IXC

Automated test failure analysis, quarantine and return-to-service worksheets

From first failure to investigation, temporary verification and evidence for return

A practical IXC resource for adoption decisions and delivery

Published 2026.09 · English edition · PDF + Excel worksheets

Using this resource

Separates the causes of automated test failures from unconfirmed risks, and records owners and return conditions.

QA leads investigating CI test failures, developers and staff taking over automated test assets

Contents

01 Record the first failure as an incident 3

02 Separate observation from cause assessment 4

03 Compare failures and corrections in our own example 5

04 Set verification for the quarantine period 6

05 Decide the return on evidence of changed conditions 7

06 Hand over recurring causes and maintenance work 8

How to complete and reuse it

Read the criteria in this PDF, then fill the blank fields with your actual conditions and anything still unconfirmed. The Excel file for the same practice area has a worksheet for each resource. Keep the worked examples separate from real customer cases and confirmed quotes.

With attribution, you may quote and adapt this material for internal review and your own work. The latest files are at www.ixc.co.kr/en/assets.

Record the first failure as an incident

Keep the record of the first failure even if a retry changes the result. Recording only the test name mixes in results from other builds or other data. Under the incident identifier, link the code and test revisions, environment, starting data, expected result and observed result.

Initial failure incident record

Item	What to record	Evidence and references
Incident and test identifiers		
Product and test revisions		
Environment and execution settings		
Starting data and expected result		
First observation and execution identifier		
Retry identifiers and changed conditions		

> **Keep three kinds of status apart**
>
> An execution's PASS, FAIL, BLOCKED or NOT RUN; an incident's received, investigating, actioned or reviewed status; and whether an automated test asset is in use, quarantined or returned to service are separate records. Being under investigation is no reason to erase an execution failure or to count a quarantined test as a pass.

Link screenshots, traces and logs by execution identifier. Do not attach account passwords or raw personal data to the record; set an evidence location the team can open, along with its access scope. If the retry count, data reset or execution order changed, first check whether it was a repeat under the same conditions.

Separate observation from cause assessment

Do not settle a failure's cause from the last error line in the log alone. Compare the agreed requirement with the observed result, then check the product code, the test's expected value, data preparation and the environment in turn. For each hypothesis, record the evidence confirmed and the causes not yet ruled out.

Candidate cause	Check first	Next check
Product defect	Gap between the requirement and actual behavior	Verify the product correction with the same input and test
Test defect	Expected value, locator and observation timing	Correct the test while preserving the requirement
Data or environment	Shared keys, resets, versions and integrations	Control the suspected condition and reproduce
Unresolved	Observed, but evidence of cause is insufficient	Assign an owner, further observation and a check time

Cause investigation record

Incident and hypothesis	Supporting or excluding evidence	Next investigation	Owner and review date
Worked example: F-DATA / shared key	Two setup operations overwrote the same key	Check the same sequence with per-run keys	Data preparation owner assigned

If you correct the product and the test at the same time, it is hard to tell which change resolved the failure. Where practical, change one condition at a time and keep before-and-after records. A cause you could not narrow down is also a result. If there is evidence of a problem users experience, do not close a product defect on the hypothesis of a test problem alone.

Compare failures and corrections in our own example

The following is a local Node.js synthetic fixture that IXC ran to prepare this material. Three causes were deliberately introduced, and each first failure was compared with the result after correction. It did not verify a customer product, a browser or operational CI, and it did not measure real-world failure frequency.

Incident	First observation	Confirmed correction	Observation after correction
F-PRODUCT	Viewer role allowed to edit	Corrected the product's role condition	3 PASS runs with the same expected value
F-DATA	Another setup operation overwrote the owner	Separated data keys per run	3 PASS runs in the same sequence
F-TEST	2 active records, but the expected value was all 3	Corrected the expected value to match the active-count requirement	3 PASS runs with no product change

> **Do not close a cause on a successful retry alone**
>
> F-DATA was PASS on a retry with clean data but FAIL when the original A-write → B-write → A-read sequence was run again. After the keys were separated per run, the same sequence was checked. This is not the result of testing a real concurrent scheduler.

The three checks after correction are an observation record for this example. They are not a stability threshold for every automated test asset. In practice, confirm that the affected environment, data and failure conditions can be run again, and have the return owner review the agreed evidence.

Set verification for the quarantine period

Quarantining an automated test asset does not mean verification of that risk is complete. Record which execution stage it leaves, and what will be checked manually or at another layer in the meantime. Without substitute verification, report it to the release owner as unverified scope.

Quarantine and substitute verification agreement

Record item	What was agreed	Owner and evidence
Quarantined test and incident identifier		
Executions it leaves and release impact		
Reason for quarantine and remaining hypotheses		
Substitute verification and when it runs		
Correction owner and next review date		
Conditions for return or closure		

> **Worked example: data collision under investigation**
>
> If a test that uses a shared key has been separated out for investigation, record the owner who will check that flow with independent data, and the execution identifier. This substitute check does not prove the stability of the entire parallel run. Record the conditions not yet checked separately.

If the only goal is fewer quarantined tests, the number can be met by deleting tests. Review together the reason for quarantine, elapsed time, the presence of substitute verification and unresolved risk. Even if you decide to stop automating a test, agree who will continue that verification and how.

Decide the return on evidence of changed conditions

A return review reads the cause hypothesis, the correction and the revalidation of the original failure conditions together. If the test code was corrected, check that it was not changed to verify less of the requirement. If the product was corrected, check whether it can be confirmed with the test's expected result kept as it was.

Return review record

Incident and correction identifiers	Revalidation conditions and execution	Check result and remaining limitations	Reviewer and decision
Worked example: F-TEST / t1	2 active and 1 inactive records kept	Requirement counts active records; product unchanged	Example observation / not an actual approval

Check item	Evidence to keep before return
Original failure conditions	Record of running the same conditions again
Impact of the correction	Impact on other scenarios, and how it was checked
What the verification means	Reasons for changing the requirement, expected value or observation method
If it fails again	Owner, intake path and location of investigation materials

Set the repetition count based on product risk and execution variation. Meeting the count while leaving out the data or execution order that caused the original failure is not sufficient evidence. Keep return approval separate from release approval, and give the release meeting the results of what actually ran on that build and the conditions left unverified.

Hand over recurring causes and maintenance work

In the monthly review, group different failure incidents by cause and decide the improvement work. As executions or retries increase, analysis time changes too. Align first failures, retries and the time spent on investigation and correction to the same period, and do not mix them with product defect counts.

Maintenance work and handover record

Recurring cause and incidents	Corrective and preventive work	Owner and next check	Execution and document locations
Worked example: shared data collision	Correct the per-run key rule and cleanup procedure	Owner assigned; review at next execution	Link setup code and execution logs

> **What to prepare before a consultation**
>
> Gather representative failure incidents, code and test revisions, how CI currently runs, data preparation and the maintenance owner. If you choose logs that can be shared without secrets, you can agree on a concrete scope for failure analysis and automation maintenance with IXC.

Tracking the time spent classifying failures, correcting scripts and preparing environments separately lets you explain what to improve in the next budget. Do not conclude that a higher pass rate after retries means better product quality or lower cost. The test assets and the investigation procedure must both stay with the next owner.

IXC

Publication and use

Based on representative execution records, your environment and data preparation method, and your internal maintenance owner, we set the scope of failure investigation, correction and handover with you.

IXC Corp.

CEO Nam Kichul | Business registration no. 109-86-56732
4F 408, Chuncheon ICT Venture Center
7 Huseok-ro 462beon-gil, Chuncheon-si, Gangwon-do, Republic of Korea
+82-33-242-0210 | info@ixc.co.kr

www.ixc.co.kr

www.ixc.co.kr/en/services/test-automation

www.ixc.co.kr/en/assets/test-automation-failure-log

References

Playwright Retries: distinguishing retries and execution states

https://playwright.dev/docs/test-retries

Playwright Best Practices: test isolation and data control

https://playwright.dev/docs/best-practices

IXC: What should test automation take first, and what stays with people?

https://www.ixc.co.kr/en/insights/test-automation-portfolio

IXC Test Automation

https://www.ixc.co.kr/en/services/test-automation