EXAMPLE 05

# An unfamiliar application. A focused repair. A checked extension.

Support receives an order that should have been rejected. A fresh AI session examines an unfamiliar 11-node application, explains the wrong shipping calculation and asks for approval. The team repairs it, checks the boundaries and extends the process without discarding the original evidence.

**11 → 12 processing nodes · 85 repair assertions · 80 extension assertions · 45 independent boundary assertions**

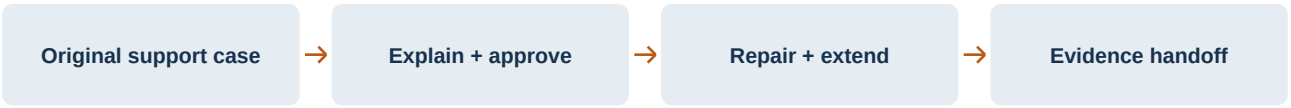



Illustration of the demonstrated workflow.

## The problem

A £105 order receives a 10% discount. Merchandise is now £94.50, below the £100 free-shipping threshold. But the application charges no shipping and releases the order against a £97 credit limit. Which rule caused that result, and can it be changed without breaking the rest of the process?

## What changed

The approved repair compares the shipping threshold with discounted merchandise: £94.50 + £5 shipping = £99.50, so the order is rejected against a £97 limit. That repair leaves the generated topology unchanged. The subsequent candidate adds a HandlingCharge node and a HandlingFeeUpdate event on the existing Policy node, producing a new 12-node graph.

MEASURED RESULTS

## Observed result

The approved repair compares the shipping threshold with discounted merchandise: £94.50 + £5 shipping = £99.50, so the order is rejected against a £97 limit. That repair leaves the generated topology unchanged. The subsequent candidate adds a HandlingCharge node and a HandlingFeeUpdate event on the existing Policy node, producing a new 12-node graph.

| Case or component | Recorded result |
|---|---|
| Original recorded decision | £94.50 total; RELEASED — inconsistent with the shipping requirement |
| Approved repair | £99.50 total; REJECTED_CREDIT |
| Repair checks | 85 assertions pass; the original fails the shipping regression |
| Independent boundary checks | 45 assertions pass |
| Extended graph | 12 processing nodes; four business event types |
| Extension capture | 9 order attempts; 10 updates; 8 publications; 23 audit records |
| Failure and recovery | Overflow publishes nothing; fee reset and the next order succeed on the same instance |

A synthetic engineering demonstration, with actual runs and saved investigations. Not a customer incident or a production-readiness claim.

Open the extended application investigation

Download the matching investigation bundle

To reopen the investigation in the Analyser, save the HTML reel and bundle in the same folder.

Three recorded model task prompts; actual owner approval for the shipping repair. CLI default model and effort not captured. Original MCP calls blocked; the supervising assistant separately used the Analyser action API.

Read the recorded diagnosis and approval

FOLLOW THE DEMONSTRATION

# 5-stage walkthrough

## 1. Open the support case

Start with the reported decision and its original investigation bundle.

## 2. Explain the mismatch

Connect £94.50 merchandise, missing shipping and the £97 limit to the Shipping.java comparison.

## 3. Approve the repair

The human authorizes the focused change and regression checks.

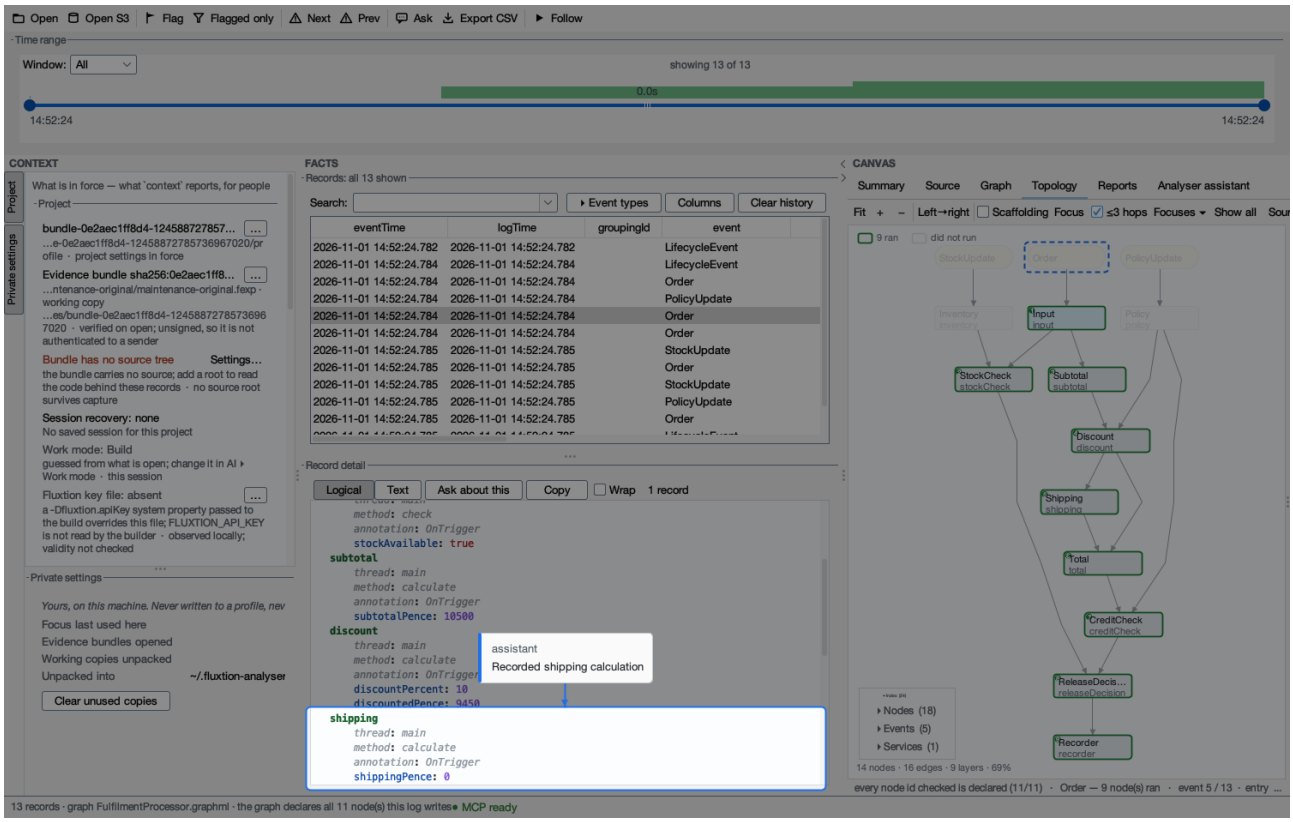
## 4. Extend and test

Add one handling-charge node and one event handler, then test boundaries, non-propagating updates and recovery.

## 5. Return the evidence

Inspect separate original, repaired and extended investigations with the source and tests.

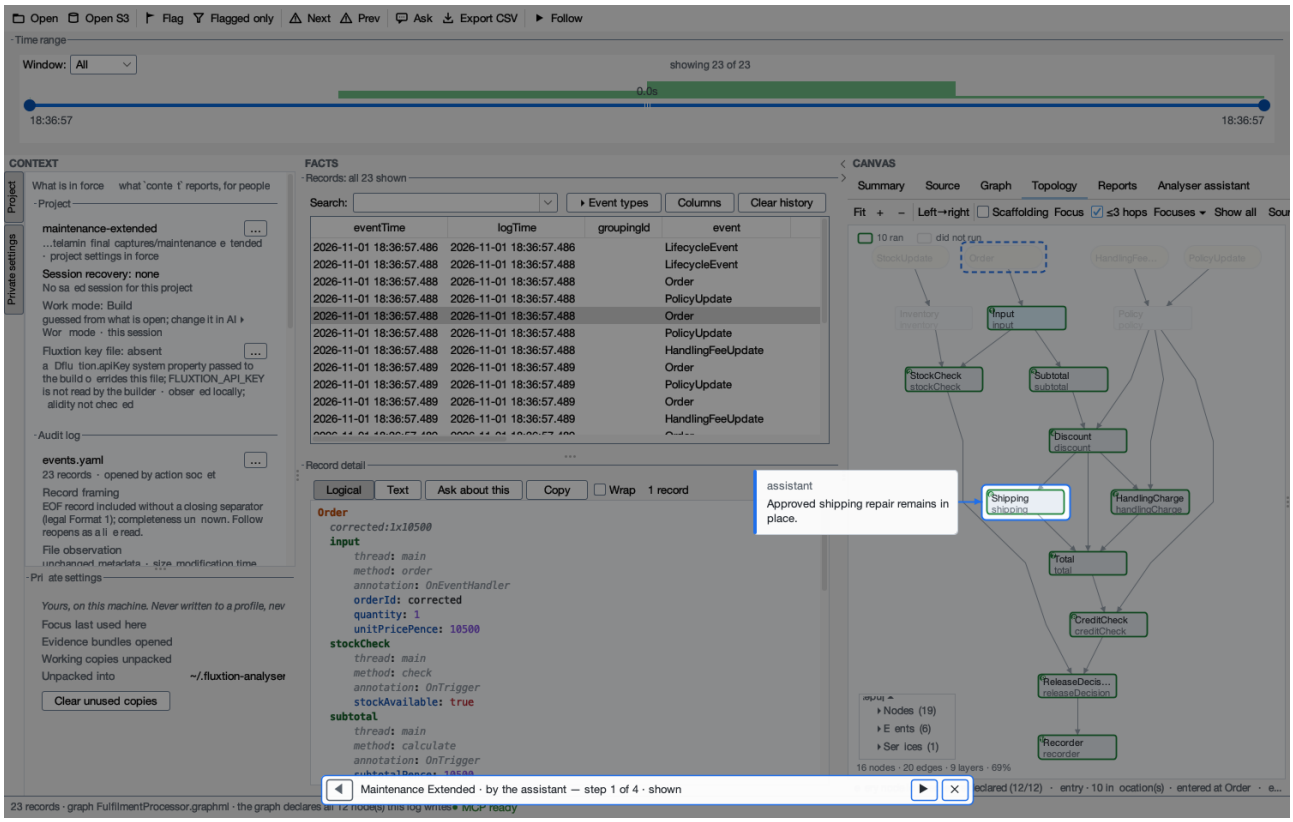
Original captured record: discountedPence 9450, shippingPence 0, outcome RELEASED. The shipping node detail is visible; the outcome is recorded in the same audit event.




Open the full-resolution screenshot

### VERIFY THE CHANGE

Extended application: the graph includes HandlingCharge beside Shipping; the approved repair remains in place.




Open the full-resolution screenshot

The original and extended screenshots are genuine Analyser captures. The smaller repair changes one rule without changing topology; the new handling node changes the graph. The extension remains a candidate for local owner review.

An evidence bundle is a portable investigation created by the Analyser from selected records, the graph and a saved walkthrough. A person, optionally with an AI assistant, guides the investigation. The bundle can be reopened; the reel and PDF are for reading and sharing.

What is an evidence bundle?

## Why this matters

The team can see both the small repair and the larger structural change, inspect why each result changed, and hand the same context to another reviewer. This demonstration measures checks and observed outcomes; it does not establish diagnostic time or cost savings.

## Apply it to your process

Bring one representative process, its expected outcomes and failure cases. Agree the checks and evidence before evaluating a candidate change; finish with an adopt, extend or stop decision.

Discuss a five-stage pilot

TECHNICAL APPENDIX

## Reproduce this result

Unzip the package and run from its complex-maintenance directory:

```
python3 run.py \
  --runtime /path/to/fluxtion-runtime-1.1.3.jar \
  --out results
```

The runner verifies the public runtime checksum, compiles the supplied generated Java and executes the checks locally. No generation credit, model request or network service is used. Use a new output directory.

Source and reproduction package for this example

Generated orchestration is included for offline execution. No hosted compiler is called by reproduction. Changing topology requires regeneration.

Runtime prerequisite: com.telamin.fluxtion:fluxtion-runtime:1.1.3. Download the JAR without an account; the runner checks its SHA-256. The source ZIP includes RUNTIME-PREREQUISITES.txt with the complete download command.

Download runtime JAR | Published POM: declares GNU AGPL version 3, separate from Apache 2.0 demo code.

SHA-256: 2bd45eb62409403422d5a676a7ad177f1984aa80dc399c20ceb6b467eab4b775

## Detailed evidence

summary.json

commands.json

conversation.json

Extended candidate investigation bundle

Approved repair investigation bundle

Original intake investigation bundle

All exported PDFs, screenshots and verification receipts

## Scope and interpretation

Synthetic unfamiliar-code exercise, not a real customer incident or a formal blinded study. The creator knew the fault; the fresh diagnosis session was not given its oracle.

The original session’s MCP access was blocked. Analyser inspection and exports were performed separately by the supervising assistant through the local action API; no successful fresh-model MCP workflow is claimed.

Handling charge is a proposed local extension, not an owner-approved business rule. Human approval in the transcript applies to the shipping repair and tests.

The failure/recovery checks cover this synchronous fixture, not transaction rollback, arbitrary failures or production resilience.

Three model task prompts and actual visible messages are available; CLI default model/effort were not recorded. No token, cost or model-level comparison is claimed.

Bundles contain selected logs, matching graphs and saved investigations. They are unsigned and do not carry the complete source or execute a replay.

Example code is released under Apache 2.0. Runtime, third-party and general customer-output terms remain separate. Source and artifact hashes compare bytes; they do not authenticate a sender or certify correctness.

Deployment and data flows | Applicable product rights