10

WHITE PAPER · AI OPERATING GOVERNANCE

# Local AI Versus Cloud AI Is a Governance Decision

Choose the operating boundary before the model

**Marco Policani**

Enterprise Portfolio · PMO · AI Operating Governance

policani.net · linkedin.com/in/marcpolicani

July 2026

EXECUTIVE SUMMARY

**The operating boundary should be chosen around data, access, action authority, resilience, and exit—not model fashion.**

Deployment debates that start with models anchor a long-lived decision to the stack's most volatile layer and smuggle in false safety equations — local is not automatically private, cloud is not automatically resilient. This paper reverses the order: seven operating boundaries, examined per workflow, decide the pattern before any model is named.

01 Trace the workflow's actual data through prompts, retrieval, telemetry, administration, and backups per option before comparing capability.

02 Specify the action envelope independently of hosting — a local model with broad permissions outranks a drafting-only cloud model on consequence.

03 Price the exit while designing the entrance: an untested exit is a commitment signed without reading.

## The argument

Local, cloud, hybrid, or no deployment should follow an operating-boundary decision about data, access, action authority, evaluation, resilience, support, suppliers, and exit. Model fashion is not a governance criterion.

The deployment debate I am asked to referee usually arrives already broken. One camp has benchmarked the frontier cloud models and wants the best one; the other has a security instinct that anything sensitive belongs on hardware the organization can touch. Both camps argue about models. Neither has yet answered the questions that actually decide the matter: where may this workflow's data move, who can reach the system and its administration, what is it permitted to do, how will its behavior be evaluated as everything underneath it changes, what happens when it fails, who operates it on the third year, and how does the organization leave. I have watched architecture reviews spend three meetings on model quality and thirty minutes on all seven of those questions combined — an inversion of effort that the resulting incidents eventually correct.

The order should be reversed. Deployment pattern is a governance outcome, not a technology preference: the right answer falls out of the boundaries the workflow requires, and different workflows in the same enterprise will legitimately land on different answers. NIST's generative-AI risk profile and CISA's cloud security architecture point the same direction — connect the technology choice to managed risk, shared responsibility, and an inspectable operating design [1][2]. This paper turns that direction into a working instrument: seven boundaries, examined per workflow, before any model is named. It is written for CIOs, architecture leaders, security leaders, AI program owners, and workflow executives; the synthesis is Marco Policani's.

## Why the model-first debate fails

Model-first reasoning fails because model properties are the most volatile element in the entire stack. Rankings shuffle quarterly, prices move monthly, and the capability edge that justified a choice erodes while the procurement paperwork is still circulating. Boundaries have the opposite lifespan: the data-residency rule, the customer commitment, the regulator's expectation, and the organization's operating capacity change slowly. Anchoring a long-lived deployment decision to the fastest-moving variable guarantees the decision is stale at delivery.

Model-first reasoning also smuggles in false safety equations. 'Local means private' survives until the endpoint audit: an on-premises model with weak access control, unpatched serving infrastructure, and administrator sprawl is less private than a well-governed tenant. 'Cloud means resilient' survives until the first regional incident or quota surprise. Neither property comes from the deployment pattern; both come from the operating discipline around it — which is precisely what the boundary review examines and the model bake-off does not.

The practical consequence of getting this wrong is not usually catastrophe. It is expensive rework: the workflow built on a pattern its data classification never permitted, discovered at audit; the local cluster sized for a pilot and abandoned when the operating load arrived; the cloud arrangement renewed under pressure because nobody had priced leaving. Each is a boundary question answered late, at the maximum possible cost.

## The seven-boundary deployment decision

The instrument examines seven boundaries for a specific workflow — not for the enterprise in the abstract. Each boundary yields evidence, and the deployment pattern is chosen to fit the assembled evidence. Run honestly, the review sometimes concludes cloud, sometimes local, often hybrid, and occasionally that no current option meets the need responsibly — which is also a governance answer, and a cheaper one than learning it in production.

EXHIBIT 1

### Seven boundaries decide the deployment pattern

1. **Data boundary**
   Deployment must fit the sensitivity, residency, lineage, retention, and secondary-use rules of the actual workflow data.
2. **Access and identity boundary**
   The decision includes who can reach models, data, tools, administrative functions, and support channels.
3. **Action-authority boundary**
   Inference location does not decide what the system is permitted to do inside a workflow.
4. **Evaluation and change boundary**
   Every option needs a way to evaluate behavior as models, prompts, data, and users change.
5. **Resilience and support boundary**
   Availability, latency, incident response, continuity, and recovery obligations differ by deployment pattern.

6 **Supplier and exit boundary**

A deployment choice creates dependencies on models, hardware, platforms, contracts, skills, and data formats.

## Data boundary

The first boundary traces the workflow's actual data through the system's actual anatomy. Not 'the model' in the abstract: prompts, retrieved documents, embeddings, logs, telemetry, fine-tuning sets, administrator access, support channels, and backups each move data somewhere, under some retention rule, visible to some population. Classification exercises that stop at 'the model is hosted in-region' routinely miss that the support pathway or the telemetry pipeline crosses a boundary the workflow's data may not.

Retention and secondary use are the map's fine print, and the fine print governs. A provider may process in-region and still retain prompts for abuse monitoring under a different clock; an embedding store may outlive the documents it was built from; a fine-tune may carry training data into a model artifact that travels on its own rules. None of these is hidden — they live in the terms and the architecture diagrams — but they only get read when a decision instrument demands the map.

The boundary resolves into a data-flow map per option — input, processing, retrieval, telemetry, administration, backup, deletion — annotated with classifications, contractual terms, and verified deletion behavior. The decision rule is not local-by-default: choose a bounded or local pattern when required controls cannot be demonstrated in the cloud, and choose cloud when its managed controls demonstrably exceed what the organization can operate itself. Both directions of that rule get exercised in real enterprises.

## Access and identity boundary

The second boundary asks who can reach the system — models, data stores, tools, administrative planes, and support channels — and whether that population matches the workflow's accountability needs. Assumptions do the damage here: a local installation presumed private while a shared service account with a year-old password reaches it; a cloud service presumed controlled because corporate login exists, while tenant administrators and vendor support sit outside the story the workflow owner was told.

The AI-specific wrinkle is that new access surfaces arrive with the technology and get missed by standing reviews: the prompt history that becomes a sensitive-data store nobody classified, the retrieval index that quietly grants broad read access to whatever was ingested, the evaluation dataset containing production samples on a data scientist's laptop. An identity review scoped to 'the model' misses all three. The boundary review scopes to the workflow's full anatomy, which is where these surfaces live.

Ordinary identity discipline, applied to this new surface, settles the question: mapped identities and privileges including service accounts, tenant and administrative boundaries, separation of duties, provisioning and offboarding records, and monitoring on privileged access. The rule is strict because everything downstream leans on it: reject any option whose identity model cannot support the accountability the workflow requires. An architecture that cannot say who did what is not eligible, whatever its benchmarks.

## Action-authority boundary

The third boundary separates where a model runs from what it may do — two questions that deployment debates persistently fuse. Hosting decides data movement and operational control; authority decides consequence. A local model wired to production systems with broad permissions is a higher-consequence arrangement than a cloud model that can only draft text for human review, yet the reflex ranks them the other way because 'local' sounds contained.

What this boundary demands is an explicit action envelope, specified independently of hosting: what the system may recommend, prepare, approve, execute, or reverse; tool scopes; transaction limits; approval records for consequential paths; and a tested kill path. No architecture selection until the permitted envelope is clear, because the envelope drives requirements the pattern must serve — audit density, rollback design, containment behavior — and choosing hosting first means retrofitting those requirements into whatever was bought.

## Evaluation and change boundary

The fourth boundary asks how the organization will know the system still behaves after everything changes — and everything changes. Cloud providers update models under their own schedules; a static approval ages beneath the workflow silently. Local deployments invert the failure: the model the team froze for stability quietly becomes stale, unsupported, and unpatched, drifting from the data it serves. Both patterns need governed change, and they need it differently.

The evidence per option: who holds update authority, what regression testing runs against the workflow's own cases, what release evidence exists, how behavior is monitored between releases, what triggers reapproval, and whether rollback is real. The rule favors honesty about capacity: prefer the option whose change model the organization can actually govern. A team that cannot staff model-lifecycle management should not own one, however attractive the control fantasy; a team whose vendor will not disclose change notices should not build reliance it cannot re-verify.

## Resilience and continuity boundary

The fifth boundary prices failure. Availability targets, latency floors, capacity peaks, incident response, and disaster recovery differ radically by pattern — and by investment within a pattern. The recurring asymmetry: local capacity is under-engineered because its costs are capital and visible, while cloud dependence is accepted without a viable degraded mode because its risks are contractual and abstract. Both errors surface in the same place — the workflow's worst week.

Latency belongs in this boundary too, because it is an operating requirement that gets mistaken for a performance preference. A workflow that needs sub-second response in a customer interaction has different eligible patterns than one that batches overnight, and quota ceilings or throttling behavior under provider load are continuity facts a contract may only whisper. The review asks the workflow for its real tolerances and tests the options against them, rather than discovering the mismatch through user complaints.

The evidence is exercised, not asserted: load tests at realistic peaks, continuity exercises with results, dependency maps including the quiet ones (identity provider, network egress, the one engineer who understands the serving

stack), recovery objectives the operation actually accepts, and a defined degraded mode the workflow can genuinely run in. The rule allows layered answers: hybrid designs, bounded functionality during outages, or non-adoption where neither option can meet the operational need responsibly.

## Support and operations boundary

The sixth boundary asks who runs the thing on an ordinary Tuesday in year three. Deployment patterns are also staffing decisions: a local stack requires serving expertise, patching discipline, hardware lifecycle, and on-call depth that most enterprises underestimate by a factor; a cloud arrangement requires vendor management, quota governance, cost observability, and the skills to translate provider incidents into workflow decisions. Neither is free; they are differently shaped bills.

The evidence is a support model with names and numbers: rosters and escalation depth, skill coverage and its concentration risk, ticket and question volume expectations, patch and upgrade cadence, and the budget line that funds all of it past the founding team's enthusiasm. The rule is the same honesty test as change governance: choose the pattern whose operating bill the organization will actually pay, not the one whose bill it prefers not to look at.

## Supplier and exit boundary

The seventh boundary examines the dependencies the choice creates — on models, hardware vendors, platforms, contracts, formats, and skills — and how the arrangement ends. Every deployment creates lock-in of some shape: cloud lock-in is contractual and data-gravitational; local lock-in is capital, skills, and the sunk cluster nobody wants to strand. The governed question is not how to avoid dependency but whether the exit has been designed and priced while designing it was cheap.

The evidence: exit clauses and their tested meaning, data export actually exercised against real volumes, replacement lead times, the asset and knowledge inventory a successor would need, and named transition ownership. The rule treats an untested exit as a material commitment the organization has signed without reading. Concentration deserves explicit statement too — a single model family, a single region, and a single irreplaceable engineer are one risk with three addresses.

## Two workflows, two verdicts

A pair of composite examples shows why the review must run per workflow. First: a claims-adjudication support tool for a regulated insurer. The data boundary binds hard — case files carry health information under residency rules, and the vendor's telemetry pipeline fails the map. The action envelope is modest (recommend and prepare only), but identity requirements are strict and the auditors will want inspectable review records. The verdict lands on a bounded pattern: a dedicated tenant with contractual telemetry controls for the general reasoning, a small local model for the extraction step that touches raw files, and the exit tested against a successor vendor's import format.

Second: a marketing-content assistant in the same enterprise. The data is low-sensitivity, the action envelope stops at drafting, and the volume is bursty. Here the resilience and support boundaries dominate: the team has no

serving expertise and no appetite for patch discipline, while the cloud provider's managed controls exceed anything the function would build. Verdict: cloud, standard tenant, quarterly evaluation samples, exit priced at nearly zero because the workflow holds no state worth migrating.

Same company, same year, opposite answers — both correct. An enterprise-wide 'AI deployment standard' that forced either workflow into the other's pattern would have manufactured either an audit finding or a wasted cluster. The unit of decision is the workflow, because the boundaries bind at the workflow.

## Reading the seven together

The boundaries are not a scorecard to average; they are constraints to satisfy simultaneously, and their interactions decide the hard cases. Data and access constraints may force a bounded pattern that resilience economics argue against — that tension is the actual decision, and it belongs to the workflow's risk owner, made visibly. Action authority multiplies every other boundary's stakes: widen the envelope and the evaluation, resilience, and identity requirements widen with it. Exit discipline keeps the whole decision revisitable, which matters precisely because the model layer changes fastest.

EXHIBIT 2

### The rule each boundary enforces

**Data boundary**

Choose a bounded or local pattern when required controls cannot be demonstrated in the cloud; choose cloud when its managed controls better fit the need.

**Access and identity boundary**

Reject any option whose identity model cannot support the required accountability.

**Action-authority boundary**

Select architecture only after the permitted action envelope is clear.

**Evaluation and change boundary**

Prefer the option whose change model the organization can actually govern.

**Resilience and support boundary**

Use hybrid, bounded functionality, or non-adoption when neither option can meet the operational need responsibly.

**Supplier and exit boundary**

Treat an untested exit as a material commitment, not a future procurement detail.

Hybrid deserves precision, because it is the answer most reviews reach and the word covers three different designs. Split-by-step: sensitive steps run bounded or local, general steps run cloud — the claims example. Split-by-tier: a cloud frontier model for complex cases, a smaller local or bounded model for routine volume, with routing rules that are themselves governed. Split-by-mode: cloud in normal operation, a degraded local capability for

continuity. Each hybrid carries its own boundary profile — routing rules need evaluation, dual patterns double the support surface — and 'hybrid' as a strategy statement, without naming which design, is a decision still unmade.

Run per workflow, the review also produces a portfolio pattern worth reading: the workflows clustering toward bounded local patterns reveal where the organization's data constraints genuinely bind; the ones comfortable in cloud reveal where managed controls dominate; and repeated 'no current option' verdicts point at capability investments — often in identity or evaluation capacity — that would open whole classes of adoption at once.

Shared responsibility deserves its own sentence in every cloud-leaning decision, because the phrase hides a moving line. The provider secures the infrastructure and publishes the controls; the customer configures them, governs identities, classifies data, and owns the workflow's behavior [2]. Most cloud AI incidents in practice are customer-side configuration and governance failures that get reported under the vendor's name. The boundary review makes the line explicit per workflow: this column the provider holds, this column we hold, and this column — usually evaluation and action authority — nobody holds until we assign it.

Cost honesty rounds out the comparison. Cloud costs are visible, metered, and politically noisy; local costs are lumpy, capitalized, and hidden in salaries and depreciation, which biases casual comparisons toward whichever bill the organization prefers not to see. The boundary review prices both patterns fully — including the support roster, the evaluation cadence, the patch discipline, and the exit — so the finance conversation happens on total operating cost rather than on the shape of the invoice.

## Running the review on two workflows first

The review installs modestly:

- Begin with two real workflows; abstract enterprise-wide scoring hides important differences.
- Complete the data and action maps before comparing models.
- Estimate standing support, evaluation, security, and recovery capacity for each option.
- Record the decision, residual risk, triggers for reconsideration, and exit obligations.

The evaluation boundary is also where whole classes of stalled adoption often get released. Organizations discover mid-review that the binding constraint is not data or hosting but the absence of any capacity to evaluate model behavior against their own cases — which no deployment pattern fixes. Building that capacity once, as shared infrastructure, converts a dozen stalled deployment debates into decidable questions. The review surfaces this because it asks the evaluation question per workflow and gets the same embarrassed answer twelve times.

Two workflows, deliberately different — one data-sensitive and narrow, one broad and low-consequence — teach the instrument faster than any template rollout. The first pass on each takes days, not months, because most of the evidence exists in fragments already: the classification policy, the identity architecture, the vendor contract. The review's work is assembling fragments into a decision, and its first visible product should be a one-page decision record the workflow's owner signs.

The record's reconsideration triggers keep the decision alive without keeping it open: a new model class that changes the capability equation, a policy or regulatory shift, a provider change notice above a threshold, an incident, a support-capacity change. Any trigger reopens the affected boundary — not the whole debate — which is how the organization tracks a fast-moving field without re-litigating its architecture quarterly.

Procurement is where the boundary review earns hard cash, because contracts are boundary instruments. A team that arrives at the vendor table with its data map, action envelope, and exit requirements negotiates specific clauses — telemetry handling, change-notice windows, export formats, deletion verification — while a team that arrives with a model preference negotiates price. The first team's contract is a governance asset for years; the second team's contract is a discount on terms it never read closely.

Regulated industries will recognize a familiar structure in all of this, and that recognition is the point. The seven boundaries are the AI-shaped version of questions their existing control frameworks already ask about any material system — data handling, access, authorization, change management, continuity, operations, third-party risk. Framing AI deployment this way lets the review ride existing governance machinery instead of building a parallel bureaucracy, and it gives risk and audit functions a vocabulary they already trust.

## What the record should show

Governance of deployment decisions is itself inspectable:

- Decision records per workflow, signed by the risk owner, with residual risks stated.
- Data-flow and action-envelope maps that match the deployed reality.
- Change events handled by the governed path — regression run, reapproval where triggered.
- Continuity and degraded-mode exercises completed on cadence.
- Exit tests actually run: exports exercised, transition inventories current.
- Reconsideration triggers fired versus decisions silently aging.

The last line is the health check for the whole practice. Deployment decisions in this field have a shelf life; a portfolio whose decision records have no recent trigger activity is not stable, it is unexamined — and the difference becomes visible at the least convenient moment available.

## The strongest counterargument

Local is not automatically private, cloud is not automatically resilient, and hybrid is not automatically balanced. Outcomes depend on implementation and operating discipline. The framework supports a reasoned choice; it does not declare one deployment pattern universally superior.

A second fair objection: small organizations cannot staff seven-boundary reviews. True — and the instrument scales down honestly. A twenty-person company runs the same questions in an afternoon, answers most of them 'cloud, standard terms, minimal envelope,' and writes half a page. The value at that scale is not the analysis; it is the two questions small organizations otherwise skip entirely — what the system may do, and how they would leave.

The objection that seven boundaries is heavyweight for a fast-moving field has the sequence backward. The boundaries are what make speed safe: a workflow with mapped data flows, a defined action envelope, and a tested exit can change models in a week, because the change touches the volatile layer while the governed layers hold. It is the organization without boundaries that cannot move — every model change reopens every question, and architecture debate becomes the standing tax on adoption.

## The shift

The shift is from 'which model should we run, and where?' to 'which boundaries must hold, and which pattern holds them?' Asked in that order, the deployment question stops being a technology argument and becomes what it always was underneath: an operating-governance decision about data, authority, failure, operations, and commitment — decided per workflow, on inspectable evidence, with the exit priced before the entrance is signed.

Model fashion will keep moving; that is the one certainty in the stack. Organizations that govern the boundaries get to treat that movement as opportunity — swap the volatile layer, keep the governed ones. The ones that argue about models get to treat every movement as a recurring crisis, because nothing beneath the argument was ever settled.

The instrument's final virtue is that it produces decisions someone signed. In a domain this noisy, the difference between an architecture and an accident is a record: which boundaries were examined, what the evidence showed, who accepted the residual risk, and what would reopen the question. That record is what turns deployment from a debate the organization keeps having into a decision it keeps improving.

# Notes and sources

[1] NIST, "Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile," July 2024. https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.600-1.pdf

[2] Cybersecurity and Infrastructure Security Agency, "Cloud Security Technical Reference Architecture, Version 2," June 2022. https://www.cisa.gov/sites/default/files/2023-02/cloud_security_technical_reference_architecture_2.pdf

**About the author**

Marco Policani is an enterprise portfolio, PMO, and AI operating-governance leader. He builds portfolio governance systems where AI carries the analytical load and named humans own every decision — an approach documented across case studies, walkthroughs, and working governance guides on his portfolio site.

Portfolio & governance library: policani.net/governance · Full portfolio: policani.net · LinkedIn: linkedin.com/in/marcpolicani