FUTURE ALPHA TECHX · 31 MARCH 2026

What Good Looks Like for Front-Office Engineering

Generate Alpha. Sooner, Safer, Stronger.

A practitioner view for front-office engineers, heads of execution systems, quant-platform leads and buy-side technologists.

Alan Russell · Buy-side & sell-side front-office engineering · 20 years

Everyone's Out By One

Are your teams out by one in the responsibility array?

Quant Researcher

HIRED TO

Build models.
Test hypotheses.
Generate alpha.

ACTUALLY DOING

Writing Python ETL.
Cleaning data.
Waiting for compute.

→

Quant Developer

HIRED TO

Build research infrastructure.
Accelerate quants.

ACTUALLY DOING

Writing infra-as-code.
Managing cloud configs.
Fighting cost alerts.

→

SRE / DevOps

HIRED TO

Reliability.
Observability.
Design out failure.

ACTUALLY DOING

Fighting production fires caused by under-engineered paths.

→

Platform / Infra

HIRED TO

Build the platform.
Own the roadmap.
Serve the customer.

ACTUALLY DOING

Servicing ad-hoc demand. Running a ticket queue.

Nobody is lazy. Nobody is stupid. The system keeps pushing scarce talent into work that should already be solved.

When did your best quant last spend a whole week on pure research?

The Assumption Mismatch

Netflix is a delivery truck. A hedge fund is a Formula 1 car.

THE DELIVERY TRUCK

Netflix · Spotify · Amazon

- Known route, known cargo
- Same road every day
- Scale is the problem
- Optimise for throughput

Cache it. Distribute it. Standardise it.

Brilliant engineering for a known problem at enormous scale. Glorified as best practice. But the problem is known.

FORMULA 1

Hedge funds · Front-office engineering

- Different track every race
- Conditions change mid-lap
- Speed of adaptation is the edge
- Opportunity decays before you scale

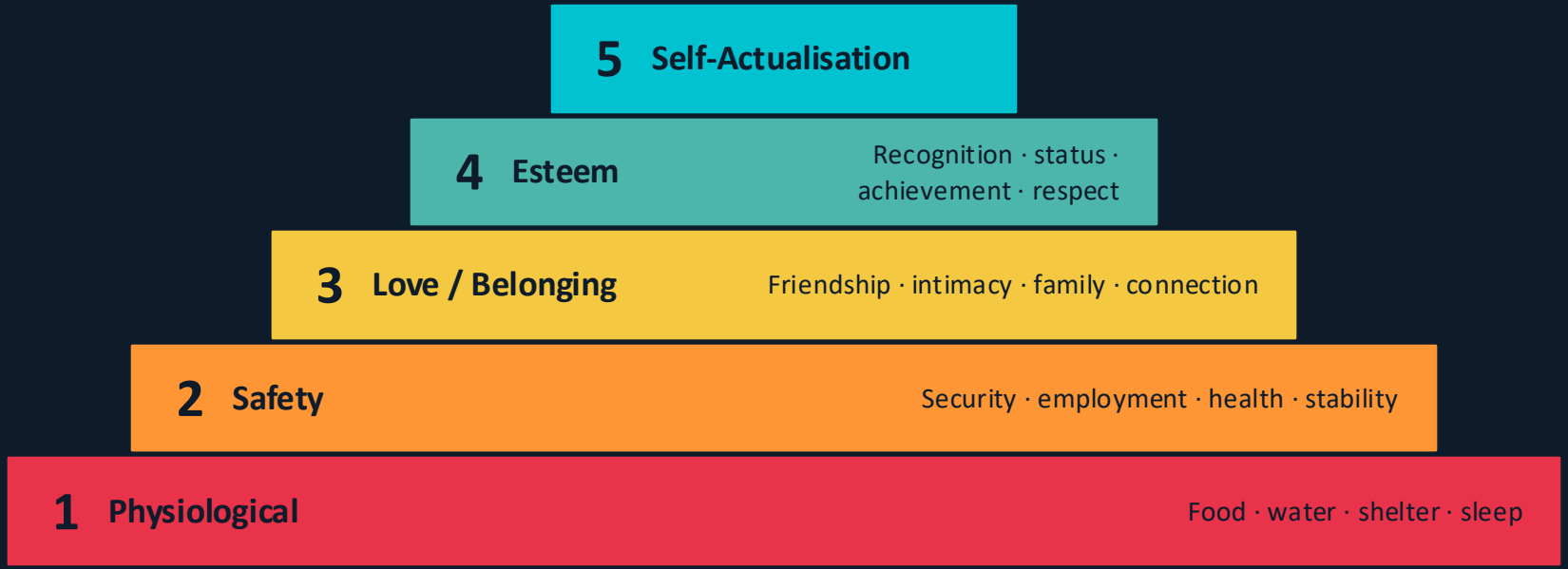
Iterate fast. Fail safe. Learn before the market moves.

You cannot win a race by optimising the delivery schedule. Different vehicle. Different engineering. Different KPIs.

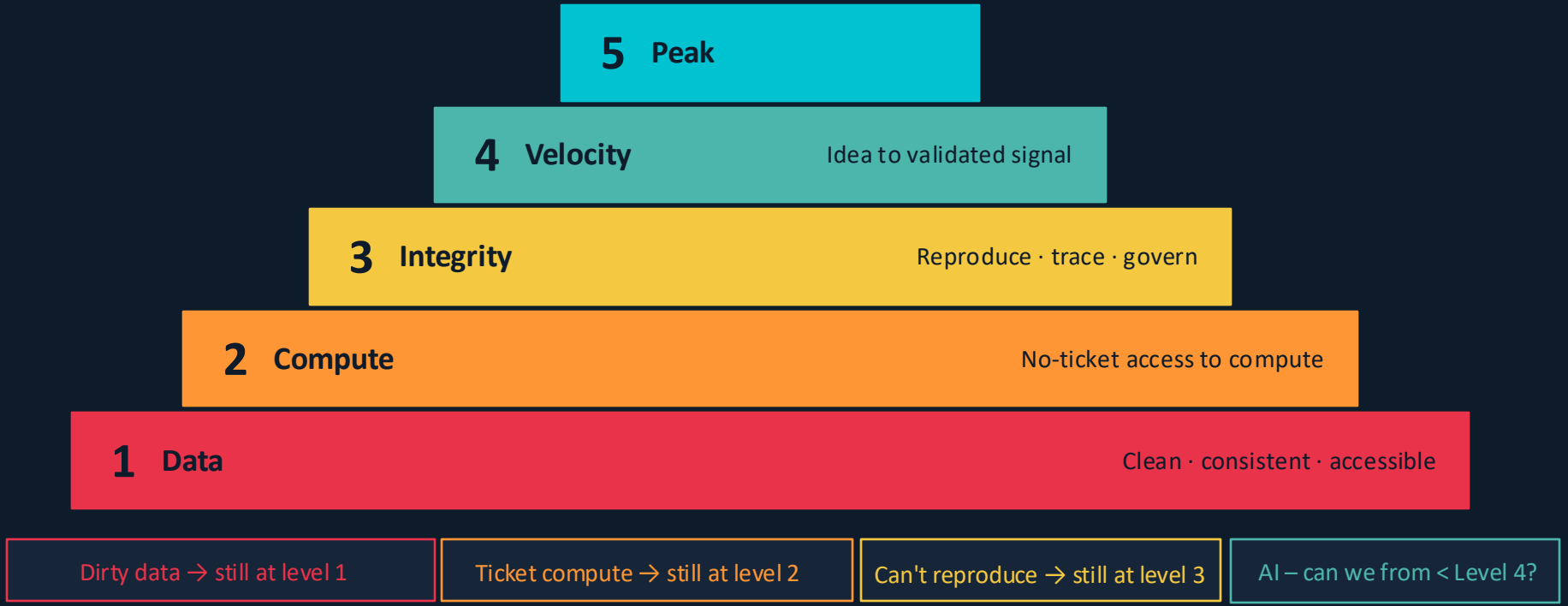
Cloud is not the mistake. Treating a volatility problem like a scale problem is the mistake.

Same tools. Different design centre.

Maslow's Hierarchy of Needs



Quant Research and Portfolio Manager Needs



SECTION 4

Do we measure the wrong KPIs?

Temptation: measuring what is easy to count. Not what counts.

WHAT GETS REPORTED

AI adoption: % of Users, Token Cost
Sounds modern

Models deployed
Visible, countable

Compute consumed
Feels like scale

Sprint velocity
Conflates pace with direction

Feels fluent. Feels correct. Feels green.

WHAT IT HIDES

- Quants waiting weeks for compute
- PM decisions no better at 8 a.m.
- Reporting optimised for reassurance
- High performers quietly disengaging
- Lead time unmeasured or unknown

The failure cascade:

Delivery hard → reporting heavier → narratives cleaner → uncertainty removed → decisions feel confident → outcomes worsen

THE STEERING METRIC

Lead Time

Shortest safe path from idea to production-validated model

- Surfaces every queue and handoff
- Cannot be gamed without fixing the system
- Forces the whole org to look at itself

SCORECARD

Revenue per quant

Flat at major banks for 20 years despite trillions in tech spend

Lead-time is the steering metric. Revenue per quant is the scorecard.

Does the QR / PM have a better week? If not, it is activity — not progress.

The Porsche Problem

The quant arrives Monday morning with a live idea. What happens next?

ACCUMULATED INFRASTRUCTURE

The Monday morning experience:

- Compute exists but needs a ticket and a queue
- Data pipelines exist but nobody told you which
- Libraries exist but are undocumented or stale
- Approval gates nobody mapped for you
- Capability hidden — known to builders, invisible to users

Efficient for the builders.
Friction for the user.
Nobody owns the gap.

PLATFORM AS PRODUCT

The quant gets in and drives:

Ownership
Someone owns the end-to-end experience

Roadmap
Shaped by customer pain, not team backlogs

Onboarding
Day-one path to first model run

Self-service
Compute, data, deploy — no / low tickets

Feedback loop
The quant's friction is the backlog

Not: is the platform up?
But: are Quant Research shipping models?

Do not make the internal customer assemble the car themselves.

Five Pillars of Platform Velocity

01 Iteration Velocity

Research to production lead time

No engineer in the model promotion loop.
Quant pushes — it deploys.

Sub-30 min param changes
Sub-4 hrs new models

02 Operational Confidence

Front office trusts the system

One failure during a market event loses trust.
It will not come back quickly.

99.99% uptime during market hours
= 52 minutes of downtime per year

03 Ingestion Resilience

Feeds break. Plans don't.

Your dashboard loads. The data is four days old.
The PM finds out in a credit committee review.

Schema version tolerance
Staleness alerts · DLQ + replay

04 Validation Speed

Remove the engineer from the critical path

Quant pushes config. CI validates.
Pipeline deploys to UAT. PM approves. Live.

10× researcher iteration rate

05 Observability

Know before they tell you

"EKS node CPU at 82%" is the wrong alert.
"Credit feed: 4 hrs stale. Last batch: 09:14." is the right one.

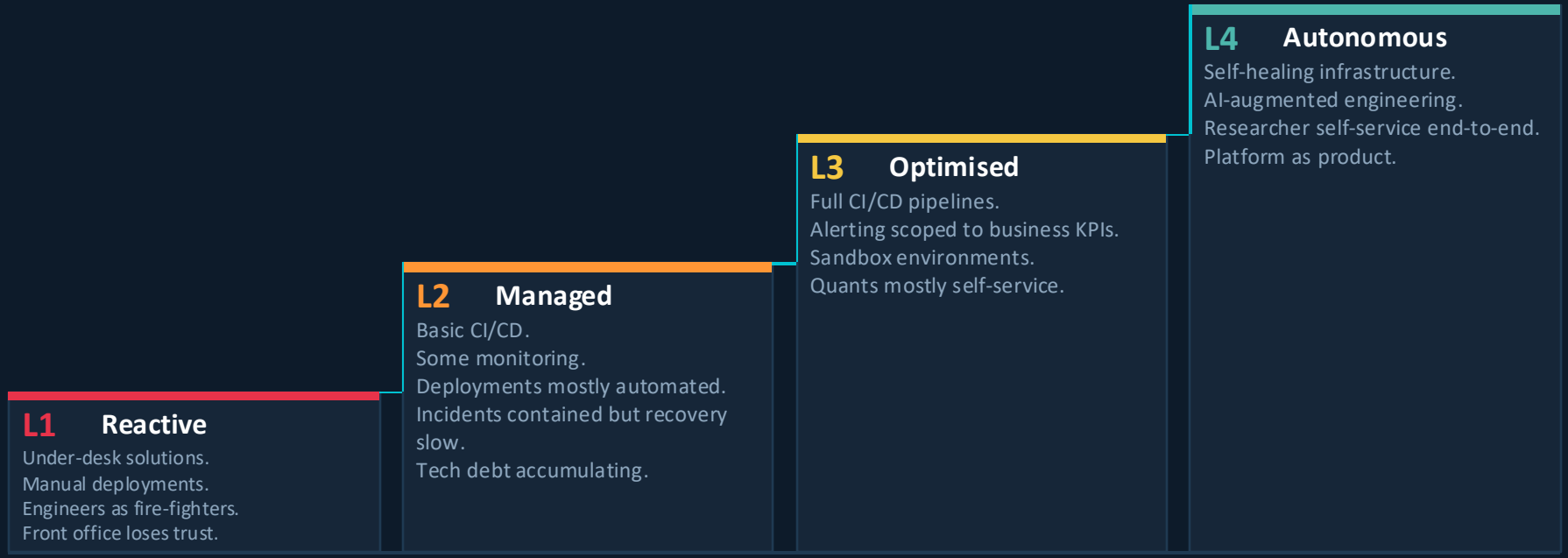
Data freshness · Pipeline success
Model version in production

If data is dirty and compute needs a ticket, you are not having an AI conversation. You are having a plumbing conversation.

Fix the pillars in order. There are no shortcuts up the stack.

MATURITY MODEL

Where Do You Place?



Many pods and engineering teams sit between L1 and L2. Elite firms are pushing L3. L4 is the emerging frontier.

Where does your team honestly sit — and what would it take to move one level?
Who is the person to elevate the platform to the next level? Do they exist yet?

Does your team consider their platform a product that you sell every day -
to your internal *customers*

...

or is it managing inherited infrastructure that has accumulated over time?

The 45-Minute Dashboard

Credit PM pod · Major macro fund · £5bn book

THE SITUATION

- PM dashboard ran on a single researcher's desktop machine
- 45-minute cold start on reboot
- No alerting, no monitoring, no disaster recovery
- Data ingestion tightly coupled to the visualisation layer
- Credit committee reviewing decisions on stale data — and didn't know it

WHAT WE DID

- Containerised with Docker, deployed on EKS
- Decoupled data ingestion into independent service
- Pre-computed intermediary tables on scheduled cron
- Schema validation on every ingestion — reject bad data, not serve it
- Alerting on data freshness, not CPU utilisation

WHAT CHANGED

45 min → 5 sec
Dashboard load time

99.9% uptime
During market hours

<30 sec auto-restart
No human intervention

Team self-sufficient
For model iteration

Credit committee
Never saw stale data again

A £5bn book was being risk-managed on a desktop machine with no alerting. Nobody had asked whether it should be.

This is what accumulated infrastructure looks like from the inside.

Six Hours to Three Minutes

Research pipeline automation · Systematic fixed income team

BEFORE — 6 hours

- Manual pipeline run every morning
- The researcher was the deployment mechanism — they ran the script
- No staging environment — changes went straight to production
- R models ran on the researcher's laptop
- No version control on model parameters — no audit trail, no rollback

AFTER — 3 minutes

- Git push triggers pipeline automatically — no human in the loop
- Staging deploys on every PR to main — changes tested before they land
- Production via release tag — Head of Research controls the promote
- Regression tests run automatically on every change — catches drift early
- Parameters versioned in git — full audit trail, instant rollback

The researcher was the single point of failure for a systematic strategy managing live capital.

Lead time dropped 120x. But the real win was removing a human from a path that should never have needed one.

The Volatility Surface Problem

Model validation velocity · Equity derivatives PM pod

THE BOTTLENECK

3 days engineering per iteration

- Researchers testing 2–3 vol model variants per month
- Each change required an engineer to deploy
- Pricing surface errors and interpolation spikes caught late — sometimes in production
- The engineer was the bottleneck, not the maths

THE UNLOCK

Self-service sandbox
Researchers deploy their own vol builder via a single parameter override. No engineer required.

Per-pod parameter overrides
Each PM pod tweaks vol fitting inputs independently. No cross-pod risk.

A/B validation dashboard
Side-by-side: current vs proposed surface. Greeks, smile quality, interpolation errors all visible before promotion.

GitOps approval flow
PM reviews in UAT. One click to approve. Release tag promotes in <5 minutes.

THE OUTCOME

2–3
iterations / month

↓

20+
iterations / month

Pricing errors caught in UAT, not production

Engineers freed from model deployment entirely

The maths was never the bottleneck. The deployment path was. Remove the engineer and the researcher iterates 10x faster.

This is what Pillar 4 looks like when you actually build it.

Source Notes

McKinsey 2024

Developer time spent coding · Banking tech spend and productivity

Broadridge 2024

Hedge funds stymied by inflexible systems

Gartner / McKinsey 2024

Global banking technology spend

Goldman Sachs 2024

Annual results — engineering productivity data

XTX / RenTech

Companies House / industry reports · Public secondary references