INVESTPUPPY UNVARNISHED

InvestPuppy

Life after live

*For anyone else who's been in **that** room.*

Serious about outcomes. Not serious about ourselves.

About InvestPuppy

InvestPuppy builds Vektor, a systematic portfolio management platform for wealth management professionals.

What this series is

Every experienced practitioner thinks it. Almost nobody publishes it: *there has to be a better way.*

We thought it long enough that we went and built one.

Before that, we spent decades in the rooms.

The same project was failing for the third time. The project plan had been signed off before anyone had understood the first requirement.

The relationship was too warm for anyone to say so.

It isn't incompetence.

It's the incentive structure doing its job perfectly.

Nobody planned it this way. Nobody needed to.

We got frustrated enough to write it down. Then frustrated enough to build something. These papers are the writing-it-down part.

The something we built is a separate conversation.

The system went live.

> Go-live is the end of the project. It says so in the project plan — the final milestone, the one the Gantt chart terminates at. It is not the end of anything. It is the beginning of a different, harder problem: the problem of an organisation built around a legacy system encountering, for the first time at production volume, the system that was supposed to replace it.

The implementation did not.

1. The Mythology of Go-Live

The go-live date functions as a destination. Everything before it is a journey. The kick-off is the departure. The UAT phase is the approach. Go-live is arrival. This mythology is maintained because it is useful: it gives the project a shape, gives the steering committee something to track, gives the vendor a milestone against which payment is triggered.

Go-live day is the first day the system is used in production, at volume, by the people who were not in the UAT environment, on the data that was not in the test dataset, for the transactions that were not in the test scripts. It is the first real test of the implementation. The project plan treated it as the finish line. It is the starting gun.

2. The Hypercare Fiction

Most project plans include a hypercare phase — a period of intensified support immediately following go-live, typically two to four weeks. Hypercare ends when the project ends. The issues do not.

The issues that surface during hypercare are the obvious ones. The project team departs. Hypercare is logged as complete.

The issues that surface after hypercare are the real ones. The reporting that does not match the legacy system output. The workflow exception that occurs twice a month. The regulatory reporting failures that emerge when the first post-live reporting period closes. These are not merely operational problems. They are reportable events. They arrive after the project manager's engagement has ended.

3. The Workaround Economy

Every organisation that has been using a legacy financial system for more than five years has built a workaround economy around it. Spreadsheets that compensate for reporting limitations. Manual processes that handle transactions the system cannot. Institutional knowledge held by specific individuals.

> *The system that was built for the specification meets the organisation that has been using workarounds for eighteen months. Between them is a gap.*
>
> *The gap is filled initially by people who know the old workarounds and have not yet learned the new ones. Then it is filled by new workarounds — built on top of the new system, invisible to the vendor, undocumented, dependent on the continued employment of the people who invented them.*
>
> ***The system went live. The implementation did not.***

4. What Comes After Live

The post-live phase is not a support problem. It is an adoption problem, a change management problem, a configuration problem, and a data quality problem.

A post-live plan adequate to the actual challenge has five components:

One. A structured process for capturing and resolving configuration issues that only appear under production load.

Two. An adoption tracking mechanism that measures not go-live usage, but correct usage.

Three. A workaround audit conducted sixty days after go-live, identifying manual processes that have emerged around the system.

Four. A regulatory reporting validation conducted at the close of the first full reporting period post-live.

Five. A data quality review at ninety days, measuring mandate compliance, data completeness, and the accuracy of the migrated data against the source.

Plan for what comes after live.

It is longer than the project.

The Unvarnished Series

Thirteen field notes on why financial software implementations fail.
Published under our own name. All freely available, no registration required.

Why Implementations Fail (UNV-01)
The same implementation failures repeat across the industry, unrecorded, because no party benefits from writing them down. This paper names the pattern openly, once, rather than let it repeat quietly again.

Nobody Owns It (UNV-02)
Ask a vendor who owns an implementation and they say the client. Ask the client and they say the vendor -- both are right, and that shared, undefined ownership is the actual failure mode, not any single decision either side made.

Complexity as Cover (UNV-03)
A consultant's real product is often the document justifying the advice, not the advice itself -- built to survive the project's failure, not prevent it. Complexity becomes the cover story for that gap.

Plan Last (UNV-04)
Project plans are written before anyone understands the workflow they claim to schedule, in columns spanning weeks one through thirty-six. Planning first and discovering later gets the sequence backwards.

The Change Request Trap (UNV-05)
A change request is a formal admission that the original specification was wrong, and also a commercial mechanism for charging separately for what should have been included from the start. Neither party is surprised when it happens.

Invisible Stakeholders (UNV-06)
Every implementation has a second set of stakeholders who never attended a workshop or reviewed a specification, yet are the ones who have to live with what was built. Their absence from the room doesn't make them absent from the consequences.

The Hammer, the Tool, and the Methodology (UNV-07)
A tool gets picked up, used for the job it suits, and put down. A methodology requires certification, governance, and a steering committee -- and the industry keeps reaching for the second when it only ever needed the first.

Life After Live (UNV-08) (you are here)
Go-live is the final milestone on the Gantt chart, but it's the beginning of the harder problem: an organisation built around a legacy system now has to actually live inside the new one. The project plan ends; the real work doesn't.

The U in UAT (UNV-09)
UAT sign-off documents carry the right signatures -- from consultants testing scripts written by other consultants, not from anyone who actually uses the system day to day. The U in UAT is the part everyone skips.

Nine Women (UNV-10)
Fred Brooks named it in 1975 and the same boardroom conversation still happens fifty years later: a project is late, so the proposal is more people. The logic feels intuitive and has always been wrong.

The AI Revolution — Automating Bad Practice at Scale (UNV-11)
AI doesn't fix a broken process -- it executes it faster, at greater scale, with fewer chances for the human hesitation that might have caught the problem. The real question isn't whether an AI implementation will work, but what it will end up working very hard to get wrong.

Model Bank — the perfect solution for banks that don't exist (UNV-12)
A 'model bank' configuration is a composite of institutions that don't exist, built from someone else's defaults before your institution was ever part of the conversation. Every subsequent customisation is just correcting an assumption that was never really about you.

The parameter that broke the camel's back (UNV-13)
A platform demo answers every question with yes, right up until one specific parameter turns out to be the one nobody actually tested. This paper is the account of exactly where that line sits, and what breaks on the other side of it.

Our Better Way

This is what we built instead.

VEKTOR by INVESTPUPPY

"So, what did you do about it?" — a fair question after thirteen papers on what goes wrong. InvestPuppy builds Vektor, a systematic portfolio management platform for wealth management professionals — the direct answer, series by series.

These four papers are the how to the thirteen whys above — not a sales pitch, but a technical account of the specific choices Vektor made where the industry's default pattern would have produced the same failures just described.

– Technical Debt (BW-01)
– Vektor House Implementation (BW-02)
– The Configurability Gap (BW-03)
– AI as Instrument (BW-04)

More at investpuppy.com