KPMG

AI-Assisted Model Validation

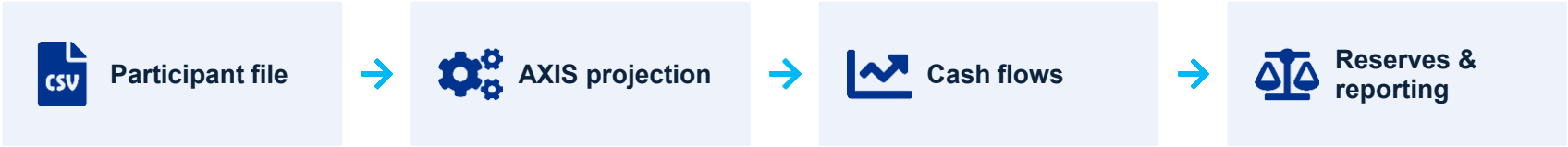
Automated input and output controls for a pension risk transfer block

SEAC | Jonathan Roy, Director, Actuarial Services | Illustrative example; no client data used

THE PROBLEM

Every quarter, a model run depends on two things being right

A pension risk transfer (PRT) block: a participant file goes into AXIS, and the results drive reserves and reporting.



1. The input file is wrong

- Duplicate people, missing birth dates
- A deceased participant still being paid
- A benefit that jumped 10x
- AXIS will happily project bad data

2. The model mishandles a good file

- A policy is missing from the output
- A terminated participant is still projected
- Totals don't add up
- Cash flows don't match an independent recalculation

THE SOLUTION

One click: pick the files, run, get a pass / fail dashboard

23 automated controls in three groups, each with a plain-English explanation of what failed and why.

13 Input checks

Is the participant file clean?

- File layout and required fields
- No duplicates, valid codes and dates
- Benefits plausible and stable vs last quarter
- Participant counts roll forward

8 Input-to-output checks

Does AXIS match the file?

- Every in-force policy is in the output
- No terminated policy is projected
- Totals equal the sum of policies
- Deaths and retirements are reasonable

2 Independent recalculation

Does AXIS get the right answer?

- Policies recalculated in validated Excel replicators
- Compared with AXIS month by month
- Sample run live; full block from stored results

VS Code + AI: I describe the rule, AI writes the code

Actuarial knowledge stays with the actuary; AI handles the programming.



Describe
Write the rule in plain English

Generate
AI writes the Python code in VS Code

Test
Run against known results from the Excel replicators

Review
Read, challenge and approve the code

Repeat in minutes, e.g. "a lump sum is not allowed once the pension has started"

Bridges the skills gap
Deep actuarial knowledge, limited coding? Describe it, get working code.

Saves time
Days of specialist work compressed into a much shorter cycle.

Fast changes
New rules or layout changes made and re-tested in minutes.

Documentation included
Help text and user guide drafted quickly, then reviewed.

KEY PRINCIPLE

AI builds the tool. Code checks. People own the results.

Actuarial work needs deterministic results. AI helps us design the program; the program, not AI, runs the checks.

Why deterministic code matters

Repeatable: same file, same answer, every time

Exact: reconciles to the cent across thousands of numbers

Auditable: each rule is short, readable, with a stated tolerance

Independent: validated replicators give a true second opinion

Secure: participant data never leaves the machine

Controlled: a rule only changes when someone approves it

Where AI adds the most value

Writing code: Python, Excel automation, file handling

Building the dashboard: charts and pass / fail screen

Creating test data: synthetic participants and model output

Iterating: turning feedback into changes quickly

Documenting: help text and user guide

LIVE DEMO

Three scenarios, one dashboard

Synthetic PRT block (~100 policies), two quarters: 6/30/2026 and 9/30/2026.

1

Input data failure

The 9/30 participant file has problems

7 red controls

Bad policies are set aside, so the remaining checks still run

2

Everything ties

Clean inputs and a matching AXIS output

23 green controls

What a clean quarter looks like

3

Input / output disconnect

Clean inputs, but AXIS disagrees with the replicators

5 red controls

Wrong payout, double-counted expense, missing and terminated policies

Live demo

PRT Validation Studio: from file selection to pass / fail dashboard

Three scenarios | Illustrative data | Questions welcome along the way