ENTERPRISE ARCHITECTURE · GOVERNANCE

Aspirational Architecture

Three generations of declared intent, and why nothing enforces any of it.

ABSTRACT

Enterprise architecture has never lacked ways to say what the environment should be. TOGAF defines the target. Platform engineering paves the golden path. Infrastructure-as-code declares the desired state. Three generations of architectural intent, each more executable than the last — and in most enterprises, nothing validates that a given change conforms to any of them before it runs. Each practice declares. Each quietly assumes something else will hold reality to the declaration. Nothing does.

1. Three layers of aspiration, one shared assumption

The target architecture. TOGAF and enterprise architecture define principles, roadmaps, and the intended shape of the estate. The blueprint lives in documents and diagrams — artifacts that describe, and that are consulted by people, not by change.

The golden path. Platform engineering paves the approved route: templates, paved pipelines, sanctioned patterns. The path is genuinely easier than the alternatives. But easier is not the same as enforced, and teams under deadline route around it — which is precisely when the routing matters.

The declared state. Infrastructure-as-code states what managed resources should be — for the resources it manages, in the repositories that stay current. Around that footprint, change arrives by console session, ticket, and script, and drift accrues exactly where nothing declares anything.

Each generation moved intent closer to execution. None of them placed intent *in the path of* execution.

2. The four ways aspiration fails

Documents are consulted by people, not by change. A change does not read the architecture repository before it executes. Conformance depends on a reviewer remembering the principle, recognizing the violation, and having the standing to stop the work — three human contingencies standing in for an enforcement point.

Adoption is voluntary exactly where it matters most. The teams most likely to leave the golden path are the ones under the most pressure, and the changes made under pressure are the ones that most need governing. A paved road with optional exits governs only the compliant.

Coverage ends at the managed footprint. IaC validates what it applies. It cannot speak for the change that reaches infrastructure through a console, an emergency fix, or a vendor tool — and it cannot see its own declared state go stale against a reality that moved without it.

Review arrives after the decision. Architecture review boards evaluate designs, not executions. By the time nonconformance is visible in an assessment, it is not a proposal to be declined — it is deployed infrastructure to be remediated, with dependencies already grown around it.

The pattern across all four is the same. Intent exists. Intent is even well-expressed. But intent is positioned where change cannot encounter it — before the work in documents, beside the work in templates, after the work in reviews. Never *in the way of* the work.

3. What an enforcement point requires

For a declaration to govern rather than aspire, four things must be true of it, and none of the three generations satisfies all four.

It must be **authoritative** — one declared statement of permitted states, ownership, and boundaries, not a federation of documents that disagree. It must be **current** — continuously reconciled against the environment, so the intent being enforced refers to the estate that exists rather than the estate as of the last update. It must be **in the execution path** — consulted by connected change before it runs, structurally, not by the discipline of whoever is making the change. And it must apply to **every actor** — people, automation, and AI — because an enforcement point with exemptions is a suggestion with extra steps.

This is not a criticism of TOGAF, platform engineering, or IaC. Each is doing its job. The target architecture is supposed to describe the destination. The golden path is supposed to make the right way easy. Declared state is supposed to make managed resources reproducible. The missing element was never inside any of them — it is the layer beneath them all: a governing function that holds declared intent and evaluates connected change against it before execution.

That layer is what the Infrastructure Operating Model standard defines. With it in place, the three generations stop being aspirations and become what they were always meant to be: the content of the declaration the enforcement point enforces. The architecture stays the architecture. The path stays the path. What changes is that, for the first time, something holds change to them.

Declared intent has been the easy half for thirty years. The standard exists because the other half — enforcement, structural and universal — is the half that was never built.

The Infrastructure Operating Model standard, including the Govern function and its relationship to existing architecture practice, is maintained at theiom.org/standard/. Related reading in the whitepaper library: theiom.org/whitepapers/.

The IOM Standard is an open, vendor-neutral specification for modeling infrastructure intent, ownership, constraints, and meaning. Learn more at theIOM.org.