

Reference Architecture for API-Led Core Modernisation

Proven patterns across payments, lending, KYC, Customer 360 and trade finance



What twenty years of point releases actually built

UK Finance calls core banking software the “museum of technology” of financial services, built up over three or four decades through mergers, product launches and regulatory patches until it resembles a geological formation more than a single application. Most cores in production today still function reliably. What has grown unmanageable is the layer around them: hundreds of point-to-point connections, each negotiated separately, each maintained by whichever team happened to own the project that built it.

There is rarely a governed API layer underneath. There is rarely a contract that a new fintech partner, a payments scheme, or an internal reporting tool can build against without a bespoke integration effort of its own. That is the problem an architecture review needs to solve, and it is a narrower, more tractable problem than replacing the core outright.



Why the rewrite instinct keeps failing

The instinct to solve integration debt with a full core replacement is understandable, and poorly rewarded on the numbers. Large-scale monolith replacement programmes routinely run past three years, and industry estimates put the failure rate for single-cutover core rewrites above 80 percent. The risk isn’t confined to budget. A bank cannot stop processing transactions while a new core is built, tested and cut over, so every big-bang programme carries operational risk on top of cost risk.

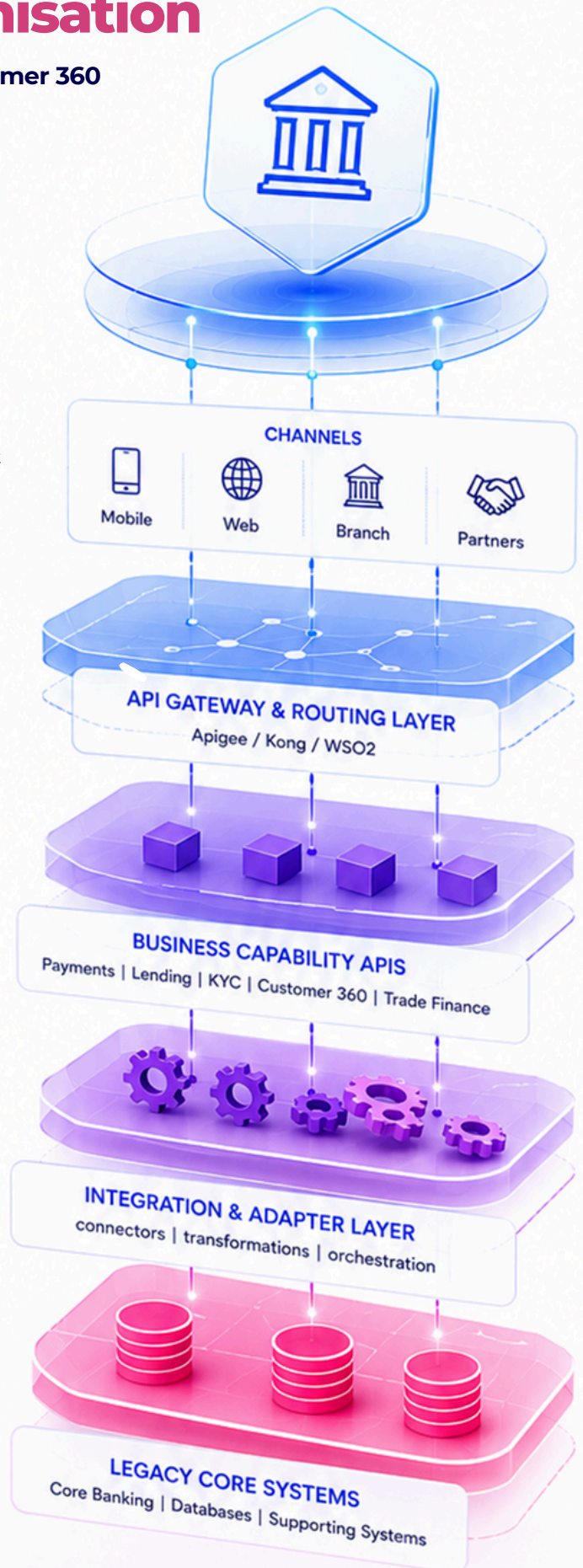
The approach that has held up under scrutiny is incremental decomposition, commonly known as the strangler fig pattern: a routing layer sits in front of the legacy core, new services are built and validated one bounded context at a time, and traffic shifts across gradually while the legacy system keeps running underneath it. Analysts now expect this to be the dominant pattern in banking. An estimated 40 percent of global banks were already running modern services alongside legacy cores in parallel by 2026, a share IDC projects will reach 70 to 80 percent by 2028.



What the reference architecture actually contains

The library behind this reference architecture spans more than 40 banking-specific API sets and 15-plus regulatory API bundles, built from work across more than 30 live banking programmes reaching upward of 10 million banked customers, rather than assembled from documentation. That distinction matters at review stage. A reference architecture that has only been drawn on a whiteboard behaves differently under load, under audit and under a live incident than one that has already carried production traffic.

The gateway layer typically sits on Apigee, Kong or WSO2, whichever a bank already runs, exposed through a backend-for-frontend per channel rather than one shared API forced to serve mobile, web, branch and partner traffic identically. Banks taking this route report roughly 40 percent less integration effort per channel launched, since experience teams stop rebuilding the same backend logic for every new surface. The Customer 360 accelerator alone integrates against core platforms including Mambu and HPS, decisioning engines like Actico, and external bureaus, through a single adapter layer rather than a separate connector for each.



Domain	What it covers	What good looks like
 Payments	Real-time and batch rails, sanctions and fraud screening hooks	Contract-first APIs, idempotent processing, clean separation from settlement
 Lending	Origination, decisioning, servicing	Product modelled as data, not hard-coded into the platform
 KYC and onboarding	Identity verification, screening, document capture	One reusable service across retail, business and correspondent flows
 Customer 360	Unified view across product silos	Event-driven synchronisation in place of batch reconciliation
 Trade finance	Letters of credit, guarantees, documentary collections	Workflows historically locked in proprietary systems exposed as APIs



Where to start

The pattern that works in practice starts narrow. Pick one bounded context, typically the one causing the most integration pain today, and prove the pattern end to end: façade in front of the legacy service, new microservice built and tested alongside it, traffic shifted incrementally, legacy component retired only once the new one has carried production load without incident.

A well-scoped first slice is demonstrable within 90 days. That gives an architecture team something concrete to bring to a board that is, reasonably, wary of committing to another multi-year transformation programme with no proof points along the way.



If this maps to what you're navigating

These are the patterns we see hold up at banking scale. If your team is weighing the same trade-offs, a Maveric architect would welcome the conversation.

Sources: UK Finance, "Core banking: getting to the heart of the matter" (2026); IDC sidecar architecture projections (2026); core-modernisation programme benchmarks, 2025-2026;

The Maveric Edge

Maveric Systems is a banking-exclusive technology specialist with 25+ years of domain expertise, engineering trust in AI-first banking. We deploy AI at the core through our proprietary AI@Scale framework and AI-powered platforms, guided by principles that engineer trust by design. Our pragmatic, outcome-driven approach, and AI services spanning advisory, CoE establishment and specialist competencies led engagements make us the trusted engineering partner for the AI era.

25+

years of domain mastery

7+

Operations and Technology AI CoEs

12+

AI powered Platforms and Solutions

AI@Scale

Proprietary Framework for AI adoption at Scale