

Most companies don't plan cloud exits too late.

They architect them away early.

The exit window closes before the exit decision.

Cloud exit becomes non-computable before it becomes operationally impossible.

The data is still there. The workloads are still running.

But the structural conditions that made exit legible are gone.

It's not the vendor. It's the design philosophy.

DESIGNED FOR

Onboarding Velocity

Faster provisioning. Less ops burden.
Managed everything.

WHAT YOU NEED

Exit Survivability

Portable data. Owned control plane.
Mappable dependencies.

Exit readiness is the absence of irreversible coupling across 4 constraint domains.

01

Data Gravity Constraint

Storage & schema coupling
to provider-native semantics

02

Dependency Graph Entanglement

Integration surface creating
unmappable blast radius

03

Control Plane Sovereignty

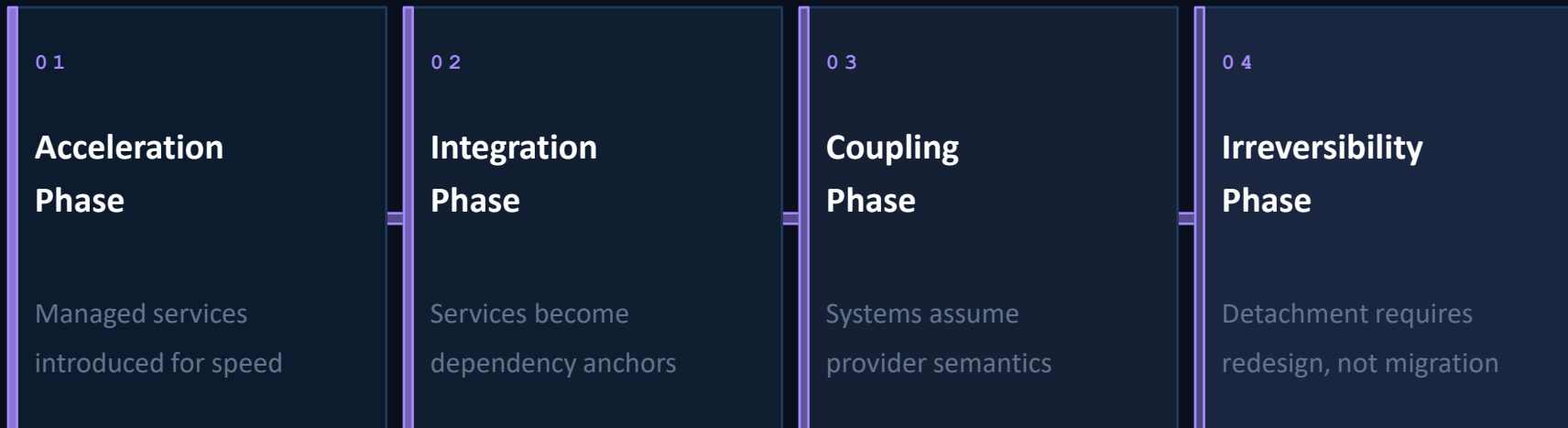
Operational independence
from provider-managed ops

04

Commercial Lock Structure

Contract & data terms that
survive provider removal

The window doesn't close in one decision. It closes in four stages.



The mechanism of lock-in is not the managed service. It's the irreversibility threshold.

IRREVERSIBILITY THRESHOLD

The point at which reversing a design decision requires rewriting adjacent systems — not replacing the original component.

IAM integrated into
application auth flows

Managed DB triggers
embedded in business logic

Observability tied to
provider telemetry schemas

Eventing dependent on
proprietary message
semantics

Exit readiness is not a migration plan. It's a set of explicit architectural rejections.

- ✗ Provider-native database behavioral semantics
- ✗ IAM delegation to provider identity plane as root auth authority
- ✗ Managed Kubernetes as operational authority for cluster governance
- ✗ Provider telemetry schema as observability and runbook backbone
- ✗ Egress pricing treated as a procurement variable, not an architecture constraint

The Exit Readiness Window Closes Before You Know It's Open.

Exit readiness is the absence of irreversible coupling across four constraint domains.

Optionality is not a feature.

It is an architectural constraint.