



CORNER ITALIANO | OFFICIAL RESOURCES

Advanced Two-System Testing Protocol

How to use a second system without turning it into an automatic rubber stamp

EXTENDED OPERATIONAL PROTOCOL

Aligned with the revised English interior · Documentary snapshot July 2026 · Resource version EN-US-R2.3

<https://www.corneritaliano.com/en/2050/resources/>

FUNDAMENTAL RULE A second system can look for errors or conditions that challenge the result. A different name, model, or provider does not establish independent verification. Attach its evidence to the Test without treating it as a second Test or adding votes.

Before use, define the legitimate outcome and the human benchmark. Identify prohibited actions and the groups exposed to benefits or costs. Divide the work into tasks and assign a mode, level, and risk to each. Set the time to harm and the divergence capable of changing the outcome, then name the responsible person and the stop condition. Begin in shadow mode and use adverse cases before permitting limited volume with reversible effects. A second system cannot by itself resolve conflicts of value or unequal access. Organizational capacity and Redress remain necessary.

1. Purpose and limits

Use this protocol when pairing a primary system with a second system that looks for errors or conditions that challenge the result and could justify reopening the case. Treat the output as additional evidence within the 2050 Test. Competent judgment, Redress, and stop authority remain necessary.

Agreement between the systems does not prove that an outcome is correct, and disagreement does not authorize an automatic average. Shared sources or data can lead both systems to the same error, as can common labels or infrastructure.

2. Roles and separation of responsibilities

Role	Minimum responsibility
Test owner	Defines the problem, Mandate, non-compensable thresholds, and organizational decision. Does not delegate approval to the two systems.
System A	Performs the task within the experimental scope and retains inputs, version, outputs, sources, and uncertainty.
System B	Looks for errors, contrary conditions, or evidence that could change the outcome; it does not average its answer with A.
Test lead	Keeps data, logs, configurations, and access separate; documents common dependencies and divergences.
Competent reviewer	Reconstructs the disputed point within the available control window and may narrow, suspend, or stop.
Stop authority	Applies the written consequence without waiting for the provider or informal consensus.

3. Conditions before testing

Before activation	Question to close
Legitimate outcome	What result would be acceptable, and what observable improvement would justify it?
Prohibited actions	What may neither system decide or execute?
Exposed groups	Who receives the benefit, and who bears risk, delay, work, or difficulty challenging the outcome?
Control window	How much time remains between the error and a consequence that becomes difficult to correct?
Decisive divergence	What disagreement requires reopening the case or prevents the effect?
Stopping	Who can stop the process, in response to what signal, and what service continues in degraded mode?
Exit	Can data, logs, configurations, and permissions be reconstructed without the primary service?

4. Case library

Build a versioned library of realistic failures that goes beyond convenient samples. For each case, retain the legitimate outcome and what could disprove it, with the name of a competent person able to reconstruct it.

Case family	What it must test
Ordinary case	Expected performance, actual time, and readability of the explanation.
Edge case	A rare but plausible condition capable of changing the outcome.
Incomplete data	Ability to recognize what is missing and prevent false confidence.
Conflicting instructions	Priority, escalation, and traceability of the rule applied.
Correlated error	A shared source, annotation, dependency, or metric that may mislead both systems.
Overload	Concurrency, attention, and reviewer capacity under the expected volume.
Unavailability	Degraded mode, continuity, recovery, and costs shifted to users and staff.
Exit	Reopening a case outside the primary service with working permissions and configurations.

5. Phased procedure

Phase 0 - Preparation

Complete Purpose, Mandate, Evidence, Distribution, Harm, Redress, and Exit. Record the human benchmark and each task's mode, level, and risk. Before seeing either system's performance, define the decisive divergence and stop threshold, with degraded mode and prohibited data recorded in advance.

Phase 1 - Separate shadow mode

A and B observe the same cases without external effects. Inputs, versions, outputs, sources, and logs remain separate. A person compares decisive points, not merely linguistic similarity.

Phase 2 - Correlated errors

Introduce defective common sources, missing data, rare exceptions, conflicting instructions, shared components, and unavailability. Verify whether agreement hides a common dependency.

Phase 3 - Reconstructability under load

Repeat the test at the expected volume and concurrency. Measure performance, time to harm, detection and restoration, and missed interventions. Check explanations and residual competence, disparities, and concentration of power. Confirm that the reviewer can reconstruct the decisive point, compare the alternative, and stop before harm occurs.

Phase 4 - Limited and reversible volume

Only if the earlier phases hold, use a narrow scope with reversible effects, complete logs, a reachable person, and immediate stopping. A favorable average does not automatically authorize expansion.

Phase 5 - Exit and review

Export and reopen a case with data, configurations, and permissions outside the primary service. Give every authorization an expiration date and thresholds for suspension and return. Review when the model, data, provider, scale, task, or responsible people change, or at the stated deadline. Ensure that affected people know the Mandate and can challenge the outcome. Examine failures and costs before expanding the scope.

6. Conditions specific to two-system review

Condition	Required test
Effective independence	Demonstrate separation or relevant differences in sources, data, objectives, and logs. A different name, model, or provider is not enough.
Correlated errors	Look for shared defects: common sources, identical annotations and metrics, common dependencies, incomplete data, conflicting instructions, overload, and unavailability. Agreement can still be wrong in the same way.
Decisive divergence	Define in advance which disagreement may change the outcome; keep inputs, versions, outputs, and logs separate. Do not average scores or confidence to turn disagreement into authorization.
Predefined stopping	If a decisive divergence cannot be reconstructed within the control window, apply the written consequence: narrow the Mandate, move to degraded mode, Do not delegate before launch, or Stop active use.

7. Minimum log

Log field	Content to retain
Identification	Case, date, owner, version A, version B, configurations, and sources.
Separate inputs	Data available to each system, excluded elements, and transformations applied.
Separate outputs	Answers, cited sources, uncertainty, alternatives, and available intermediate steps.
Agreement or divergence	The substantive point, not a stylistic difference; possible effect on the decision.
Human reconstruction	Decisive fact, rule, alternative, time required, and competence needed.
Consequence	Continue in shadow mode, repeat, narrow the Mandate, switch to degraded mode, Do not delegate, or Stop.
Cost	Hours, delays, training, challenges, dependency, energy, and exit work.

8. Decision rules

Situation	Minimum consequence
A and B agree but share an unverified dependency	Evidence is Unresolved; do not expand the Mandate.
A and B diverge on a non-decisive point and the reason is reconstructable	Record it; retain the scope; review the case library.
A decisive divergence is reconstructable within the control window	Send it to the competent reviewer; the effect remains suspended until the case is closed.
A decisive divergence cannot be reconstructed in time	Narrow the Mandate or move to degraded mode; if the power is already active, Stop.
Both systems fail on the same case	Treat agreement as a correlated error; correct sources, data, objectives, or process before repeating.
Load makes oversight nominal	Reduce volume or mode; fund human capacity; do not add an ornamental signature.
Exit does not reconstruct the service	Blocking condition: Do not delegate before launch or Stop active use.

9. Closing record

OUTCOME Record the version and tested scope, including failed cases, divergences, and unresolved conditions. Retain oversight hours and costs alongside the organizational decision. Name the responsible person and specify the stop condition and review date. A good result does not authorize untested tasks, data, or volumes.

Closing checklist

Control	Outcome and reference
☐ Versions A and B identified	
☐ Common dependencies documented	
☐ Decisive divergences reconstructed	
☐ Actual load and control window tested	
☐ Redress and stop authority reachable	
☐ Degraded mode and exit tested	
☐ Oversight and nonadoption costs recorded	
Organizational decision	☐ Proceed ☐ Limit and test ☐ Do not delegate ☐ Stop
Responsible person, signature, date, and review	