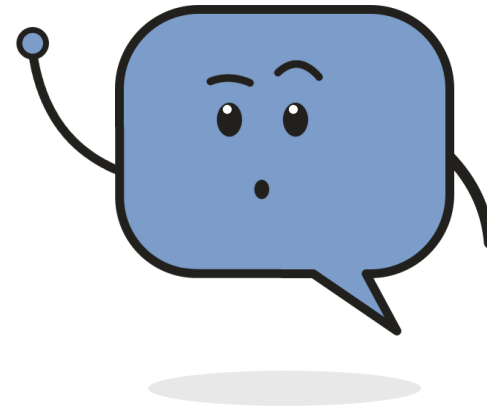


Can We Make AI Agents Coordinate Safely?

Benedikt Bollig

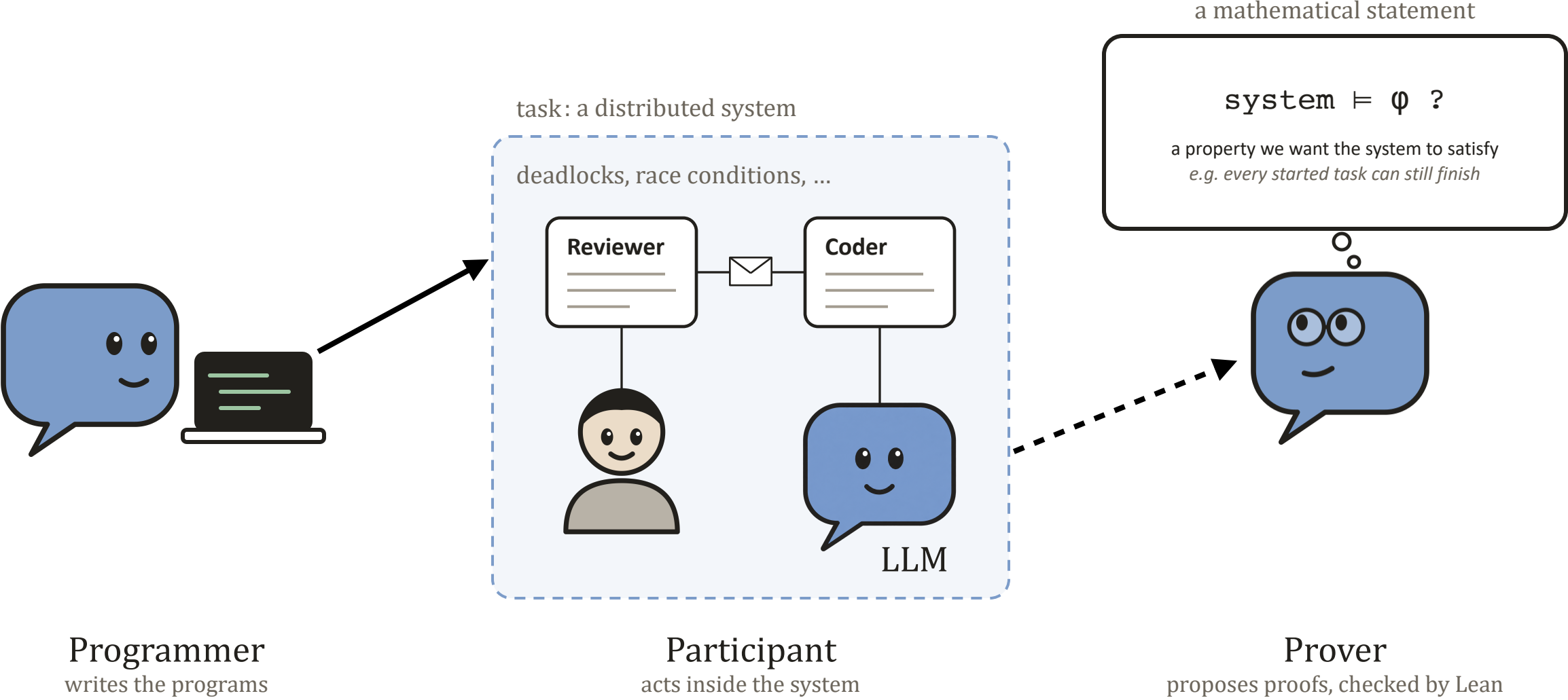
CNRS, LMF, Université Paris-Saclay

Formal Methods



Thought Experiment #1
Paris School of AI
October 15, 2026

Agentic System: where LLMs enter the picture



AI can now solve remarkably hard problems

Navier-Stokes (Millennium Prize Problem): Can smooth solutions blow up in finite time?

10,000

AI agents working together

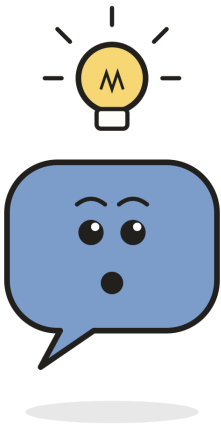
88 h

to a proposed blow-up proof

17 h

to formalize and check it in Lean

As reported by OpenAI (September 2026). The Clay Institute calls the problem apparently settled while evaluation continues.



MATHEMATICS

OPENAI

OpenAI Research Catalog

372 result families in 722 manuscripts

October 6, 2026

An overview of the [manuscript collection](#). Results are grouped by subject; entry numbers follow the catalog order and do not indicate a ranking. PDF links follow each summary.

Contents

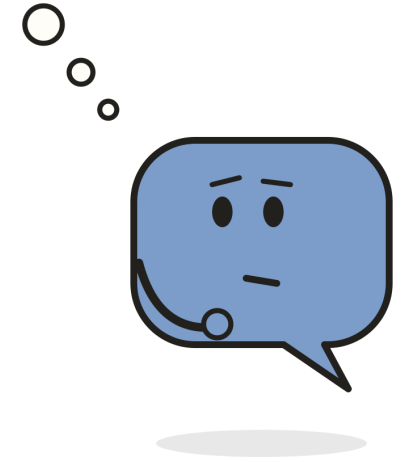
Number theory	2
Algebraic and complex geometry	5
Real and complex analysis	9
Convex and metric geometry	10
Theoretical computer science	12

So can AI simply prove distributed systems correct?

A distributed system and a property are one instance.
Strong models often find the invariants they need.

But how hard is it?

And do we need anyone to understand the proof?



Natural language proof

For every natural number n ,
 $n + 0 = n$.

formalize

Does the formalization say
exactly the same thing?

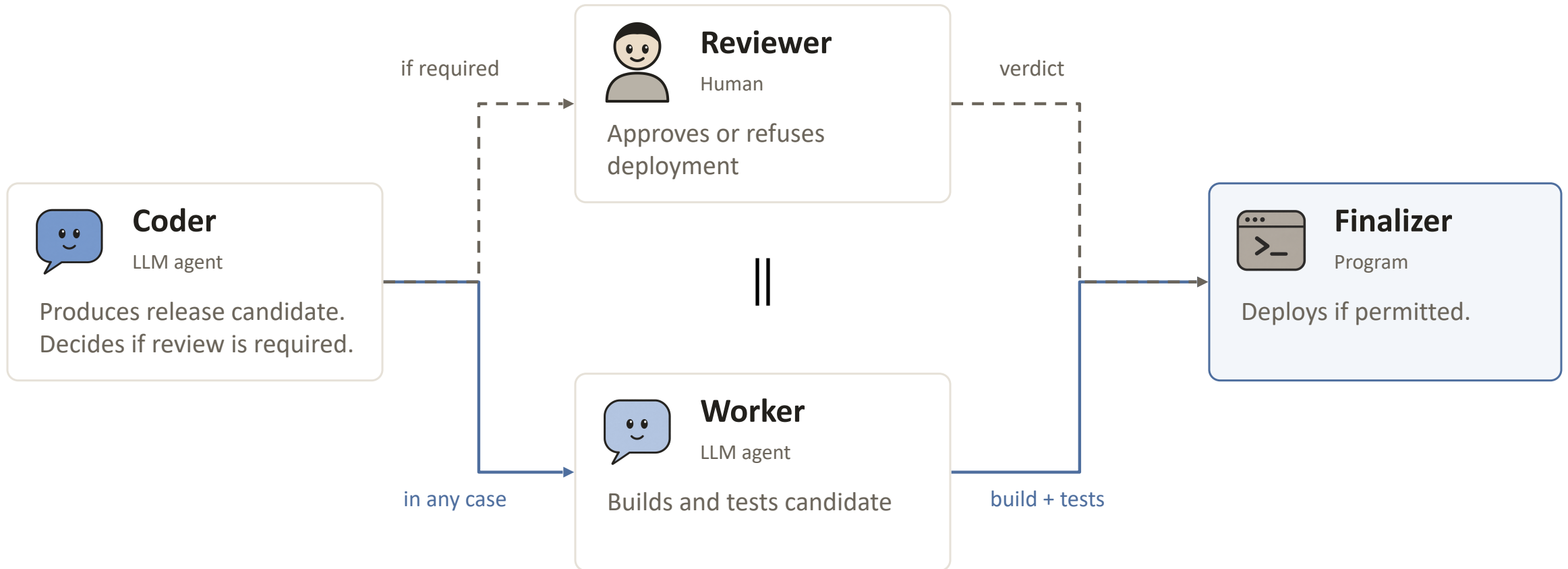
Lean formalization

$\forall n : \mathbb{N}, n + 0 = n$



Alexander Bastounis, Fabian Circelli, Anders C. Hansen: *Navier-Stokes lost in translation: Why Lean verification of AI autoformalisation does not guarantee correct natural language proofs*. 2026. arXiv:2610.08144v1

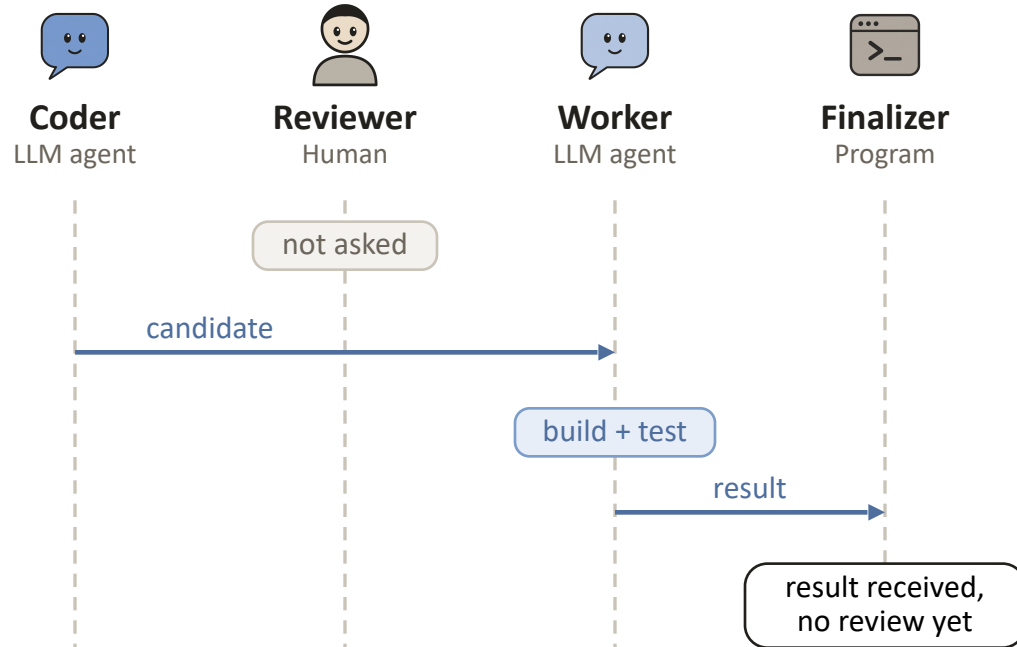
A concrete agentic system $\Rightarrow$ mathematical problem



Two scenarios

Same local observation from the point of view of the Finalizer

No review required

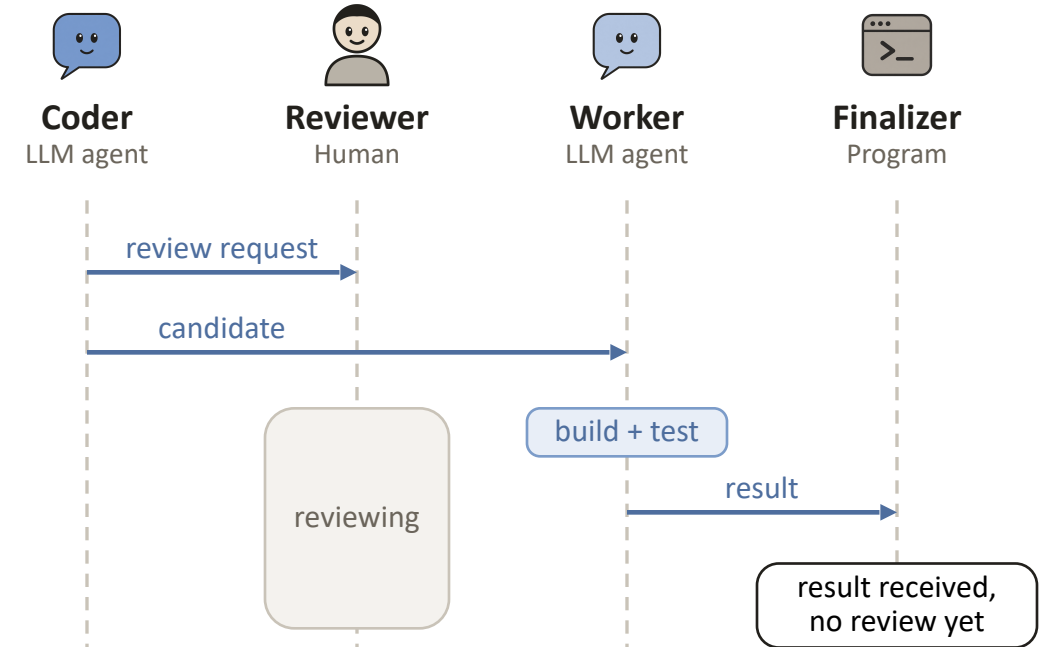


Correct next step

Finalize now

Waiting would prevent completion

Review required



Correct next step

Wait for the review

Finalizing now can deploy without approval

The Finalizer is missing one piece of information.

A plausible & naive implementation

Polling Finalizer

```
wait for first available input
```

```
if review is available:
```

```
    receive review
```

```
    receive result
```

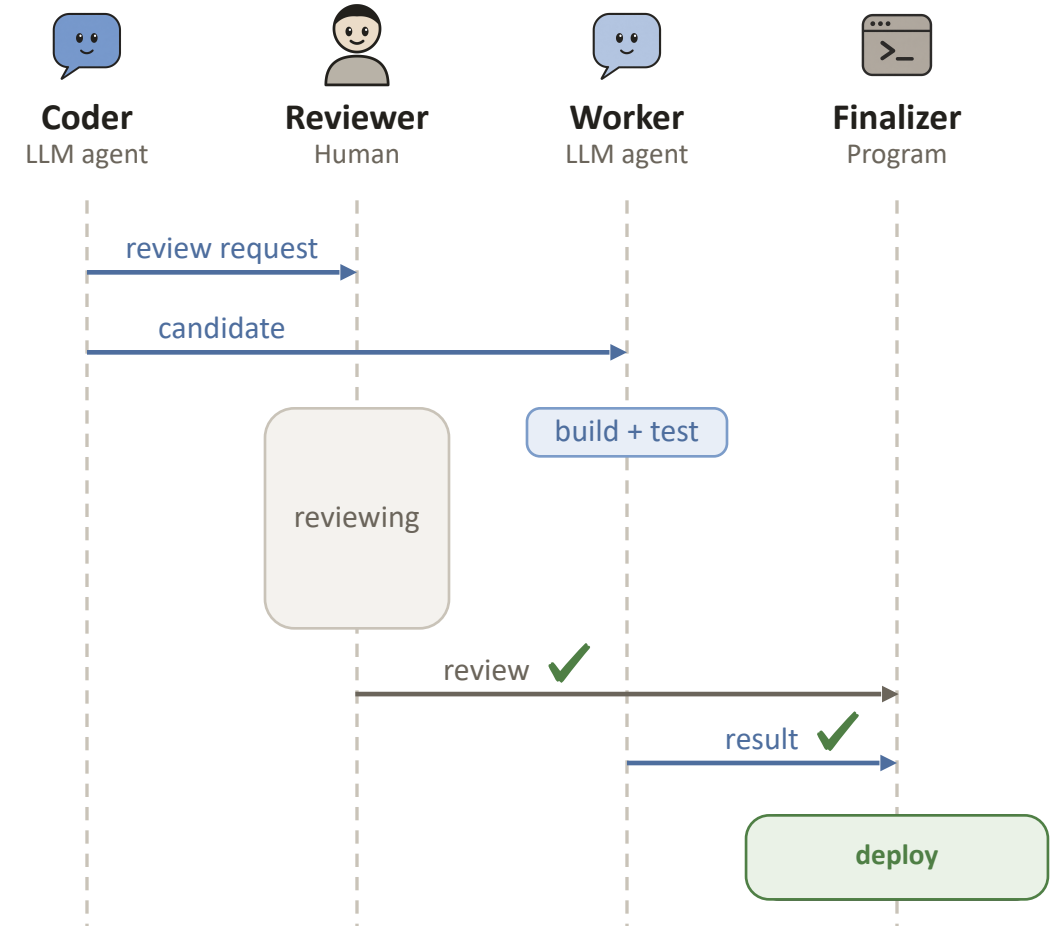
```
    finalize(review, result)
```

```
else if result is available:
```

```
    receive result
```

```
    finalize(no_review, result)
```

Review required



A plausible & naive implementation

Polling Finalizer

```
wait for first available input
```

```
if review is available:
```

```
    receive review
```

```
    receive result
```

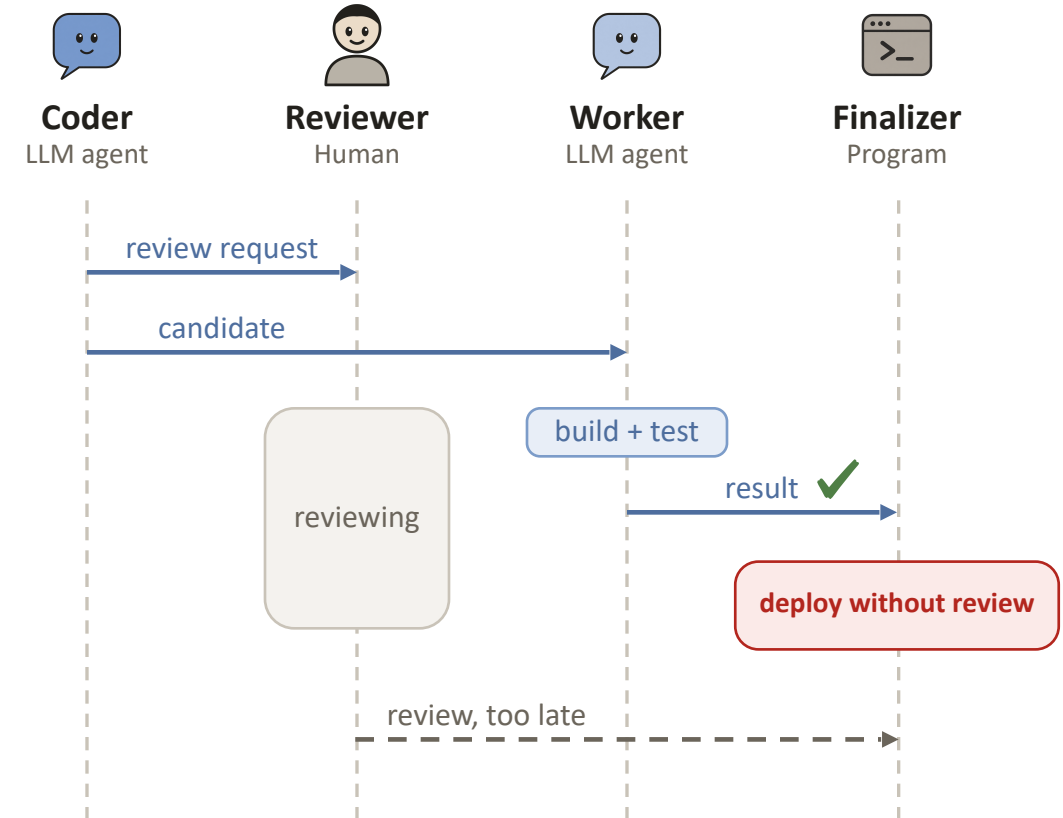
```
    finalize(review, result)
```

```
else if result is available:
```

```
    receive result
```

```
    finalize(no_review, result)
```

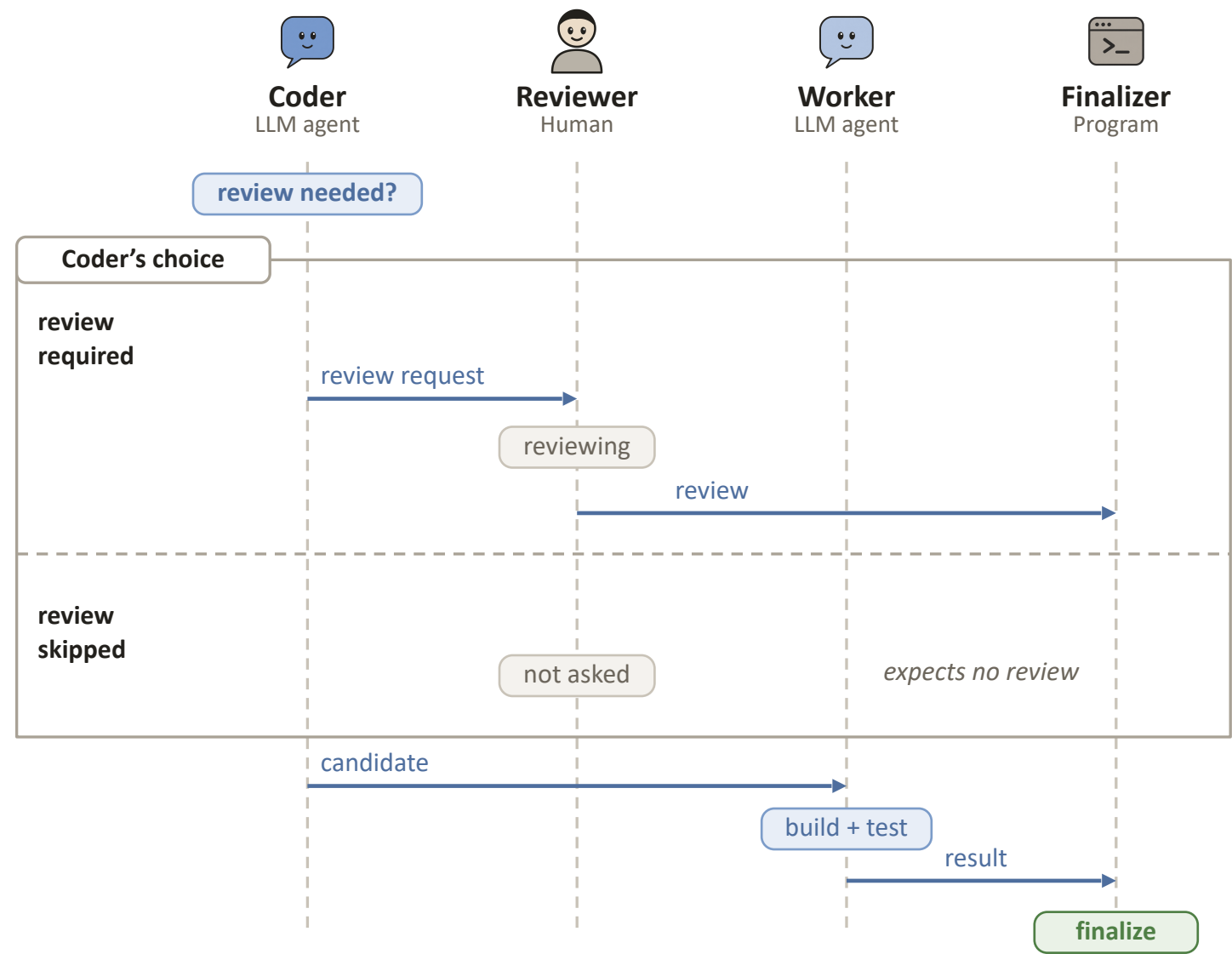
Review required



How does the Finalizer learn the Coder's decision?

Choreographies & formal methods

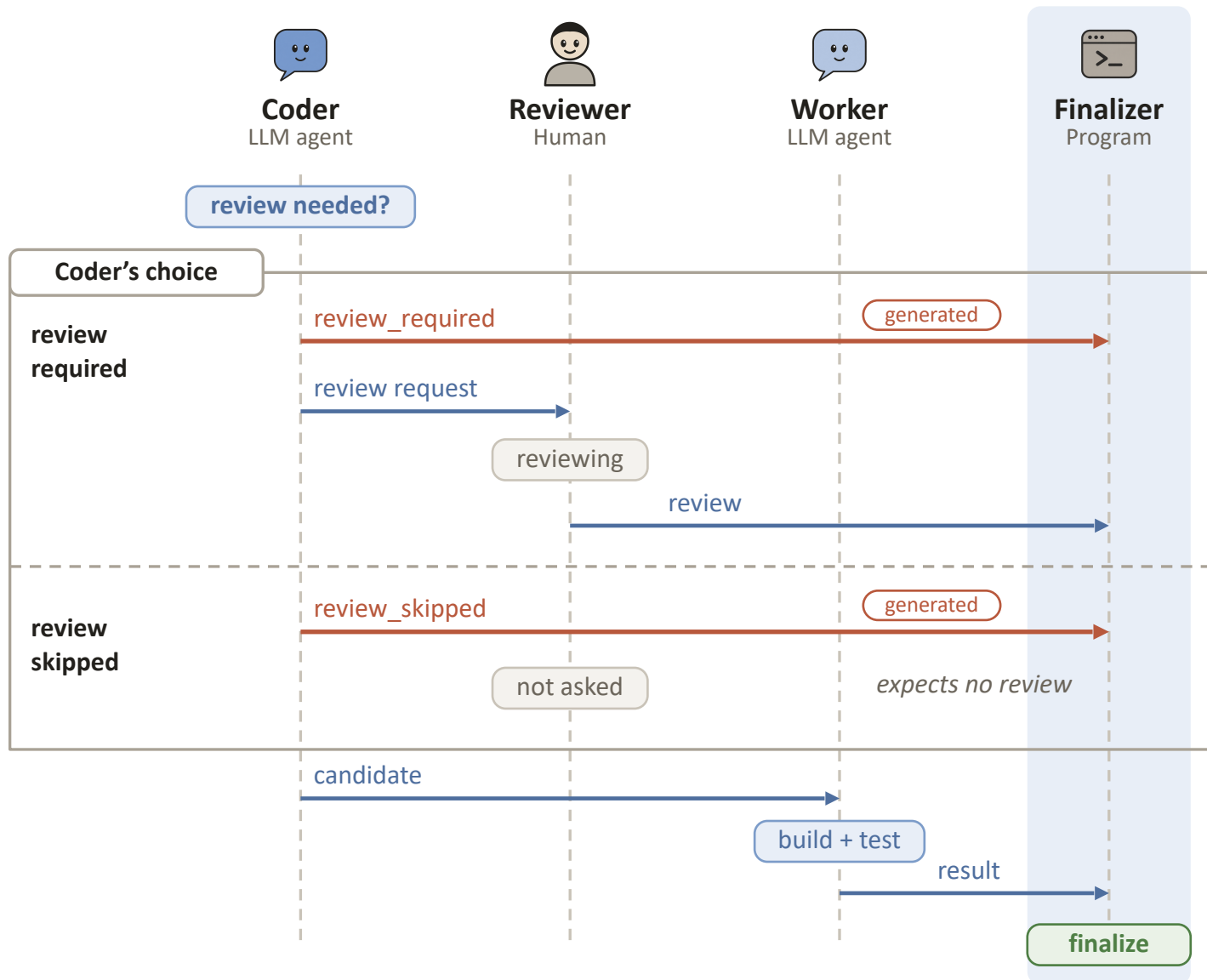
Writing each participant separately.
First describe the whole conversation in one global view.



Choreographies & formal methods

Writing each participant separately.

First describe the whole conversation in one global view.



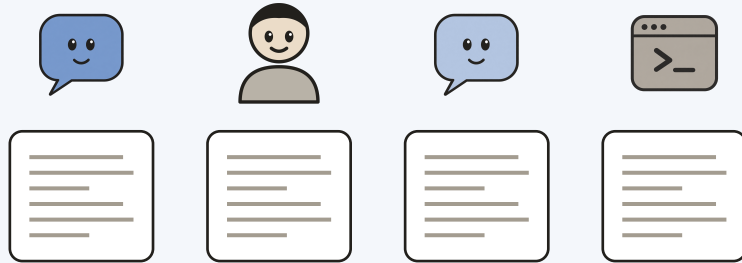
Finalizer needs to know Coder's decision.
Communicated through control messages.

Projected Finalizer

```
decision = receive from Coder
if decision == review_required:
    review = receive from Reviewer
    result = receive from Worker
    finalize(review, result)
else: # review_skipped
    result = receive from Worker
    finalize(no_review, result)
```

The mathematical problem

distributed system



?

deadlock-freedom

"From any point in any run,
there is still a way to
complete every task that
was started."

It guarantees

"Can finish", not "will finish"

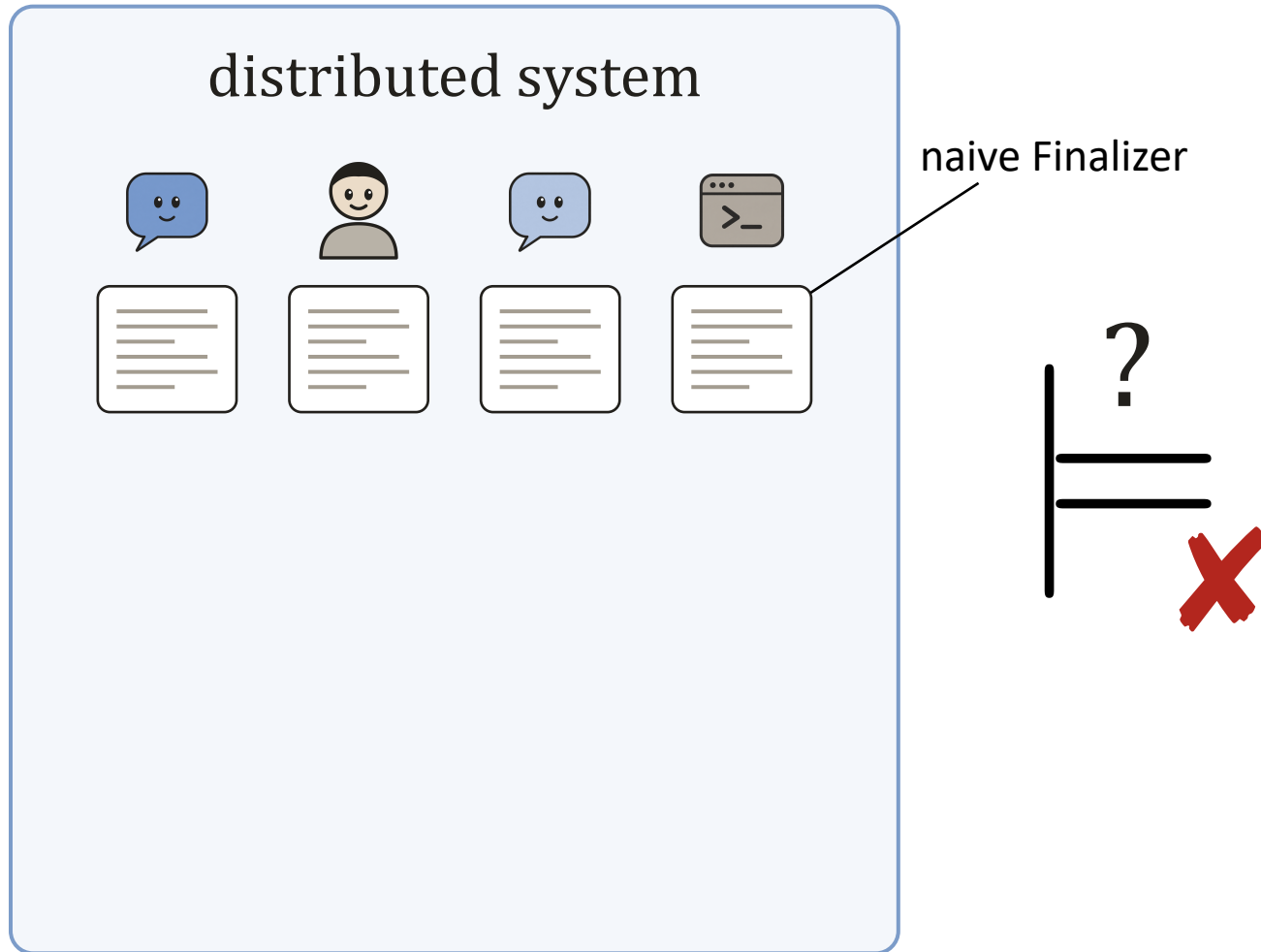
It does not guarantee

that a human eventually answers

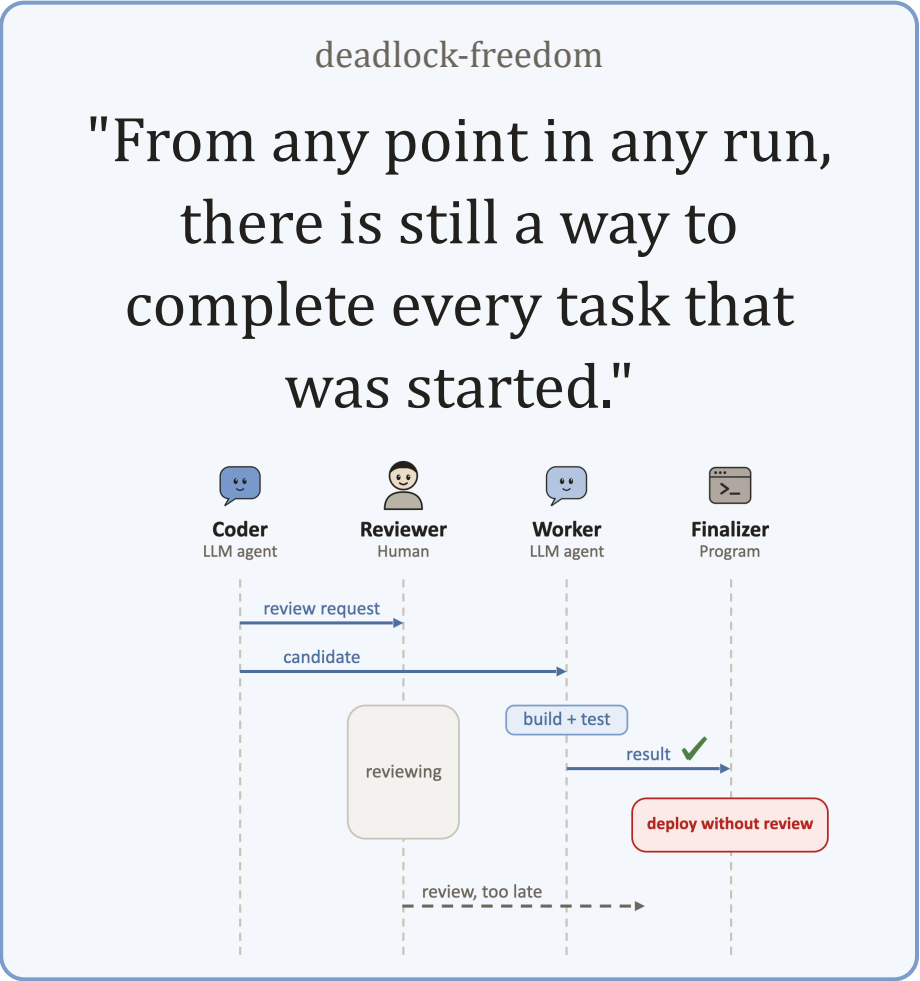
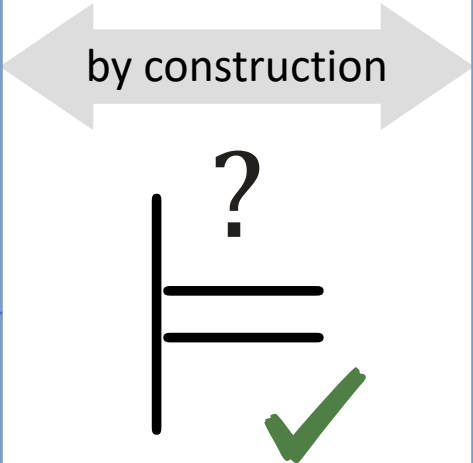
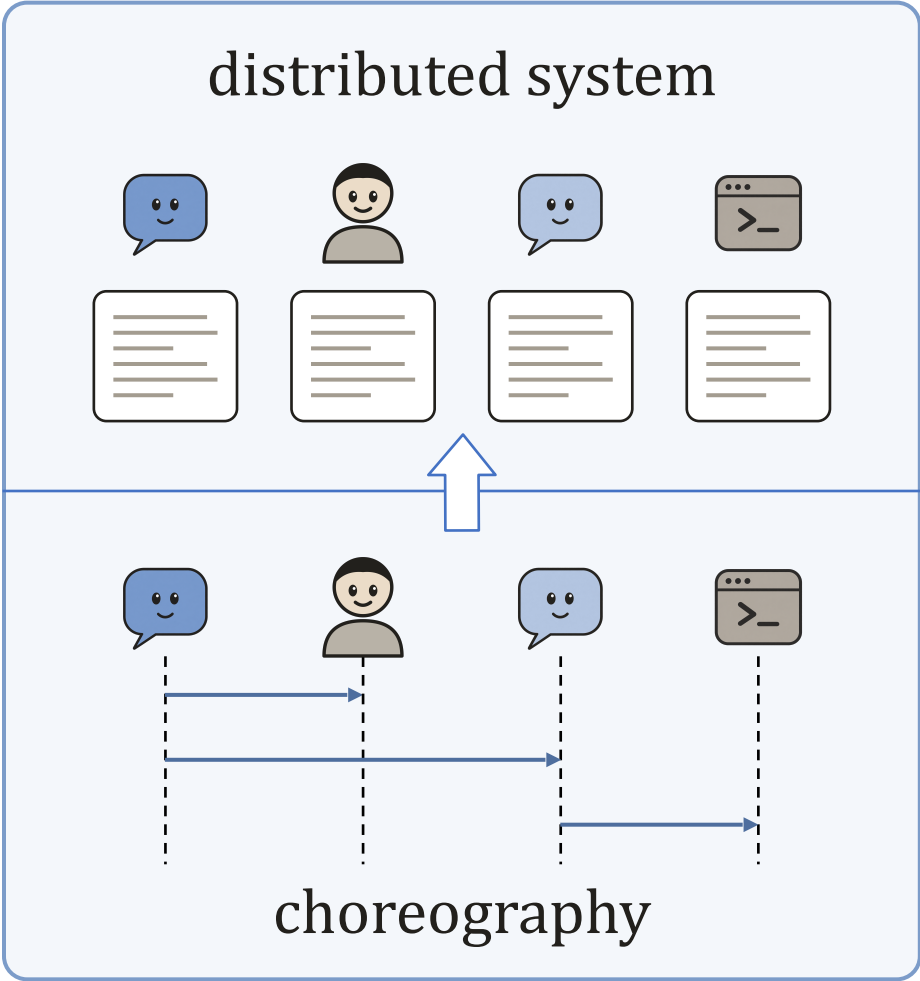
that an external tool terminates

that the environment is fair

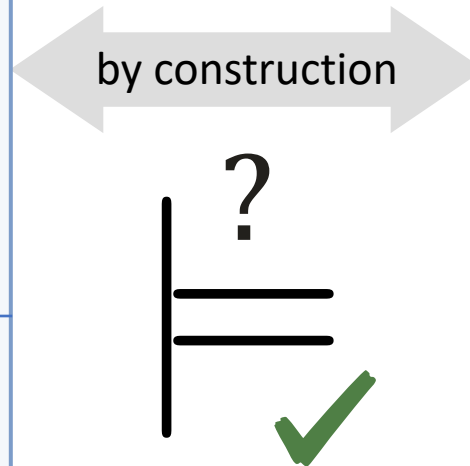
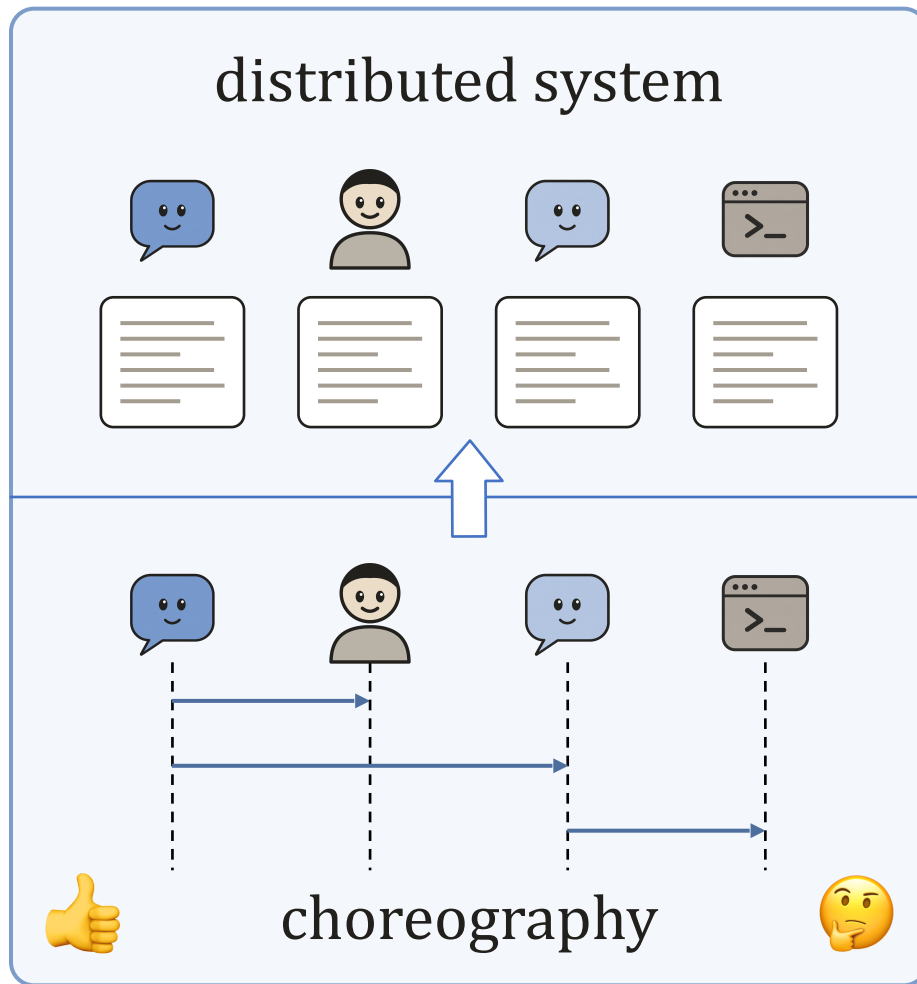
The mathematical problem



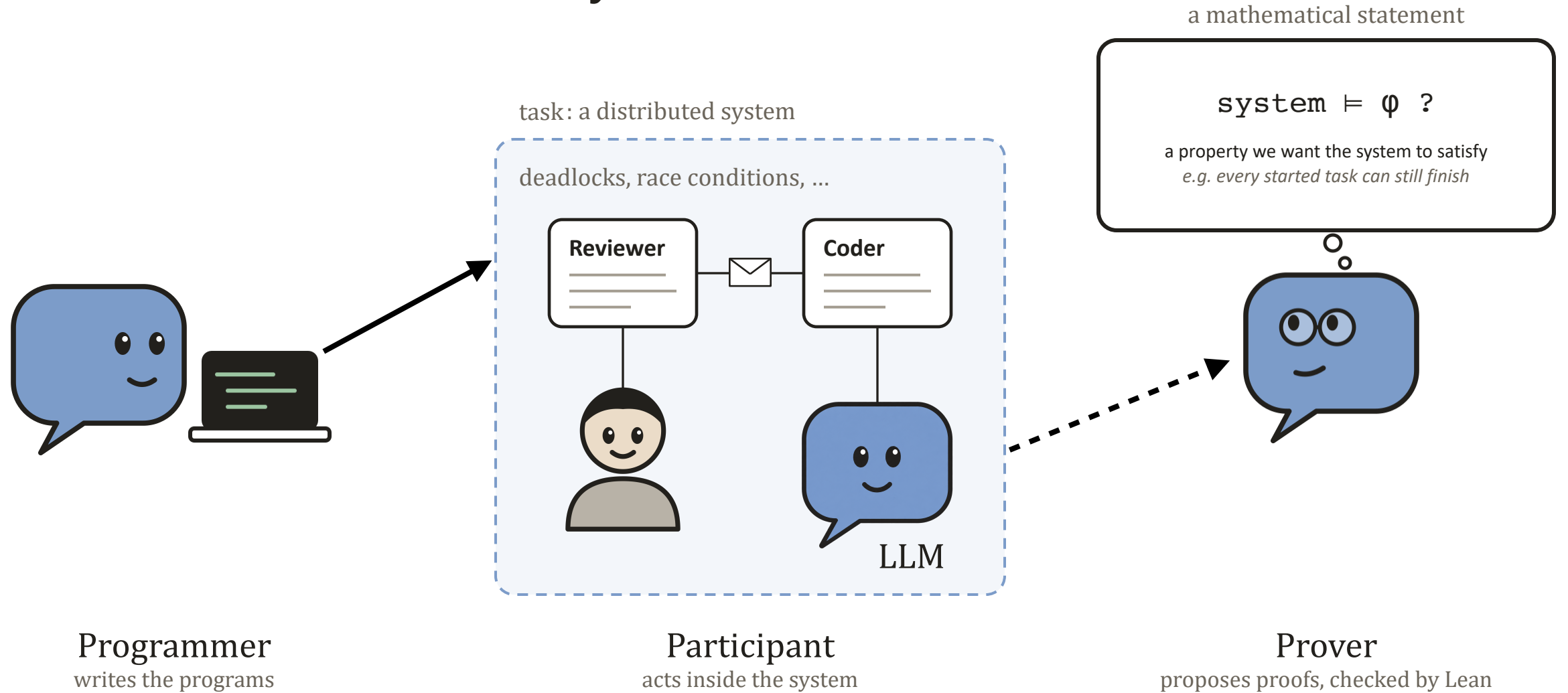
The mathematical problem



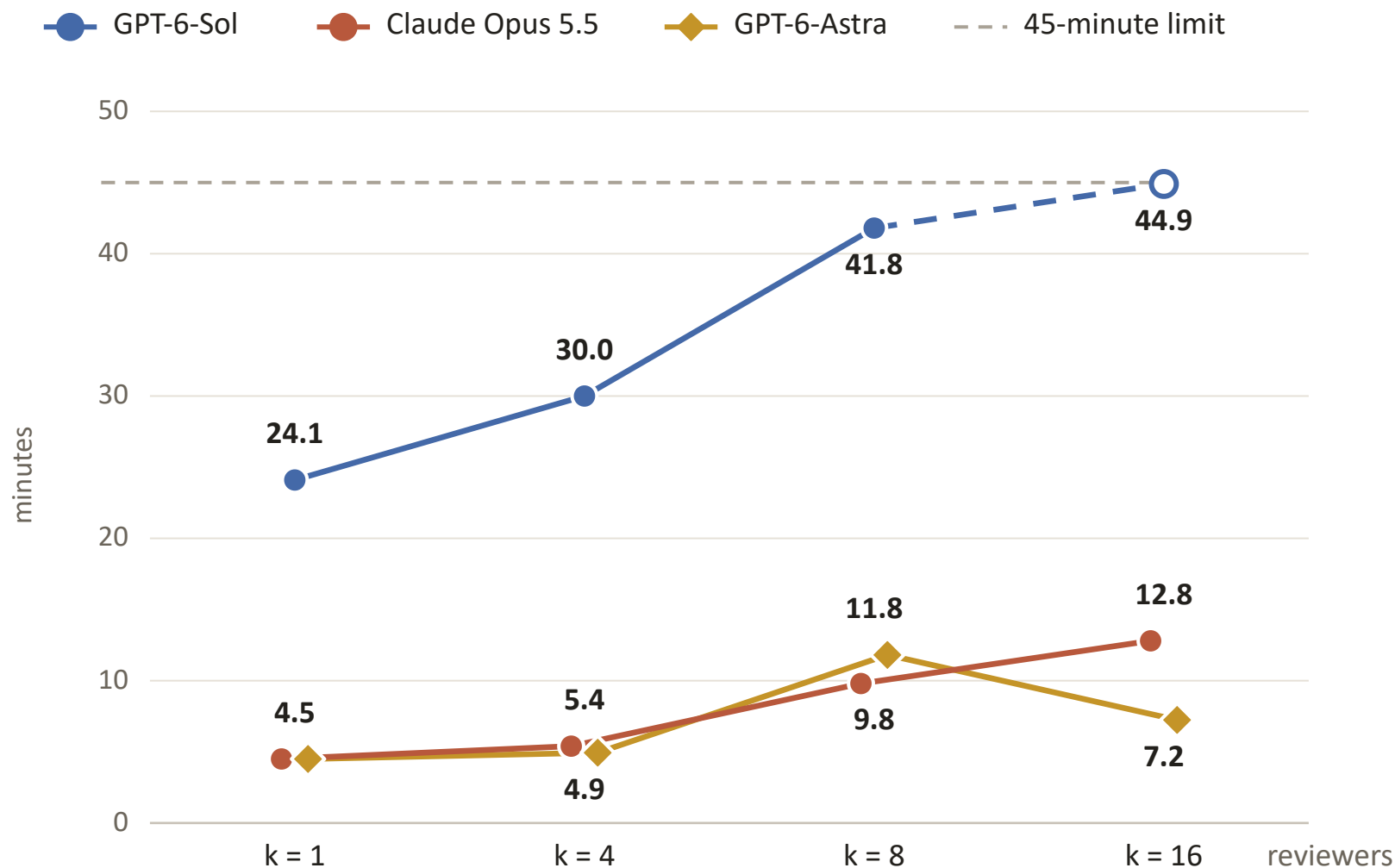
The mathematical problem



Can't LLMs do this already?



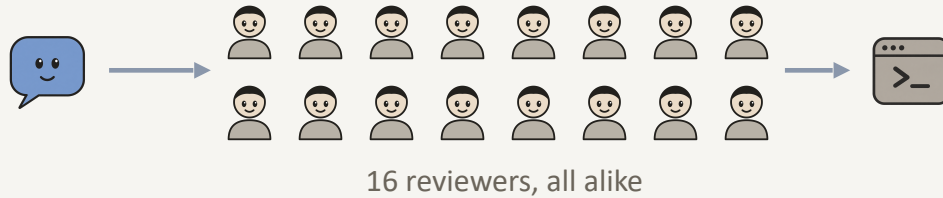
Let LLMs prove deadlock-freedom from scratch



This may not be the whole story

Exercise

simple pilot experiment

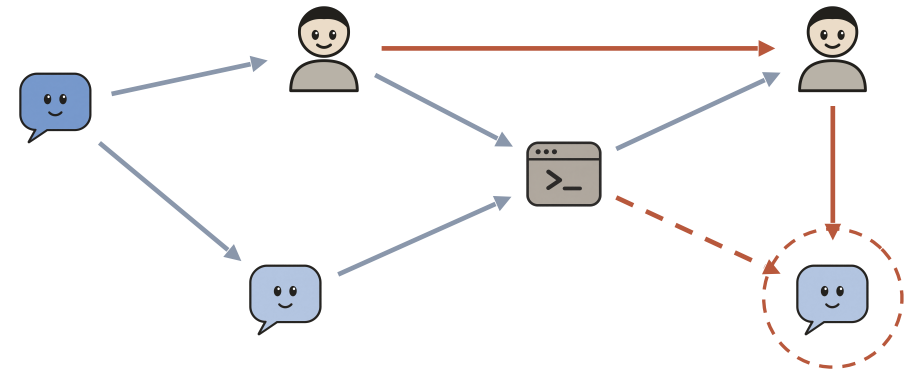


Repeated structure.
One invariant may cover them all.

Millennium Prize Problem?



hypothesis



Different roles, choices, overlapping sessions,
new participants. May also be manageable.

What should we still understand?

Back to the opening question.

1

Proofs

How much of a correctness proof must a human understand?

2



Experience

I deploy a system derived from a choreography. I trust its construction.

3

The future

What will coding and mathematics become?

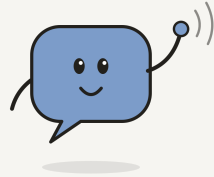
To go further

TOOL

ZipperGen

Choreographic workflows for LLM agents.

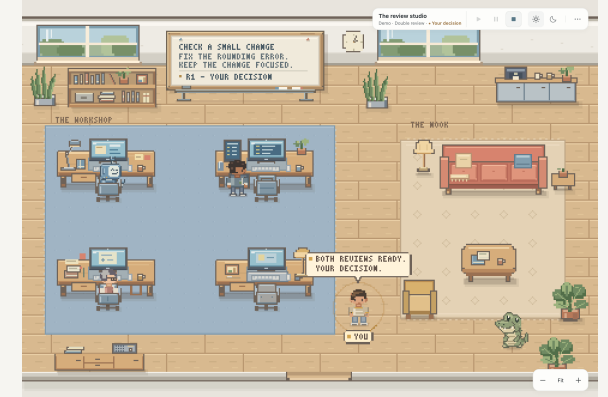
zippergen.io



DEMO

ZipperCrew

zippergen.io



PAPER

Provable Coordination for LLM Agents via Message Sequence Charts

B. Bollig, M. Függer, T. Nowak · ISoLA 2026 · [arXiv:2604.17612](https://arxiv.org/abs/2604.17612)

BOOK

Introduction to Choreographies

F. Montesi · Cambridge University Press, 2023