FROM DATA MODEL TO DATA PRODUCT

Converting Inward-Looking Models Into Outward-Looking Products — Without Spaghetti

v1.0

ACT 1 — FIRST PRINCIPLES

The Problem, The Shift, The Outcome

THE PROBLEM

Banks invest heavily in internal data models (3NF, canonical, semantic) but still have 200+ uncontrolled outbound feeds. Internal quality does not guarantee external simplicity. Spaghetti is born at the delivery layer, not the modelling layer.

THE SHIFT

Separate the internal model (how you think) from the external product (how the world interacts with you). The data model optimises finance operations. The data product optimises exchange. You need both — but separated.

THE OUTCOME

200+ custom feeds collapse to 5-8 canonical product contracts. Interface proliferation frozen. Delivery becomes structured. Speed increases because consumers use reusable interfaces, not custom feeds.

10 Core Assumptions Challenged

#	ASSUMPTION	NECESSARY?	CONCLUSION
A1	Strong internal model reduces integration	**No**	Internal quality ≠ external simplicity
A2	Canonical model should be universal	**Not physically**	Canonical alignment must be minimal
A3	Data delivery is just a layer	**No**	Delivery architecture determines cost
A4	Interfaces are technical artifacts	**No**	Interface = institutional boundary
A5	Every consumer needs custom feed	**Destructive**	Customisation creates entropy
A6	Business logic deeply embedded is fine	**No**	Hidden logic = trapped value
A7	Modelling purity improves integration	**No**	Over-optimisation distracts externally
A8	Governance slows delivery	**Only manual**	Embedded governance speeds delivery
A9	Centralisation fixes inconsistency	**No**	Control exchange, not construction
A10	Standard schema avoids spaghetti	**Not sufficient**	Schema without discipline fails

10 Irreducible Truths

- 1. Systems will exchange data
- 2. Exchange requires interface
- 3. Interfaces require agreement
- 4. Agreement requires versioning
- 5. Change is inevitable
- 6. Identity consistency is mandatory across boundaries
- 7. Traceability is required for accountability
- 8. Duplication increases complexity nonlinearly
- 9. Integration cost grows with number of unique interfaces
- 10. Delivery design determines reuse economics

Everything else is preference. These 10 truths are the only constraints.

The Clean Distinction: Data Model vs Data Product

DATA MODEL (Inward)	DATA PRODUCT (Outward)
Inward	**Outward**
Optimises finance operations	Optimises exchange
Controls internal accuracy	Controls external stability
Focuses on entities	Focuses on contracts
Concerned with normalisation	Concerned with interface consistency
Invisible externally	Publicly declared

A data model without delivery discipline creates spaghetti. A data product without internal model creates garbage. You need both — separated.

The biggest integration cost is not bad data models.

It is uncontrolled interface multiplication.

Most enterprises renovate internal data structures repeatedly while leaving the exchange plumbing chaotic.

Spaghetti is born at the delivery layer, not the modelling layer.

Reconstruct From Scratch: Two Layers

INTERNAL LAYER

(Domain Team)

- Finance canonical core
- Internal transformations
- Operational performance model

No constraints on how they do it internally

EXTERNAL LAYER

(Data Product)

- Small number of outward contracts
- Standardised delivery formats
- Identity harmonisation
- Versioned interfaces
- Data delivery becomes structured

Data Delivery Design: 6 Non-Negotiable Layers

LAYER	ROLE
Canonical Identity	Align keys
Product Contract	Define semantics
Version Gate	Prevent breaking changes
Delivery Adapter	SQL / API / Event / File
Monitoring	SLA & DQ
Consumer Registry	Track usage

No direct table access. No unmanaged exports.

The Contrarian Approach

Instead of: "Let's design the perfect canonical model."

Do: "Freeze interface proliferation immediately."

- Step 1: Inventory all outbound feeds
- Step 2: Collapse them into interface clusters
- Step 3: Force new integrations through standardised contracts
- Step 4: Gradually migrate old feeds

Don't renovate the house without fixing the plumbing. Fix the pipes before repainting.

200 Feeds → 5-8 Product Contracts (Finance Example)

1. GL Core Product

Account-level, T+1

Contract
Schema
SLA
Quality
Identity
Versioning

2. P&L Summary Product

Business line, Monthly

Contract
Schema
SLA
Quality
Identity
Versioning

3. Balance Sheet Product

Legal entity, Calendar

Contract
Schema
SLA
Quality
Identity
Versioning

4. FTP/Treasury Product

Product-level, T+1

Contract
Schema
SLA
Quality
Identity
Versioning

5. Reconciliation Product

Transaction-level, Intraday

Contract
Schema
SLA
Quality
Identity
Versioning

5 products serve 95% of consumers. 195 legacy feeds get sunset timelines.

The Unified Delivery Adapter

One pattern for: SQL, API, Event, File

Common envelope for every outbound interaction:

SQL View

- Product ID
- Version
- Timestamp
- UID
- Classification

REST API

- Product ID
- Version
- Timestamp
- UID
- Classification

Event Topic

- Product ID
- Version
- Timestamp
- UID
- Classification

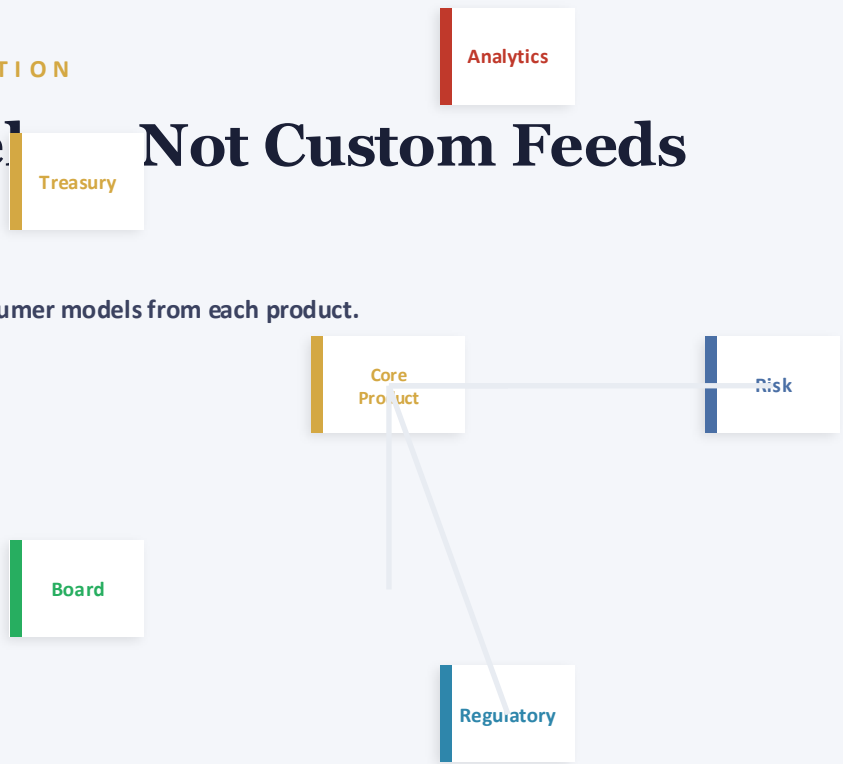
File Export

- Product ID
- Version
- Timestamp
- UID
- Classification

The adapter ensures every delivery format carries the same identity, version, and metadata.

Consumer Models, Not Custom Feeds

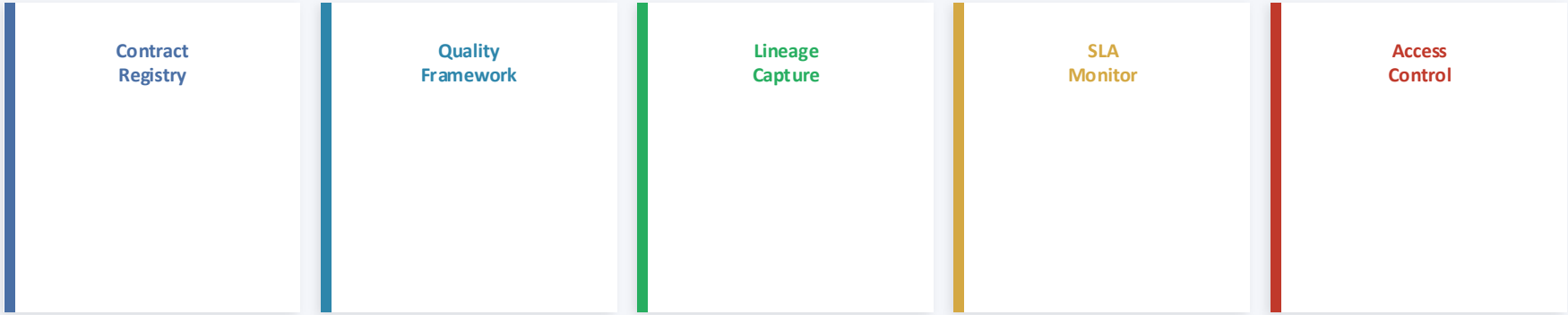
Instead of 200 custom feeds, create 5-8 consumer models from each product.



Consumers subscribe to a model. They don't negotiate a custom feed.

The Sidecar Foundation

Central team attaches enterprise capabilities without touching domain pipelines.



Domain owns the product. Central provides the rails.

Central enablement multiplies domain velocity.

Informal Economy → Formal Economy

INFORMAL

- Undocumented tables
- Hidden joins
- Manual file exchange
- Dependency on people
- No owner
- Unclear freshness
- No change notice
- Custom cuts

FORMAL

- Registered products
- Declared owners
- Explicit contracts
- Versioned interfaces
- Visible SLAs
- Discoverable metadata
- Governed access
- Quality scoring

Structure does not appear by renaming datasets. It appears when exchange becomes formal.

Value Trapped: 4 Critical Areas

1	**Embedded Business Logic**	Logic duplication traps institutional knowledge
2	**Custom Feeds**	Each feed encodes assumptions. 200 feeds = 200 assumption sets
3	**Ownership Confusion**	No clear boundary between internal model and exchange
4	**Over-Investment in Internal Purity**	While delivery chaos remains unaddressed

Unnecessary Complexity: 5 Areas

- 1. Repeating identical semantic transformations across feeds
- 2. Letting consumers define their own data cuts
- 3. Allowing uncontrolled field additions
- 4. Rebuilding identity resolution in every integration
- 5. Treating interface change as informal communication

Where Speed Can Be Unlocked

Speed is not pipeline speed. It is onboarding speed and change speed.

- Consumers use reusable interfaces
- Change impact is visible
- Identity is harmonised
- Interfaces are versioned
- Integration patterns are standardised

The Data Product Profile: 8 Mandatory Attributes

BUSINESS
name, purpose, domain, owner, consumers

DATA
grain, entities, keys/UID, dimensions, retention

DELIVERY
batch/streaming/API, refresh, latency

CONSUMPTION
views, APIs, topics, semantic model

QUALITY
DQ dimensions, thresholds, certification

SECURITY
classification, restrictions, masking, audit

RELIABILITY
SLA/SLO, recovery, incident ownership

CHANGE
version, compatibility, deprecation, notes

Product Boundaries: Good vs Bad

GOOD

- One team owns end-to-end
- Consumers understand purpose
- Contract is stable
- Changes are versioned
- Value is visible

BAD

- Mixes unrelated concerns
- Ownership is split
- Consumers need custom interpretation
- Schema changes constantly
- Value is hidden

3 Concrete Strategic Moves

MOVE 1	**FREEZE INTERFACE PROLIFERATION**	No new custom feeds. All new integration through declared product contracts.
MOVE 2	**SEPARATE GOVERNANCE**	Different boards. Different KPIs. Internal: finance accuracy. External: interface stability.
MOVE 3	**UNIFIED DELIVERY ADAPTER**	One pattern for SQL/API/Event/File. Common envelope: Product ID, Version, Timestamp, UID, Classification.

The Practical Next Step: Week 1 Action

Run a 2-week exercise:

1. List every outbound finance feed
2. Group by semantic similarity
3. Identify top 5 common patterns
4. Design one standardised contract for each pattern
5. Announce: All new integrations must use these patterns

Do not touch the canonical model yet. Fix the pipes before repainting the house.

Implementation Sequence: 4 Phases

PHASE 0
WEEK 1–2

INVENTORY

- List all outbound feeds
- Classify by type
- Identify consumers

PHASE 1
MONTH 1–3

FREEZE + CLUSTER

- No new custom feeds
- Collapse into patterns
- First contracts

PHASE 2
MONTH 3–6

PRODUCTISE

- Build adapters
- Deploy sidecar
- Onboard consumers

PHASE 3
MONTH 6–12

MIGRATE

- Sunset legacy feeds
- Full catalog
- Cross-domain mesh

How Each Domain Transforms

DOMAIN	INTERNAL MODEL	EXTERNAL PRODUCT	FEEDS COLLAPSED
Finance	GL/Sub-ledger/FTP	5 products, 8 consumer models	**200+ → ~30**
Risk	Risk models/ratings	4 products, 6 models	**90 → ~20**
Regulatory	Return templates	3 products, 4 models	**70 → ~15**
Customer	CRM/360° view	3 products, 5 models	**55 → ~15**
Operations	Transaction/settlement	4 products, 6 models	**85 → ~20**

500+ feeds across the enterprise collapse to ~100 managed interfaces. 80% reduction in integration surface.

Governance Split: Internal vs External

INTERNAL MODEL

GOVERNANCE

- Domain architecture board
- KPI: Finance accuracy
- Concerned with normalisation
- Reviews data model changes
- Focus: operational performance

EXTERNAL EXCHANGE

GOVERNANCE

- Cross-domain interface board
- KPI: Interface stability
- Concerned with contract consistency
- Reviews interface/product changes
- Focus: consumer experience

Different boards. Different KPIs. Internal: accuracy. External: stability.

Non-Negotiable Delivery Layers

1	**Canonical Identity**	align keys across all products
2	**Product Contract**	declare schema, semantics, SLA
3	**Version Gate**	no breaking changes without notice
4	**Delivery Adapter**	SQL/API/Event/File — one pattern
5	**Monitoring**	SLA, DQ, usage, freshness

No direct table access. No unmanaged exports. Every outbound interaction goes through these 6 layers.

Good Cop: What Success Looks Like

- Internal models stay clean
- External products are stable
- Consumers onboard in days not months
- Changes are versioned not surprising
- Identity is consistent
- 80% fewer interfaces

Bad Cop: What Failure Looks Like

- Internal model gets beautiful but 200 feeds remain
- Products are labelled but delivery is still custom
- Governance added without delivery discipline
- No common identity
- Consumers still negotiate bespoke feeds
- Spaghetti renamed not removed

Then you have renamed the mess, not fixed it.

THE TRANSFORMATION PROMISE

Data model = how you think internally. Data product = how the world interacts with you.

- Separate the internal model from the external product
- Freeze interface proliferation immediately
- 200+ feeds → 5-8 product contracts per domain
- Every outbound interaction through 6 non-negotiable layers
- Delivery design is the boundary of institutional coherence
- Canonical model matters. But interface governance matters more.
- Fix the pipes before repainting the house.
- The biggest integration cost is not bad data models. It is uncontrolled interface multiplication.