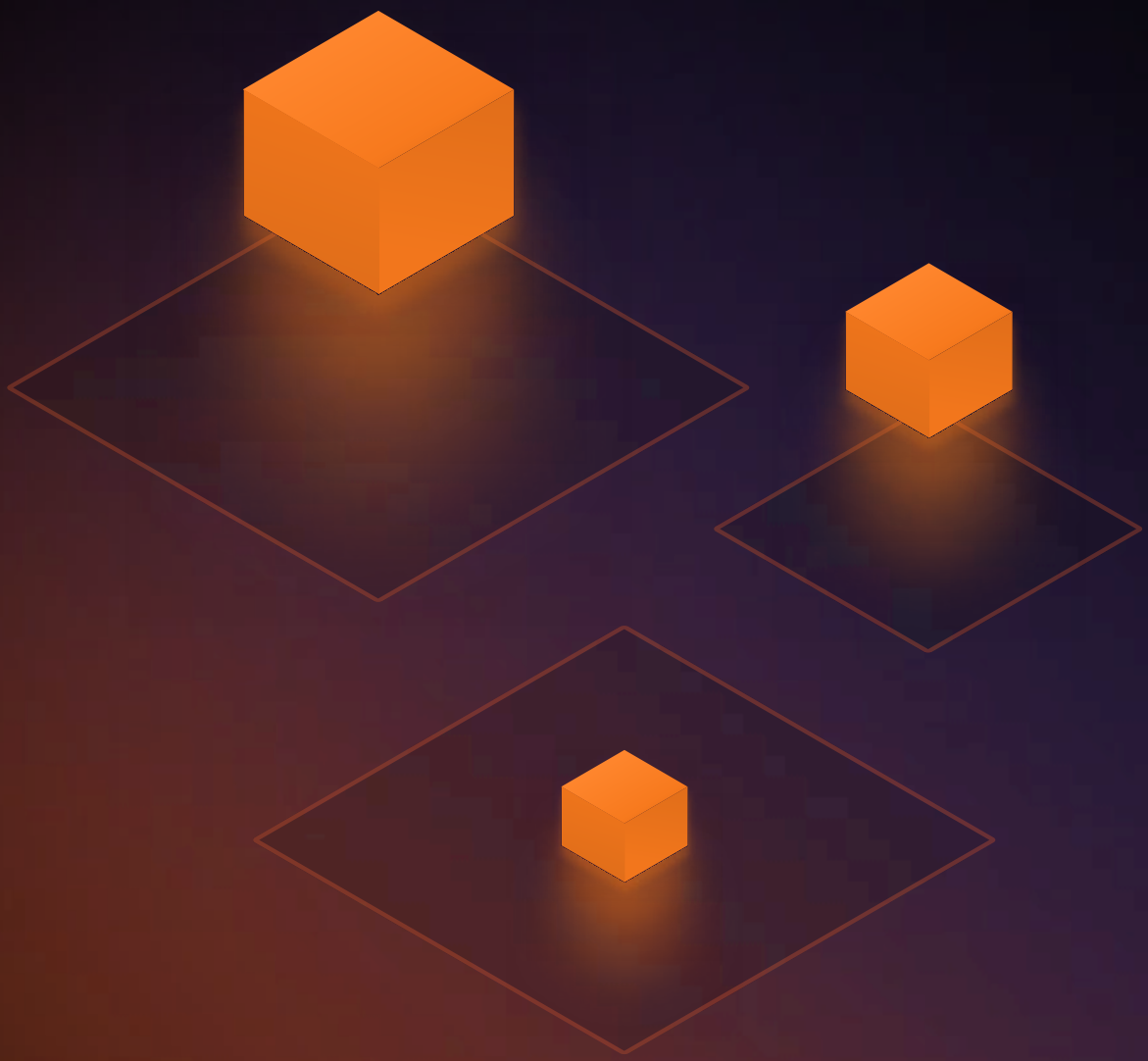
OpenLegacy

Modernizing with Confidence: A Phased Co-Existence Strategy for Legacy Transformation

By Leigh-Ann Silver



1

Executive Summary

Enterprises that modernize mission-critical legacy systems can unlock dramatic gains—**up to 353 % ROI and 45 % lower IT operating costs**—yet most initiatives struggle with skills shortages, hidden dependencies, and an uncompromising need for uptime. [1]

This whitepaper discusses **OpenLegacy's phased and agile modernization strategy - handling the co-existence practice**: an evidence-based framework that balances innovation with operational continuity. The framework combines automated analysis, a structured migration planner, and a library of decoupling patterns that enable legacy and cloud workloads to run side-by-side while change proceeds incrementally.

↗

353%

ROI

↓

45%

lower IT operating costs

Built to Support Any Modernization Methodology

Whether an enterprise chooses to **refactor, re-platform, rewrite, repurchase, retain, or retire** applications, OpenLegacy's phased migration approach adapts easily. Service façades allow legacy and modern modules to coexist so teams can deliver new digital capabilities—APIs, real-time data feeds, AI integration, etc.—**immediately while progressively moving workloads to the cloud**. This ensures today's innovations aren't delayed by tomorrow's migrations.

KEY TAKEAWAYS FOR EXECUTIVES

- Reduce modernization risk through **pre-migration dependency mapping** and automated service generation.
- Adopt a **step-by-step strategy** that delivers measurable value early—without the cost or disruption of a "big-bang" cut-over.
- Future-proof investments by **exposing core business logic and data** as cloud-ready APIs that fuel analytics and AI initiatives.

KEY TAKEAWAYS FOR TECHNICAL LEADERS

- Leverage **internal, external, data-layer, and logic-layer decoupling patterns** to decouple safely at multiple layers.
- Bypass middleware with **direct, standards-based connectivity** to mainframe and mid-range platforms.
- Accelerate delivery with **AI-assisted code & test generation**, CI/CD-ready artifacts, and automated regression testing.
- Accelerate connectivity with an extensive library of **pre-configured connectors for legacy systems** spanning CICS, AS/400, DB2, VSAM, Oracle, and more—eliminating months of custom adapter effort.

2

The Modernization Imperative:
Challenges, Impacts & OpenLegacy Response

Despite decades of new investment, legacy platforms still underpin mission-critical workloads in banking, insurance, manufacturing, and the public sector. Modernization is therefore not optional—but it must be executed without jeopardizing daily operations.

CHALLENGE	IMPACT	HOW OPENLEGACY RESPONDS
System Complexity & Hidden Dependencies	Decades-old code, proprietary protocols, and undocumented inter-module calls raise the risk of regression failures if changes are made blindly.	Hub Planner automatically maps program calls, database interactions, and batch dependencies; generated façade APIs encapsulate complexity while automated contract tests provide guardrails, relieving you from the burden of system complexity and hidden dependencies.
Skills Shortage & Tribal Knowledge	Retiring experts leave gaps that are hard to fill, extending project timelines.[2]	Low-code wizards, AI-assisted code templates, and a rich connector catalog reduce reliance on scarce mainframe specialists, empowering your existing development teams to deliver changes safely.
Manual Integration Effort	Traditional middleware and bespoke adapters create long feedback cycles (4–6 months per interface).	Automated API generation and pre-configured connectors compress interface delivery to **days or weeks**, accelerating your digital initiatives and keeping you ahead in the competitive market.
Zero-Downtime Mandate	Regulated industries cannot tolerate service disruption; modernization must proceed *in flight*.	The **Phased Co-Existence Framework** keeps legacy and cloud workloads running side-by-side, supported by dual factories (Integration & Migration) and rolling, validated releases.

OpenLegacy removes the traditional blockers—complexity, skills gaps, slow integration cycles, and downtime risk—allowing enterprises to innovate immediately while modernizing on their own schedule.

At the same time, competitive pressures and cloud economics are driving boards to demand faster release cycles, AI-readiness, and cost rationalization. A modernization strategy that **de-risks coexistence** is therefore essential.

3

A Phased Co-Existence Approach

OpenLegacy's approach focuses on **co-existence**. The goal is to create a **unified landing zone** where legacy and modern components interact through generated APIs while workloads migrate in manageable slices.

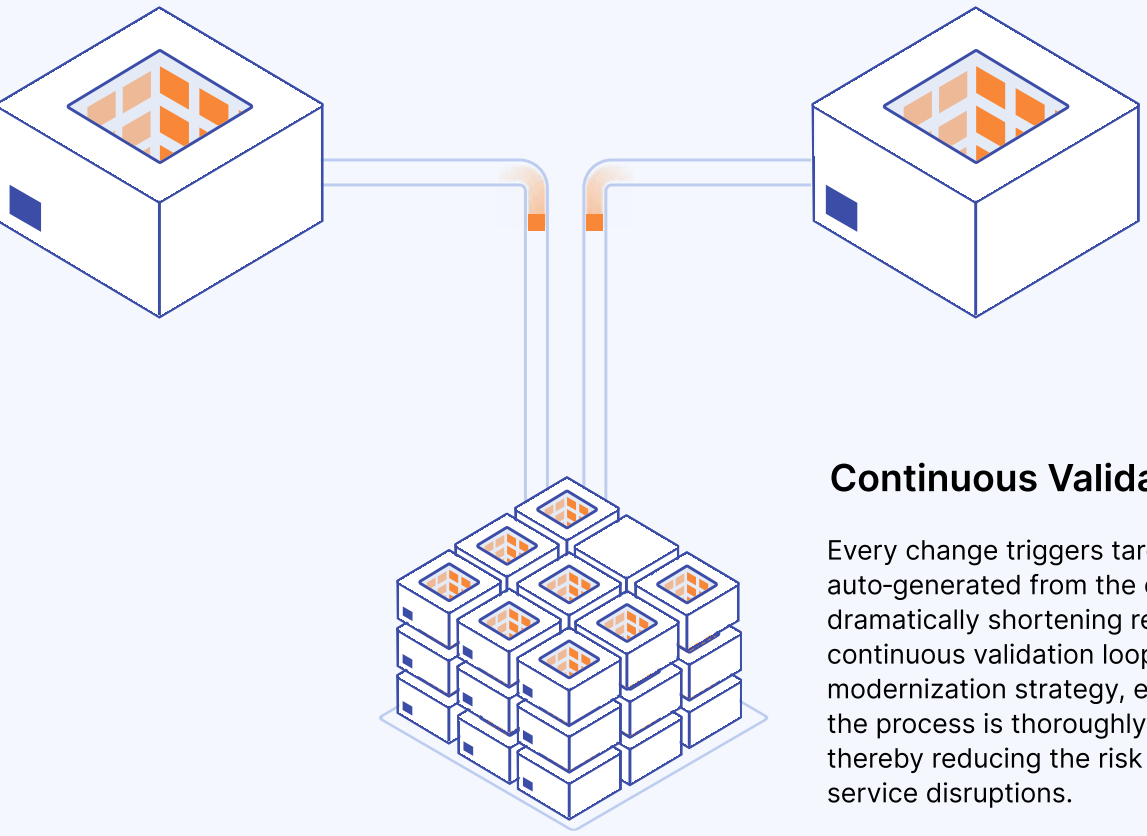
Unified Landing Zone

The Unified Landing Zone, a secure, cloud-native substrate—container or serverless—hosts lightweight service façades generated directly from legacy artifacts. These façades expose data & logic via REST/JSON, gRPC, or events, eliminating extra middleware hops. This approach not only simplifies the architecture but also accelerates the integration process, reducing the risk of service disruption during the modernization process.

Dual Factory Model

- **Integration Factory** – Automates API generation for legacy endpoints so new digital services can call them unchanged.
- **Migration Factory** – Generates modernized code & data access layers when components are ready to move off-platform.

Both factories feed a single CI/CD pipeline, ensuring consistent governance, security scans, and automated testing.



Continuous Validation Loop

Every change triggers targeted test suites auto-generated from the original legacy contracts, dramatically shortening regression cycles. This continuous validation loop is a key component of the modernization strategy, ensuring that each step of the process is thoroughly tested and validated, thereby reducing the risk of unexpected issues or service disruptions.

4

Decoupling Patterns: Building Blocks for Risk-Free Transformation

OpenLegacy operationalizes complementary patterns that can be applied independently or in combination:

PATTERN	PURPOSE	TYPICAL TRIGGER
Internal Decoupling	Replace program-to-program calls with remote APIs so individual modules can be re-written or re-hosted.	Need to refactor a high-change module without touching the full monolith.
External Decoupling	Intercept existing MQ/CTG/IMS gateways, transforming payloads for new cloud services.	Front-end channels must remain intact while back-end logic migrates.
Data-Layer Decoupling	Extract embedded SQL and call remote data services; supports hybrid DB topologies.	Move data to cloud databases without rewriting COBOL.
Logic-Layer Decoupling	Isolate business rules from screen-handling code, exposing them as reusable micro-APIs.	Unlock core algorithms for reuse in mobile or AI apps.
Data Extraction & Sync	eal-time CDC or batch ETL keeps legacy and cloud data stores aligned.	Analytical workloads require cloud-native scale; regulatory constraints demand dual writes.

Guiding Principle: Decouple only as much as necessary for the next modernization step; avoid unnecessary re-writes. This principle guides the modernization approach, ensuring that the focus is on incremental and manageable changes that deliver immediate value, rather than large-scale rewrites that can introduce unnecessary complexity and risk.

5

Implementation Roadmap & Best Practices

1. **Portfolio Assessment & Dependency Mapping** – Use OpenLegacy Hub Planner to visualise inter-program calls, database touchpoints, and batch schedules.

2. **Choose Decoupling Points** – Apply patterns where they deliver immediate value with minimal risk.

3. **Establish the Integration Factory** – Generate and deploy façade APIs; integrate security and observability early.

4. **Iterate with Migration Factory** – Modernize selective components; validate via automated contract tests.

5. **Retire Legacy Interfaces Incrementally** - Decommission gateways or modules once traffic fully migrates.

↓

Success Metrics may include release frequency, mean-time-to-restore (MTTR), cloud consumption, and reduction in manual hand-offs.

6

Technical Deep Dive: Architecture & Automation

Direct Connectivity Layer

Generated connectors, drawn from OpenLegacy's extensive catalog of pre-configured connectors—including native CICS, IMS, VSAM, DB2, AS/400, SAP, and Oracle access—use native protocols to avoid the performance penalties of middleware stacks.

Security & Compliance Controls

- OAuth2/OIDC wrappers, field-level encryption, role-based access mapped to RACF or LDAP.
- Audit logs streamed to SIEM; lineage preserved for regulatory evidence.

API & Service Generation Pipeline

- **Model Extraction** – Parsers read copybooks, PCML, RPG, or PL/I definitions.
- **Code Templates** – AI-assisted templates emit Java Spring-Boot micro-services.
- **Test Harness** – Request/response pairs captured from production are converted into JUnit or PyTest suites.

DevOps Alignment

Artifact repositories, container registries, and IaC modules integrate with standard pipelines (GitHub Actions, Jenkins, Azure DevOps). Rolling upgrades enable zero-downtime releases.

7

Illustrative Case Snapshots*

ENTERPRISE	CHALLENGE	PATTERN MIX	OUTCOME
Tier-1 European Bank	Achieve real-time payments while core remains on z/OS	External + Data-Layer Decoupling	24×7 instant-payment rail with zero core outages; cut CTG licensing costs.
Global Insurer	Separate document management from policy admin	Internal + Logic-Layer	8–10× faster delivery; retired CICS gateway.
Multinational Life Company	Multiple mainframes, phased cloud move	All 5 patterns	Deployed APIs on AWS Fargate; maintained policy service continuity.

**All company names anonymised or used with permission.*

8

Future Outlook

Emerging AI copilots will further automate code comprehension, test generation, and optimization. Meanwhile, multi-cloud demands will increase the value of a **landing-zone-agnostic** approach that abstracts core logic into portable APIs.

Net result: You don't have to finish a multi-year migration before your teams can start experimenting with AI/ML, real-time analytics, or future innovations —the decoupled slice of your legacy estate is ready on day one.

[1] Forrester Consulting, The Total Economic Impact™ of OpenLegacy, 2023.

[2] Gartner, “Assessing the Mainframe Talent Gap,” 2024.

About OpenLegacy

OpenLegacy is an AI-driven platform purpose-built for continuous, phased, structured legacy modernization. By combining automated analysis, instant API generation, and a library of proven decoupling patterns, OpenLegacy enables enterprises to modernize at their own pace—without the risk.