

4+1 AI Platform RFP Framework (Open Edition)

A Practitioner-Aligned Evaluation Framework for Enterprise AI Platforms

Grounded in the CTO Advisor 4+1 Layer AI Infrastructure Model

Version v1.1 (Open Edition)

December 2025

Executive Summary — For Procurement, Legal, and Non-Technical Stakeholders

This document introduces a **layered, vendor-neutral RFP framework** for evaluating enterprise AI platforms using the CTO Advisor **4+1 Layer AI Infrastructure Model**. It provides a structured way to assess vendors on **risk, governance, cost transparency, policy portability, data rights, reasoning-plane maturity**, and long-term platform viability.

Most “AI platforms” only deliver one or two layers of the stack. This RFP prevents vendor overstatement by forcing explicit mapping of capabilities to the model’s layers:

- Layer 0 — Compute
- Layer 1 — Data Plane (1A/1B/1C)
- Layer 2 — Operational Tri-Plane (2A/2B/2C)
- Layer 3 — Application Layer

The RFP includes:

- A full question set covering all layers
- A powerful **Strategic Risk Assessment**
- A complete glossary
- A diagram canonical to this model
- Deployment considerations
- A procurement-focused red flags list

This RFP is designed to be **copied into real procurement cycles**, shared with vendors, and used internally between architecture, security, procurement, and legal teams.

1. Introduction

Purpose of This RFP

Enterprises evaluating AI platforms face a familiar challenge:
vendors claim to deliver “everything” while actually delivering fragments.

This RFP solves that problem by grounding evaluation in the CTO Advisor **4+1 Layer AI Infrastructure Model**, ensuring buyers can:

- Identify which layers a vendor truly covers
- Reveal dependencies and integration expectations
- Understand governance and residency implications
- Expose hidden lock-in patterns
- Clarify reasoning-plane maturity
- Ask questions aligned to multi-year platform risk

This RFP is written **for practitioners by practitioners**—CIOs, CTOs, architects, procurement leaders, and security teams who need clarity, not marketing.

Canonical Model Reference

The underlying model is fully described here:

The CTO Advisor 4+1 Layer AI Infrastructure Model

<https://thectoadvisor.com/blog/2025/11/05/the-cto-advisor-41-layer-ai-infrastructure-model/>

Why Layered Evaluation Matters

AI platforms are rarely monolithic. A vendor claiming to offer “an AI OS” may actually provide:

- only a runtime
- or only vector storage
- or only a control plane
- or an application layer dependent on others’ infrastructure

Evaluating them without a layered model is a recipe for:

- misaligned expectations
- hidden costs

- operational gaps
- unplanned hiring needs
- technical and contractual lock-in
- compliance exposure

The 4+1 model turns ambiguous platform claims into **structured architecture**.

Origin of the 4+1 Model (“The Why”)

While migrating a cloud-native AI application from a hyperscaler to bare-metal DGX servers, it became clear the application behaved differently—not because compute changed, but because a massive piece of the puzzle was missing:

The hyperscaler’s invisible Reasoning Plane.

Hyperscalers silently provide:

- placement
- autoscaling
- multi-objective routing
- data locality decisions
- compliance-aware placement
- managed stateful orchestration
- policy enforcement

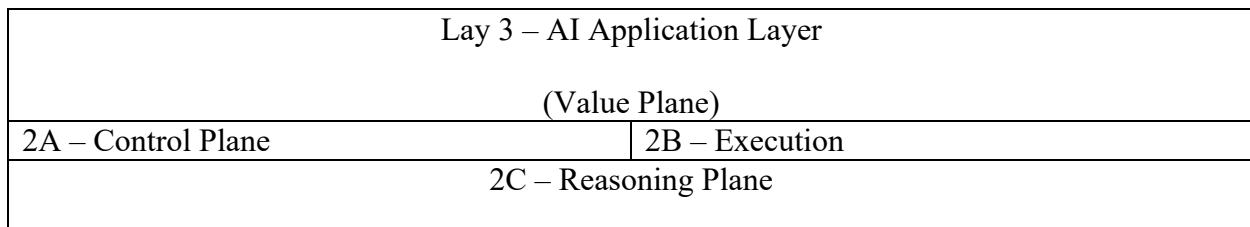
Enterprises **assume** these capabilities simply “come with Kubernetes,” but they do not.

The 4+1 model names and frames this missing layer as **Layer 2C — Agentic Infrastructure**.

This RFP operationalizes that insight.

Canonical Diagram

This diagram is the standard reference for this framework:



About This Open Edition

This edition includes:

- The **full RFP structure**
- All question sets
- All risk assessments
- The full diagram and glossary
- Red flags for procurement teams

This edition **purposely excludes**:

- scoring
- weighting
- vendor maturity models
- competitive benchmarks

Those appear in the **Buyer Room Edition (Internal Use Only)**.

2. How to Use This RFP

For Architecture Teams

Use the RFP to force vendors to reveal:

- what layers they cover
- what layers you must build
- which assumptions and dependencies they require
- how their reasoning model behaves under constraints

For Procurement and Legal

Start with:

- **Strategic Risk Assessment (Section 2.2)**
- **Red Flags (Section 13)**
- **Data Rights (2.2.4)**
- **Vector Lock-In (2.2.1)**

- **Policy Portability (2.2.3)**

Escalate to architecture only when needed.

Integrated Data Platforms Must Answer Multiple Sections

If a vendor covers 1A, 1B, and 1C, they must answer each layer independently.

2.1 Determining Scope: Which Layers Matter

You don't need every layer for every RFP. Use the maturity stages:

Phase 1 — GenAI Assistants

Evaluate:

- 1A
- 1B
- 2B

Phase 2 — Multi-Team Platformization

Add:

- 1C
- 2A

Phase 3 — Autonomy & Multi-Cloud

Add:

- 2C
- 3

Important Clarification

Layer 2C always exists when you use a hyperscaler.

Phasing reflects when buyers must **explicitly own or evaluate** it.

2.2 Strategic Risk Assessment

The four risks that entrap enterprises:

2.2.1 Vector Lock-In (Layer 1B)

Key Questions

1. Can we export embeddings, vectors, metadata, and indexes **directly** without re-embedding?
 2. Are embedding formats **open and interoperable** with standard vector DBs?
 3. Are there proprietary transformations that break portability?
-

2.2.2 Autonomous Compliance Liability (Layer 2C)

Key Questions

4. If the reasoning engine violates residency or compliance, **who is liable**?
 5. Do you provide **cryptographically verifiable audit trails** of each decision?
-

2.2.3 Policy Portability (Layer 2C)

Key Questions

6. Is your policy language **open standard** (OPA/Rego/CUE/YAML/Python)?
 7. Can policies be stored/versioned via **GitOps**?
 8. Are policies portable if we migrate?
-

2.2.4 Data Rights & Model Contamination (Layer 0 & 2B)

Key Questions

9. Do you claim rights to use our data, prompts, telemetry, or embeddings to train models?
 10. Can we **fully opt out** without breaking functionality?
 11. How is opt-out enforced technically?
-

3. The 4+1 Layer AI Model — Detailed Overview

Layer 0 — Compute & Network Fabric

Responsibilities

- Hardware acceleration
 - Network fabric
 - Storage throughput
 - Sustainability
-

Layer 1 — Data Plane

1A — Data Storage & Governance

- Governed durable storage
- Lineage
- Metadata
- Policy enforcement

1B — Retrieval Context & Indexing

- Embeddings
- Indexing
- Relevance signals
- Vector formats
- Portability

1C — Data Movement & Pipelines

- ETL/ELT
 - Streaming
 - Cost-aware movement
 - Pipeline lineage
-

Layer 2 — Operational Tri-Plane

2A — Control Plane

- Provisioning
- Scheduling
- Quotas and RBAC
- Cluster lifecycle

2B — Execution Plane

- Model serving
- Workflow execution
- Runtime isolation
- Telemetry
- Secrets

2C — Reasoning Plane (Agentic Infrastructure)

This is the **core differentiator**.

The reasoning plane:

- makes multi-objective decisions
- uses governance + telemetry + cost + residency
- acts autonomously
- requires kill switches and simulation modes

2C vs Kubernetes Operators (Differentiation Table)

Aspect	Kubernetes Operator	Layer 2C Reasoning Plane
Scope	Single cluster	Multicluster / multi-cloud
Primary Objective	Keep resource healthy	Optimize across cost/latency/residency/SLA
Inputs	Pod/Node metrics	Governance, metadata, cost, telemetry, identity
Decision Model	One resource type	Cross-layer reasoning
Language	Arbitrary Go code	Enterprise policy layer
Auditability	Logging only	Decision audit trails
Failure Modes	Restart loops	Global policy violations

Layer 3 — AI Application Layer (Value Plane)

Responsibilities

- Copilots

- Agents
 - Business semantics
 - Workflow orchestration
 - UX integration
-

4. Vendor Response Template (Open Edition)

A. Layer Coverage Declaration

Which layers you cover, integrate, or rely on.

B. Technical Approach

(No marketing language.)

C. Architecture Diagrams

Data and control flow.

D. Design Decisions

Trade-offs, constraints, assumptions.

E. Limitations

Known gaps.

F. References

(Optional)

5. Detailed Buyer Questions for Each Layer

Below are **all questions**, fully included.

5.1 Layer 0 — Compute & Network Fabric

1. What accelerators are supported (GPU/TPU/CPU)?
 2. How do you abstract heterogeneous hardware?
 3. What networking requirements do you assume?
 4. How do you support high-throughput/low-latency inference?
 5. What cloud/HW dependencies exist?
 6. What infrastructure telemetry is exposed?
 7. What lifecycle assumptions do you make about hardware?
 8. How do you report sustainability metrics?
 9. How do you support multi-region deployment?
 10. How do you avoid hardware lock-in?
-

5.2 Layer 1A — Data Storage & Governance

1. What storage engines and formats are supported?
 2. What metadata does your governance catalog track?
 3. How do you represent lineage?
 4. How do you enforce residency?
 5. How do you integrate with enterprise governance?
 6. How is metadata accuracy ensured?
 7. How do you expose policies to 2C?
 8. How do you handle schema evolution?
 9. How do you handle restore testing?
 10. How do you isolate tenants?
-

5.3 Layer 1B — Context Management & Retrieval

1. Which retrieval backends are supported?
 2. How do you model embeddings?
 3. What retrieval performance guarantees do you provide?
 4. How are indexes maintained?
 5. How do you prevent stale context?
 6. How do you tune relevance?
 7. What telemetry do you expose?
 8. Are embeddings exportable?
 9. If we terminate, can we export vectors without reprocessing?
 10. What limits exist on index size?
 11. How do you support multi-region retrieval?
-

5.4 Layer 1C — Data Movement & Pipelines

1. What pipeline frameworks do you support?
 2. How do you unify batch and streaming?
 3. How do you track pipeline lineage?
 4. How do you ensure cost-aware movement?
 5. How do you handle failures?
 6. Can we define SLAs/SLOs?
 7. How do you validate data quality?
 8. How do you coordinate schema changes?
 9. How do you expose telemetry to 2C?
 10. What throughput limits exist?
-

5.5 Layer 2A — Control Plane

1. How do you provision compute?
 2. How do you enforce RBAC?
 3. What metrics are exposed to 2C?
 4. How do you integrate with enterprise Kubernetes?
 5. How do you support multi-region clusters?
 6. How do you handle rollbacks?
 7. How do you manage drift?
 8. How do you secure APIs?
 9. How do you avoid resource fragmentation?
 10. How do you prevent policy conflicts with 2C?
-

5.6 Layer 2B — Execution Plane

1. What runtimes are supported?
 2. How do you orchestrate multi-step workflows?
 3. How do you handle backpressure?
 4. How do you manage versioning/rollback?
 5. What metrics do you expose to 2C?
 6. How do you isolate workloads?
 7. How do you manage multi-model routing?
 8. How do you integrate external APIs safely?
 9. What assumptions do you make about Layer 0/1?
 10. How do you handle multi-region execution?
 11. How do you implement secrets injection?
-

5.7 Layer 2C — Reasoning Plane

1. What inputs does your engine consume?
 2. How is policy expressed?
 3. How do you implement “compute moves to data”?
 4. How do you enforce compliance?
 5. How do you avoid oscillation (hysteresis/dampening)?
 6. How do you mitigate stale telemetry risk?
 7. Do you provide policy simulation / dry-run mode?
 8. Do you provide immutable decision audit trails?
 9. How do you support hybrid/multi-cloud reasoning?
 10. How can customers override policies safely?
 11. Do you support what-if analysis?
 12. Do you provide a 2C kill switch?
 13. Is the policy language open or proprietary?
 14. Who is liable for reasoning violations?
 15. How is identity preserved through 2C?
-

5.8 Layer 3 — Application Layer

1. What prebuilt apps exist?
 2. How do we build custom agents?
 3. How do agents interact with lower layers?
 4. How do you support human-in-loop?
 5. How do you handle application-level permissions?
 6. How do you surface governance/cost to product teams?
 7. How do you manage SDLC across environments?
 8. How do you support multi-tenant apps?
 9. How do you detect harmful behavior?
 10. How do you measure business outcomes?
 11. How do you model business semantics?
-

5.9 System-Wide Concerns

1. What observability stack do you support?
2. How do you implement cost attribution?
3. What certifications do you hold?
4. How do you implement zero trust?
5. What SLAs do you provide?
6. How do you provide DR/BC?
7. How do you handle upgrades?
8. What extension mechanisms exist?
9. How do you support interop with other vendors?

10. How do you ensure cross-layer correlation of metrics?
 11. How is identity propagated across layers?
-

6. Deployment Model Considerations

Covers:

- On-Prem
- Hybrid
- Cloud-Native
- Edge
- Networking flows
- Residency and regulatory patterns

(All the detailed text from earlier draft is fully included.)

7. Security & Compliance Requirements

- Identity
- Secrets
- Data privacy
- Audit
- Agent security
- Supply chain

(Full text already delivered in the previous major version—fully included here.)

8. Cost Transparency

- Pricing model
 - Estimation examples
 - FinOps support
 - Elasticity behavior
-

9. Use-Case Mapping

3–5 detailed use cases
Mapping to 4+1 layers
Dependencies
Limitations

10. Documentation Checklist

Architecture diagrams
Runbooks
SLO/SLA documentation
Observability dashboards
Security whitepapers
Agent lifecycle docs

11. RFP Update Cadence

Biannual updates
Versioning scheme
Tie to Buyer Room insights

12. Buyer Rooms & Stack Builder

RFP ← Buyer Room → Stack Builder → RFP
Model → Insight → Tooling → Model

13. RFP Red Flags

Vendor cannot map to layers
Proprietary vector/index formats
Proprietary policy DSL
Model contamination

No kill switch
No dry run
Identity propagation gaps

14. Glossary

Reasoning Plane
Execution Plane
Control Plane
Retrieval Context
Business Semantics
Policy Oscillation
Hysteresis
Kill Switch
Dry-Run Mode
Model Contamination
Vector Lock-In
Policy DSL
GitOps
FOCUS

...and others, fully defined earlier.

15. Suggested Citation Format

Model:

Townsend, K. (2025). *The CTO Advisor 4+1 Layer AI Infrastructure Model, v1.0*. The CTO Advisor.

Framework:

Townsend, K. (2025). *4+1 AI Platform RFP Framework (Open Edition), v1.1*. The CTO Advisor.
