

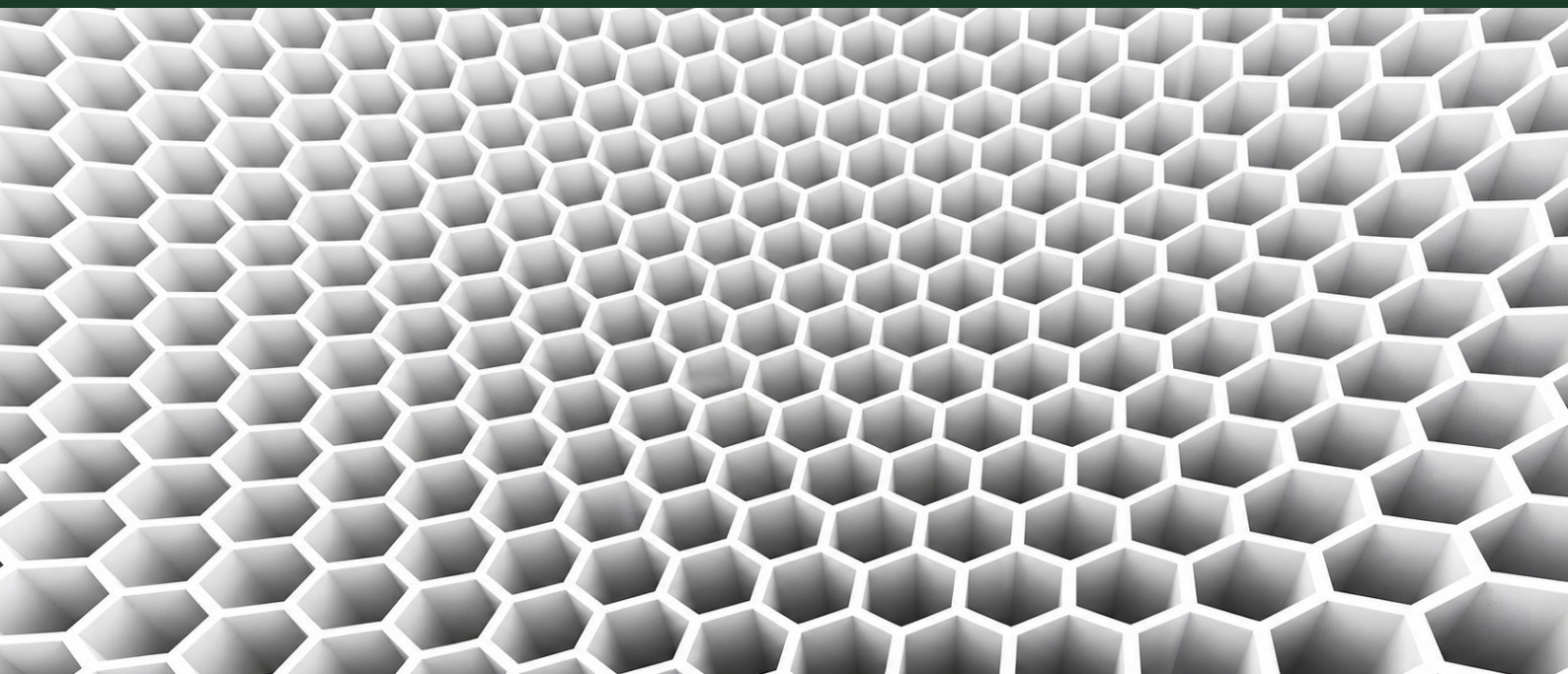
BEYOND THE PROJECT

Data Products in Banking: Moving Beyond Projects Without Losing Control

A Practitioner's Perspective on the Data Product Operating Model in Regulated Banking

Giovanni Leonardi

May 2022



Contents

1. Executive Summary	3
2. The Project Reflex And Why It Fails Data	3
3. The Evidence: What Decay Looks Like	4
4. Project Versus Product: The Distinction That Matters	5
5. The Operating Model: Five Components, Five Owners	5
6. The Hard Ambiguities	6
7. Keeping Control While Moving Beyond Projects	7
8. A Recommended Adoption Path For A Mid-Size Regulated Bank	7

Data Products in Banking: Moving Beyond Projects Without Losing Control

Programme · No. 210

Originally written May 2022 | Re-edited and re-rendered for the WhereBeyond Edition, 2025

“What was missing was an owner, a funded support line and a governing body – the three things a project, by its nature, removes at exactly the moment a product needs them most.”

1. Executive Summary

For two decades banks have delivered data capability through projects. A need is identified, a business case is written, funding is approved for a fixed scope over a fixed period, a team is assembled, the thing is built, the team is disbanded, and the budget line closes. This machinery is well understood and, for the delivery of a discrete outcome, it works. The difficulty is that the thing most banks now need to deliver is not discrete. A data product – a curated, owned, reusable set of data with defined quality, defined consumers and a defined service level – does not end. It persists, it is depended upon, and it must be supported, funded and governed for as long as anyone relies on it.

This white paper sets out the problem created when a persistent product is delivered by machinery designed for temporary projects, and it proposes a practical operating model to resolve it. The model names five components that must each have a clear owner: product ownership, programme and portfolio governance, technical accountability, business adoption, and lifecycle funding. It then confronts the ambiguities that the project model leaves unresolved and that quietly destroy value after go-live: who funds the product once the project budget has closed, who owns its support, and how governance persists when the programme that built it has stood down.

The recommendation is that a mid-size regulated bank should not attempt a wholesale reorganisation around data products. It should adopt the product operating model deliberately, on a small number of high-value data products first, proving the funding and governance mechanisms on real examples before extending them. Moving beyond projects is necessary. Doing so without losing control is the harder and more important achievement.

2. The Project Reflex And Why It Fails Data

The pattern I have observed repeatedly is that a bank recognises the value of treating data as a product, endorses the idea at the most senior level, and then delivers it through

exactly the project machinery the idea was meant to supersede. The reflex is understandable. Projects are how the organisation knows how to fund, staff and govern change. But the reflex imports a set of assumptions that do not hold for data products, and each broken assumption becomes a source of decay.

A project assumes a definable end. A data product has none; it is valuable precisely because it goes on being consumed. A project assumes a temporary team that disbands on completion. A data product needs someone who remains accountable for it indefinitely. A project assumes funding for a scope, after which the budget closes. A data product incurs cost for as long as it lives – support, quality management, change, infrastructure – and that cost does not stop when the project does. A project measures success by delivery to time, cost and scope. A data product is successful only if it is adopted, trusted and used to make decisions, which is a judgement that can only be made after the project has ended.

The consequence is a predictable failure mode. The product is built and launched to applause. The project closes and the team moves on. Within months the data quality drifts because no one owns it, consumers lose trust and revert to their own extracts, support requests fall between teams, and the next necessary change has no budget and no sponsor. The bank concludes that the data product did not deliver, when in fact the delivery machinery was never designed to keep it alive.

A project is a promise to finish. A data product is a promise to keep going. Delivering the second with the machinery built for the first is the single most common reason data products decay after launch.

3. The Evidence: What Decay Looks Like

The failure this paper describes is not hypothetical, and it follows a consistent shape across institutions. A bank builds a data product to give a business function a single, trusted view of something it previously assembled by hand. The build succeeds. At go-live the product is accurate, the consumers are enthusiastic, and the project is closed as a success against its time, cost and scope.

The decay begins quietly. In the first quarter after launch a source system changes and the product's figures drift, but the team that built it has moved to the next initiative and no one is accountable for the fix. A consumer notices a number they cannot reconcile, raises it, and finds there is no clear owner to raise it to. Rather than wait, they rebuild the extract they used before, because they have a decision to make and cannot make it on a figure they do not trust. Within two quarters several consumers have quietly reverted to their own copies. The product is still running, still costing money to host, but it is no longer the source of truth it was built to be.

By the time anyone looks, the diagnosis is usually mistaken. The product is judged to have underdelivered, or the data to have been poor. In fact the data was fine at launch and the product was well built. What was missing was an owner, a funded support line and a governing body – the three things a project, by its nature, removes at exactly the moment

a product needs them most. The evidence, in other words, points not at the build but at the machinery that closed around it.

4. Project Versus Product: The Distinction That Matters

Before proposing the operating model it is worth making the distinction explicit, because much of the confusion in practice comes from using project language for product realities.

Dimension	Project	Data Product
Time horizon	Fixed, ends at delivery	Persistent, ends only when retired
Team	Temporary, disbands on completion	Standing accountability, indefinite
Funding	Approved for a scope, then closes	Lifecycle funding for as long as it lives
Success measure	On time, on cost, in scope	Adoption, trust, decision impact
Governance	Programme board for the duration	Persistent product and portfolio governance
Change	Re-scoped or re-projected	Managed continuously as versions
Support	Handed to run, often undefined	Owned, with a defined service level

The value of stating it this way is that it exposes where the project model is silent. It has no answer to who funds the product next year, no answer to who owns its support, and no answer to how it is governed once the board that oversaw its build has dissolved. Those silences are not details. They are where control is lost.

5. The Operating Model: Five Components, Five Owners

A workable operating model for data products in a regulated bank does not require a new organisation. It requires that five distinct accountabilities are named, assigned and held. The failure mode is almost always that one or more of these is assumed rather than owned.

- **Product ownership.** Every data product has a single accountable owner, on the business side, who is responsible for its purpose, its consumers, its quality standard and its priorities. This is not a technical role. It is the person who can say what the product is for, decide what changes matter, and answer for whether it is trusted and used. Without a named product owner, a data product is an orphan the moment the project ends.
- **Programme and portfolio governance.** Individual products are built within programmes, but the estate of products must be governed as a portfolio. Portfolio governance decides which products are worth funding, resolves overlaps and

dependencies between them, and takes the persistent decisions – continue, invest, retire – that no single product owner can take alone. This is where control over the whole is exercised.

- **Technical accountability.** Someone must be accountable for how the product is built and run: its architecture, its pipelines, its resilience, its lineage and its cost to operate. This accountability persists beyond the build, because the product persists. Separating technical accountability from product ownership, while keeping them in close partnership, prevents the common error of a business owner who cannot answer for the platform and a technical team that cannot answer for the purpose.
- **Business adoption.** A data product delivers no value until it is used in place of the thing it replaces. Adoption is an accountability in its own right: retiring the shadow extracts, migrating the consumers, and confirming that decisions are now being made on the product. Where adoption is left to happen by itself, it does not, and the bank pays to build a product that is used by no one.
- **Lifecycle funding.** The product must be funded for its life, not only its build. This means a funding mechanism that continues past go-live to cover support, quality, change and infrastructure. It is the component most often missing, and its absence is the clearest signal that a bank has adopted the language of products without the substance.

6. The Hard Ambiguities

The operating model above resolves the ambiguities the project reflex leaves open, but only if the bank confronts three of them directly. These are the questions that, left unanswered, allow a well-built product to decay while everyone believes it was delivered successfully.

1. **Post-go-live funding.** When the project budget closes, who pays for the product? The honest answer is that a bank cannot run a persistent product estate on project funding alone. Some portion of funding must become recurring, attached to the product rather than to a time-boxed initiative, and owned by the portfolio. The mechanism can vary – a run budget, a charge to consuming functions, a central data operating budget – but the principle is fixed: a product with no source of ongoing funding has been set up to fail, and the failure is simply deferred to the first year in which it needs money and has none.
2. **Support ownership.** When a consumer finds a figure they do not trust, or a pipeline stops, who answers? In the project model support is frequently undefined, handed vaguely to "run" without a named owner or a service level. For a data product, support ownership must be explicit before go-live: who triages, who fixes, what the service level is, and how it is resourced. Ambiguity here is not merely operational friction; it is the fastest route to lost trust, because a consumer who cannot get a wrong number fixed will stop using the product and go back to their own copy.
3. **Persistent governance.** The programme board that oversaw the build will stand down. What governs the product then? Persistent governance is the answer: portfolio-level oversight that continues to take decisions about the product across its life – prioritising change, approving investment, deciding retirement, and holding

the product owner and technical owner to account. Without it, the product drifts, because the only body that ever governed it has ceased to exist.

“A data product with no source of ongoing funding has not been delivered. It has been set up to fail, with the failure deferred to the first year it needs money and has none.”

7. Keeping Control While Moving Beyond Projects

The phrase "without losing control" is not rhetorical. A regulated bank cannot treat data products as free-standing artefacts that individual teams spin up and run as they see fit. The move to products can, if handled carelessly, fragment ownership and weaken the very governance that regulation requires. Control is preserved by three disciplines that the operating model makes possible.

The first is that governance is designed to be embedded and persistent rather than attached to a project and then removed. Ownership, quality standards, lineage, access and accountability are properties of the product itself, defined at its creation and carried through its life. The operational resilience expectations that now apply to important business services make this not merely good practice but a supervisory requirement: a service that persists must have owners, controls and resilience that persist with it.

The second is that the portfolio, not the individual team, holds the authority to create, continue and retire products. This prevents the sprawl of unmanaged products that no one funds or governs, which is the characteristic failure of a poorly executed product model. A product exists because the portfolio agreed it should, and it is retired when the portfolio agrees it should be.

The third is that technical accountability includes cost, resilience and third-party dependency. As products increasingly rely on shared platforms and external providers, concentration risk and third-party dependency become persistent product risks, not one-off project considerations. Naming a technical owner who answers for them across the life of the product is how the bank keeps control of a risk that the project model, which closes at go-live, was never structured to manage.

8. A Recommended Adoption Path For A Mid-Size Regulated Bank

The evidence points to a clear recommendation, and it is deliberately not a transformation of the whole organisation. A mid-size regulated bank has neither the surplus capacity nor the risk appetite for a wholesale reorganisation around data products, and attempting one is the surest way to lose the control this paper is concerned with. The path that works is incremental and proof-driven.

1. Select a small number of high-value products. Choose two or three data products where the value is clear and the consumers are known - a trusted client or exposure view, a regulatory dataset, a core reference set. These become the proving ground for the operating model.
2. Assign the five accountabilities explicitly. For each chosen product, name the product owner, the technical owner, the portfolio governance forum, the adoption owner and, critically, the lifecycle funding source. Do not proceed until the funding source past go-live is real, not notional.
3. Establish persistent governance before the first build completes. Stand up the portfolio-level forum that will govern these products after their projects close, and give it authority over continue, invest and retire decisions. It must exist before the programme boards stand down, or there will be a gap in which control is lost.
4. Prove the hard ambiguities on real examples. Use these first products to make post-go-live funding, support ownership and persistent governance concrete. Run them for a full cycle past launch, so the bank learns what recurring funding and owned support actually cost and require.
5. Extend deliberately, not wholesale. Only once the model has been proven on the first products should it be extended to the next. The portfolio grows product by product, each with its five accountabilities in place, rather than through a reorganisation that declares everything a product overnight and controls none of it.

The conclusion is straightforward. Banks are right to move beyond projects, because the project model cannot keep a persistent product alive. But the move is not a matter of renaming teams or relabelling initiatives. It is a matter of assigning the five accountabilities, funding the product for its life, and governing the estate as a persistent portfolio. A bank that does this moves beyond projects and keeps control. A bank that adopts the language without the accountabilities moves beyond projects and loses it - and discovers the loss only when the products it was proud to deliver quietly cease to be trusted.