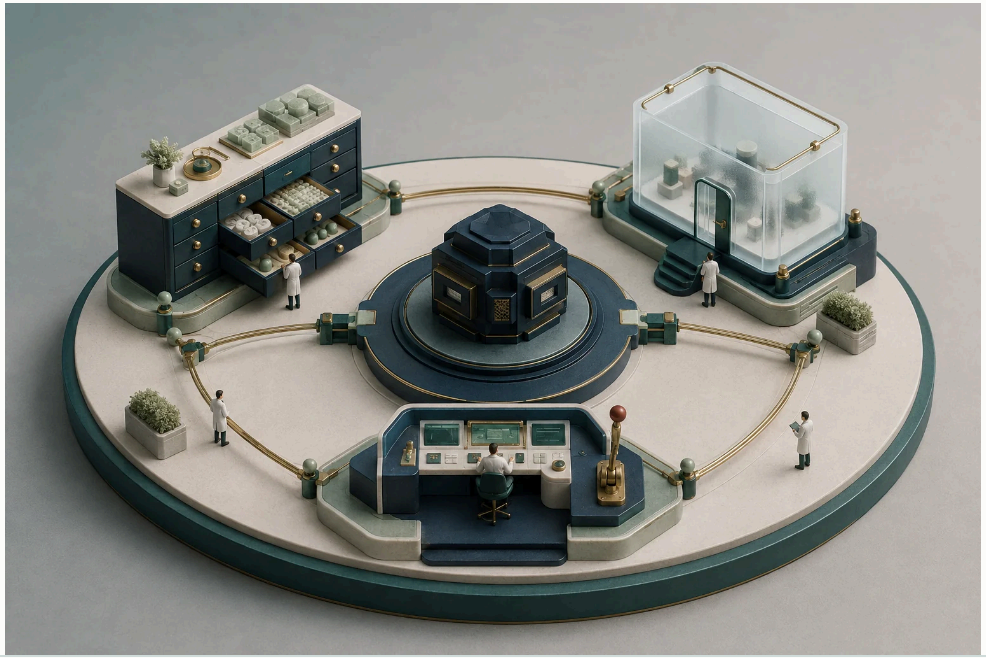
Assurance across the AI lifecycle

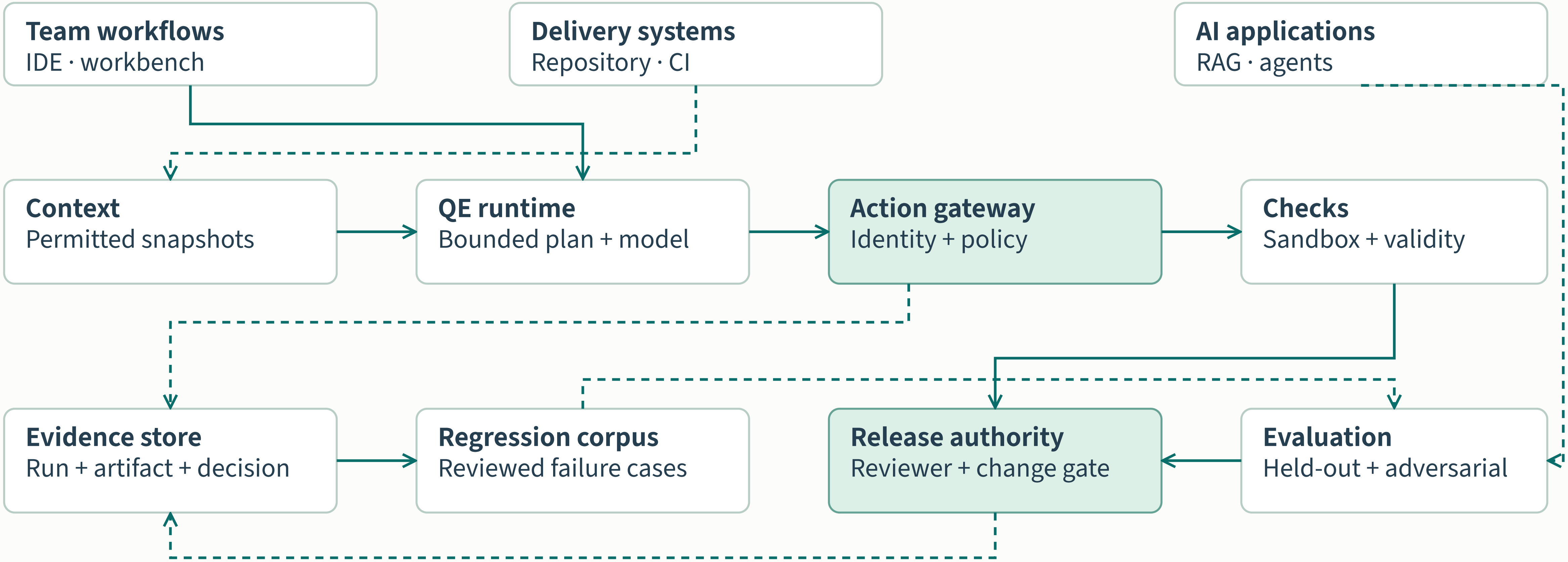
AI × quality engineering · Industry research

For AI-assisted payment testing:
Start with the banking workflow →



Prepared by Tom Wu · AI-generated illustrations · Synthetic English narration in the web presentation.

The enterprise assurance platform



A payment test has four joined contracts

Task envelope
PAY-042 · payments-api
Versioned input hashes
Scope · expiry · budget

→

Policy decision
DEC-042 · policy v7
Allow sandbox test only
Identity + expiry binding

→

Evaluation manifest
EVAL-042 · corpus v3
Original passes
Mutant fails
Artifact + runner hashes

→

Evidence record
RUN-042 · review held
Links all three artifacts
No self-approval

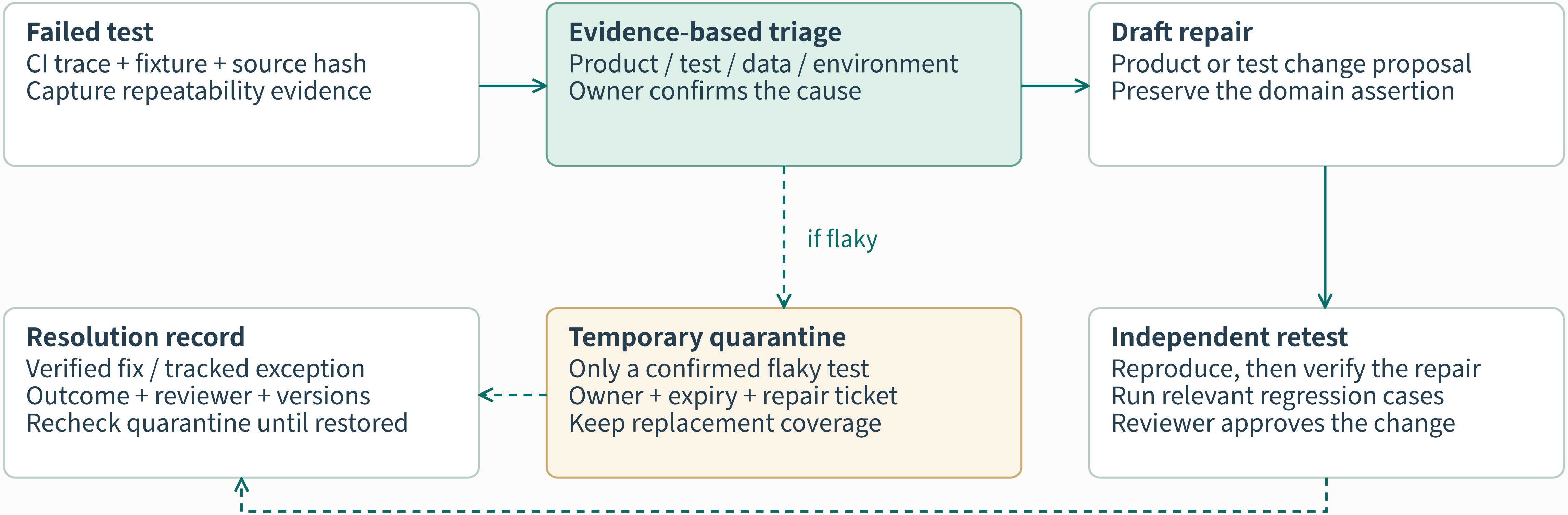
Every boundary verifies identity, versions, expiry, resource scope and the same artifact digest.

Mismatch, unavailable authorization or missing durable evidence → hold; no promotion.

Failure semantics are part of the contract

FAILURE	REQUIRED BEHAVIOR	PAYMENT EXAMPLE
Timeout / unknown result	Cancel new actions; query status by idempotency key before retry.	Reuse RUN-042; never launch duplicate runners blindly.
Stale policy / auth outage	Deny execution; renew policy and re-authorize.	Expired DEC-042 cannot authorize a test run.
Evidence unavailable	Hold promotion; bounded durable outbox or halt.	No complete evidence receipt means no review-ready change.
Digest / oracle mismatch	Invalidate results; regenerate evidence against the new artifact.	Changed payment assertion cannot reuse EVAL-042.
Recovery	Human queue; exercise the failing case before an owner resumes.	Model rollback does not undo completed external actions.

Repair the failure or track the exception



Tools occupy different assurance layers

LAYER	REPRESENTATIVE TOOLS	LOCAL PROOF
Test assets	Tosca · Applitools	Test validity; baseline quality
Change review	GitHub Copilot	Issues found and missed; reviewer effort
AI evaluation	LangSmith · Microsoft Foundry	Dataset fit; judge calibration; exportability
Adversarial testing	Promptfoo	Coverage of actual tool authority

Reuse approved CI, identity and artifact storage; add thin adapters and owned evaluation datasets where required. Representative documentation, reviewed September 2026. Not a ranking or independent product benchmark. [T01][T02][T03][T04][T05][T06]

Platform prerequisites for the selected workflow

Adoption decision

Required evidence accepted · owner accountable · foundation work funded

↑

Business & people

Expected outcomes · reviewer capacity · offshore handoffs · measured baseline

Runnable engineering

Infrastructure · CI/CD · test data · dependency control · testable applications

Supported services

Frameworks & devices · AI access · run evidence · platform ownership

Authored adoption assumptions. Validate each application and workflow; unknown requirements remain open. Assumptions and evidence →

Assess infrastructure, pipelines, dependency control, data reset and application testability separately. Add approved AI access, observability, framework ownership and operating capacity.

Controlled dependencies and real integration

Reviewed payment test

Same key · retry · expected balances

→

Payment service

Pinned build and isolated fixture

→

Controlled provider

Timeout · delay · duplicate callback

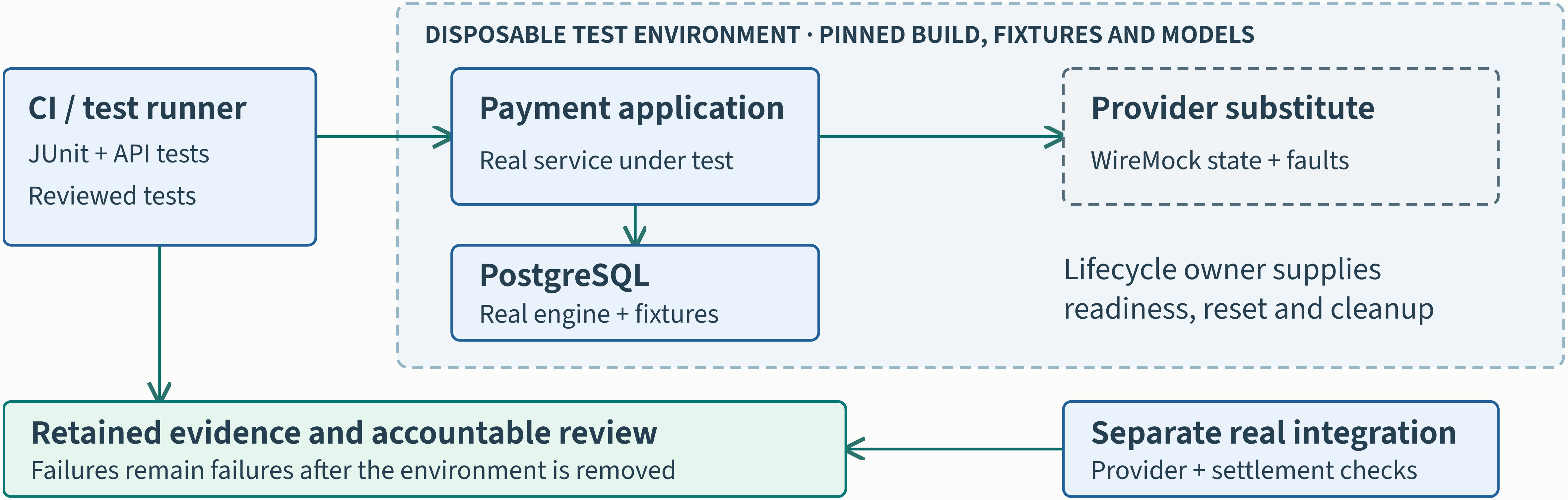
Separate integration gate

Versioned contract + provider sandbox / real batch validation ↔ Refresh the substitute when the real dependency changes

Proposed payment test architecture. A passing substitute is evidence about the modeled behavior, not proof of compatibility with every real dependency. Dependency strategy and sources →

An integration engineer owns contract fidelity, state reset and refresh triggers. AI may draft HTTP mappings; message, mobile and legacy dependencies need their own test strategy.

A repeatable test environment



Run real application and database software, model selected provider behavior and retain original evidence beyond environment cleanup.

Containerization, virtualization and contracts

RUN

Containerization

Package and run real software with repeatable dependencies.

EXAMPLE

A PostgreSQL instance created for one test run.

SIMULATE

Service virtualization

Replace selected dependency behavior with a controlled model.

EXAMPLE

A WireMock provider with an intentional timeout.

CHECK

Contract testing

Check whether consumer and provider interactions agree.

EXAMPLE

A provider change fails a reviewed Pact contract.

These capabilities complement each other. They do not establish complete business correctness or replace all real integration checks. Technology choices and limitations

Select a supported runtime for the application. Contract compatibility and simulated behavior need complementary domain and real-integration checks.

Value within quality and control floors

DECISION	RULE, APPLIED IN THIS ORDER	OWNER / ACTION
Stop / hold	Any quality or authorization breach. Economic route: adequate evidence puts every selected case below 10% net effort saving.	Control owner holds access; sponsor stops or re-scopes.
Insufficient evidence	Missing approved outcome, baseline, exposure or review evidence; uncertainty crosses the agreed target.	Measurement lead reports unknown; collect evidence within a capped extension.
Go, limited scope	Meet the charter's outcome, adoption ≥50%, floors and funded cost cap. Economic route: one case ≥15%. Cash claims also need Finance proof.	Sponsor + control owner authorize the evidenced scope.
Extend / redesign	Agreed target, adoption or cost criterion missed without a breach; economic result in 10–15% band.	One new hypothesis; one extension ≤4 weeks.