Abstract

This paper examines architecture as an activity rather than a role, artifact, or diagrammatic outcome. It argues that architecture is best understood as the construction and continual revision of structural forms that constrain, enable, and stabilise complex systems over time. Drawing on lessons from software engineering, organisational design, and distributed systems, the paper reframes architectural success and failure in terms of historical adequacy: the capacity of an architectural model to accommodate evolving variables without collapse, excessive mediation, or loss of semantic coherence.

Architecture as an Activity, Not a Position

In many organisations, architecture is treated as a positional role or a phase in a delivery lifecycle. Individuals are designated as architects, expected to produce designs, standards, or target-state diagrams that guide subsequent implementation. This framing obscures the nature of architecture as an ongoing activity embedded within system evolution.

Architecture, properly understood, is not something that is produced and then handed off. It is the continuous act of shaping structural constraints: defining what kinds of change are possible, where variation is permitted, and where invariants must be preserved. These constraints operate across technical, organisational, and semantic dimensions. Treating architecture as a role separates responsibility for structure from responsibility for consequence, encouraging speculative design disconnected from operational reality.

Architecture and Ontological Commitment

All architectural work entails ontological commitments. To architect a system is to assert what kinds of things exist within it, how they relate, and which distinctions matter. In software systems, these commitments appear as data models, interface boundaries, classification schemes, ownership rules, and lifecycle definitions.

Crucially, these ontologies are not neutral descriptions of reality. They are constructed from an embodied, historically situated perspective. They encode assumptions about scale, failure, trust, coordination, and authority that reflect the conditions under which they were created.

Architectural ontologies should therefore be treated as hypotheses rather than truths. Their validity is not established at design time, but tested through sustained use. A successful architectural ontology is one that continues to explain system behaviour and accommodate new requirements without requiring ad hoc exceptions or compensatory human coordination.

Structure, Form, and Constraint

Architecture operates by introducing structure in order to reduce uncertainty. Structure constrains freedom of action, but in doing so enables predictable composition and reuse. The relevant question is not whether constraints exist, but whether they are correctly placed.

Well-placed constraints:

- localise change
- make dependencies explicit

- stabilise shared semantics
- reduce the need for cross-context negotiation

Poorly placed constraints displace rather than eliminate complexity. They force actors to route around structure through informal agreements, duplication, or exception handling. In such cases, the architecture remains formally intact while the system becomes operationally incoherent.

Historicity and the Limits of Design-Time Authority

Because architectural structures are historically situated, no design can anticipate all future conditions. Attempts to do so typically result in over-specification: rigid models that perform well under initial assumptions but fail catastrophically as those assumptions erode.

This failure mode is often misdiagnosed as poor implementation or lack of discipline. In reality, it reflects an architectural overreach: the elevation of contingent design decisions to invariant principles.

Architectural authority should therefore be understood as provisional. Its legitimacy derives not from positional mandate or conceptual elegance, but from demonstrated explanatory power over time. When an architectural model no longer accounts for observed behaviour, it must be revised or abandoned, regardless of prior investment.

Architecture as Alignment to Concrete Artefacts

Architecture does not exist independently of the systems it structures. At any moment, the architecture of a software estate is fully determined by the artefacts that exist, the dependencies between them, and the constraints imposed by data, interfaces, and operational behaviour.

Architectural improvement proceeds by engaging directly with these artefacts. Boundaries become meaningful only when they are exercised repeatedly under change. Ownership emerges where responsibility and capability align around concrete systems rather than abstract roles.

Attempts to impose architectural structure in advance of artefact reality invert this relationship. They substitute conceptual boundaries for operational ones, increasing coordination cost without reducing complexity.

In practice, this misalignment is reinforced by static architectural documentation. Design records, target-state diagrams, and post-hoc rationales frequently describe an idealised system rather than the one that actually exists. Over time, these artefacts drift away from code and runtime behaviour, while retaining institutional authority. The result is not architectural clarity but narrative substitution: explanation replaces observation, and intent is confused with outcome.

Success Criteria for Architectural Activity

From this perspective, architectural success cannot be measured by diagram completeness, standard adherence, or target-state convergence. Appropriate success criteria include:

- Reduction in cross-team coordination required to complete changes
- Stability of core semantics under incremental extension

- Ability to introduce new functionality without revisiting foundational assumptions
- Declining reliance on informal exception handling
- Increased predictability of change impact

These indicators reflect whether the architecture is performing its primary function: absorbing complexity structurally rather than delegating it to human negotiation.

Crucially, architectural success must be observable. Claims about structure that cannot be traced to concrete artefacts, runtime behaviour, or change history lack evidential grounding. An architecture that cannot explain itself through the system's own operation has ceased to function as architecture and has become commentary.

Failure Modes and Their Symptoms

Architectural failure is rarely sudden. More often it manifests as:

- proliferating special cases
- growing dependency on tacit knowledge
- repeated re-litigation of boundary decisions
- increasing mediation by senior individuals
- divergence between documented architecture and lived reality

These symptoms indicate that the existing ontological model no longer aligns with practice. Persisting with the model under such conditions compounds failure by obscuring its cause.

A common contributing factor is reliance on static decision artefacts that are disconnected from system evolution. When architectural decisions are recorded as isolated texts rather than as properties of running systems, they become immune to falsification. The organisation retains the language of control while losing the ability to observe where and how that control has eroded.

Architecture, Adaptation, and Revision

Because architecture is hypothesis-like, revision is not a sign of weakness but of health. However, revision must be structural rather than cosmetic. Superficial adjustments that preserve underlying assumptions rarely restore coherence.

Effective architectural revision:

- re-examines core classifications and boundaries
- distinguishes invariants from conveniences
- retires abstractions whose compensatory value has expired
- re-aligns responsibility with operational control

For revision to be possible, architectural decisions must remain connected to observable system behaviour. “Living Decision Architecture” provides a practical mechanism for this connection. By grounding architectural understanding in telemetry, dependency graphs, and actual code change, decisions become continuously testable rather than historically frozen.

Such revision is disruptive, but less so than the unmanaged drift it replaces.

Conclusion

Architecture is an activity concerned with the creation and maintenance of structural forms that make complex systems governable over time. Its products are not designs or diagrams, but enduring constraints that stabilise meaning, localise change, and reduce the need for continuous human coordination.

Architectural ontologies are necessarily historical and provisional. Their success can only be judged through their continued capacity to accommodate evolving variables without collapse or distortion. This judgement must be grounded in evidence drawn from running systems rather than from static representations.

Living Decision Architecture reinforces this stance by replacing speculative justification with observable fact. It allows systems to explain themselves, making architectural structure auditable, revisable, and accountable to reality.

Architects who treat structure as timeless truth risk producing systems that are elegant in conception but brittle in practice. Architects who treat structure as a living hypothesis, continuously tested against embodied operation, contribute to systems capable of sustained coherence under change.