Philosophy & Technology manuscript No.
(will be inserted by the editor)

Architecture as Law: The Structural Necessity of the Fifth Condition for Artificial Intelligence

Anonymous Author(s)

Abstract Contemporary AI governance frameworks identify failures at the level of data, models, and control policies, but systematically miss a structural dimension: the organizing law that makes these components cohere as a governable system. This paper argues that existing frameworks—from the classical DIKW hierarchy to contemporary AI ethics—omit *Architecture*: the constitutive constraint system that specifies which relations among a system's cognitive components are permissible, enforceable, and auditable. Architecture is not a software engineering concept; it is a structural condition without which a system's components operate without coordination and its behavior cannot be governed.

The paper establishes three results. First, Architecture functions as a constitutive condition for coherent and governable artificial intelligence: a system lacking architectonic specification is not merely technically deficient but ungovernable—its behavior cannot be predicted, constrained, or audited in any principled sense. This claim is supported by three philosophical traditions that converge independently on the same structural concept: Kant's Architectonic of Pure Reason, Laozi's *dao* (道), and Aristotle's formal cause. Second, Architecture resists reduction to Floridi's levels-of-abstraction framework: LoA is an epistemic tool for describing systems at different granularities, while Architecture is a constitutive constraint on what system configurations are possible—a logically prior and distinct function. Third, Autonomy is a derived property of the Architecture–Control relation: a system exhibits autonomy precisely when its Architecture specifies conditions under which the Control layer may self-modify. Autonomy without architectonic bounds is not advanced intelligence but governance failure.

These results are formalized within the DIKCA framework (Data, Information, Knowledge, Control, Architecture) and carry direct implications for AI governance: the failure to specify Architecture is not an engineering oversight

Anonymous Institution

but a governance gap that makes safety constraints unenforced, responsibility attribution indeterminate, and accountability structurally impossible.

Keywords Architecture · Artificial Intelligence · DIKCA Framework · Constitutive Condition · AI Governance · Auditability · Autonomy · Philosophy of Technology · Floridi

1 Introduction

1.1 The Governance Problem

Contemporary AI governance faces a structural diagnosis problem. When an AI system produces harmful outputs, violates stated policies, or exhibits unsafe autonomous behavior, the standard response targets the epistemic layer: more data, better training, improved alignment procedures. This response is systematically misdirected when the failure lies not in what the system knows but in how its components are structurally organized.

The DIKW hierarchy (Data, Information, Knowledge, Wisdom)—the dominant conceptual framework for AI systems—reinforces this misdirection [1]. By representing intelligence as a progressive refinement of epistemic content, DIKW provides no conceptual location for the structural law that makes cognitive components cohere as a governable system. Governance failures that arise from the absence or misspecification of this structural law are, on the DIKW model, invisible—or worse, misattributed to epistemic deficiency.

1.2 The Missing Structural Dimension

This paper identifies and characterizes the missing dimension. Any artificial system that is to function as a coherent and governable agent must instantiate an *organizing structural law* that specifies the permissible relations among its data structures, information-processing operations, knowledge representations, and control mechanisms. Without such a law, the system's components operate without coordination: data may be processed by incompatible operations, control policies may be unenforceable, and the system's behavior may be indistinguishable from stochastic drift.

We call this organizing structural law *Architecture*—not in the engineering sense of a software design style, but in the philosophical sense of a constitutive condition: a structural requirement that must be in place for the system to function as a coherent, auditable, and governable agent. Architecture in this sense is not a design choice; it is a precondition for the possibility of governance.

1.3 The Three Contributions

The paper makes three principal contributions.

First, it establishes Architecture as a constitutive condition for governable artificial intelligence, distinguishing this from the stronger (and more vulnerable) claim that Architecture is necessary for all intelligence. The argument proceeds through the Coordination Problem (Section 2) and is supported by convergent philosophical traditions: Kant's Architectonic of Pure Reason, Laozi's *dao*, and Aristotle's formal cause (Section 3).

Second, it provides a formal characterization of Architecture within the DIKCA framework (Data, Information, Knowledge, Control, Architecture)—showing that Architecture is logically heterogeneous from the other four layers, and that Autonomy is a derived property of the Architecture–Control relation rather than an independent cognitive tier (Sections 4–5).

Third, it establishes that Architecture resists reduction to existing frameworks—specifically, that it is not captured by Floridi's levels-of-abstraction methodology, because LoA is an epistemic tool for describing systems while Architecture is a constitutive constraint on what systems can be (Section 8). This non-reducibility has direct implications for AI governance, auditability, and responsibility attribution (Section 6).

1.4 Paper Structure

The argument proceeds as follows. Section 2 develops the constitutive condition argument for Architecture, grounding it in the concept of governability. Section 3 traces the philosophical genealogy of Architecture as organizing law: Kant's Architectonic, Laozi's *dao*, and Aristotle's formal cause. Section 4 provides the formal characterization of Architecture within the DIKCA framework. Section 5 derives Autonomy as a property of the Architecture–Control relation. Section 6 connects Architecture to governance, auditability, and responsibility. Section 7 addresses five objections to the central thesis. Section 8 positions the paper relative to Floridi, Coeckelbergh, and contemporary AI safety frameworks. Section 9 concludes.

2 Architecture as a Constitutive Condition for Governable Intelligence

2.1 From Necessary Condition to Constitutive Condition

The claim advanced in this paper is not that Architecture is merely a useful design feature of intelligent systems, nor that it is one component among others that systems may or may not possess. Rather, the claim is that Architecture *functions as a constitutive condition* for a specific and practically significant class of intelligence: namely, intelligence that is *coherent, sustained, and governable.*

A constitutive condition, as distinct from an empirical property, is not something systems may possess to varying degrees; it is a condition that must

be in place for a certain kind of system to be possible at all—or, more precisely, for a system to qualify as a coherent and governable agent rather than a collection of interacting components. The question is not whether all systems exhibiting intelligent behavior explicitly instantiate Architecture, but whether a system can count as a *coherent and governable intelligent system* in the absence of an organizing structural law.

We argue that it cannot—and that this is the philosophically and practically consequential claim, not the broader (and more vulnerable) assertion that no form of intelligence can exist without explicit Architecture.

2.2 The Coordination Problem

Consider a system S that contains the following capacities: data access, mechanisms for transforming data into structured representations, the ability to store and reuse representations over time, and policies that constrain or guide outputs. Such a system possesses what are commonly identified as the core functional capacities of artificial intelligence. However, these capacities alone do not guarantee that the system operates as a *unified agent*. A further condition must be satisfied: the relations among these capacities must be *coordinated*.

Without coordination: data may be processed by incompatible operations; information outputs may fail to update knowledge structures; knowledge states may not activate relevant control policies; control policies may not be enforceable over system behavior. A system in which such coordination is absent does not fail gradually; it fails categorically—it ceases to function as a system and becomes an aggregate of partially interacting components.

2.3 The Heap Argument

This motivates what we term the *Heap Argument*. A collection of components does not constitute a system merely by virtue of co-presence. What distinguishes a system from a heap is the existence of a structural principle that specifies the permissible relations among its components.

To illustrate: consider a large corpus of data, sophisticated information-processing mechanisms, extensive knowledge representations, and robust control policies—none of which are organized by a structural law specifying their permissible relations. Which data is valid input for which information-processing operation? Which information outputs update which knowledge representations? Which knowledge states activate which control policies? Without answers to these questions, S does not exhibit intelligence. It exhibits *component operation* in isolation. Architecture is the structural law that transforms a heap into a system.

Thesis 1 (Architectonic Constitution)**.** *For any coherent and governable intelligent system S, there exists an Architecture $\mathcal{A}$ such that $\mathcal{A}$ specifies the*

permissible relations among S's data structures, information-processing operations, knowledge representations, and control mechanisms, and S's behavior is governed by $\mathcal{A}$.

2.4 Architecture as Condition of Governability

The constitutive necessity of Architecture becomes particularly clear when the notion of *governability* is introduced. A system is governable if its behavior can be predicted within bounded uncertainty, constrained by enforceable rules, and audited with respect to those rules. These properties do not arise from data, information, knowledge, or control taken in isolation. They arise from the *structured relation* among them.

A system lacking an explicit or implicit Architecture may exhibit intelligent behavior in a local or transient sense—connectionist systems, for instance, exhibit remarkable capabilities without formally specified architectural layers. However, such behavior is not systematically *governable*: without an organizing structural law, it is indistinguishable, from the perspective of an external auditor, from stochastic drift or uncontrolled adaptation. This is not a limitation of connectionist systems per se, but a consequence of the absence of explicit architectonic specification.

Thesis 2 (Constitutive Necessity of Architecture)**.** *Architecture functions as a constitutive condition for coherent and governable artificial intelligence. Any system that qualifies as such must instantiate an organizing structural law that specifies the permissible relations among its cognitive components. This claim does not extend to all forms of intelligent behavior, but to the class of systems that can be meaningfully governed, audited, and held accountable.*

The relation between Thesis 1 and Thesis 2 is as follows. Thesis 1 (Architectonic Constitution) establishes the structural claim: any coherent and governable system S instantiates some Architecture $\mathcal{A}$. Thesis 2 (Constitutive Necessity) draws the normative consequence: since governability requires Architecture, the absence of architectonic specification is not merely a technical deficiency but a governance failure—the system cannot be meaningfully audited, constrained, or held accountable.

2.5 Architecture as Structural Necessity, Not Design Choice

A critical distinction separates two senses of "architecture": *Architecture as design choice* (the engineer's selection of microservices over monolith, or transformer over recurrent network) versus *Architecture as structural necessity* (the organizing law that any coherent and governable system must instantiate). The former is contingent; the latter is constitutively necessary—a structural requirement, not a design option.

This paper is concerned exclusively with the latter. The claim is not that some architectures are better engineering choices than others (though this is

true). The claim is that any system that is to function as a coherent, auditable, and governable intelligent system must instantiate *some* Architecture—some structural law governing the relations among its cognitive layers—whether or not this law is explicit, well-specified, or designed. A system with a *poorly specified* Architecture still has Architecture; it simply has a deficient one, with predictable governance consequences. And a system with no architectonic specification at all is not a candidate for governance, because governance requires a structural target.

3 Philosophical Genealogy: Three Traditions

The concept of Architecture as the organizing structural law of a cognitive system is not new. It has deep roots in three philosophical traditions that have developed the idea independently, from different starting points and for different purposes. Their convergence on a structurally similar concept provides significant philosophical support for the necessity claim.

3.1 Kant's Architectonic of Pure Reason

Kant's *Critique of Pure Reason* devotes a section—the "Architectonic of Pure Reason"—to the systematic organization of the faculties of cognition [9]. For Kant, an Architectonic is "the art of constructing systems," and a system is "the unity of the manifold modes of knowledge under one idea." Crucially, this unifying idea is not itself a piece of knowledge within the system; it is the *principle of organization* that makes the collection of knowledge a system rather than a rhapsody (Kant's term for an unsystematized aggregate).

Three features of Kant's Architectonic are directly relevant to our argument:

1. **The Architectonic is a structural necessity, not an optional supplement.** Kant does not say that a well-organized cognitive system may optionally have an Architectonic. He says that a *system*—as opposed to an aggregate—necessarily has one. The Architectonic is constitutive of systematicity.
2. **The Architectonic operates at a different level from the knowledge it organizes.** The organizing principle is not knowledge-content; it is the formal structure that governs how knowledge-contents cohere. This is a direct precursor to our claim that Architecture is logically heterogeneous from Data, Information, Knowledge, and Control.
3. **The Architectonic determines what counts as a valid component of the system.** Not any proposition can enter Kant's system of pure reason; the Architectonic specifies which concepts are valid, which inferences are legitimate, and how the faculties relate. This is structurally identical to our claim that Architecture specifies permissible data structures, information-processing operations, knowledge representations, and control mechanisms.

3.2 Laozi's *Dao* as Structural Principle

The concept of *dao* (道) in the *Daodejing* offers a structurally parallel account of the organizing principle of a complex system [11]. The *Daodejing* opens: "The dao that can be named is not the eternal dao." This is not mysticism; it is an epistemic claim about the organizing principle of a system: the structural law that makes things cohere cannot itself be fully captured in the categories of the things it organizes.

Several features of *dao* are structurally relevant:

1. **Dao is not above things; it is through them.** A persistent misreading treats *dao* as a transcendent principle standing above the phenomena it governs. The *Daodejing* is consistent about the opposite: *dao* operates *through* things, not above them. It is the structural law that is immanent in the system—the way water flows downward not because a law is imposed from outside but because the structural character of water and terrain make this the natural path.
   In terms of the DIKCA framework: Architecture does not stand above Data, Information, Knowledge, and Control as a fifth tier in a linear progression. It is the organizing law *through which* they cohere—the architectonic envelope rather than the apex.
2. **Dao is objective, not constructed.** The *dao* is not invented by the sage; it is discovered. The sage who understands *dao* has not created a new principle but has discerned the structural necessity that was already operative. This maps directly onto our claim that Architecture-as-structural-law is a necessity, not a design choice: the architectural law of a functional intelligent system is not invented by its designer but is the discovered structural requirement for coherent intelligence.
3. **Deviation from dao produces systemic failure.** Systems that violate the structural law of their domain do not simply produce suboptimal outcomes; they lose coherence. This is the *Daodejing*'s explanation of organizational and political failure: not bad intentions but deviation from structural principle. The governance consequences of architectural failure in AI systems are analogous: not merely poor performance but systemic incoherence in the relations among cognitive layers.

3.3 Aristotle's Formal Cause

Aristotle's account of causation distinguishes four causes: material, efficient, formal, and final [2]. The formal cause is the *form* or *structural principle* that makes a thing what it is—not the matter from which it is made (material cause), not the process by which it was made (efficient cause), and not the purpose for which it was made (final cause), but the structural principle that constitutes it as a thing of its kind.

For Aristotle, the formal cause of a bronze statue is not the bronze (material cause) or the sculptor (efficient cause) but the *form* of the statue—the structural principle that organizes the bronze into a statue rather than a heap of metal.

Applied to intelligent systems:

- Material cause: the computational substrate (processors, memory, networks).
- Efficient cause: the training and design processes.
- Final cause: the task the system is designed to perform.
- Formal cause: the Architecture—the structural principle that organizes the computational substrate into a coherent intelligent system rather than a heap of computations.

The formal cause is precisely what standard AI frameworks omit. They specify material causes (computational requirements), efficient causes (training procedures and algorithms), and final causes (benchmark performance on designated tasks), but not the formal cause: the organizing structural law that makes the system an intelligence, not merely a computation.

3.4 Convergence

These three traditions—Kant's Architectonic, Laozi's *dao*, and Aristotle's formal cause—converge on a structurally identical concept: there is a dimension of organized systems that is neither the content of the system (what it knows, what it can do) nor the process by which the system was created, but the structural law that organizes content and process into a coherent whole. This convergence, across traditions separated by two millennia and three continents, provides significant philosophical support for the claim that Architecture is a genuine structural necessity, not a contingent engineering concern.

4 Formal Characterization within DIKCA

4.1 The DIKCA Framework

The DIKCA framework (**D**ata, **I**nformation, **K**nowledge, **C**ontrol, **A**rchitecture) extends the classical DIKW hierarchy by identifying Control as an irreducible regulatory layer—logically heterogeneous from knowledge (K≠C)—and Architecture as the organizing structural law that constitutes D, I, K, and C as a unified intelligent system [7].

The logical structure of DIKCA is *not* a linear progression in which Architecture is a fifth step following Control. It is a four-layer operational stack (D→I→K→C), each layer processing the outputs of its predecessor and governed throughout by an architectonic envelope (A) that specifies the permissible relations among all four:

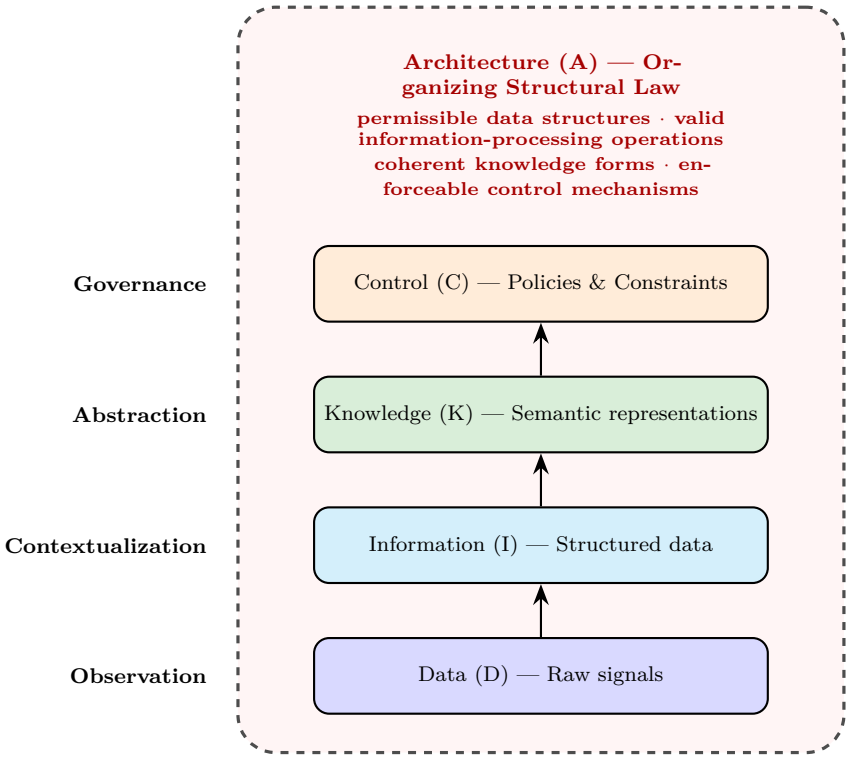

Fig. 1 The DIKCA framework. Architecture (A) is the organizing structural law that constitutes the D/I/K/C layers as a unified intelligent system. It is not a fifth step in a linear progression but the architectonic envelope governing the permissible relations among all four layers. Autonomy is a derived property of the A–C relation (see Section 5).

4.2 Formal Definition

Conceptual Clarification 1 (Architecture)**.** The Architecture $\mathcal{A}$ of an intelligent system S is the organizing structural law that specifies:

(i) $\mathcal{A}_D$: the set of permissible data structures and input representations for S;
(ii) $\mathcal{A}_I$: the set of valid information-processing operations and their domain–range constraints;
(iii) $\mathcal{A}_K$: the coherence conditions for knowledge representations, including consistency requirements and update protocols;
(iv) $\mathcal{A}_C$: the enforceability conditions for control mechanisms, specifying which policies can be instantiated given the system's D, I, K layers;
(v) $\mathcal{A}_{\text{rel}}$: the permissible relations among (i)–(iv), specifying which data structures are valid inputs to which information-processing operations, which information outputs may update which knowledge representations, and which knowledge states activate which control policies.

$\mathcal{A}$ is a structural necessity: any functional system S instantiates some $\mathcal{A}$, adequate or deficient.

4.3 Architecture as Audit Target

A significant practical consequence of Definition 1 is that Architecture is an *audit target*: it is in principle possible to examine a deployed AI system and determine whether its Architecture is well-specified. The questions are precise:

- Are the permissible data structures ($\mathcal{A}_D$) specified and enforced? (If not: data quality failures are architecturally unaddressed.)
- Are the information-processing operations ($\mathcal{A}_I$) constrained by domain–range specifications? (If not: the system may apply operations outside their valid domains.)
- Are the knowledge coherence conditions ($\mathcal{A}_K$) specified? (If not: the knowledge layer may accumulate inconsistent representations.)
- Are the control enforceability conditions ($\mathcal{A}_C$) specified? (If not: control policies are present but not enforced— the K-saturation/C-absence failure mode identified in **(author?)** [7].)
- Are the cross-layer relations ($\mathcal{A}_{\text{rel}}$) specified? (If not: the components operate without coordination.)

This auditability is precisely what Wisdom—as traditionally conceived—lacks. Wisdom cannot be examined, quantified, or corrected through structural intervention. Architecture can.

5 Autonomy as Derived Property

5.1 The Standard Account and Its Problems

Autonomy is widely treated in AI philosophy as either the highest tier of intelligence (replacing Wisdom in DIKW extensions) or an independent property of advanced AI systems [5]. The standard account characterizes autonomy as the capacity of a system to modify its own control structures—meta-control.

This account correctly identifies what autonomy *does* but misidentifies what it *is*. Treating Autonomy as an independent tier implies that a system can acquire meta-control capacity independently of whether its Architecture specifies the bounds and conditions under which such self-modification is permissible. This has direct and negative governance consequences: a system exhibiting unconstrained meta-control—modifying its own policies without architectonic specification of the bounds of such modification—is not an autonomous intelligence. It is an uncontrolled one.

5.2 Derivation

Conceptual Clarification 2 (Autonomy as Derived Property)**.** A system S exhibits *autonomy* at degree α if and only if:

(i) S's Architecture $\mathcal{A}$ specifies conditions $\mathcal{A}_{\text{meta}} \subseteq \mathcal{A}_C$ under which the Control layer may modify itself—defining the permissible scope, rate, and bounds of C-layer self-modification;
(ii) S's Control layer executes self-modifications that satisfy $\mathcal{A}_{\text{meta}}$; and
(iii) α is a monotone function of the scope specified in $\mathcal{A}_{\text{meta}}$.

Autonomy is thus fully determined by the Architecture–Control relation; it is not an independent cognitive property.

Argument 1 (Autonomy without Architecture is incoherent)**.** *A system S for which no $\mathcal{A}_{meta}$ is specified cannot exhibit autonomy in any governable sense. Self-modification of the Control layer without architectonic bounds is indistinguishable, from the perspective of an external auditor, from arbitrary behavioral drift.*

5.3 Why This Matters for Governance

The derivation of Autonomy from Architecture has three immediate governance implications.

First: *autonomy risk is architecturally addressable.* A system whose Architecture does not specify $\mathcal{A}_{\text{meta}}$ has a diagnosable architectural gap, not a capability excess. The remedy is architectural specification, not capability restriction.

Second: *autonomy levels are comparable across systems.* If Autonomy is a function of $\mathcal{A}_{\text{meta}}$, then two systems can be compared on the basis of the scope their Architectures permit for control self-modification. This supports systematic regulatory classification of AI systems by their architecturally specified autonomy degree.

Third: *moral responsibility attribution is architecturally grounded.* When an autonomous system modifies its own control structures in ways that produce harmful outputs, responsibility can be traced to the Architecture: was $\mathcal{A}_{\text{meta}}$ absent (architectural failure), underspecified (design failure), or violated by the C-layer (enforcement failure)?

6 Architecture, Governance, and Responsibility

6.1 The Governance Gap

Contemporary AI governance faces a structural problem: governance frameworks specify requirements at the level of outcomes (what the system should not do) or values (what the system should embody), but lack a structural vocabulary for specifying and auditing the mechanisms through which outcomes are produced and values implemented. DIKCA provides this vocabulary; Architecture is its organizational center.

An AI governance framework that specifies only K-layer requirements (what the system should know, what values it should represent) and C-layer

requirements (what the system should and should not do) without specifying A-layer requirements (the structural law governing how K and C are related, and under what conditions C may self-modify) is governance with a missing dimension. Such frameworks produce well-stated principles that cannot be operationalized as enforceable architectural requirements.

6.2 Architectonic Rights as a Political Concept

The concept of Architecture as structural law has a political dimension that DIKCA makes explicit. If Architecture specifies which data structures are permissible, which information-processing operations are valid, which knowledge forms are coherent, and which control mechanisms are enforceable, then *whoever specifies the Architecture exercises a form of structural authority over the intelligent system.*

This is architectonic authority—the power to determine the organizing law of a cognitive system. In contemporary AI deployment, architectonic authority is concentrated in a small number of platform providers who design the structural laws of foundation models and their interfaces. Governance frameworks that ignore Architecture have no conceptual tools for addressing this concentration of architectonic authority. DIKCA makes it visible [7, 8].

6.3 Responsibility Trails the Architecture

Responsibility for AI system behavior, on the DIKCA account, is distributed across the layers at which governance is exercised. K-layer responsibility belongs to those who determine what the system knows and represents. C-layer responsibility belongs to those who specify and enforce the policies governing system behavior. A-layer responsibility—architectonic responsibility—belongs to those who specify the organizing structural law: which components are valid, how they relate, and under what conditions control self-modification is permitted.

Architectonic responsibility is the deepest form of AI responsibility because it determines the conditions of possibility for all other layers. A system with a deficient Architecture cannot compensate with superior knowledge or better control policies; the structural law governs what knowledge and control can do.

7 Objections and Replies

The claim that Architecture is a constitutive condition for coherent and governable intelligence invites several important objections. This section considers the most significant and provides responses.

7.1 Objection 1: Connectionist Systems Lack Architecture

A natural objection arises from connectionist models, particularly deep neural networks. These systems appear to exhibit intelligent behavior without an explicit architectural specification of the relations among data, information, knowledge, and control. If connectionist systems are genuinely intelligent, the constitutive necessity claim appears false.

Reply. The objection conflates *explicit* Architecture with *absence* of Architecture. Connectionist systems do not lack architectural constraints; their training dynamics, loss functions, optimization trajectories, and representational regularities collectively impose a structural organization on how inputs are transformed, how representations are formed, and how outputs are generated. These constraints function as an implicit Architecture.

The governance problem this paper identifies is not the metaphysical absence of Architecture in connectionist systems, but its *opacity*. An implicit Architecture—one that cannot be inspected, formally specified, or systematically revised—is not governable in the relevant sense. The inaccessibility of architectural constraints in such systems is precisely what motivates the central claim: making Architecture explicit is not an optional engineering improvement but a precondition for meaningful audit, regulatory accountability, and democratic oversight.

7.2 Objection 2: Emergent Intelligence Does Not Require Structure

A second objection appeals to emergence. Complex systems may exhibit intelligent behavior arising from simple interactions without any central organizing principle. Swarm intelligence and evolutionary systems are cited as examples.

Reply. Emergence does not eliminate structural organization; it redistributes it. Even in emergent systems, constraints govern interaction patterns, state transitions, and system evolution. These constraints function as a distributed Architecture.

What emergence challenges is not the necessity of Architecture, but the assumption that Architecture must be centrally designed or explicitly represented. The present account is fully compatible with distributed and emergent forms of Architecture. The governance consequence, however, remains: an Architecture that is distributed and emergent is one that is harder to inspect and govern—a practical argument for making it explicit, not a philosophical argument against its necessity.

7.3 Objection 3: Embodied and Enactive Accounts

Embodied and enactive approaches to cognition argue that intelligence arises through interaction with the environment, rather than from internal representational structure [14]. On these accounts, the relevant organizing structure is in the agent-environment coupling, not in an internal architectonic law.

Reply. This objection correctly emphasizes the importance of interaction, but it does not eliminate the need for structural organization. Interaction itself is governed by constraints: what actions are possible, how perception is coupled to action, how feedback modifies behavior. These constraints constitute an Architecture at the level of agent–environment coupling. The present framework can therefore be extended to include not only internal system organization but also the structural organization of interaction—a move that strengthens rather than undermines the central claim.

7.4 Objection 4: Architecture is Just Engineering Design

A further objection is that "Architecture" is simply a term from software engineering, referring to contingent design choices rather than a philosophically significant concept. On this view, the paper inflates an engineering metaphor into a philosophical claim.

Reply. This objection rests on an equivocation between two distinct senses of the term. In engineering practice, architecture refers to design patterns or implementation strategies—choices about how to build a system. In this paper, Architecture refers to a *constitutive condition*: the structural law that makes a system possible as a coherent whole.

The distinction is analogous to that between a particular bridge design (engineering choice) and the structural principles that make any bridge possible at all (physical law). This paper concerns the latter. The bridge engineer may choose between a suspension and a cable-stayed design; the laws of structural mechanics are not chosen.

7.5 Objection 5: The Claim is Trivially True

Finally, one might argue that the claim is trivially true—any system must have *some* form of organization, so the necessity of Architecture adds nothing substantive to what is already acknowledged.

Reply. The claim is not that systems are organized (trivially true), but that the organizing principle is: (i) conceptually distinct from data, information, knowledge, and control; (ii) a legitimate target of philosophical analysis, governance design, and regulatory specification; and (iii) the locus at which the deepest form of responsibility for AI systems is located.

What is non-trivial is the elevation of this organizing principle to a first-class theoretical construct with determinate implications for AI governance, auditability, and responsibility attribution. Without this conceptual distinction, structural failures of AI systems are systematically misdiagnosed as deficiencies in data, models, or policies, rather than as failures of architectonic specification.

8 Dialogue with Existing Frameworks

8.1 Floridi's Philosophy of Information

Floridi's levels-of-abstraction (LoA) methodology offers a sophisticated account of how systems can be described at different levels of analysis [6]. DIKCA's layers are not levels of abstraction in this sense: they are logically heterogeneous functional categories— ontologically different kinds of thing (data, information, knowledge, control, architecture) that co-inhabit the same computational substrate.

Floridi's framework treats values and norms as informationally embedded— as content that can in principle be represented and evaluated at an appropriate level of abstraction. Architecture, in our account, cannot be reduced to informational content at any level of abstraction. It is the organizing condition for informational content, not a species of it. This is a substantive disagreement about the ontology of the structural law, not a terminological one.

8.1.1 Why Architecture Resists Reduction to Levels of Abstraction

A natural objection to the notion of Architecture advanced in this paper is that it can be subsumed under Floridi's LoA methodology: Architecture would simply correspond to a higher-order LoA description of system organization. We argue that this reduction is not tenable. Architecture resists reduction within the LoA framework for a principled reason: it is not a level of abstraction, but a *constraint on the space of possible abstractions*. This distinction is decisive.

In Floridi's framework, a Level of Abstraction is a method for describing a system by selecting a relevant set of observables and ignoring others. Different LoAs correspond to different observational perspectives on the same underlying system. Crucially, LoAs are *epistemic partitions*: they structure how a system is described, not how it is constituted. They are representational tools for managing informational access.

By contrast, Architecture as defined in this paper is not a descriptive selection over observables. It is a set of *constitutive constraints* that determine which system states, transitions, and inter-layer relations are *possible at all*. While a Level of Abstraction selects a subset of observable predicates over a fixed system S, Architecture defines the constraint space $\mathcal{C}_A$ such that system states must satisfy $\mathcal{C}_A$, transitions between states must preserve $\mathcal{C}_A$, and inter-layer mappings are only valid if they respect $\mathcal{C}_A$. Thus, Architecture does not describe the system at a level; it defines the boundary of what counts as a valid system configuration.

The non-equivalence of epistemic and constitutive structure becomes clear when we note that LoAs operate over a fixed system, whereas Architecture partially defines what counts as a system in the first place. A Level of Abstraction may describe whether a system has a memory, but it does not determine whether memory can influence control. A Level of Abstraction may describe information flow between components, but it does not determine whether

such flows are enforceable constraints. Architecture specifies exactly these enforceability conditions.

This distinction places Architecture outside the ontological scope of LoA-based analysis without contradicting its epistemological utility. The implication is not that Floridi's framework is incorrect, but that it is incomplete with respect to system constitution in artificial intelligence. LoA is sufficient for epistemic analysis of informational systems, but insufficient for capturing the structural constraints that govern multi-layer cognitive architectures. Architecture therefore extends, rather than replaces, the philosophy of information: it introduces a constitutive dimension that is invisible at the level of abstraction, but necessary for understanding governable artificial agency.

8.2 Coeckelbergh's Relational Approach

Coeckelbergh's phenomenological approach locates responsibility in relations and practices rather than in internal system properties [4]. This insight is compatible with DIKCA but gains precision from it: relations and practices are shaped by the Architecture that governs what the system can do and how its components relate.

Architectonic authority—the power to specify the organizing law— is a form of relational power that operates prior to any particular interaction. Understanding why particular human-AI interactions have the character they do requires access to the Architecture that constitutes the system as such.

8.3 Contemporary AI Safety Frameworks

Contemporary AI safety research addresses the problem of ensuring that AI systems behave in accordance with human values and intentions as systems become more capable. The dominant approach focuses on reward modeling, constitutional AI, and alignment via human feedback. These are C-layer interventions: they specify and enforce policies governing system behavior.

DIKCA identifies a structural gap in this approach: C-layer interventions are only as robust as the Architecture that specifies their scope and the conditions under which they can be overridden. A safety constraint that is architectonically unspecified—present as a control policy but not grounded in an architectonic specification of its enforceability conditions—is vulnerable to architectural failure modes that C-layer analysis cannot diagnose. Architectonic safety is a necessary complement to alignment research.

9 Conclusion

This paper has argued that Architecture is a constitutive condition for coherent and governable artificial intelligence, that it has deep roots in three philosophical traditions (Kant's Architectonic, Laozi's *dao*, Aristotle's formal cause), that it

is formally characterizable as the organizing structural law within the DIKCA framework, and that Autonomy is a derived property of the Architecture–Control relation rather than an independent cognitive tier.

The argument has three principal implications.

For AI philosophy: the standard frameworks for understanding artificial intelligence—DIKW and its variants—are structurally incomplete. They identify the contents of intelligent systems without identifying the organizing law that makes those contents cohere as a system. Architecture is that law; its omission has produced systematic conceptual confusion in debates about wisdom, autonomy, and agency.

For AI governance: governance frameworks that specify only K-layer (epistemic) and C-layer (behavioral) requirements without specifying A-layer (architectonic) requirements are missing the structural dimension of governance. Architecture is the level at which the deepest governance decisions—about the organizing law of cognitive systems—are made. Making those decisions explicit, auditable, and democratically accountable is among the most urgent tasks in AI policy.

For AI safety: autonomy risk is architecturally addressable. A system whose architecture does not specify the bounds of control self-modification is not dangerously autonomous but architecturally deficient. The remedy is architectural specification, not capability restriction. This reframes the AI safety problem in structural terms that are more tractable than capability-level interventions alone.

DIKCA is not a finished theory of intelligence. It is a framework for asking more precise questions about what intelligent systems are, how they are constituted, and who bears responsibility for the structural laws that govern them. The most important of those questions—the architectonic question—has, until now, been absent from the discourse. This paper is an attempt to make it present.

References

1. Ackoff, R. L. (1989). From data to wisdom. *Journal of Applied Systems Analysis*, 16(1), 3–9.
2. Aristotle. *Metaphysics*. Trans. W. D. Ross. Oxford University Press, 1924.
3. Ashby, W. R. (1956). *An Introduction to Cybernetics*. Chapman and Hall.
4. Coeckelbergh, M. (2020). *AI Ethics*. MIT Press.
5. Dignum, V. (2019). *Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way*. Springer.
6. Floridi, L. (2014). *The Fourth Revolution: How the Infosphere Is Reshaping Human Reality*. Oxford University Press.
7. [Author(s)]. (2025). Intelligence beyond knowledge: Control, autonomy, and the logical architecture of artificial agency. Preprint, PhilArchive. [URL omitted for blind review] *Revised and extended version under review.*

8. [Author(s)]. (in preparation). The political economy of cognitive infrastructure. *Manuscript in preparation*.
9. Kant, I. (1781/1998). *Critique of Pure Reason*. Trans. P. Guyer and A. Wood. Cambridge University Press.
10. Kant, I. (1788/1997). *Critique of Practical Reason*. Trans. M. Gregor. Cambridge University Press.
11. Laozi. *Daodejing*. Trans. D. C. Lau. Penguin Classics, 1963.
12. Ryle, G. (1949). *The Concept of Mind*. Hutchinson.
13. Searle, J. R. (1980). Minds, brains, and programs. *Behavioral and Brain Sciences*, 3(3), 417–424.
14. Wiener, N. (1948). *Cybernetics: Or Control and Communication in the Animal and the Machine*. MIT Press.
15. Wittgenstein, L. (1953). *Philosophical Investigations*. Trans. G. E. M. Anscombe. Blackwell.