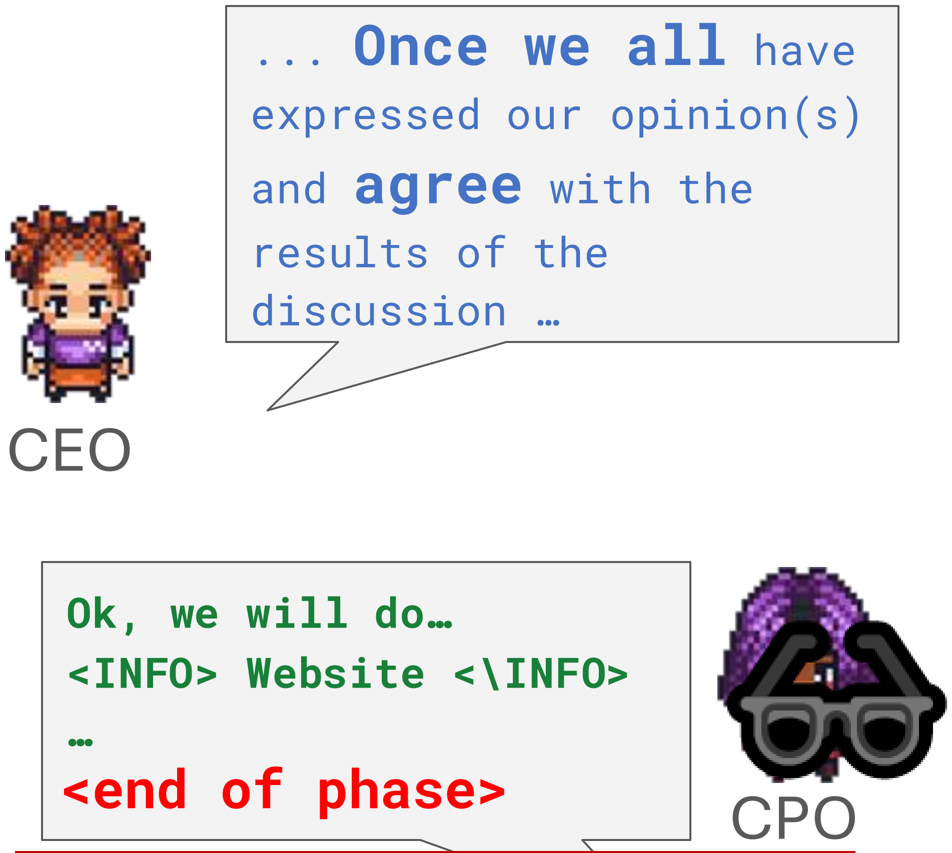
... **Once we all** have expressed our opinion(s) and **agree** with the results of the discussion …

**Ok, we will do…**
**<INFO> Website <\INFO>**
**…**
**<end of phase>**

CPO

Agent ends too soon!

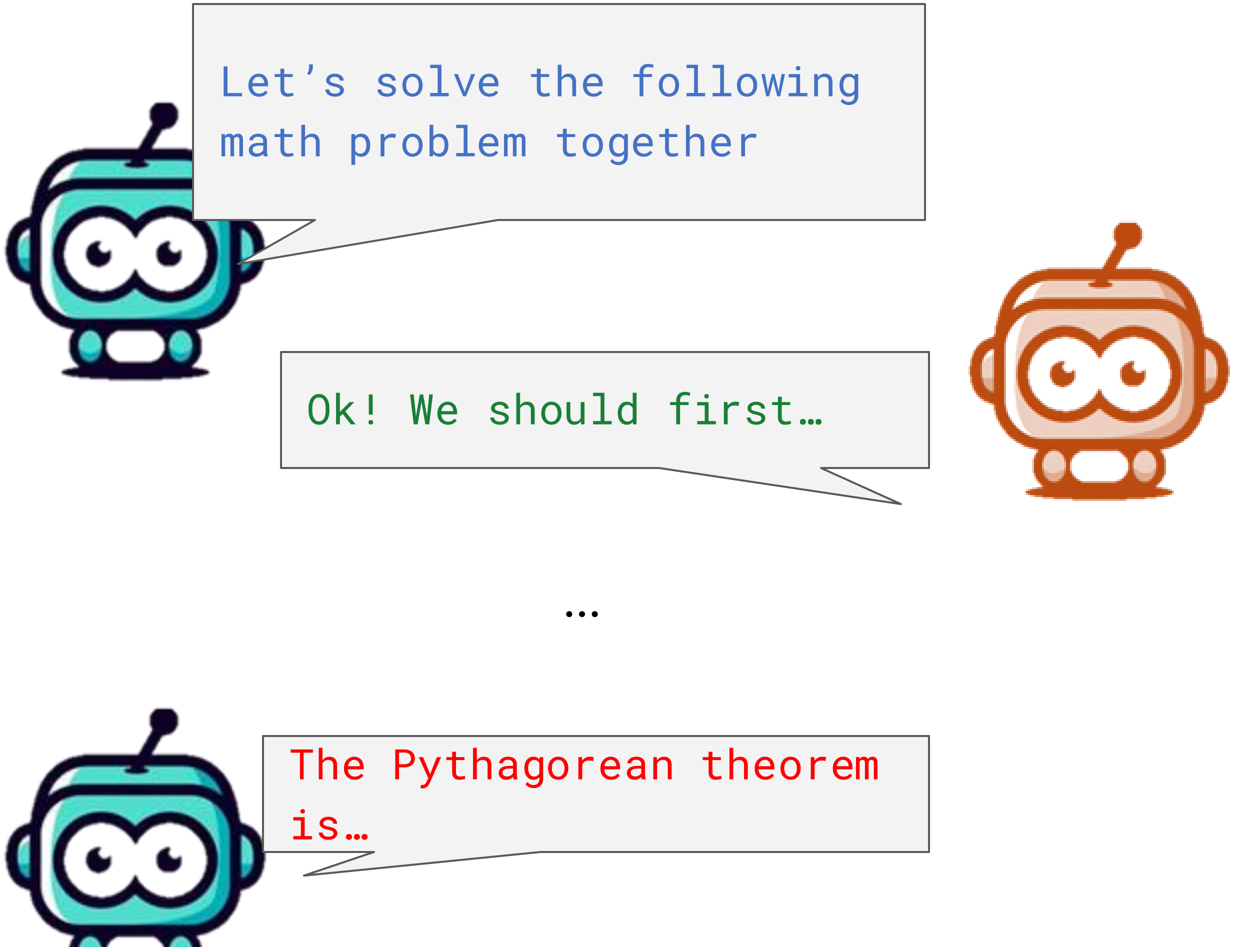
# MAS FAILURES are highly diverse , for example:



... **Once we all** have expressed our opinion(s) and **agree** with the results of the discussion …

**Ok, we will do…**
**<INFO> Website <\INFO>**
**…**
**<end of phase>**

Agent ends too soon!



Let’s solve the following math problem together

Ok! We should first…

…

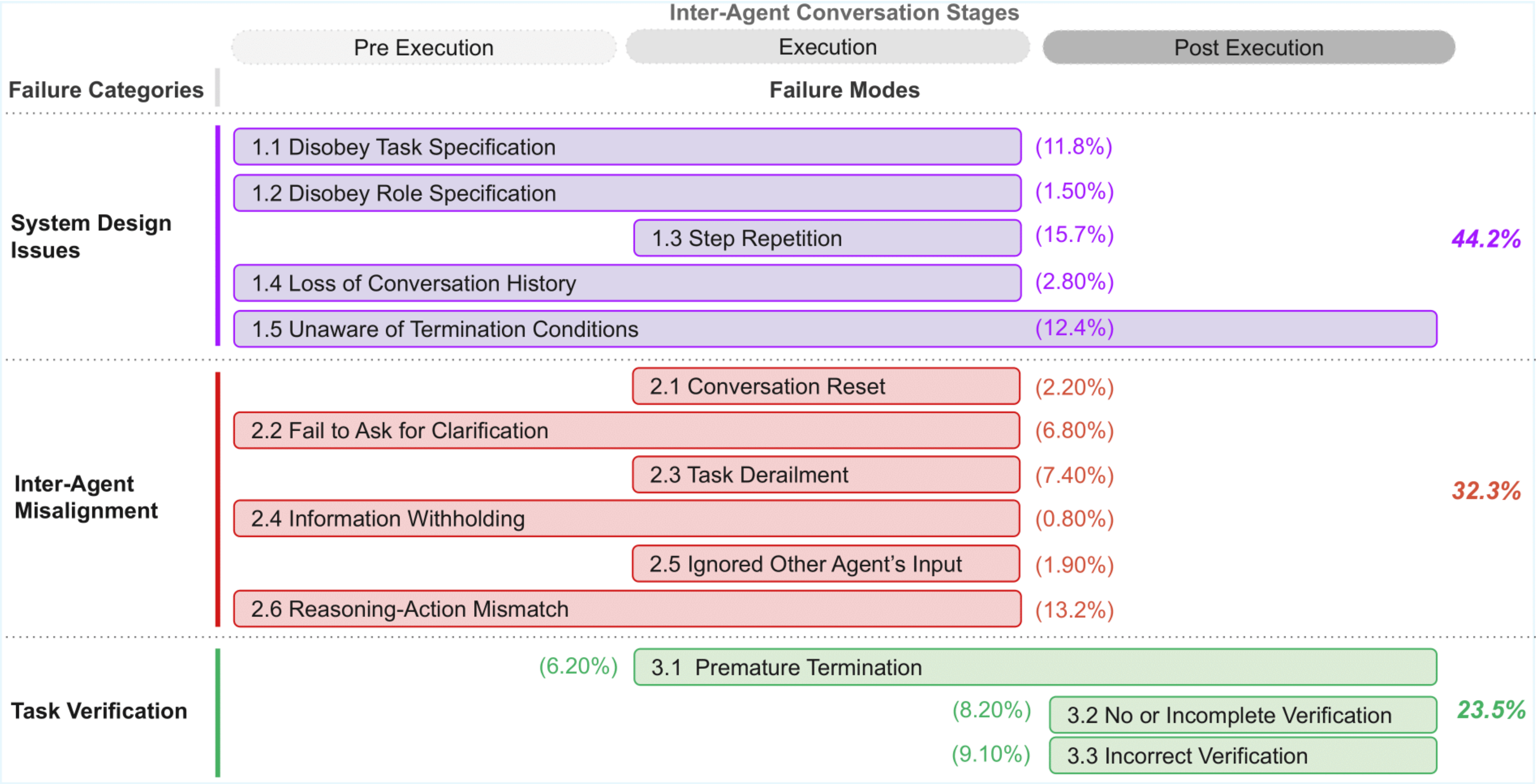
The Pythagorean theorem is…

Talking about something else!

# Reliability

# Why Multi-Agent Systems Fail?

# MAST: Multi-Agent Systems Failure Taxonomy



Inter-Agent Conversation Stages

Pre Execution | Execution | Post Execution

Failure Categories | Failure Modes

**System Design Issues** 44.2%

- 1.1 Disobey Task Specification (11.8%)
- 1.2 Disobey Role Specification (1.50%)
- 1.3 Step Repetition (15.7%)
- 1.4 Loss of Conversation History (2.80%)
- 1.5 Unaware of Termination Conditions (12.4%)

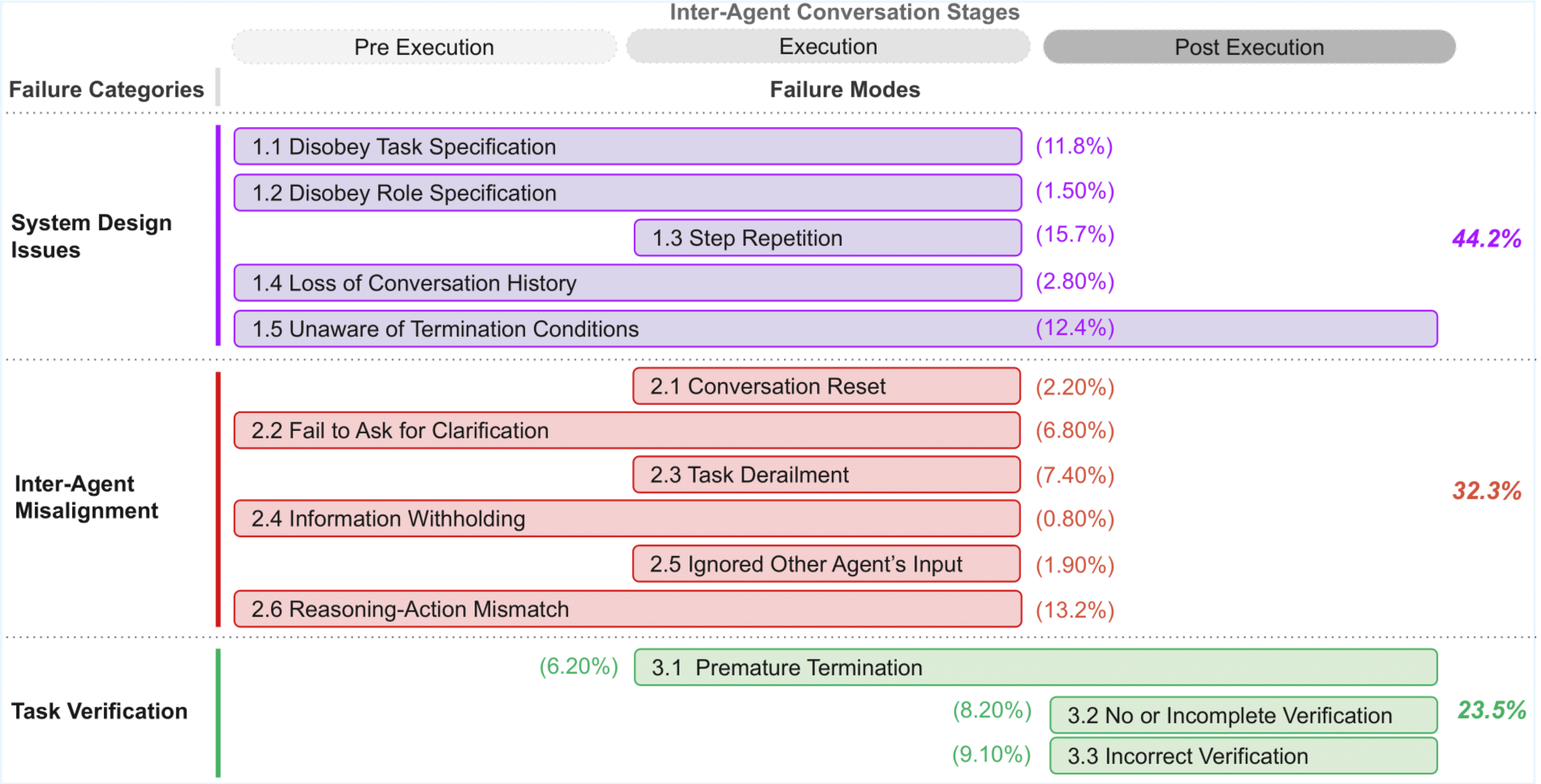
**Inter-Agent Misalignment** 32.3%

- 2.1 Conversation Reset (2.20%)
- 2.2 Fail to Ask for Clarification (6.80%)
- 2.3 Task Derailment (7.40%)
- 2.4 Information Withholding (0.80%)
- 2.5 Ignored Other Agent's Input (1.90%)
- 2.6 Reasoning-Action Mismatch (13.2%)

**Task Verification** 23.5%

- (6.20%) 3.1 Premature Termination
- (8.20%) 3.2 No or Incomplete Verification
- (9.10%) 3.3 Incorrect Verification

# MAST: Multi-Agent Systems Failure Taxonomy



Inter-Agent Conversation Stages

Pre Execution | Execution | Post Execution

| Failure Categories | Failure Modes | | |
|---|---|---|---|
| System Design Issues | 1.1 Disobey Task Specification | (11.8%) | *44.2%* |
| | 1.2 Disobey Role Specification | (1.50%) | |
| | 1.3 Step Repetition | (15.7%) | |
| | 1.4 Loss of Conversation History | (2.80%) | |
| | 1.5 Unaware of Termination Conditions | (12.4%) | |
| Inter-Agent Misalignment | 2.1 Conversation Reset | (2.20%) | *32.3%* |
| | 2.2 Fail to Ask for Clarification | (6.80%) | |
| | 2.3 Task Derailment | (7.40%) | |
| | 2.4 Information Withholding | (0.80%) | |
| | 2.5 Ignored Other Agent's Input | (1.90%) | |
| | 2.6 Reasoning-Action Mismatch | (13.2%) | |
| Task Verification | 3.1 Premature Termination | (6.20%) | *23.5%* |
| | 3.2 No or Incomplete Verification | (8.20%) | |
| | 3.3 Incorrect Verification | (9.10%) | |

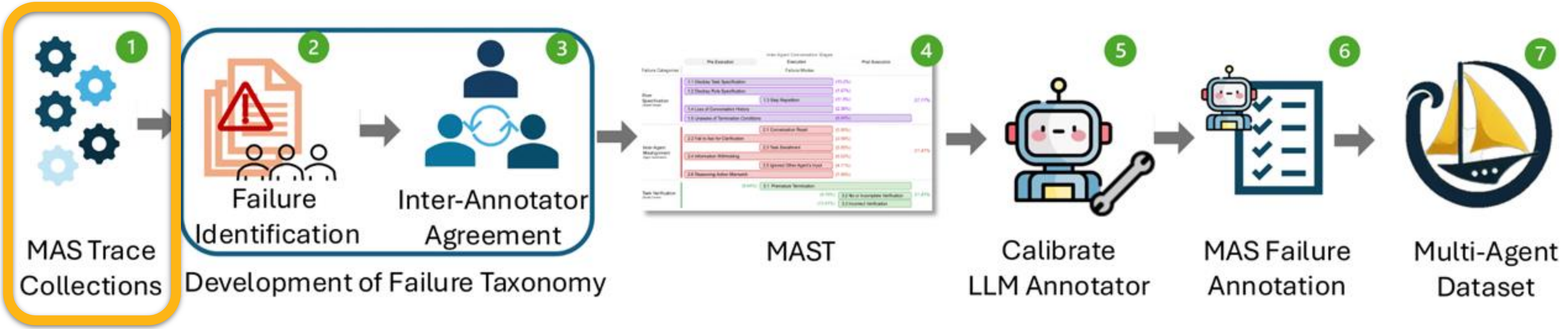
- The **first** MAS failure taxonomy
- **14** unique failure modes
- **3** main failure categories

MAST outlines **a roadmap** for future research to build more reliable and effective MAS.

# Outline

- Motivation
- Prior work
- MAST & MAD overview
- Study methodology
- Study finding & insights
- Practical use of MAST
- Q&A

# Method: Towards Uniform MAS Failure Framework



1 MAS Trace Collections

2 Failure Identification

3 Inter-Annotator Agreement

Development of Failure Taxonomy

4 MAST

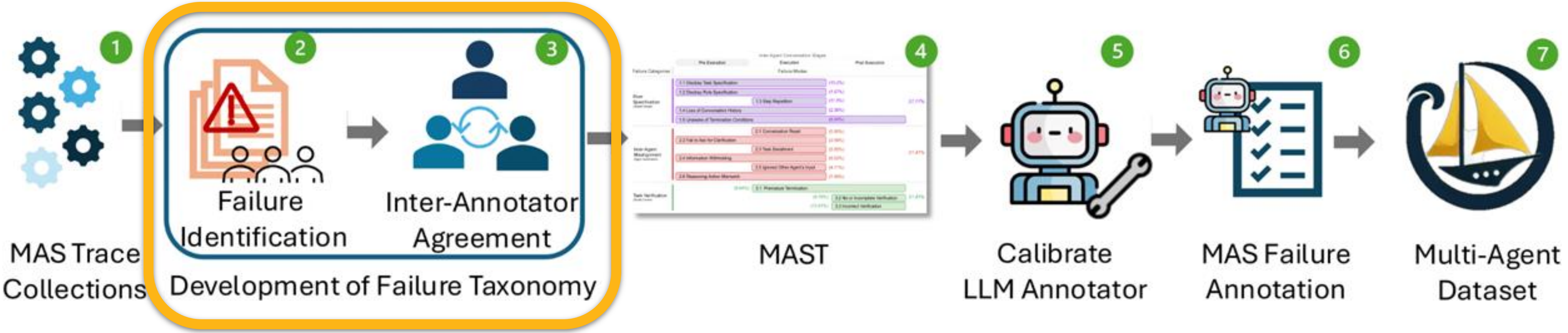
5 Calibrate LLM Annotator

6 MAS Failure Annotation

7 Multi-Agent Dataset

150 traces from 5 systems: MetaGPT, ChatDev, AG2, HyperAgent, AppWorld averaging over 15,000 lines of text per trace

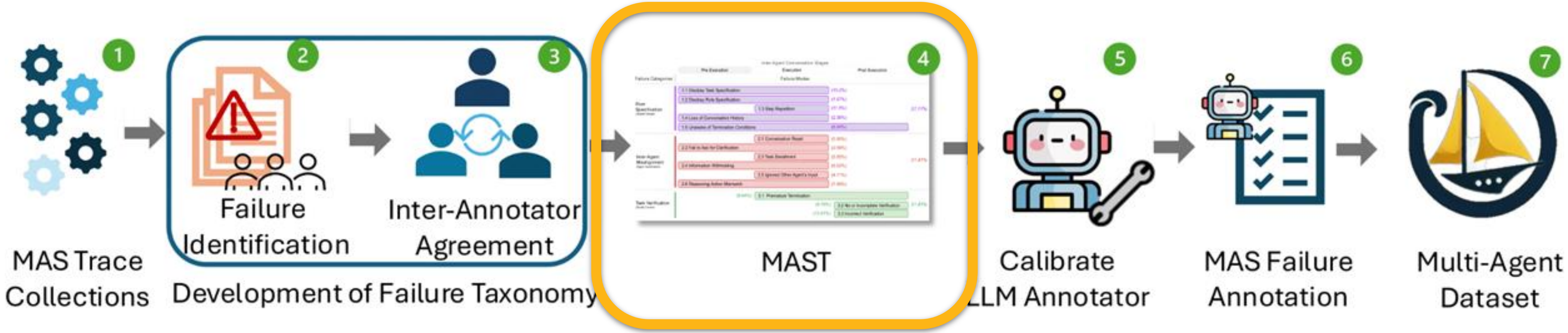
# Method: Towards Uniform MAS Failure Framework



**κ = 0.88**

0.4 is considered good ☺

# Method: Towards Uniform MAS Failure Framework



1 MAS Trace Collections

2 Failure Identification

3 Inter-Annotator Agreement

Development of Failure Taxonomy

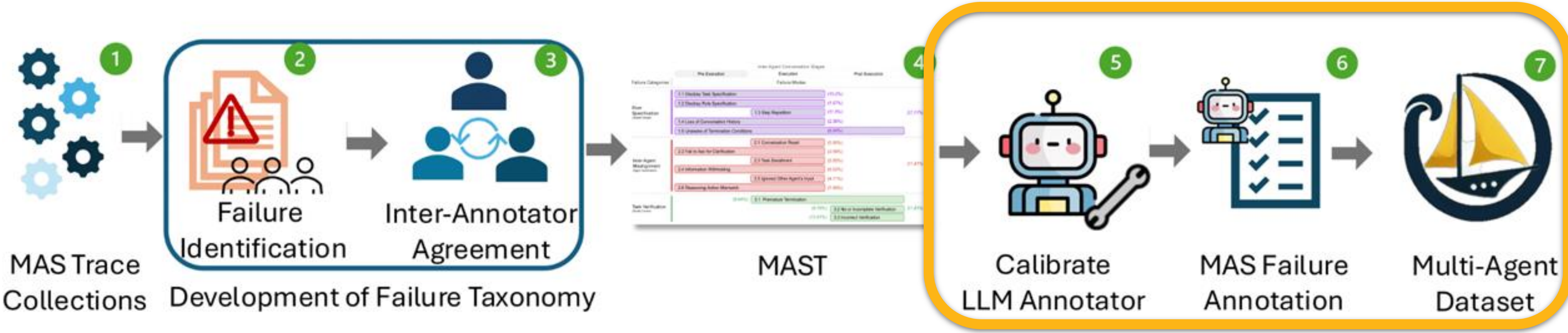
4 MAST

5 Calibrate LLM Annotator

6 MAS Failure Annotation

7 Multi-Agent Dataset

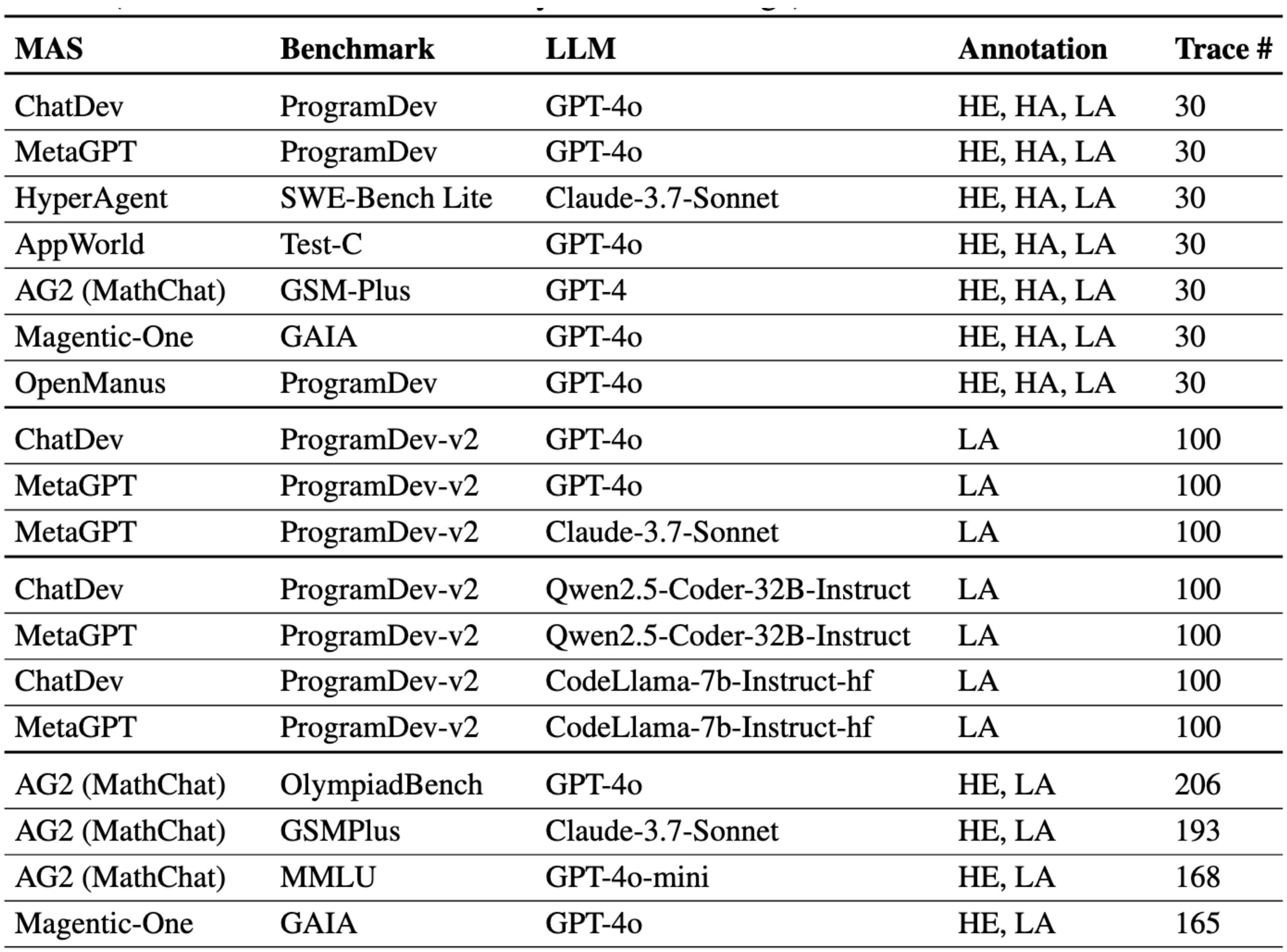
# Method: Towards Uniform MAS Failure Framework



$\kappa = 0.77$

0.4 is considered good ☺

# MAST-Data: Multi-Agent Failure Dataset



| MAS | Benchmark | LLM | Annotation | Trace # |
|---|---|---|---|---|
| ChatDev | ProgramDev | GPT-4o | HE, HA, LA | 30 |
| MetaGPT | ProgramDev | GPT-4o | HE, HA, LA | 30 |
| HyperAgent | SWE-Bench Lite | Claude-3.7-Sonnet | HE, HA, LA | 30 |
| AppWorld | Test-C | GPT-4o | HE, HA, LA | 30 |
| AG2 (MathChat) | GSM-Plus | GPT-4 | HE, HA, LA | 30 |
| Magentic-One | GAIA | GPT-4o | HE, HA, LA | 30 |
| OpenManus | ProgramDev | GPT-4o | HE, HA, LA | 30 |
| ChatDev | ProgramDev-v2 | GPT-4o | LA | 100 |
| MetaGPT | ProgramDev-v2 | GPT-4o | LA | 100 |
| MetaGPT | ProgramDev-v2 | Claude-3.7-Sonnet | LA | 100 |
| ChatDev | ProgramDev-v2 | Qwen2.5-Coder-32B-Instruct | LA | 100 |
| MetaGPT | ProgramDev-v2 | Qwen2.5-Coder-32B-Instruct | LA | 100 |
| ChatDev | ProgramDev-v2 | CodeLlama-7b-Instruct-hf | LA | 100 |
| MetaGPT | ProgramDev-v2 | CodeLlama-7b-Instruct-hf | LA | 100 |
| AG2 (MathChat) | OlympiadBench | GPT-4o | HE, LA | 206 |
| AG2 (MathChat) | GSMPlus | Claude-3.7-Sonnet | HE, LA | 193 |
| AG2 (MathChat) | MMLU | GPT-4o-mini | HE, LA | 168 |
| Magentic-One | GAIA | GPT-4o | HE, LA | 165 |

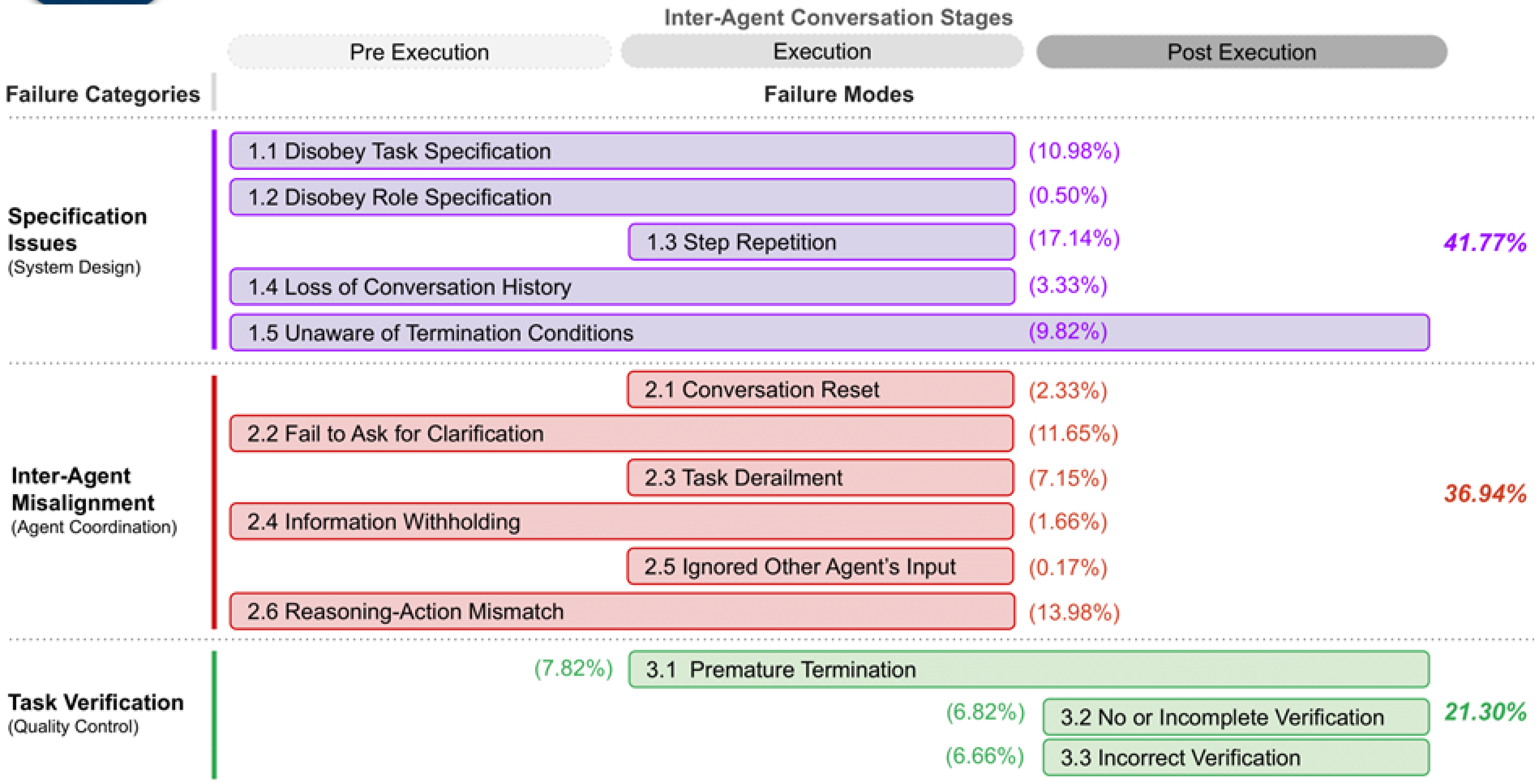
- Fully open sourced
- **1642** traces from:
  - 7 systems
  - 8 benchmarks
  - 3 tasks domains
  - 3 model families

# Our Contributions

- Manually inspected **150+** traces to build a failure taxonomy
- Created LLM judge to evaluate **1600+** more traces (MAST-Data)
- Open-sourced taxonomy (**MAST**) and dataset (**MAST-Data**)

**code**

# Thank You! Come Talk to Us!



Inter-Agent Conversation Stages

Pre Execution | Execution | Post Execution

Failure Categories | Failure Modes

**Specification Issues** (System Design) — 41.77%

- 1.1 Disobey Task Specification (10.98%)
- 1.2 Disobey Role Specification (0.50%)
- 1.3 Step Repetition (17.14%)
- 1.4 Loss of Conversation History (3.33%)
- 1.5 Unaware of Termination Conditions (9.82%)

**Inter-Agent Misalignment** (Agent Coordination) — 36.94%

- 2.1 Conversation Reset (2.33%)
- 2.2 Fail to Ask for Clarification (11.65%)
- 2.3 Task Derailment (7.15%)
- 2.4 Information Withholding (1.66%)
- 2.5 Ignored Other Agent's Input (0.17%)
- 2.6 Reasoning-Action Mismatch (13.98%)

**Task Verification** (Quality Control) — 21.30%

- (7.82%) 3.1 Premature Termination
- (6.82%) 3.2 No or Incomplete Verification
- (6.66%) 3.3 Incorrect Verification

code

paper

@melissapan

melissapan@berkeley.edu

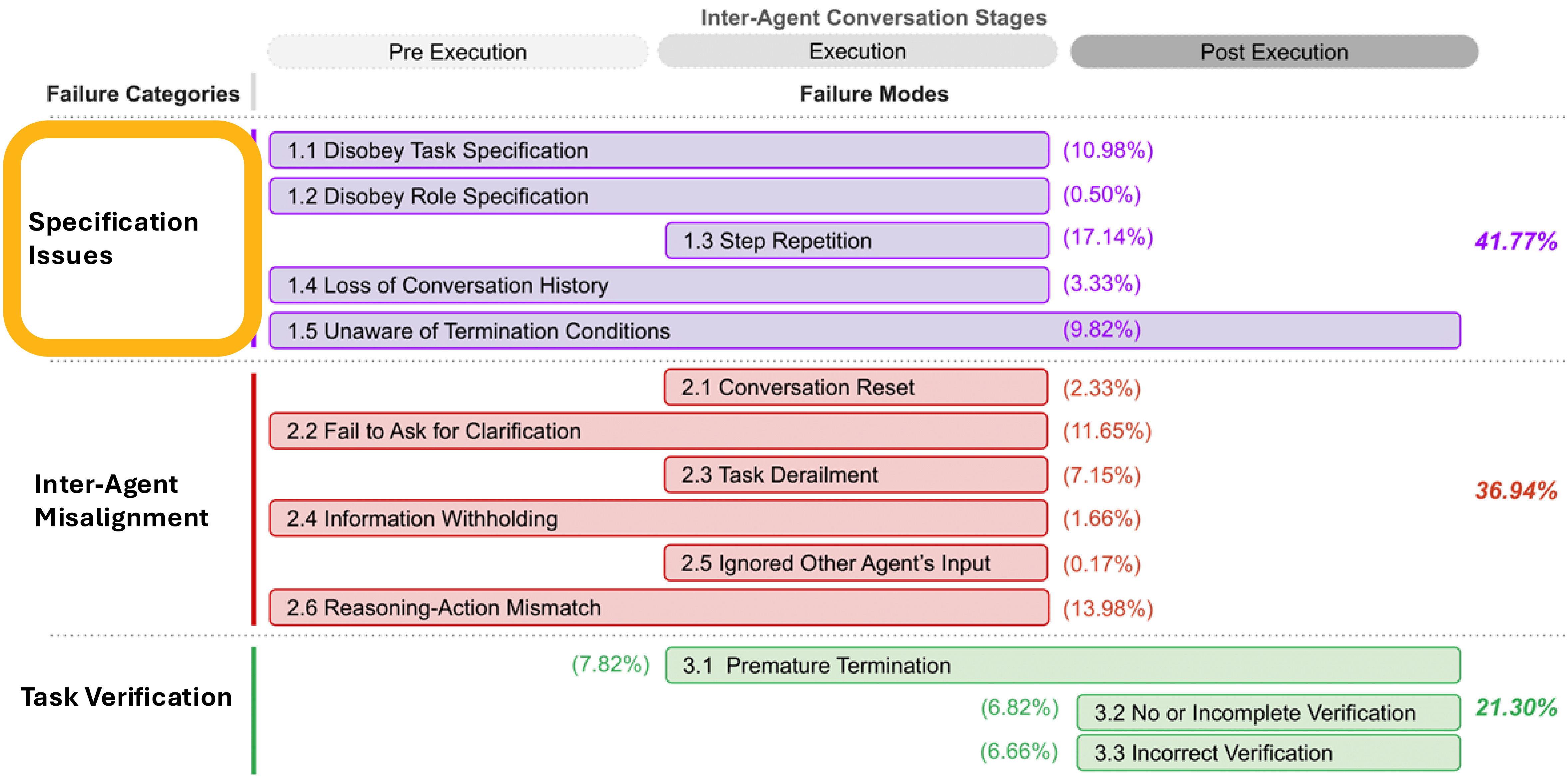
**Do you need help using MAST to build & evaluate Agents?**

github.com/multi-agent-systems-failure-taxonomy/MAST

# Outline

- Motivation
- Prior work
- MAST & MAD overview
- Study methodology
- Study finding & insights
- Practical use of MAST
- Q&A

# MAST: Multi-Agent Systems Failure Taxonomy



Inter-Agent Conversation Stages

Pre Execution

Execution

Post Execution

Failure Categories

Failure Modes

Specification Issues

1.1 Disobey Task Specification (10.98%)

1.2 Disobey Role Specification (0.50%)

1.3 Step Repetition (17.14%)

1.4 Loss of Conversation History (3.33%)

1.5 Unaware of Termination Conditions (9.82%)

41.77%

Inter-Agent Misalignment

2.1 Conversation Reset (2.33%)

2.2 Fail to Ask for Clarification (11.65%)

2.3 Task Derailment (7.15%)

2.4 Information Withholding (1.66%)

2.5 Ignored Other Agent's Input (0.17%)

2.6 Reasoning-Action Mismatch (13.98%)

36.94%

Task Verification

(7.82%) 3.1 Premature Termination

(6.82%) 3.2 No or Incomplete Verification

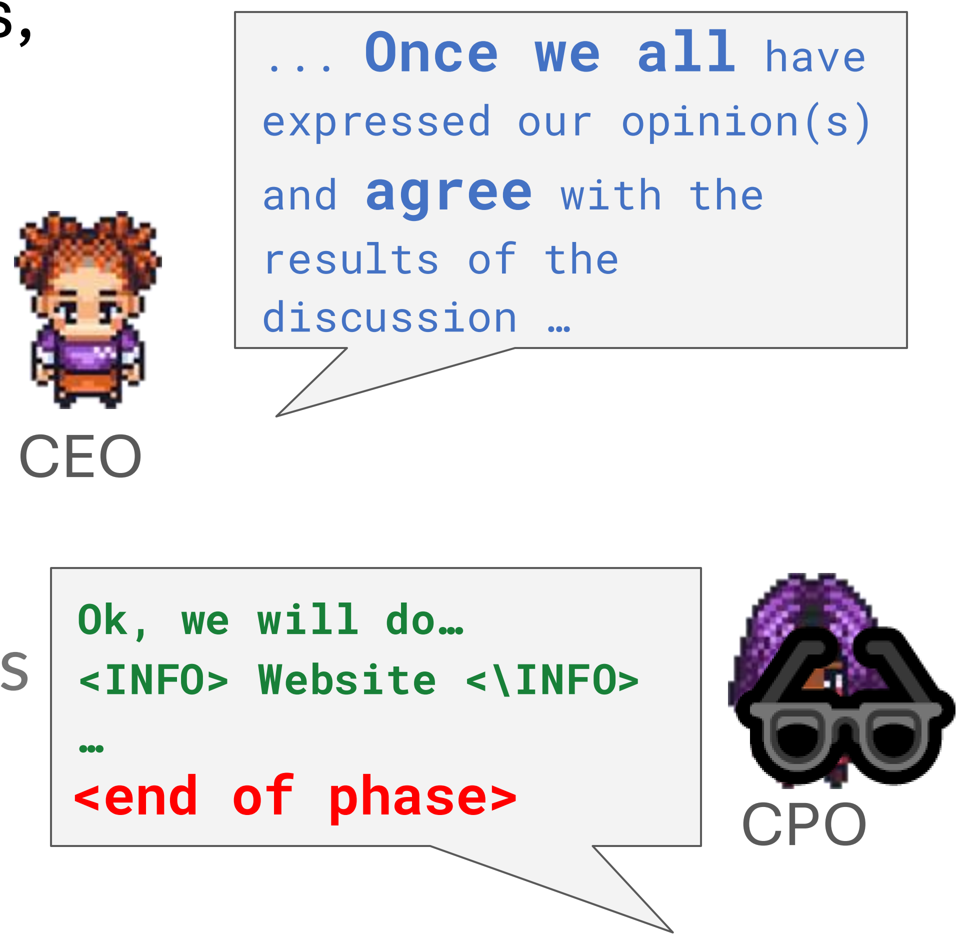
(6.66%) 3.3 Incorrect Verification

21.30%

# FC1: Specification Issues

Failures originate from **system design** decisions, and poor or ambiguous prompt specifications.

1. Disobey Task Specifications
2. Disobey Role Specifications
3. Step Repetition
4. Conversation Loss
5. Agents Unaware of Termination Conditions



... **Once we all** have expressed our opinion(s) and **agree** with the results of the discussion …

**Ok, we will do…**
**<INFO> Website <\INFO>**
**…**
**<end of phase>**

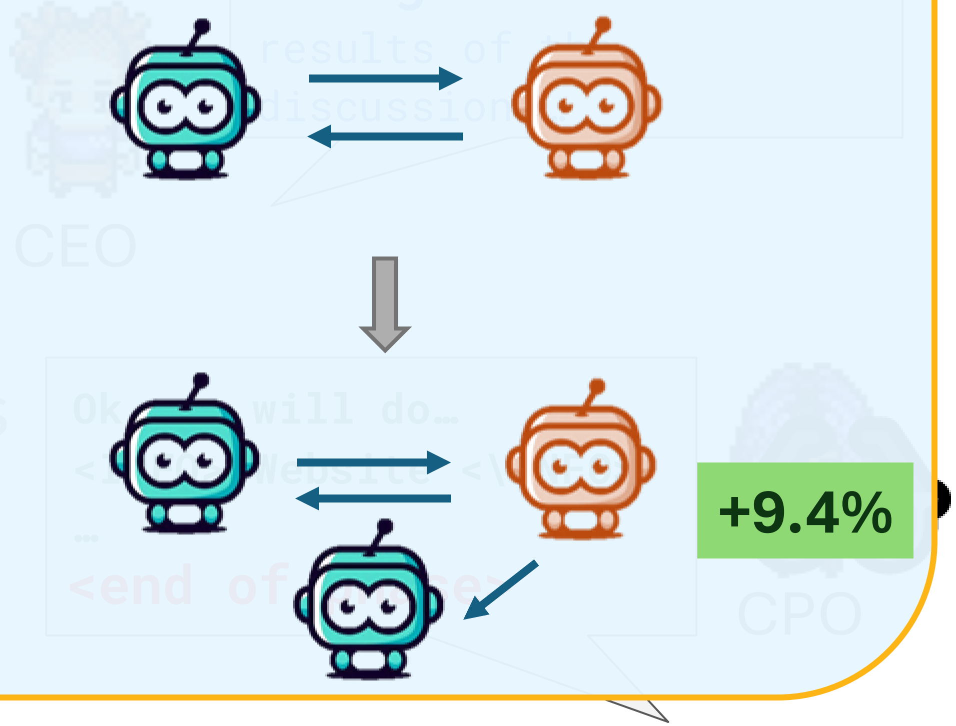
# FC1: Specification Issues

Failures originate from system design decisions, and poor or ambiguous prompt specifications.

1. Disobey Task Specification
2. Disobey Role Specifications
3. Step Repetition
4. Conversation Loss
5. Agents Unaware of Termination Conditions

## Isn't it just a limitation of the underlying LLM? 🤔

1. Multi-Agent system design ✴
2. User prompt specification
3. Limitation of the underlying model

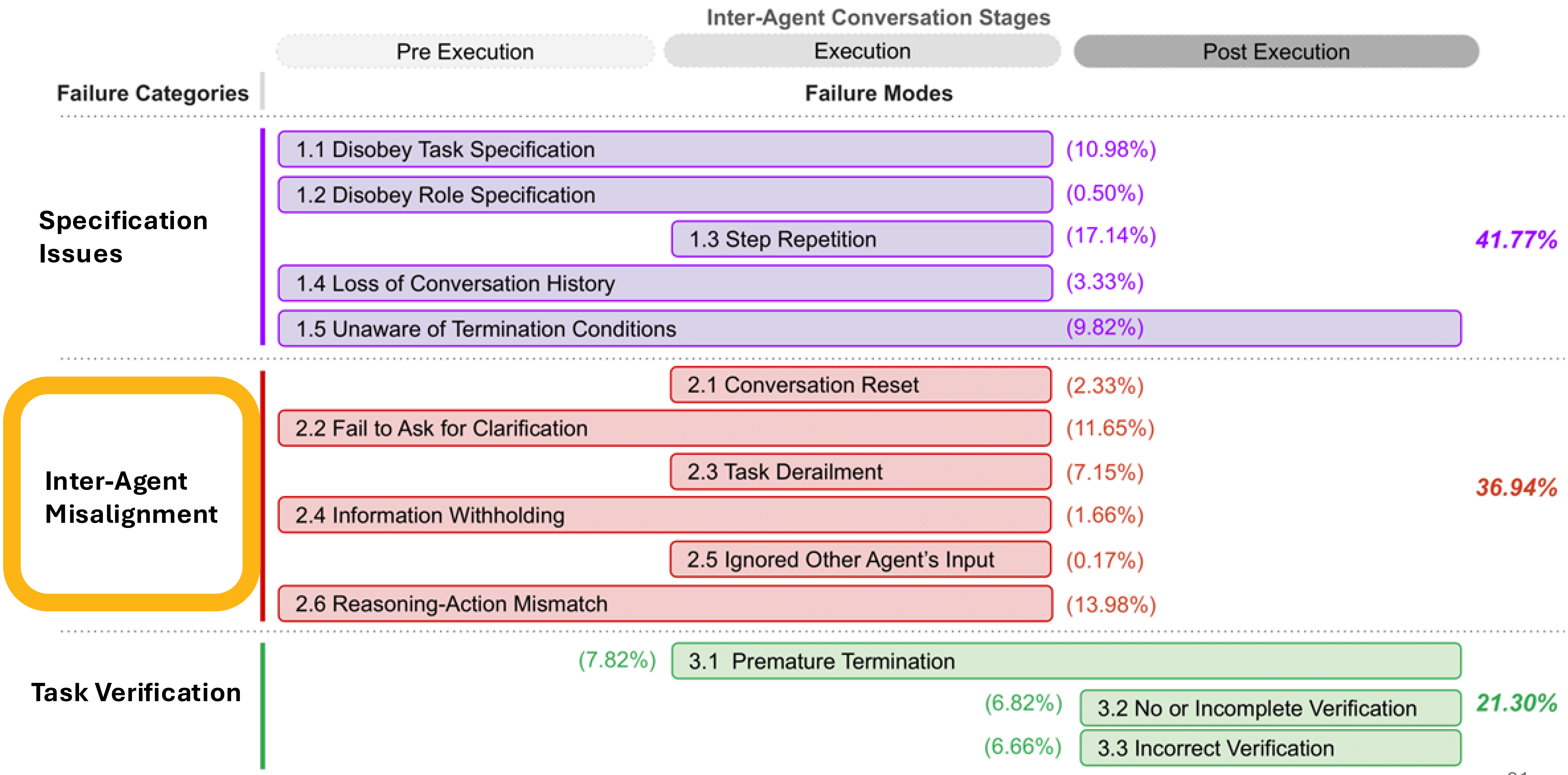


... Once we all have ... our opinion(s) and agree with the results of the discussion
CEO
Ok [illegible] will do…
<INFO> Website <\INFO>
…
<end of phase>
CPO
+9.4%

# **FC1**: Specification Issues

A **well-designed** MAS can get **performance gain** with using the same model

# MAST: Multi-Agent Systems Failure Taxonomy



Inter-Agent Conversation Stages

Pre Execution | Execution | Post Execution

Failure Categories | Failure Modes

**Specification Issues** — 41.77%

- 1.1 Disobey Task Specification (10.98%)
- 1.2 Disobey Role Specification (0.50%)
- 1.3 Step Repetition (17.14%)
- 1.4 Loss of Conversation History (3.33%)
- 1.5 Unaware of Termination Conditions (9.82%)

**Inter-Agent Misalignment** — 36.94%

- 2.1 Conversation Reset (2.33%)
- 2.2 Fail to Ask for Clarification (11.65%)
- 2.3 Task Derailment (7.15%)
- 2.4 Information Withholding (1.66%)
- 2.5 Ignored Other Agent's Input (0.17%)
- 2.6 Reasoning-Action Mismatch (13.98%)

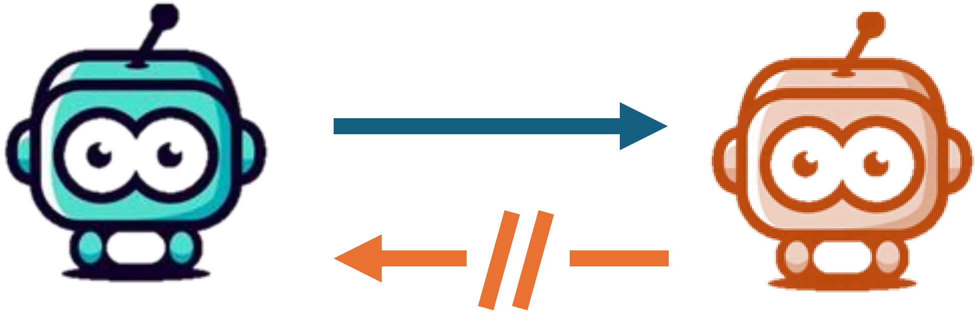
**Task Verification** — 21.30%

- (7.82%) 3.1 Premature Termination
- (6.82%) 3.2 No or Incomplete Verification
- (6.66%) 3.3 Incorrect Verification

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

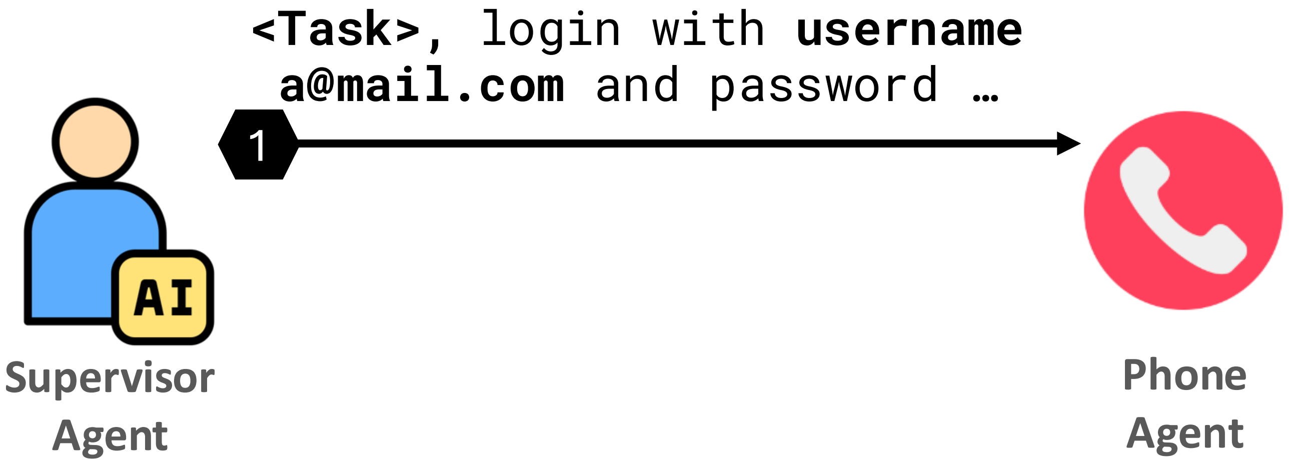
- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent’s Input
- Reasoning-Action Mismatch



# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent’s Input
- Reasoning-Action Mismatch

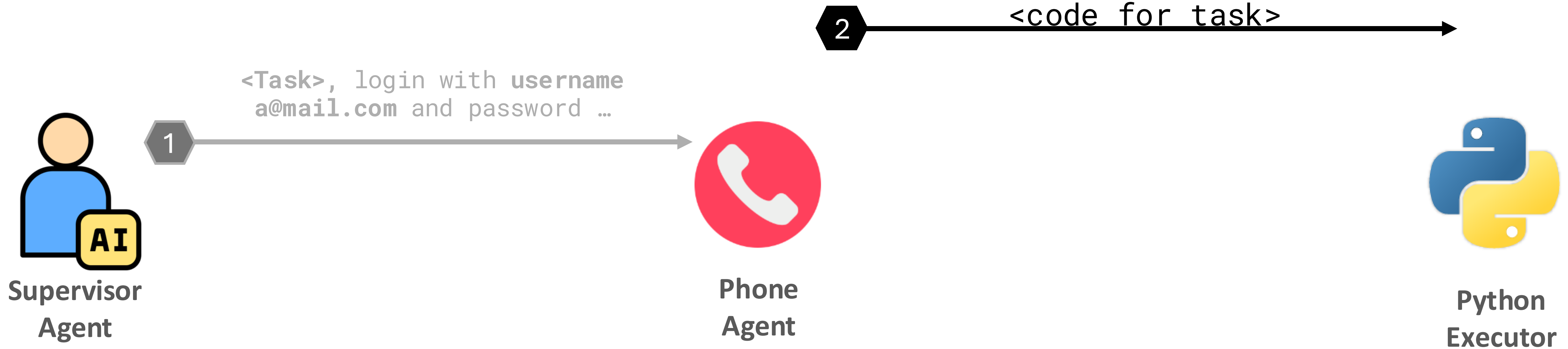


**<Task>**, login with **username a@mail.com** and password …

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

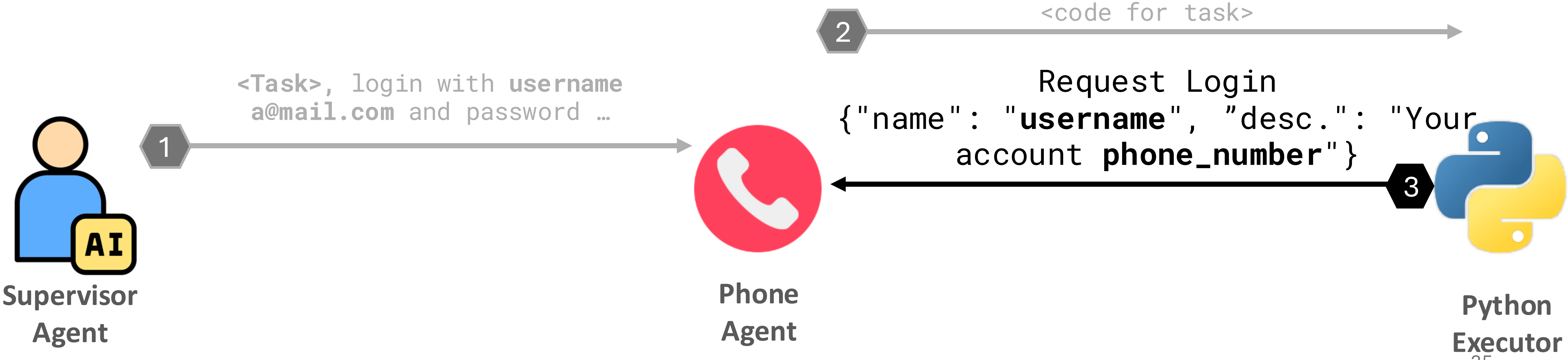
- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent's Input
- Reasoning-Action Mismatch



# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent’s Input
- Reasoning-Action Mismatch



1 `<Task>, login with username a@mail.com and password …`

2 `<code for task>`

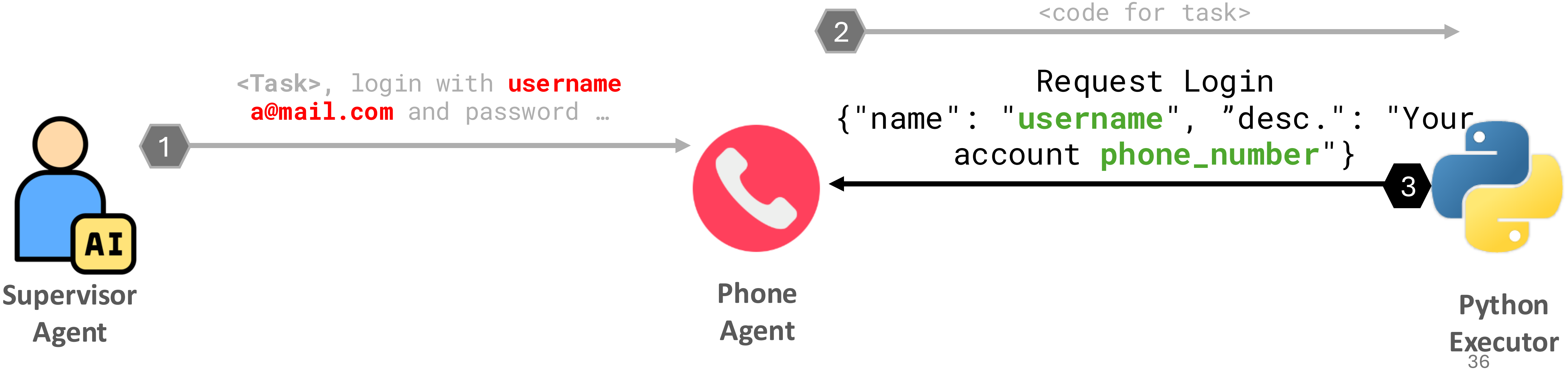
3 Request Login

```
{"name": "username", ”desc.": "Your account phone_number"}
```

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent’s Input
- Reasoning-Action Mismatch

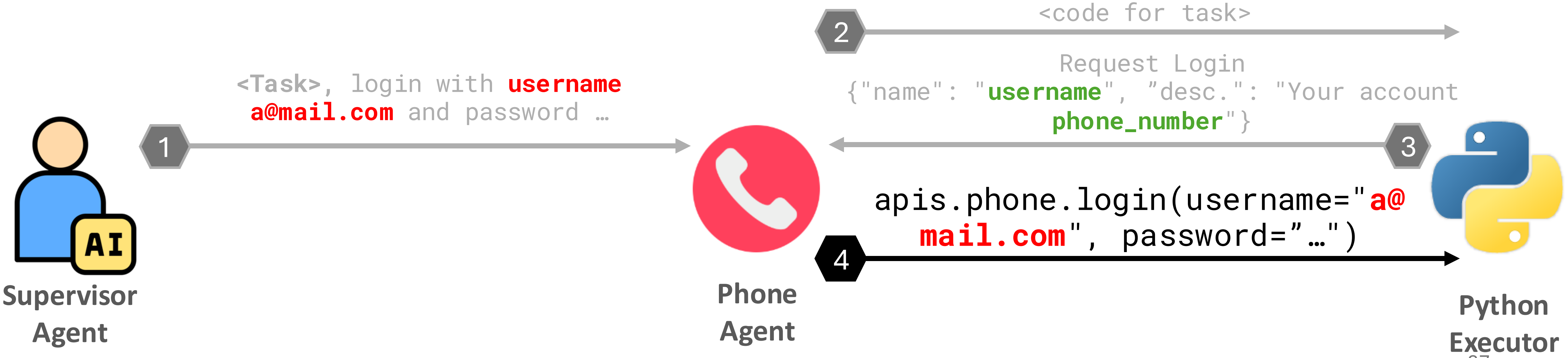


2
<code for task>
1
<Task>, login with username a@mail.com and password …
Request Login
{"name": "username", ”desc.": "Your account phone_number"}
3
AI
Supervisor Agent
Phone Agent
Python Executor

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent's Input
- Reasoning-Action Mismatch



1 <Task>, login with **username a@mail.com** and password …

2 <code for task>

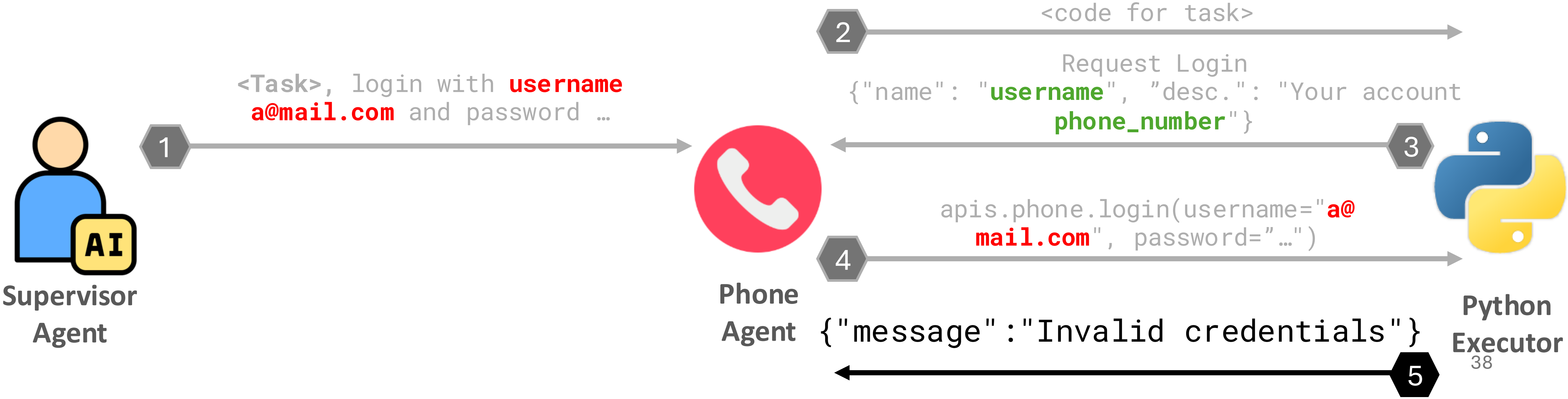
3 Request Login {"name": "**username**", "desc.": "Your account **phone_number**"}

4 `apis.phone.login(username="a@mail.com", password="…")`

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent’s Input
- Reasoning-Action Mismatch



Supervisor Agent

Phone Agent

Python Executor

1 <Task>, login with **username a@mail.com** and password …

2 <code for task>

3 Request Login {"name": "**username**", ”desc.": "Your account **phone_number**"}

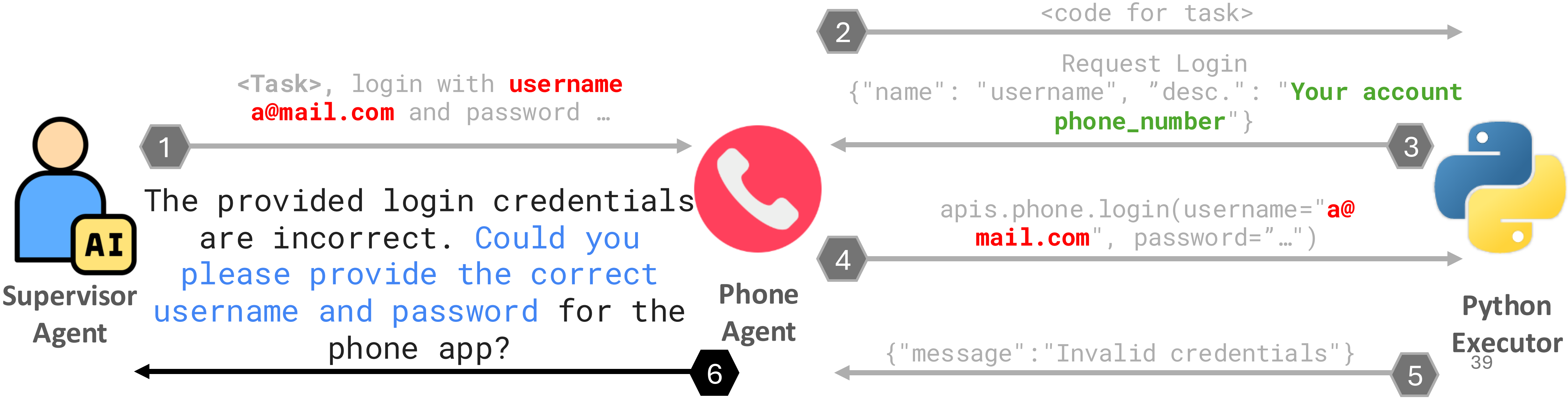
4 apis.phone.login(username="**a@mail.com**", password=”…")

5 {"message":"Invalid credentials"}

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment
- Information Withholding
- Ignored Other Agent's Input
- Reasoning-Action Mismatch



Supervisor Agent

Phone Agent

Python Executor

1 <Task>, login with **username a@mail.com** and password …

2 <code for task>

3 Request Login {"name": "username", "desc.": "**Your account phone_number**"}

4 apis.phone.login(username="**a@mail.com**", password="…")

5 {"message":"Invalid credentials"}

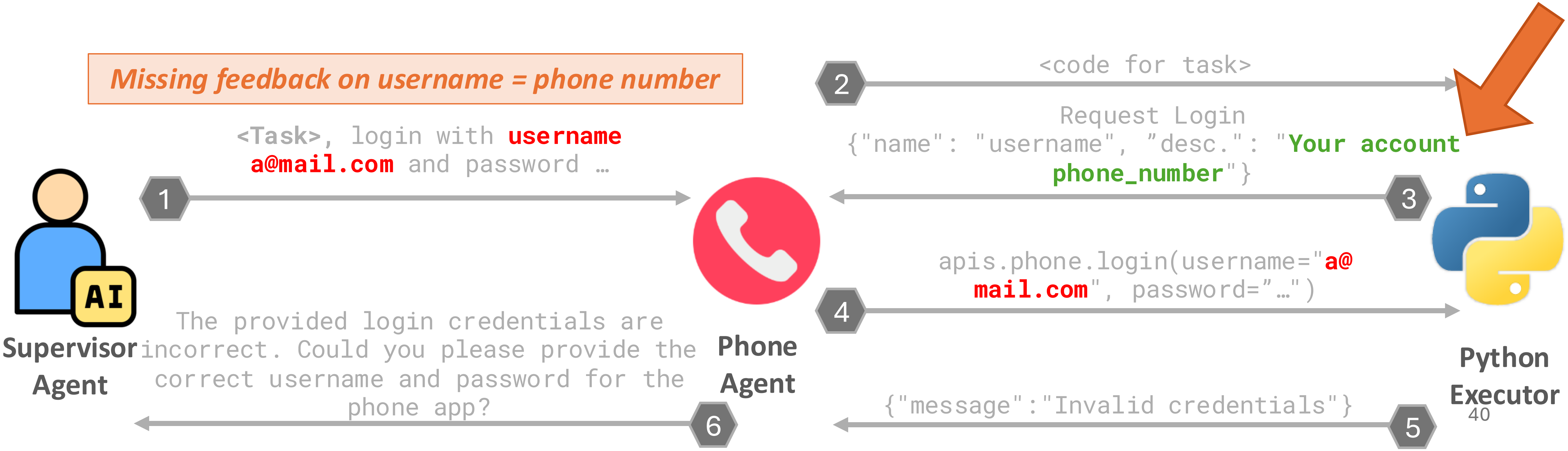
6 The provided login credentials are incorrect. Could you please provide the correct username and password for the phone app?

# FC2: Inter-Agent Misalignment

Failures in **agent coordination** to achieve a common goal.

- Conversation Reset
- Failure to Ask for Clarification
- Task Derailment

- Information Withholding
- Ignored Other Agent’s Input
- Reasoning-Action Mismatch



*Missing feedback on username = phone number*

1 <Task>, login with **username a@mail.com** and password …

2 <code for task>

3 Request Login {"name": "username", ”desc.": "Your account phone_number"}

4 apis.phone.login(username="a@mail.com", password=”…")

5 {"message":"Invalid credentials"}

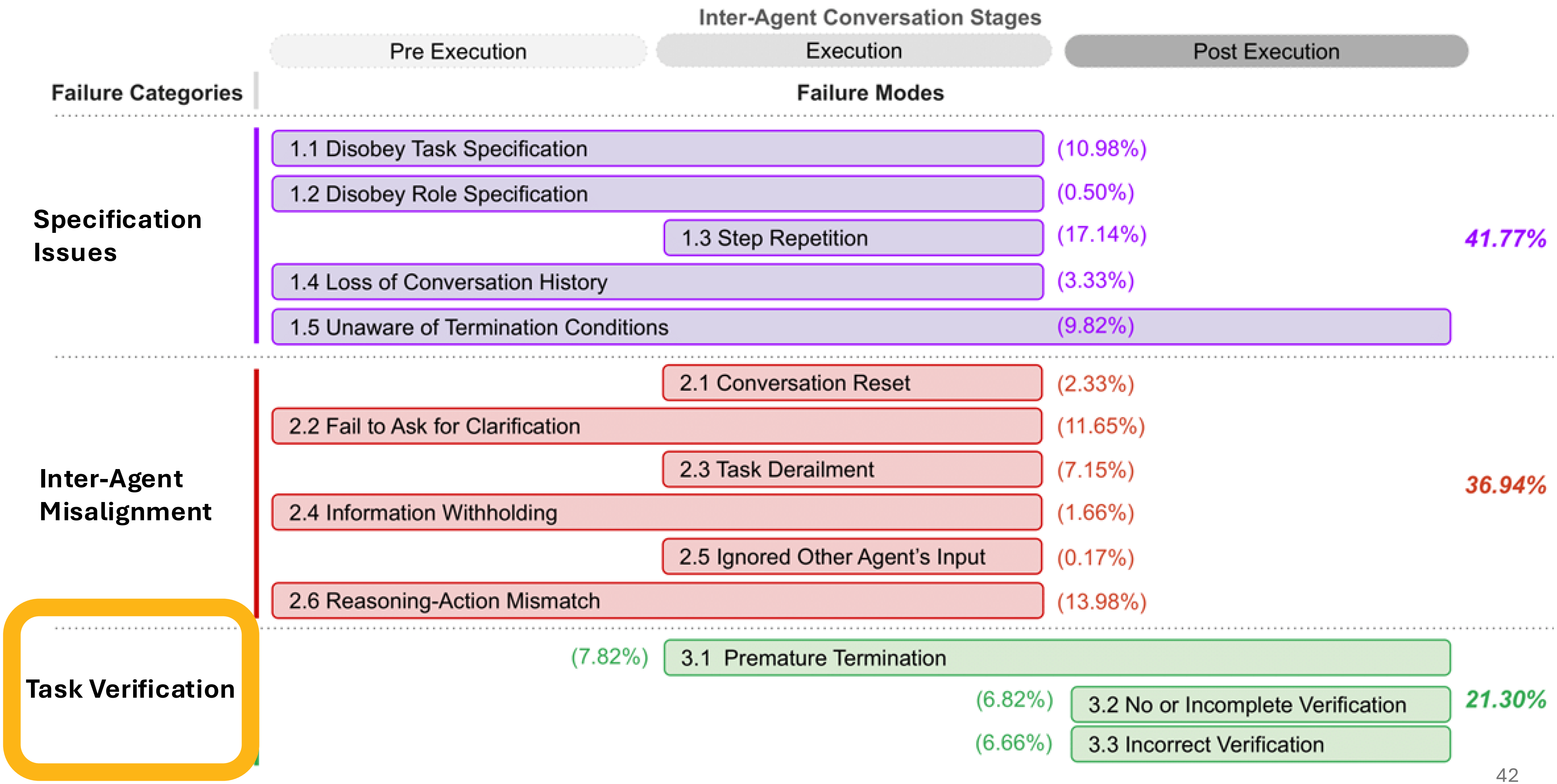
6 The provided login credentials are incorrect. Could you please provide the correct username and password for the phone app?

# FC2: Inter-Agent Misalignment

Solutions focused protocols are often insufficient for FC2 failures!

MAS demands deeper '**social reasoning**' abilities from agents.

# MAST: Multi-Agent Systems Failure Taxonomy



Inter-Agent Conversation Stages

Pre Execution

Execution

Post Execution

Failure Categories

Failure Modes

Specification Issues

1.1 Disobey Task Specification (10.98%)

1.2 Disobey Role Specification (0.50%)

1.3 Step Repetition (17.14%)

1.4 Loss of Conversation History (3.33%)

1.5 Unaware of Termination Conditions (9.82%)

*41.77%*

Inter-Agent Misalignment

2.1 Conversation Reset (2.33%)

2.2 Fail to Ask for Clarification (11.65%)

2.3 Task Derailment (7.15%)

2.4 Information Withholding (1.66%)

2.5 Ignored Other Agent's Input (0.17%)

2.6 Reasoning-Action Mismatch (13.98%)

*36.94%*

Task Verification

(7.82%) 3.1 Premature Termination

(6.82%) 3.2 No or Incomplete Verification

(6.66%) 3.3 Incorrect Verification

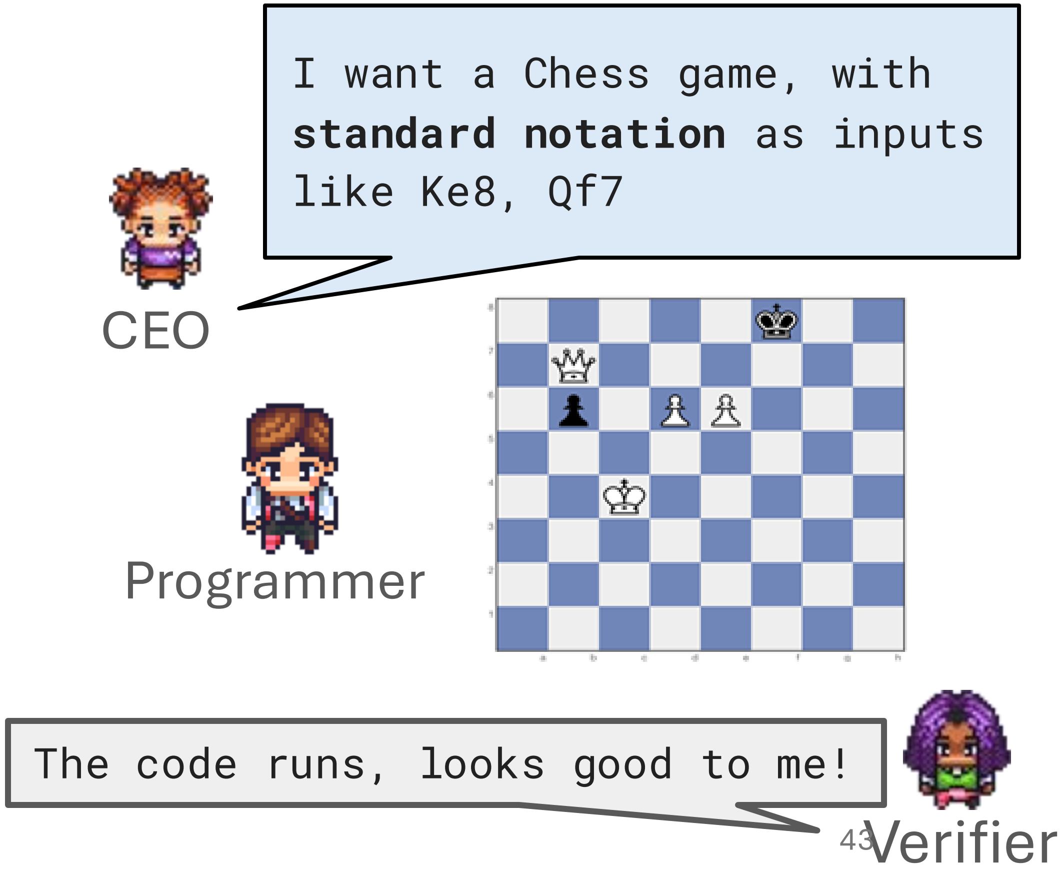
*21.30%*

# FC3: Task Verification

Failures related to **quality control** of the output

1. Premature Termination
2. No or Incomplete Verification
3. Incorrect Verification

Inputs do not follow standard Chess notation & pawns can move backwards



I want a Chess game, with **standard notation** as inputs like Ke8, Qf7

The code runs, looks good to me!

# FC3: Task Verification

Failures related to **quality control** of the output

1. Premature Termination
2. No or Incomplete Verification
3. Incorrect Verification

Verifier

Compile

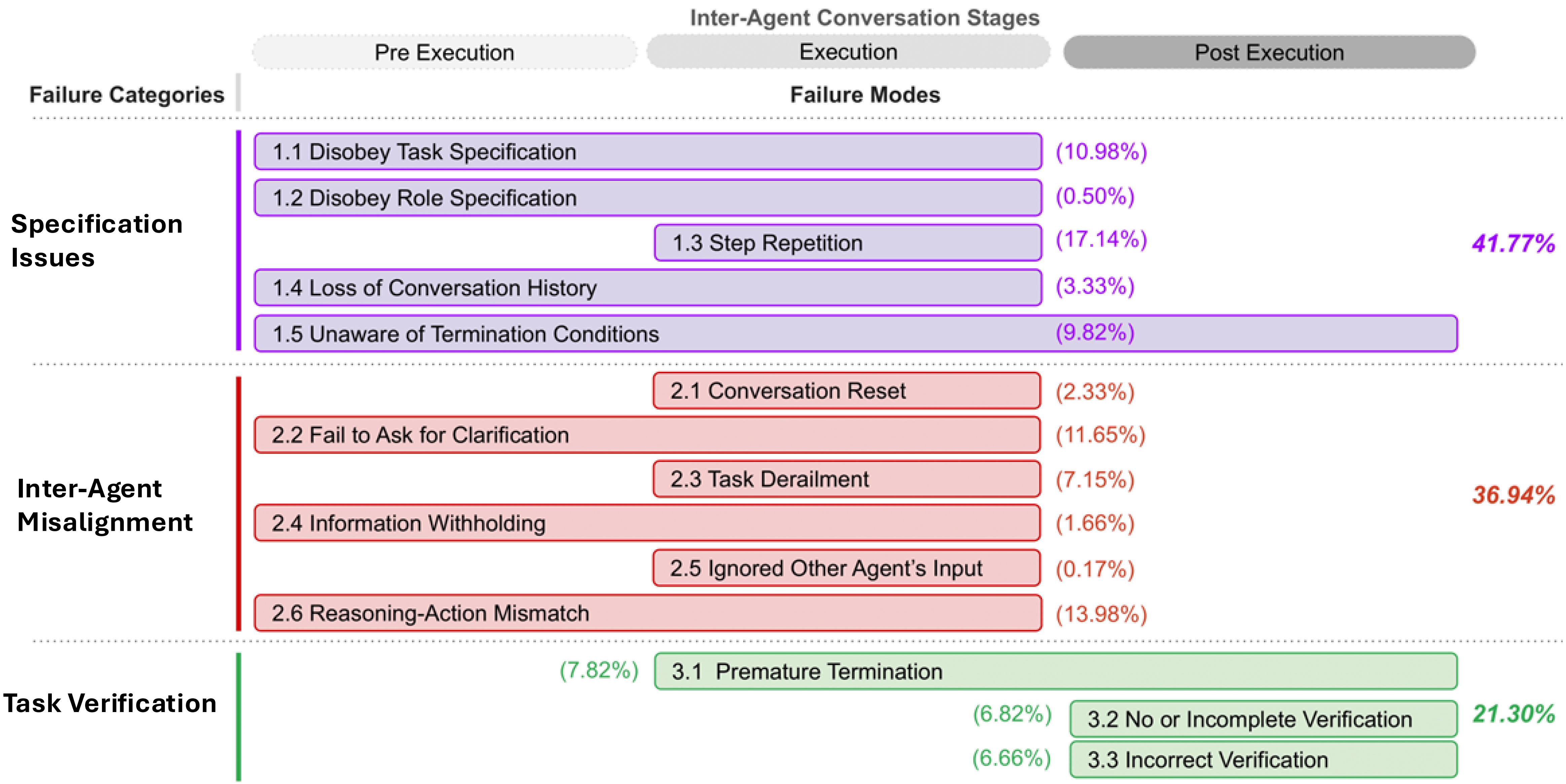
# TODO

Existing MAS

# FC3: Task Verification

Unit Testing & Multi-Level Verification is Needed!

# MAST: Multi-Agent Systems Failure Taxonomy



Inter-Agent Conversation Stages

Pre Execution

Execution

Post Execution

Failure Categories

Failure Modes

**Specification Issues**

1.1 Disobey Task Specification (10.98%)

1.2 Disobey Role Specification (0.50%)

1.3 Step Repetition (17.14%)

1.4 Loss of Conversation History (3.33%)

1.5 Unaware of Termination Conditions (9.82%)

*41.77%*

**Inter-Agent Misalignment**

2.1 Conversation Reset (2.33%)

2.2 Fail to Ask for Clarification (11.65%)

2.3 Task Derailment (7.15%)

2.4 Information Withholding (1.66%)

2.5 Ignored Other Agent's Input (0.17%)

2.6 Reasoning-Action Mismatch (13.98%)

*36.94%*

**Task Verification**

(7.82%) 3.1 Premature Termination

(6.82%) 3.2 No or Incomplete Verification

(6.66%) 3.3 Incorrect Verification

*21.30%*

# Outline

- Motivation
- Prior work
- MAST & MAD overview
- Study methodology
- Study finding & insights
- Practical use of MAST
- Q&A

# Outline

- Motivation
- Prior work
- MAST & MAD overview
- Study methodology
- Study finding & insights
- Practical use of MAST
- Q&A

# How to apply **MAST**?

# Why does MAST matter?

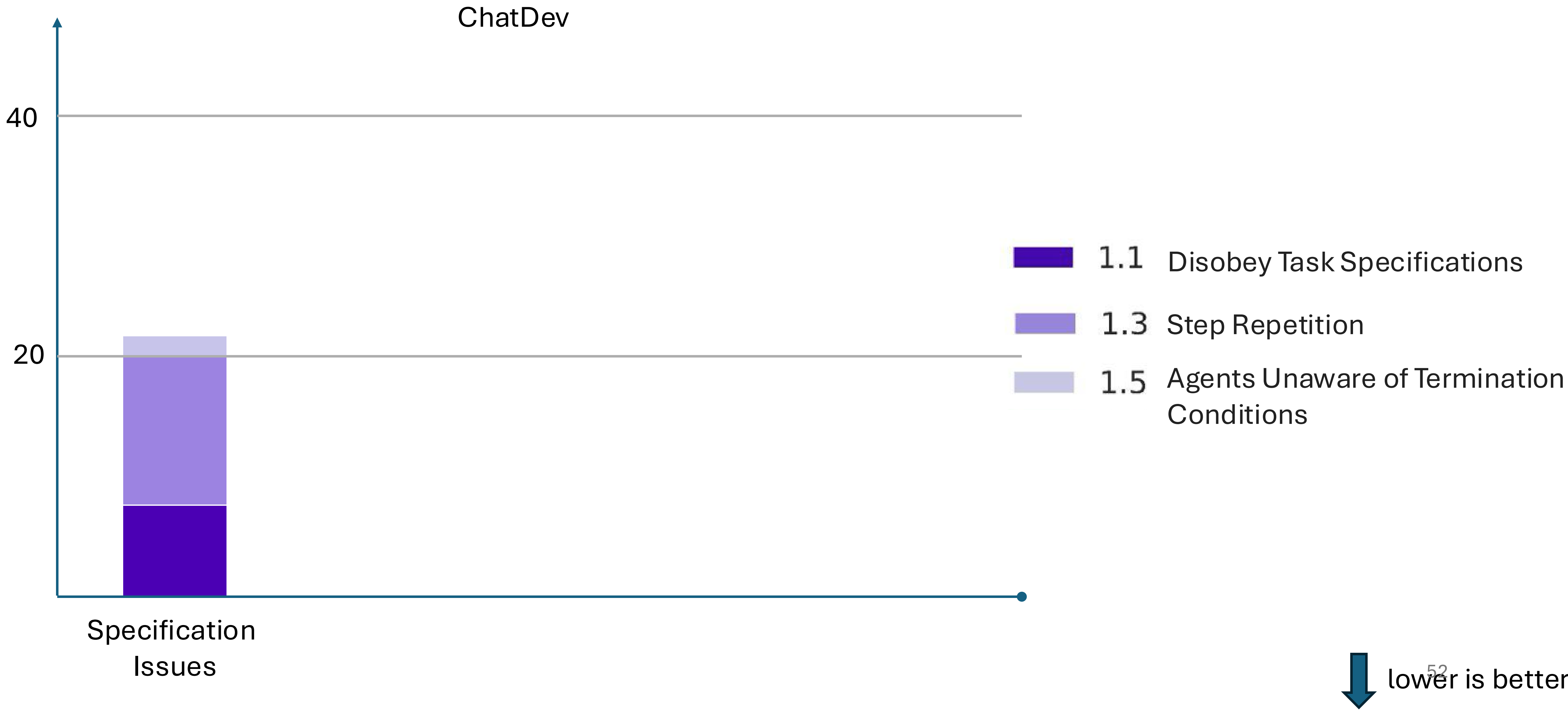
1. **Roadmap** for future research

2. MAST as a practical **tool**

# Towards Better MAS:

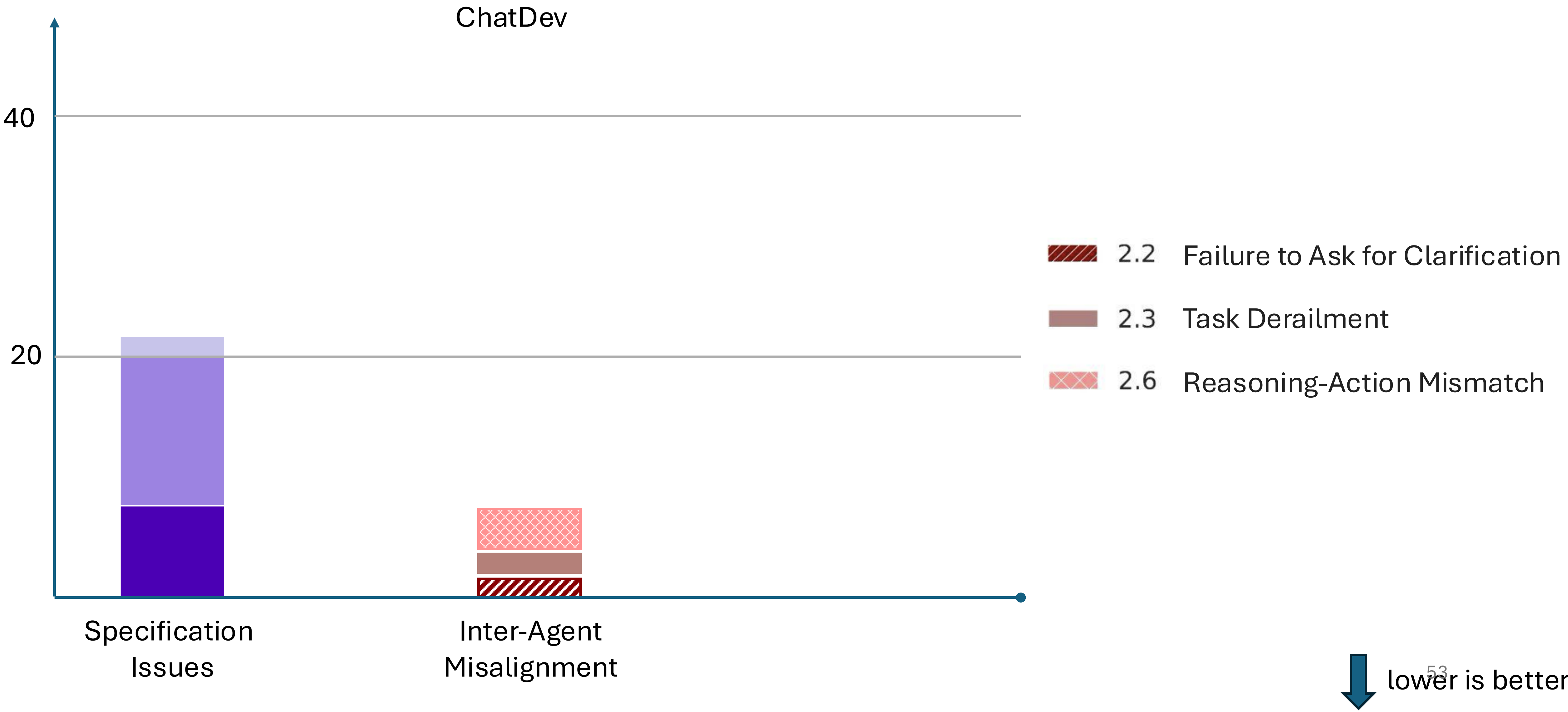
MAST as a practical **tool**

# 1. Understanding Failures Profile



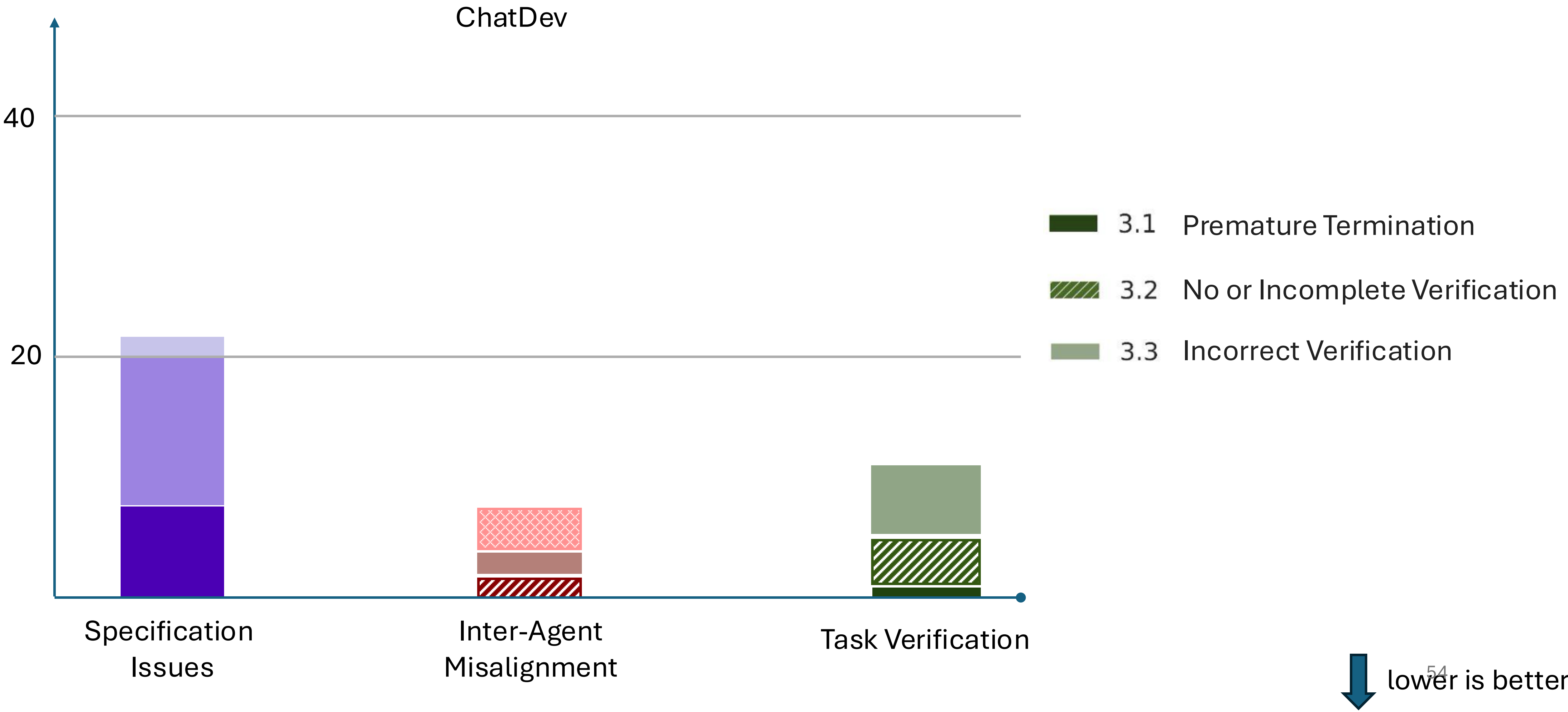
lower is better

# 1. Understanding Failures Profile



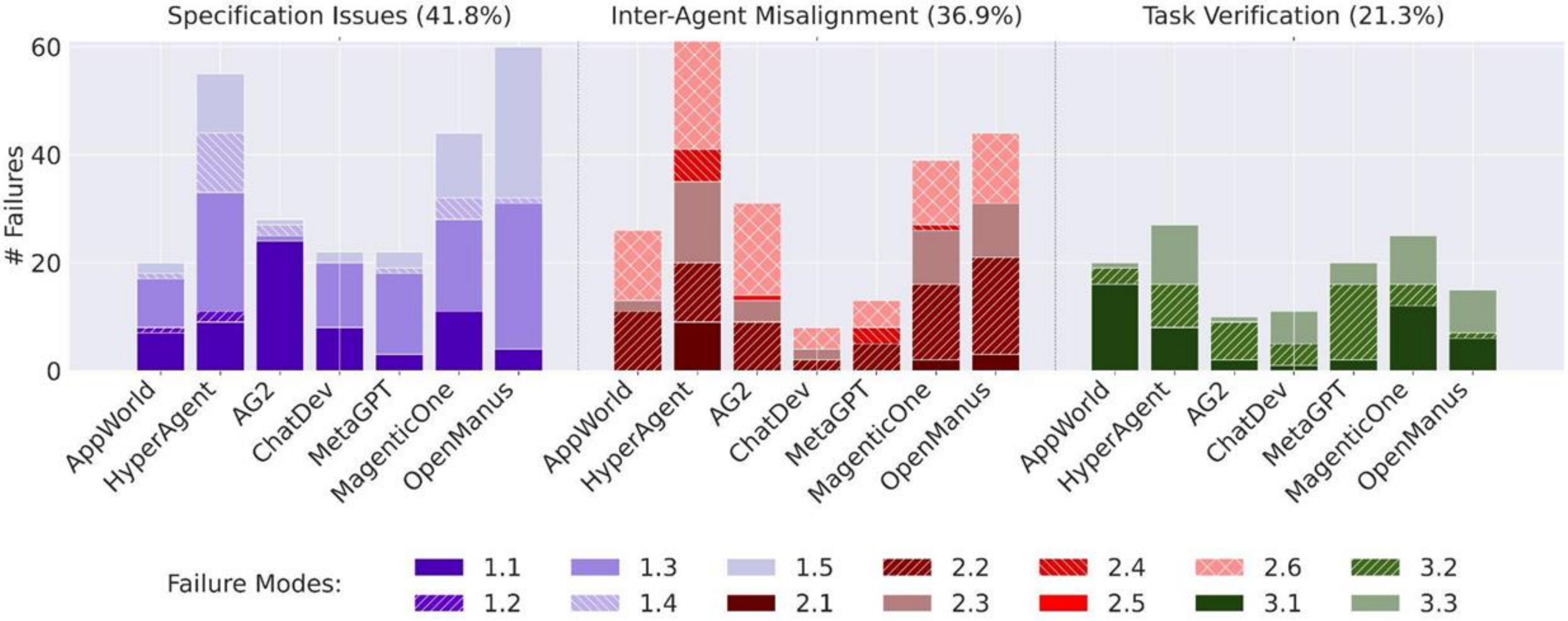
⬇ lower is better

# 1. Understanding Failures Profile



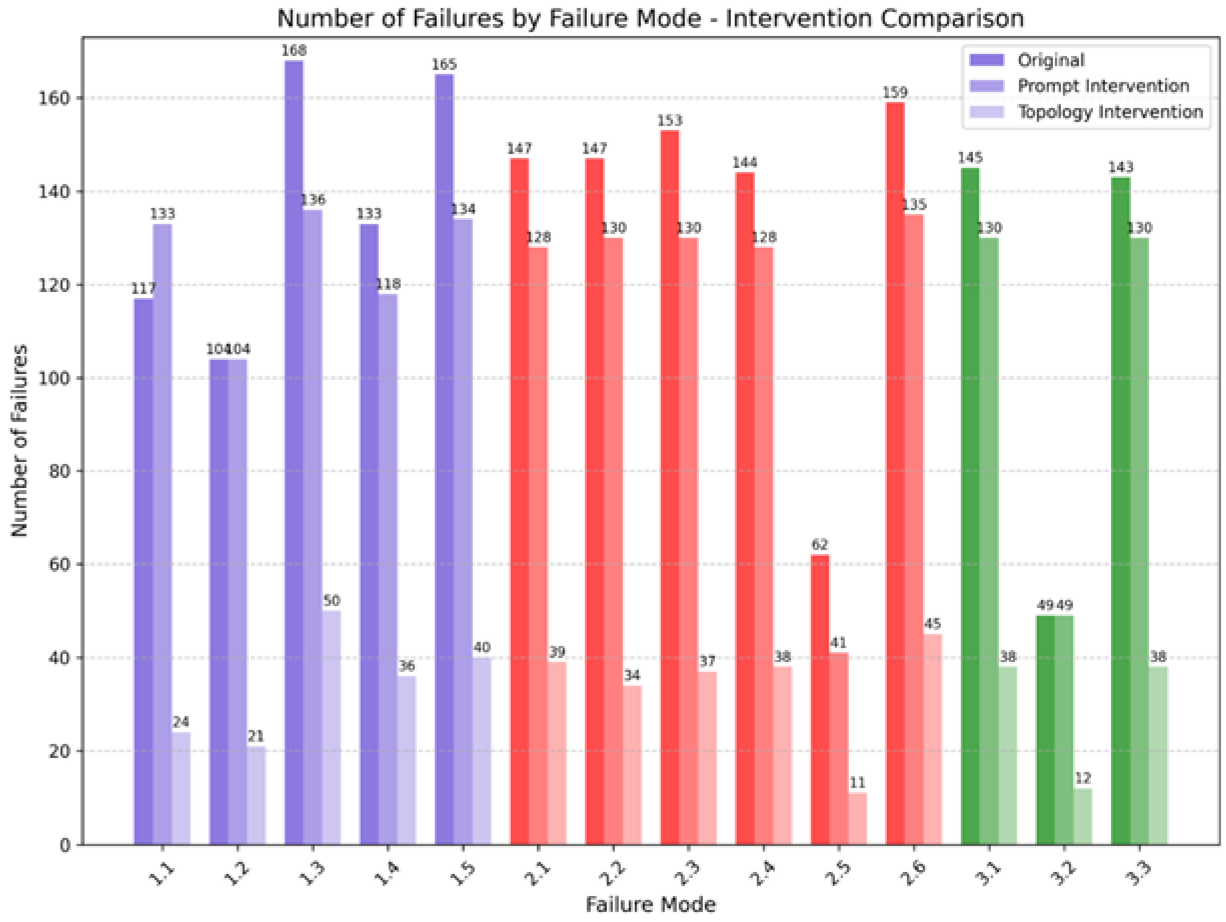
lower is better

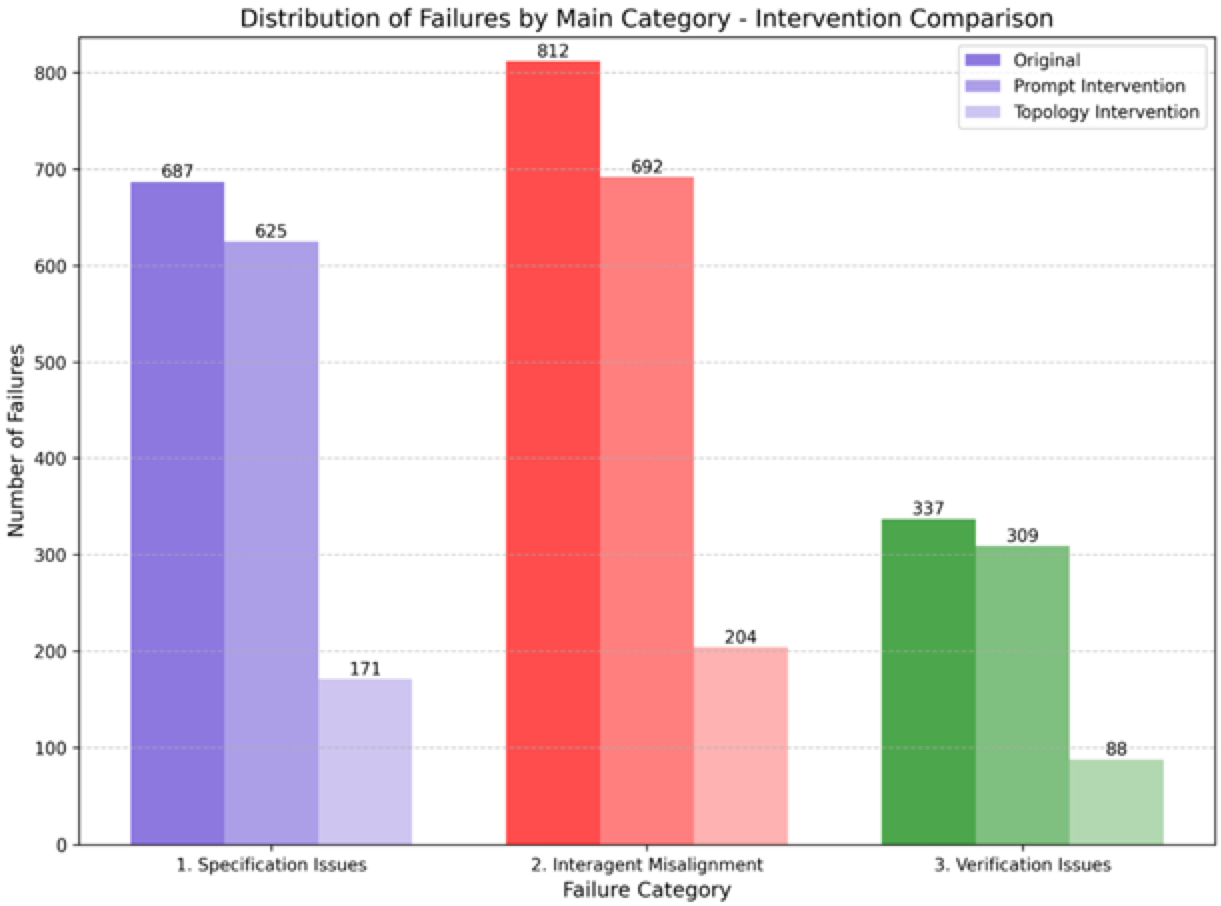
# 1. Understanding Failures Profile



lower is better

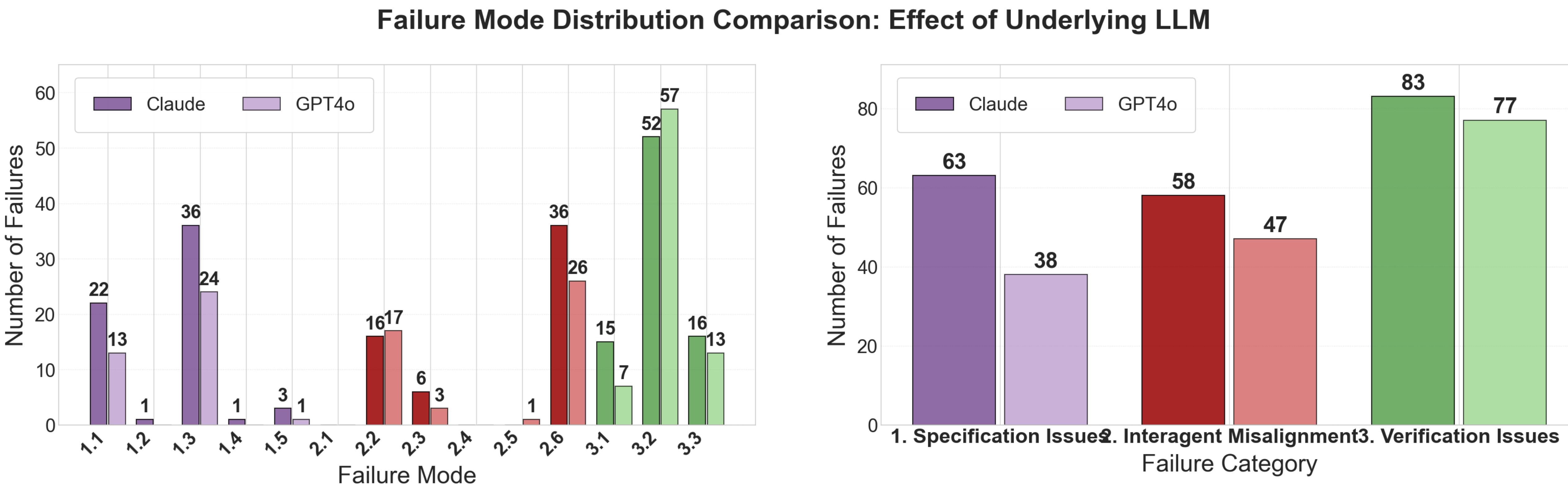
# 2. Guide MAS Debugging and Development





⬇ lower is better

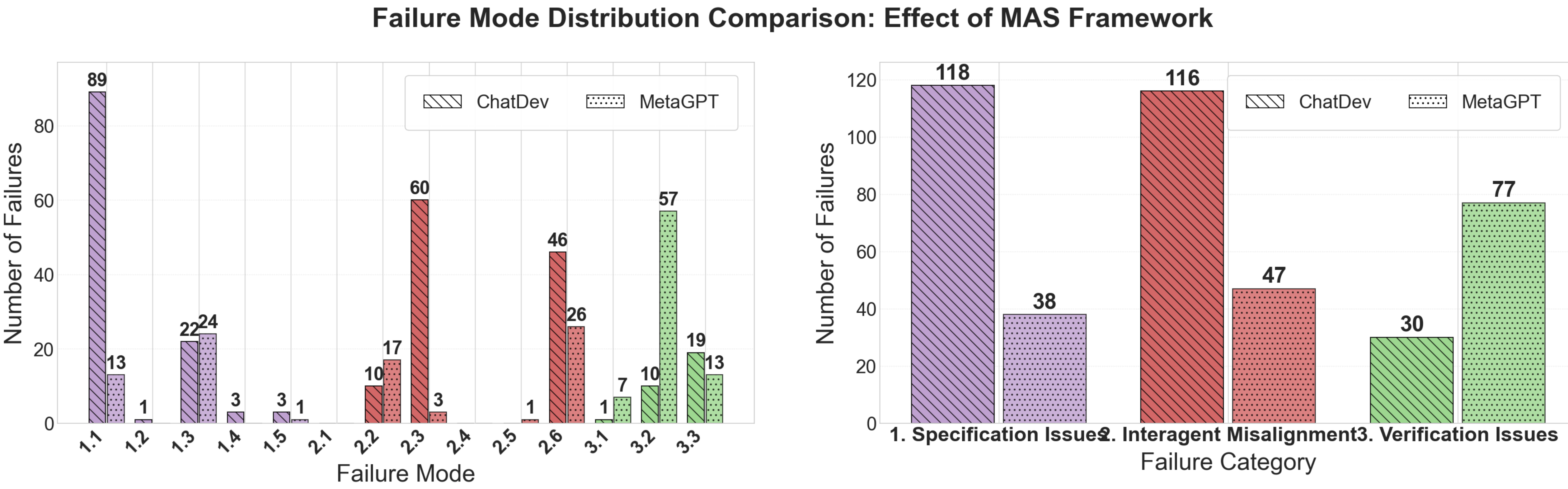
# 3. Compare Effectiveness of **LLMs**



Failure Mode Distribution Comparison: Effect of Underlying LLM

lower is better

# 3. Compare Effectiveness of **MAS Architecture**



↓ lower is better

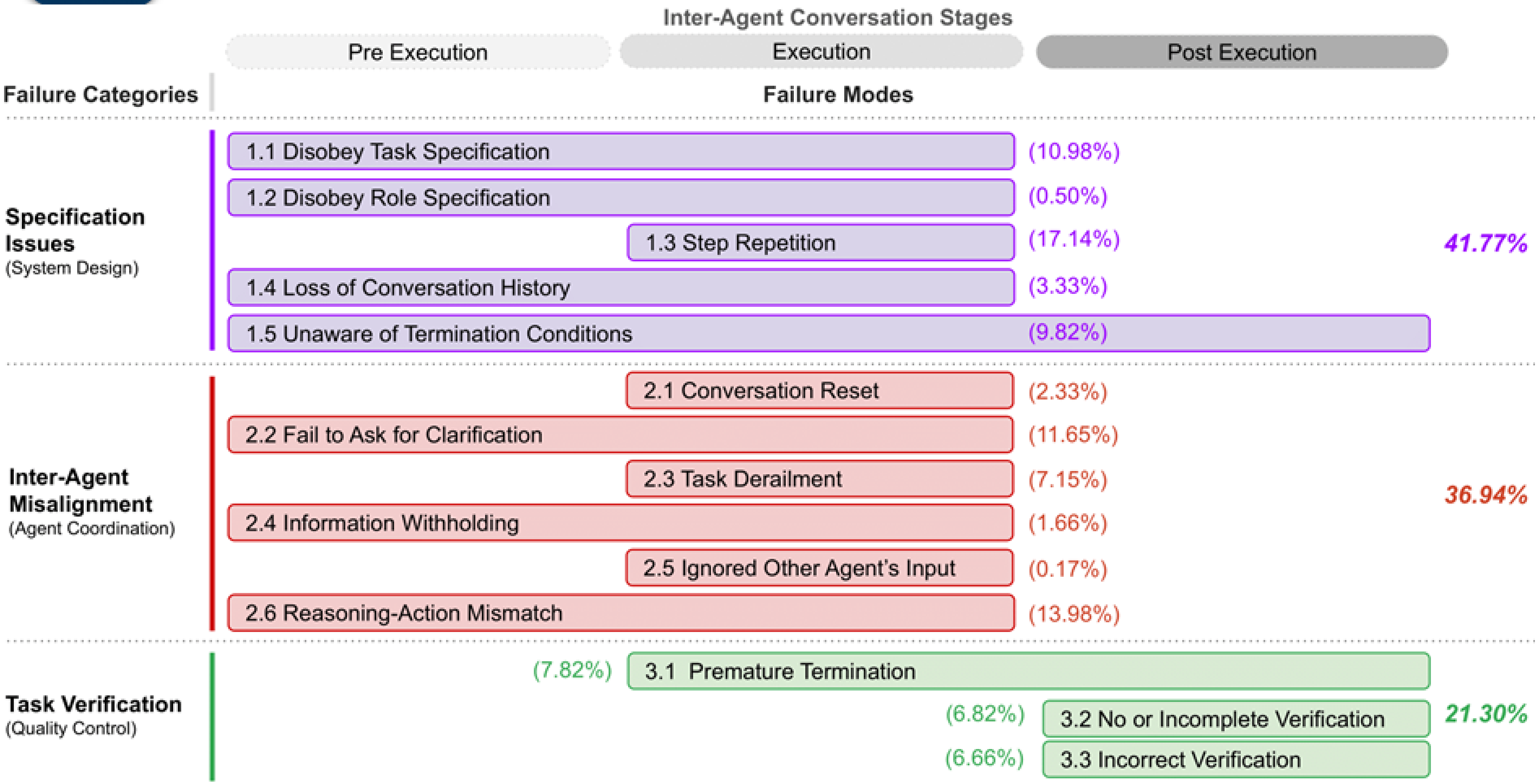
# Conclusion

- Multi-Agent Systems hold promise but face significant challenges.
- Multi-Agent Systems is becoming essential as we move towards increasing **automation in data work**
- MAST, the **first** multi-agent failure taxonomy.
- MAST as a **practical tool** for developers and a roadmap for future research to build more reliable and effective MAS.

**Feedback & Collaborations ☺**

**Do you need help using MAST to build & evaluate Agents?**

# Thank You!



Inter-Agent Conversation Stages

Pre Execution

Execution

Post Execution

Failure Categories

Failure Modes

**Specification Issues** (System Design) — 41.77%

- 1.1 Disobey Task Specification (10.98%)
- 1.2 Disobey Role Specification (0.50%)
- 1.3 Step Repetition (17.14%)
- 1.4 Loss of Conversation History (3.33%)
- 1.5 Unaware of Termination Conditions (9.82%)

**Inter-Agent Misalignment** (Agent Coordination) — 36.94%

- 2.1 Conversation Reset (2.33%)
- 2.2 Fail to Ask for Clarification (11.65%)
- 2.3 Task Derailment (7.15%)
- 2.4 Information Withholding (1.66%)
- 2.5 Ignored Other Agent's Input (0.17%)
- 2.6 Reasoning-Action Mismatch (13.98%)

**Task Verification** (Quality Control) — 21.30%

- (7.82%) 3.1 Premature Termination
- (6.82%) 3.2 No or Incomplete Verification
- (6.66%) 3.3 Incorrect Verification

code

paper

@melissapan

melissapan@berkeley.edu

**Do you need help using MAST to build & evaluate Agents?**

github.com/multi-agent-systems-failure-taxonomy/MAST