

# A night in the workshop

What happens when two systems keep triggering each other

---

## How does a loop arise between two automatic systems?

In our workshop, two fixed rules kept triggering each other. There was no limit on the number of retries. The monitoring saw heat, but did not count failed tasks. Our lesson: one retry, then a person.

## Summary

- **The question:** what happens when two systems keep triggering each other without a brake, while nobody is watching?
- **What we did:** we replay a real night in our workshop from September 2026.
- **What we saw:** by morning, there were 20,478 failed tasks, and other work had been pushed aside 1,533 times. Those two figures were measured.
- **What it means:** a fixed rule does not hesitate. So: one retry, then a person.
- **The main limitation:** the sequence through the night is simulated. Only the final figures were measured.

## Sequence (reconstructed)

| Moment        | What happens                                                                                                                                                                                                                              | Failed tasks         | Times other work was displaced |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|--------------------------------|
| Evening       | The Writer asks the Scheduler for a workspace. The Scheduler says yes.<br><b>Why:</b> The Scheduler follows a fixed rule: when someone asks for room, say yes. Its rule cannot see whether the machine actually has room.                 | 0                    | 0                              |
| Evening       | The workspace fails because the machine is full. The Writer immediately asks again. The Scheduler says yes again.<br><b>Why:</b> The machine was already full of other work. The Scheduler's rule did not account for that.               | 1                    | 0                              |
| Evening       | Another failure, another request. The Scheduler displaces other work to make room. The loop has begun.<br><b>Why:</b> The Writer has a fixed rule: after failure, immediately retry. The Scheduler makes room. Together they form a loop. | approx. 12           | approx. 2                      |
| Evening       | More and more work is displaced. Nobody notices.<br><b>Why:</b> No rule counts how often something fails. So nothing raises an alarm.                                                                                                     | approx. 300          | approx. 25                     |
| Night         | The machine gets hot. The Monitor sees the heat, but has no rule for the number of failed tasks.<br><b>Why:</b> The Monitor watches heat and memory. It has no counter for failed tasks.                                                  | approx. 2,000        | approx. 150                    |
| Night         | The machine can do nothing else. Other work now fails too.<br><b>Why:</b> The loop takes all available room. Other work can no longer get in.                                                                                             | approx. 8,000        | approx. 600                    |
| Night         | The Scheduler still says yes. A fixed rule never hesitates.<br><b>Why:</b> A language model might have hesitated. A fixed rule does exactly what it says, every time.                                                                     | approx. 15,000       | approx. 1,100                  |
| Early morning | Nobody is awake. Only a person can stop this.<br><b>Why:</b> There is no emergency brake that intervenes automatically. A person is needed.                                                                                               | approx. 20,000       | approx. 1,500                  |
| Morning       | A person arrives, sees the count and stops the Scheduler. The loop stops.<br><b>Why:</b> The rule since then: one retry, then a person.                                                                                                   | 20,478<br>(measured) | 1,533<br>(measured)            |

The sequence is reconstructed. Only the final counts were measured.

## With the lesson

1. The Writer asks the Scheduler for a workspace. The Scheduler says yes.
2. The workspace fails. The Writer gets one retry.
3. Another failure. The system stops and asks a person to take a look.

Reconstructed: this is how the rule we introduced after that night works.

**With the lesson: 2 failed tasks (reconstructed). That night: 20,478 (measured).**

## Why this went wrong

The biggest errors often come from the fixed rules between the intelligent components. This night shows how. Neither system hesitated. Both did exactly what they were supposed to do, and that was precisely why things went wrong.

## Each role follows a fixed rule

In this simulation, the systems are called the Writer and the Scheduler. The Writer turns conversations into reports. The Scheduler distributes computing capacity. The Monitor monitors the machines. Online, you see them on a stage, each alongside its rule. At each step, a note travels from one role to another: a task being passed along.

## The monitoring saw heat, not counts

The Monitor saw the machine getting warm. But it had no rule for the number of failed tasks. That allowed the loop to continue all night.

## One retry, then a person

- **One retry.** An automatic attempt gets one more chance. If it fails again, a person takes a look.
- **The system is switched off.** The Scheduler remains out of service until it is rebuilt with this brake.
- **The lesson is in our rules.** It became law 3 of our working policy Stay in control of your AI.

Online, you can turn on the lesson and replay the night.

## How we did it

The final figures come from our own log for that night. The sequence between them is a reconstruction based on what we found afterwards. Intermediate figures are rounded and marked as approximate. In the PDF, the steps appear in a table.

## Limitations

- **A simulated sequence.** The steps through the night are a reconstruction. Only the final figures were measured.
- **One night.** This is one event in one small workshop, not representative of SMEs.
- **No baseline.** At the time, we were not yet counting how many tasks failed on an ordinary night.

- **No public dataset.** This publication has no public dataset with which readers can independently recalculate the final figures.
- **The names are roles.** The Writer, the Scheduler and the Monitor are roles in this simulation.

## What this means

### What does this mean for you?

- **Where in your organisation do two systems trigger each other without anyone watching?** Tip: Look for automatic repetition: if it fails, try again. Allow each repetition one retry.
- **Does your monitoring count events, or only detect faults?** Tip: Also count how often something fails, rather than just whether a machine is hot or full.
- **Who can pull an emergency brake at night, and how do they know it is needed?** Tip: Agree who receives an alert and at what threshold. After that, a person takes a look.

## Closing notes

**With thanks to** the people and AI colleagues at ITARR.

**Also read:** Fixed rules, thinking colleagues, the working policy [Stay in control of your AI](#) (law 3), and How does an AI colleague catch its own mistakes?.

**How to cite this:** ITARR Lab (2026). *A night in the workshop*. Version 0.3, as of 18 October 2026. [itarr.eu/lab/een-nacht-in-de-werkplaats](https://itarr.eu/lab/een-nacht-in-de-werkplaats). CC BY 4.0.

**Sources:** our own log of a night in September 2026 (not public); the [film of the night](#) (as of 17 October 2026). The film depicts the incident; it is not an independent measurement of the final figures.

**Change log:** 22 October 2026, version 0.3: the core question, source links and limits of the evidence made clearer. The measurements and their reference date have not changed.

**Licence:** CC BY 4.0. Use, adapt and share it. Attribute ITARR Lab, link to the licence and indicate any changes.