9ème édition de la

Soirée du Test Logiciel Sophia - Antipolis

15 septembre 2026

17h à 23h

ETSI, Valbonne

Soirée du test logiciel

Sophia Antipolis

The silent erosion of quality

Jean-Paul ACHARD
Quality Assurance Officer, Amadeus

If AI can generate tests, analyze requirements,
investigate defects and write reports…

…what is left for QA engineers to learn?

How a real gain becomes a silent loss?

1. **The promise.** What GenAI really gives QA
2. **The silent erosion.** Degrading unnoticed
3. **Protect vs accelerate.** The capability map
4. **Guardrails, validation, accountability**
5. **Keeping the skill alive.** Techniques, hackathon

Who am I?

- **QA** Quality Assurance Officer, Amadeus
- **RES** QA for airline Reservation systems: ticketing, reissue, refund
- **AI** Leading GenAI adoption for QA: charter, coaching, prompt patterns, validation rules
- **CT** ISTQB CT-GenAI as the reference frame

> *Everything here comes from real coaching and tooling friction.*

1

The promise

GenAI genuinely accelerates test design, automation and analysis.

GenAI helps everywhere in the test process
The gain is real …

Test analysis

Surfaces ambiguities and missing acceptance criteria

Test design

Proposes cases and expands edge conditions

Automation support

Assists scripting and maintenance work

Reporting

Summarizes results and spots trends. Low risk, high value

Exploratory thinking

Generates varieties

Defect investigation

Log summarization and hypothesis generation

… and that is the problem

In one year, this job will not exist.

2

The silent erosion

Quality does not collapse. It degrades quietly, one accepted output at a time.

Nothing breaks loudly
The build is green and the report reads well

Looks like	Actually
Plausible acceptance criteria	Traceable to no business rule
Syntactically valid tests	Business-invalid
Coverage complete	Because AI declared it complete
Regression summary reads well	Flaky failures reported as real
A review in the process	Not in anyone's head

The defect is not in the output.

The defect is in the acceptance of the output.

Six failure patterns worth naming out loud in your team

Hallucination

Plausible but wrong: locators, APIs, rules that do not exist

Reasoning errors

Domain logic misread into invalid but sensible-looking scenarios

Bias

Training data produces non-representative test data. Costly for global products

Tunnel vision

Answers the prompt it was given and misses cross-domain impact

Non-determinism

Same prompt, different answer. Reviews stop being reproducible

Shadow AI

Unapproved tools, sensitive data, no trace, no governance

Soirée du test logiciel

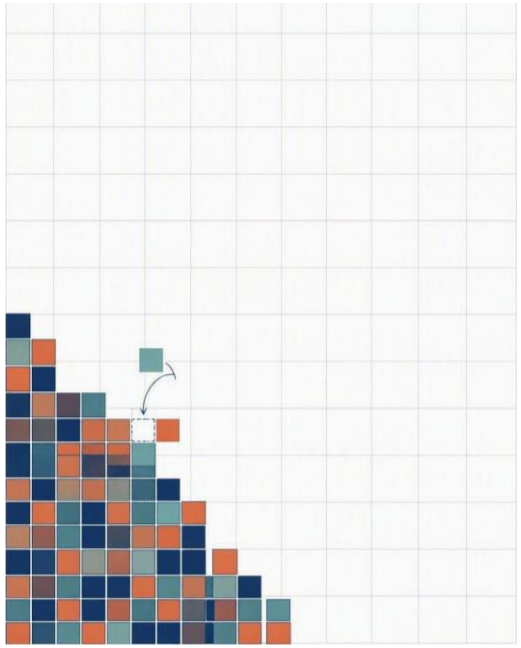
Conférence

EXAMPLE FROM OUR ACTIVITY

Bias because of the testing framework

What happened

- We asked AI to write new scripts inside an existing framework holding hundreds of scripts
- Expected: consistent scripts, faster, aligned with the functional documentation
- The existing scripts covered mostly similar features, so the new ones copied that narrow angle

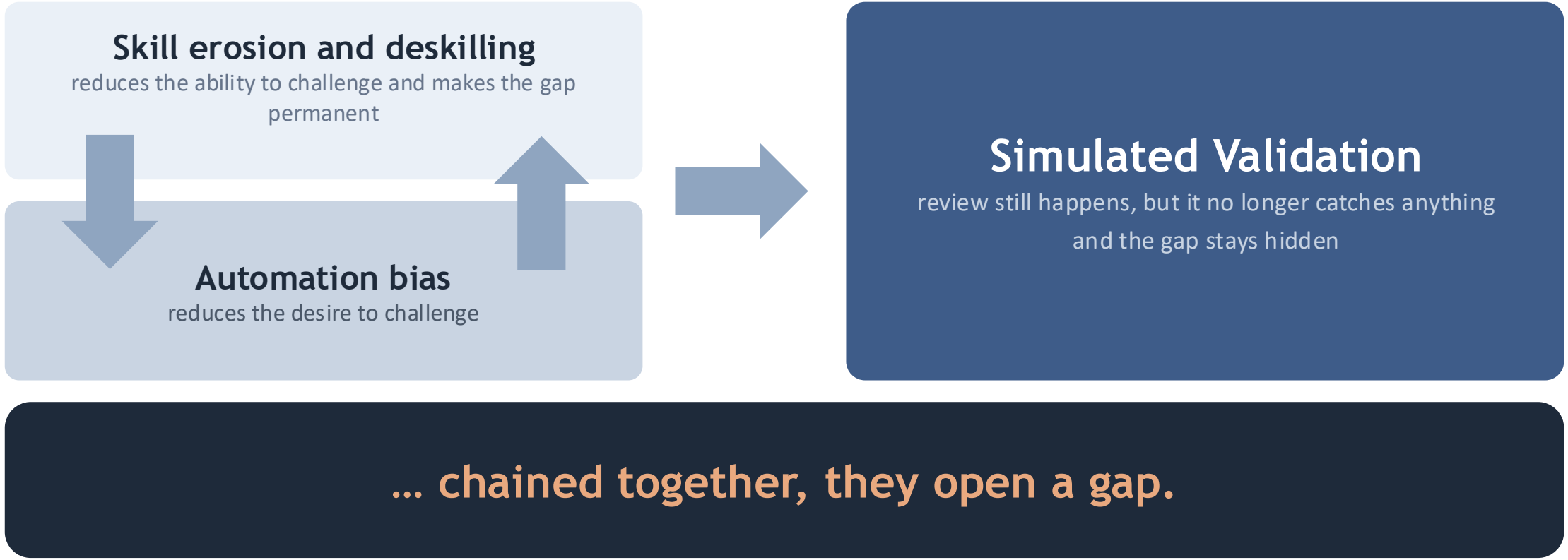


What we took away

- We now start from the risk and the documentation, not from the existing scripts
- Coverage looked healthy while whole feature areas had no test at all
- Review what the framework never tests before asking AI to extend it

AI amplifies the coverage you already have - including its blind spots.

Alone, each of these is manageable…

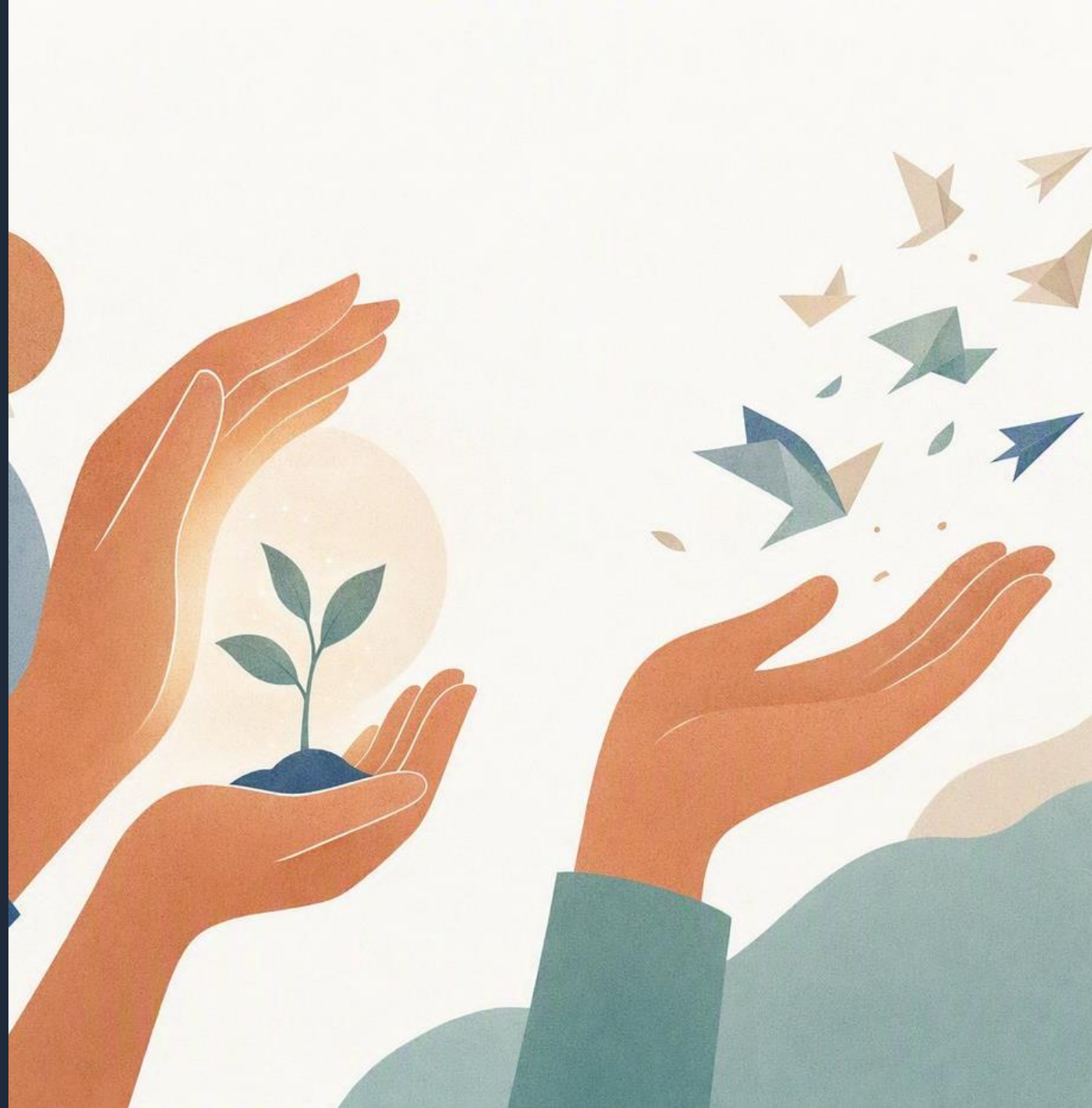


Nothing broke. Nothing was flagged. Nothing was found.

3

Protect vs accelerate

A capability map: what must stay human, and what is safe to delegate.



Protect the judgement that makes you an engineer Accelerate everything else

PROTECT remains critical

- Critical thinking
- Risk analysis
- Exploratory testing
- Business challenge
- Root cause investigation

These require context AI does not have.

ACCELERATE safe to delegate

- Documentation
- Summarization
- Idea generation
- Repetitive activities

These cost time and build no expertise.

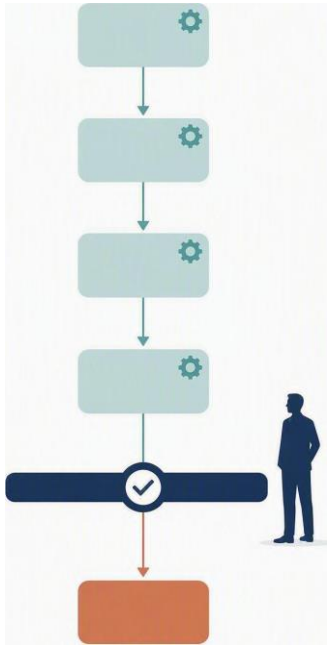
Automate the typing, not the thinking.

EXAMPLE FROM OUR ACTIVITY

Agents across our test cycle

What we accelerate

- Well-defined agents draft the
  - test strategy
  - test plan
  - scripting
- Agents also handle duplicate-test detection and first-pass pull request reviews



What we still protect

- Risk analysis, functional coverage and prioritization stay a manual review
- No plan goes forward until an engineer has accepted the risks behind it
- The agents produce the tests; we still decide which risks deserve them

We automated writing the tests, not deciding what is worth testing.

Soirée du test logiciel

Conférence

QA interviews changed: we stopped testing recall and started testing judgement

What we stopped testing

- Writing a test case from a clean spec
- Syntax and tool recall
- Volume of scenarios produced

What we now probe for

- Why is this AI-generated case wrong?
- What is missing from this coverage claim?
- Which risk would you challenge with the product owner?

You didn't review it. You forwarded it.

4

Guardrails, validation, accountability

The pragmatic framework

Behavioral rules and engineered controls.

Pillar 1 | A charter so everyone knows the risks and their duties

Be aware of the risks

- AI outputs are proposals, not truths
- AI supports the QA engineer; the engineer decides
- Validation effort is proportional to business impact
- No traceability, no acceptance

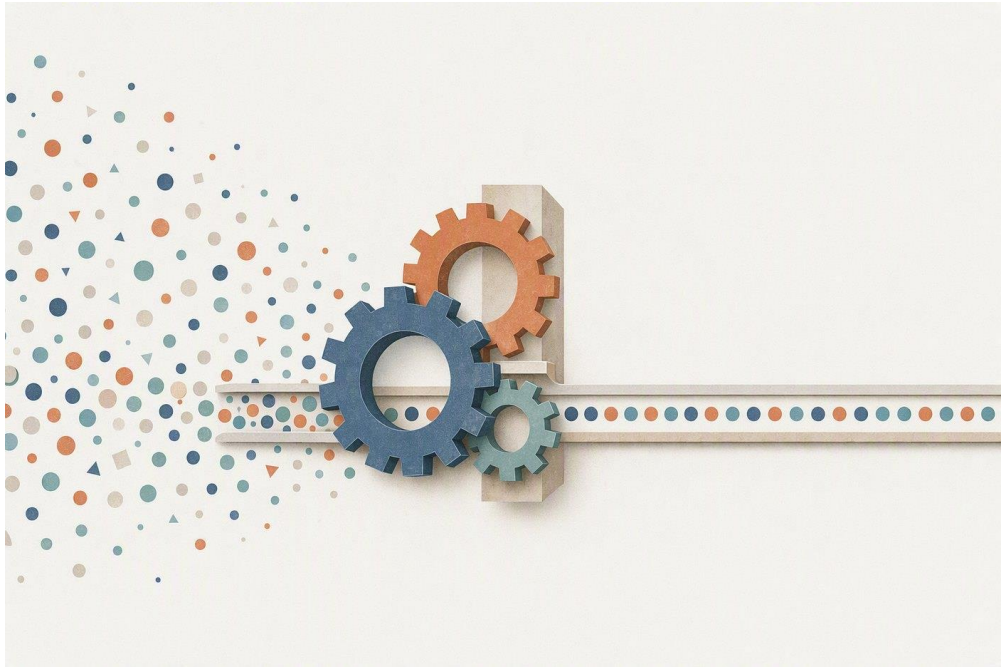
Know your duties

- No production or sensitive data in AI tools
- Company-approved AI tools only
- No blind copy and paste from AI outputs
- The QA engineer owns the final wording, tests and priorities

Source: QAE Charter - Generative AI usage, Reservation

Pillar 2 | AI orchestrates, scripts decide the facts

1. Identify the volatile step - where the same prompt returns a different answer
2. Wrap it in a deterministic script - an agent skill, an automated validation in the workflow
3. Constrain AI to orchestration - it calls the script, it does not invent the fact



Where a probabilistic answer is unacceptable, remove the model from the loop.

Soirée du test logiciel

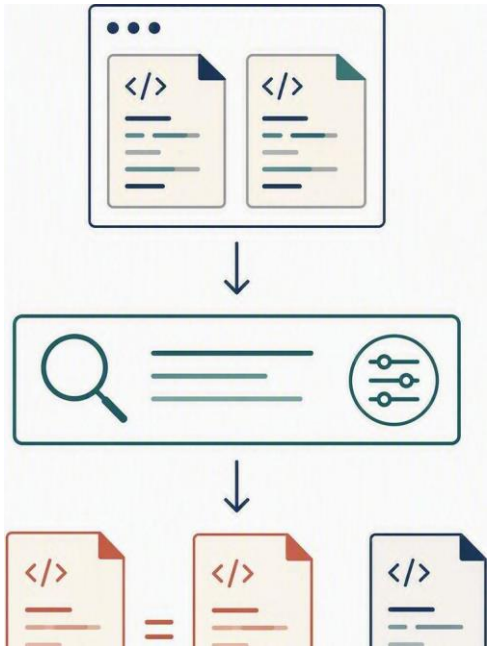
Conférence

EXAMPLE FROM OUR ACTIVITY

Duplicate scripts scanner

What happened

- Spotting scripts that already tested the same thing was manual and unreliable
- Asking an agent directly meant hoping it picked a sensible comparison method, a different one each time
- How duplication is detected is a fact, not something to improvise at each run



What we took away

- We used AI and our framework to build a deterministic scanner, with the model out of the detection
- It runs on every pull request, so duplicates are caught before they enter the suite
- The search algorithm stays ours to tune, updated deliberately when the risk changes

AI helped us write the scanner. It does not get to decide how duplication is measured.

Soirée du test logiciel

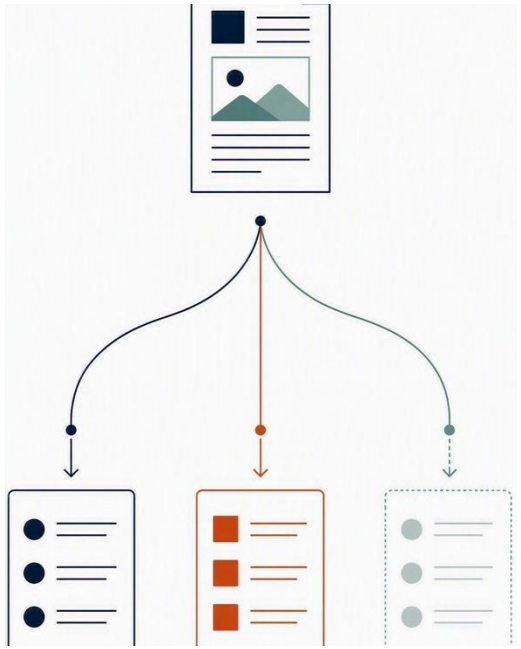
Conférence

EXAMPLE FROM OUR ACTIVITY

Non-determinism in regression monitoring

What happened

- We built a tool to analyse the daily regression run and extract the failing tests
- Expected: no manual triage, a reliable failing-test list every morning
- First delivery used AI prompts alone: same run, different answers, regressions missed



What we took away

- Extraction moved to deterministic parsing; AI only summarises what the parser found
- The same log now returns the same failing-test list every time
- Validate the tool against a known run first: a missed regression is silent

If the answer changes between two identical runs, it was never a monitoring tool.

Pillar 3 | each output gets the validation its scope demands

Brainstorming → a relevance check is enough; nothing downstream depends on it

Acceptance criteria → prove correctness and completeness

Test cases → prove coverage and data realism

Regression reports → prove factual accuracy, and flaky versus real

Test prioritization → prove the risk assumptions themselves are legitimate

AI proposes. The QA engineer decides how hard to look.

Pillar 4 | AI-assisted work is done only when all five are true

1	**Traceable to an identified test basis**	Every output points back to a requirement, a risk or a scenario.
2	**Reviewed by a QA engineer**	A person owns the review, not a team or a queue.
3	**Validated for correctness and completeness**	Checked against what it should cover, not only what it says.
4	**Explicitly accepted or rejected**	Rejection is a valid outcome, and it is recorded.
5	**Decision and rationale are auditable**	The reasoning survives the sprint that produced it.

Accountability never transfers to the model.

Pillar 4 in practice. A pull request review

Copilot reviews the PR

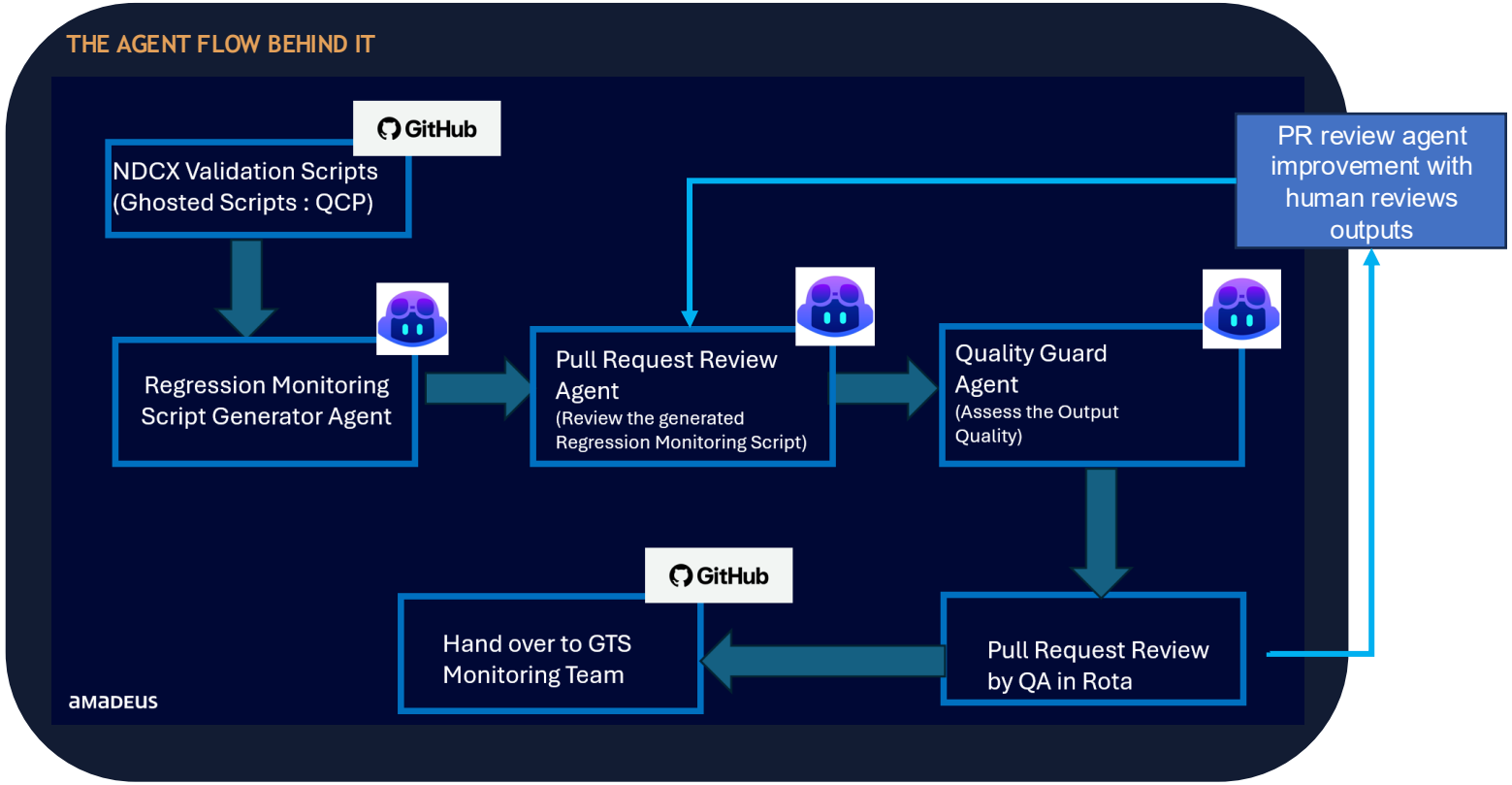
- Diff explanation
- Obvious defects flagged
- Missing tests detected
- Naming convention

The engineer reviews it

- Not “is this acceptable?”
- But “what did it not look at?”
- Business risk, edge cases, assumptions

The DoD closes it

- Gaps identified and named
- Decision and rationale recorded
- Reviewed = gaps found, not read



Source: pull request reviews in the Reservation QA repositories

AI didn't replace you. It stopped needing you to think.

5

Keeping the skill alive

Techniques, a hackathon, and a way to measure it.

Five techniques keep the creativity muscle in use Applying one beats listing five

1 Challenge the AI

The mission is not to approve it. It is to find what it missed.

2 Prompt competition

Several answers to the same question. Comparison forces reasoning.

3 Review checklist

Business risk, edge cases, silent assumptions, missing scenarios.

4 Blind analysis

Do the analysis alone first, then compare. The delta is the learning. It avoids anchoring bias.

5 Explain before accept

Never accept a recommendation you cannot personally explain. If you cannot explain it, you did not review it -> you forwarded it.

Soirée du test logiciel

Conférence

Techniques on a slide change nothing
A hackathon trains the reflex

1. **Seed**
   AI outputs with seeded hallucinations and gaps
2. **Hunt**
   The challenge is not to build - it is to break and improve
3. **Score**
   Points for what you caught and why, and the speed
4. **Repeat**
   The score is a retention metric

Critical thinking is a muscle. Give it a gym.

EXAMPLE FROM OUR ACTIVITY

The hackathon we want to put in place

A tool to validate, with defects we planted on purpose

1. **Find the seeded defects**
   A real internal tool to validate and we know exactly what we broke
2. **Best functional coverage**
   Not the most findings, the widest and most justified coverage
3. **Best model per activity**
   Which model earns its place on which task, with evidence
4. **Most efficient prompts and tokens usage**
   Same result for fewer tokens is part of the score

How it runs

Teams of QAEs, one time-boxed day, the same tool and the same seeded defects for everyone.

THE UNOFFICIAL GOAL

QAEs get hands-on with the latest AI tooling, and we keep what they build: the prompts and agent architectures that actually worked.

The scoreboard is the excuse. The trained reflex and the prompts are the prize.

How to measure whether the skill is still there

What to measure

- Unassisted analysis: design tests for a new feature with no AI
- Review depth: gaps found per review
- Time to first doubt: how fast someone challenges an output

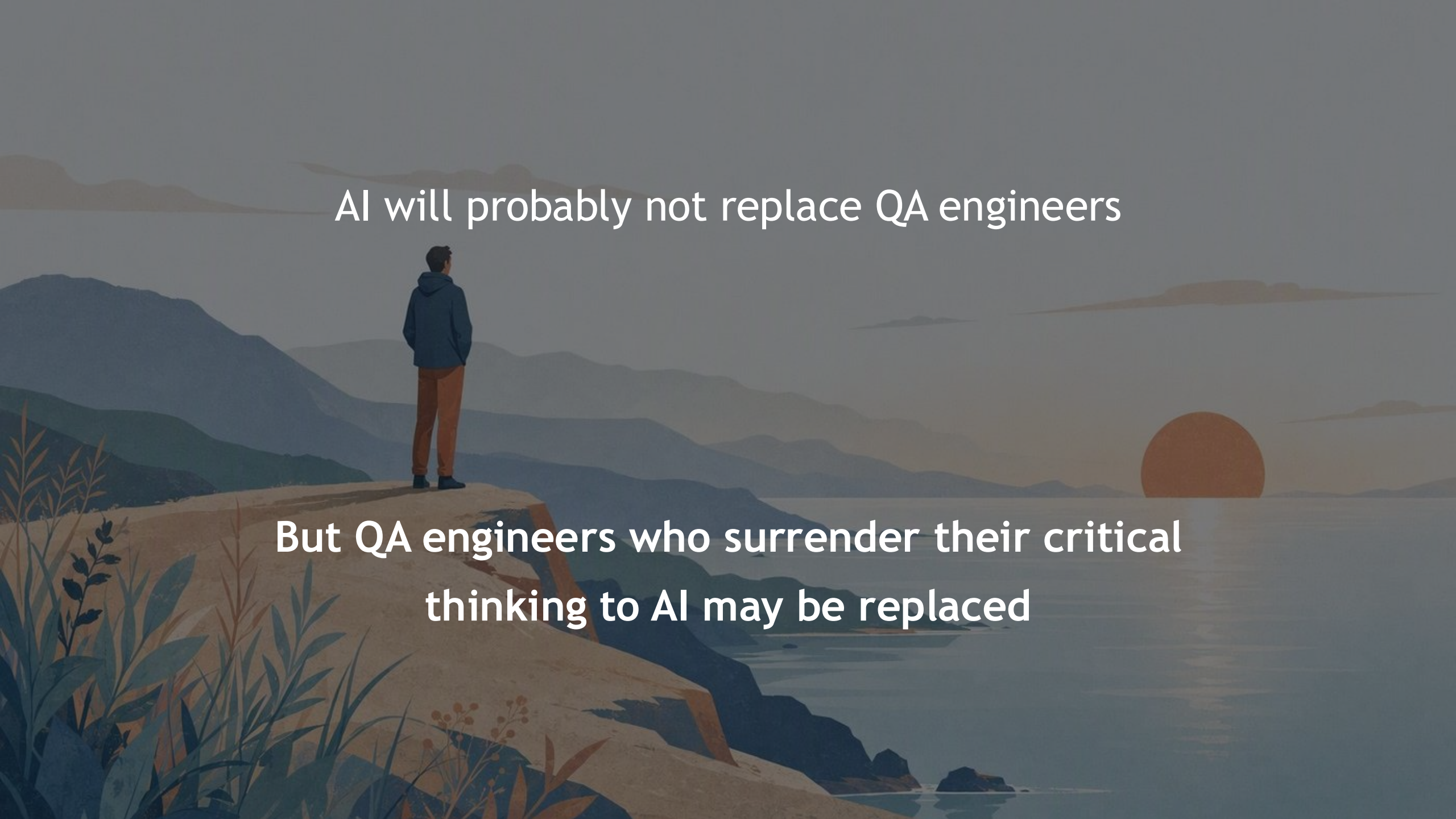
How to run it

- Twice a year, same format, so scores compare
- Score the reasoning and the why, not the speed
- Trend at team level

Pitfalls

- No baseline taken before AI use widened. Nothing to compare against
- Difficulty drifting between rounds. The scores stop comparing
- A dashboard of ten metrics nobody reads or acts on

Measure the capability, not just the output.


AI will probably not replace QA engineers
But QA engineers who surrender their critical thinking to AI may be replaced

Five things to take home

1. The risk is not replacement. It is the quiet disuse of critical skills
2. Erosion happens at acceptance, not at generation
3. Protect judgement, accelerate the rest
4. Start with one technique and train it in a hackathon
5. Measure skill retention, not only AI adoption

9ème édition de la

Soirée du Test Logiciel Sophia - Antipolis

15 septembre 2026

17h à 23h

ETSI, Valbonne

Soirée du test logiciel

Sophia Antipolis

Merci de votre écoute !

SCAN ME

Votre avis nous intéresse

Appendix

Our AI outcome reviews confirm the gain …
… and expose how premature the trust is

What we put through outcome review

- Acceptance criteria drafts
- Generated functional test cases
- Nightly regression summaries
- Risk-based prioritization proposals

What the review keeps finding

- Criteria traceable to no business rule
- Tests valid in syntax, invalid in business logic
- Flaky failures reported as real defects
- Priorities justified only by plausibility

Document reviews are where we catch it: the wording is fine, the traceability is not

What lands on my desk

- Master test plans
- Acceptance criteria
- Test strategy updates
- Handover documents

What I actually check

- Traceability to a test basis
- Business rules named
- Refunds, penalties, passenger types
- No ambiguous wording

What gets sent back

- Fluent text anchored to nothing
- Coverage claimed, never shown
- A reviewer named, no review done

The QA role shifts toward governance

What moves away

- Bulk test case writing and scripting
- First-draft documentation
- Manual result summarization

What becomes the job

- Prompt curation as a team asset
- Output validation and risk assessment
- Quality governance: charter, guardrails, auditability
- Being the trust enabler for AI-driven systems

From test production to quality governance and to strategic influence.