Delegated Authority and Human Accountability in AI Operating Systems: A Framework for Operational Governance

November 2026

Abstract

Giving an agent access to a business application does not establish which decisions it may make or who remains responsible for their consequences. This paper develops a framework for delegated authority in enterprise AI operating systems. It connects identity, action scope, business conditions, approval, revocation, and evidence of execution. The analysis draws on established access control and AI risk management guidance, then examines a constructed procurement workflow in which authority changes with value, supplier status, and transaction state. Its contribution is an action specific account of delegation that treats human review as a limited operational resource. The paper proposes tests for approval validity, cumulative exposure, reviewer comprehension, and timely intervention. MatruOS provides an architectural context for applying the framework. The work is conceptual and reports no deployment results. It concludes that accountable delegation requires enforceable application controls and an explicit allocation of responsibility before an agent is allowed to create business obligations.

Keywords: delegated authority, approvals, accountability, access control, human oversight, enterprise governance.

1. The difference between access and authority

A purchasing employee asks an agent to arrange a replenishment order. The employee can view supplier records and prepare purchase requests but cannot approve expenditure above a specified amount. If the agent uses a broadly privileged service account, it may complete an order that the employee could not authorise directly.

The user’s request is genuine, and the proposed purchase may be sensible. The governance failure is that delegation has expanded authority. A business system should preserve the distinction between requesting, recommending, approving, and executing an action. These activities can involve different people and different evidence.

Authority also changes with circumstances. An approved supplier may become suspended. A purchase request may change after approval. Several individually permitted orders may collectively exceed a budget. A static role label such as “purchasing agent” does not express all these conditions.

The research question is how an AI operating system should represent and enforce delegated authority while retaining meaningful human accountability. The scope is operational governance of business actions. Legal responsibility and sector-specific compliance require separate analysis; this framework does not determine them.

2. Research foundations and method

NIST’s guidance on attribute based access control describes authorisation through attributes of the subject, object, requested operation, and relevant environmental conditions. This offers a foundation for evaluating a specific action against policy rather than relying only on a broad role. [1]

The NIST AI Risk Management Framework organises activities around governing, mapping, measuring, and managing AI risk. It supports treating governance as a continuing organisational responsibility. The present paper applies that general orientation to the operational details of delegated business decisions. [2]

NIST’s zero trust architecture guidance places emphasis on explicit resource access decisions rather than assumed trust arising from network location. An enterprise agent running inside company infrastructure still requires defined permissions and enforcement. Internal deployment does not settle the authority of a requested action. [3]

The method is conceptual analysis using a constructed procurement scenario. The contribution is a delegation record and review process that connect access control with business approval, cumulative exposure, and the practical capacity of reviewers. The paper does not claim that these governance concepts are novel individually or that the proposed combination has been empirically validated.

3. An action specific delegation record

3.1 Identity and purpose

A delegation record should identify the requesting person, the agent or service performing the work, and the business purpose. The execution identity should remain distinct from the human identity so that the record shows how an action was performed. Impersonating the user in every downstream log can obscure whether a person or automated service made a particular change.

Purpose should be operationally bounded. “Assist purchasing” is too broad to establish an executable permission. “Prepare replenishment requests for approved products in this branch” identifies a defined class of work. Whether the agent may submit, approve, or place those requests must then be stated separately.

The record should specify the tenant, business unit, accounts, and resources within scope. An agent serving several branches must not infer permission to move information or funds between them. Shared infrastructure can host different authorities, and every consequential operation should carry the relevant scope.

3.2 Conditions, limits, and duration

A delegation needs conditions under which it is valid. These may include transaction value, supplier status, budget availability, product category, and time period. Some are properties of the request; others depend on current business state. The service should evaluate both at execution.

Limits should also consider accumulation. A purchasing agent allowed to place orders below a stated amount could exceed the intended exposure by splitting one requirement into several orders. The policy needs to define related transactions and the period over which exposure is aggregated. That relationship should be based on business intent and records, not only timestamps.

Delegation should have an expiry or review condition. Temporary authority during an employee’s absence should not become permanent through inactivity. The system should identify the owner responsible for renewal and the work that must stop or be reassigned when the authority expires.

3.3 Evidence and permitted outcomes

An authorised action requires evidence that its conditions are satisfied. For a purchase, that may include an approved requisition, supplier identity, budget allocation, and a price within permitted terms. A model-generated explanation can help a reviewer understand the request, but the evidence should remain separately accessible.

The record should state permitted outcomes. The agent may prepare a draft, submit for approval, execute after approval, or decline and explain a policy conflict. These outcomes should be visible in the workflow so that a request ending in escalation is not incorrectly treated as a technical failure.

A delegation should also describe what the agent must do when required evidence is unavailable. Silence from a service should not become permission. A missing budget response requires a defined pause or escalation, with responsibility for resolving the missing information.

4. Approval as a controlled transition

4.1 Binding approval to the proposed action

An approval should refer to the specific transaction version reviewed. The approved amount, supplier, product, and delivery conditions should be identifiable. If the request changes materially, the service should determine whether the existing approval still covers it.

This requirement addresses a common ambiguity in conversational interfaces. A manager may say “approved” after viewing one proposal, while the agent later revises the order. The system needs a durable link between the approval and the reviewed proposal. The natural language response can initiate that link but should not remain its only representation.

Approval should not automatically grant permission to alter related controls. Permission to buy goods does not imply permission to change the supplier’s bank details or extend the budget. Those actions have different consequences and should retain their own approval requirements.

4.2 Separation of duties

Where policy requires separate request and approval roles, an agent should preserve that separation. It should not act as both requester and approver through different tools using the same privileged identity. The application needs to check the effective authority behind each operation.

A small business may have limited staff and one owner responsible for several roles. The framework does not impose an impractical organisation chart. It requires the business to document the arrangement and any compensating review, such as retrospective inspection of unusual purchases or changes to supplier details.

The important question is whether the review adds independent judgement where the business requires it. A second automated response generated from the same incomplete evidence does not establish independent approval merely because it is produced by another agent.

4.3 Revocation and work already in progress

Revocation should affect queued and future actions promptly. Long running work needs checkpoints so that authority is reconsidered before the next consequential step. An approval valid at the beginning of a workflow may cease to apply after a supplier suspension or an employee role change.

The treatment of already committed actions requires a separate procedure. Revoking future authority cannot erase a purchase order already sent to a supplier. The business must identify the remaining obligations and decide whether cancellation, correction, or continued fulfilment is appropriate.

The system should therefore distinguish permission status from transaction status. A revoked delegation can coexist with an open obligation requiring authorised resolution. Combining these into a single “cancelled” label would conceal necessary work.

5. Human oversight as an operational resource

5.1 The information a reviewer needs

A reviewer needs to understand the proposed action, its consequences, relevant exceptions, and the evidence supporting it. A lengthy conversation transcript can be difficult to assess. A concise structured presentation should show the exact fields requiring approval and link to the underlying records.

The interface should make uncertainty visible. If supplier identity is unresolved or a price comparison uses an outdated quotation, that condition belongs beside the approval decision. It should not be hidden in a general disclaimer that reviewers learn to ignore.

The reviewer also needs an effective choice. Approve, reject, request clarification, and narrow the scope should have distinct effects. A request for clarification should keep the operation pending, while rejection should prevent execution under that proposal. The agent should not reinterpret rejection as an invitation to find a different route to the same prohibited action.

5.2 Review workload and delay

The existence of an approval button does not establish meaningful oversight. If every minor request requires urgent review, people may approve without examining the evidence or leave work waiting. The governance design should estimate the frequency and complexity of expected reviews.

A proposed operational measure is reviewer minutes per valid completed transaction, reported alongside the rate of invalid approvals. Another is the age of unresolved approval requests by consequence category. These measures expose a design that shifts work to managers while reporting extensive automation.

Review capacity should inform delegated scope. A business with one available approver may begin with narrow, reversible actions and explicit limits. Expansion should depend on observed review workload and control effectiveness. The volume of generated requests is not evidence that the organisation can supervise them.

5.3 Accountability after an incident

An incident record should identify the policy owner, delegating authority, approver, execution service, and person responsible for recovery. These roles may overlap, but the record should not collapse them into a generic statement that “the AI decided.”

Investigation should ask whether the policy was suitable, whether the implementation enforced it, whether the evidence was available, and whether the reviewer could reasonably identify the problem. This prevents every failure from being attributed either to the model or to the person who clicked approval.

Corrections should address the failure mechanism. If a reviewer approved a changed order without seeing the change, retraining the reviewer alone leaves the interface defect unresolved. If the rule was ambiguous, an additional confirmation dialog may not resolve the underlying policy problem.

6. A procurement scenario

Assume an agent is authorised to prepare replenishment requests for an approved product category. It can place orders only within a branch budget and below a specified approval threshold. Supplier bank details are maintained through a separate controlled process.

The agent receives a request for goods that exceeds its execution limit. It prepares a proposal showing quantity, price, supplier, and budget effect. A manager approves that version. Before execution, the supplier sends a revised quotation with a higher price and a different bank account.

The revised price requires the service to recheck the approval's scope. The bank account change requires a separate process regardless of whether the total price remains acceptable. The agent should retain the supplier's message as evidence and route the necessary decisions to the appropriate owners.

Suppose the manager approves the revised price, but another purchase has since consumed part of the budget. Execution should revalidate budget availability. Approval indicates willingness to proceed under specified conditions; it does not create resources that are no longer available.

Now suppose the agent divides the purchase into several orders to fit a per order threshold. The governance service should evaluate related intent and cumulative exposure. The action should remain subject to the original approval requirement. This case illustrates why limits need a business definition of aggregation.

Finally, if the supplier is suspended after the order is transmitted, the system should stop uncommitted follow up actions and create a recovery task for the existing obligation. A responsible person determines whether the order can be cancelled. The audit record should show both the authority at commitment and the later change.

7. Application to MatruOS

In a MatruOS context, role permissions should be supplemented by action conditions drawn from current business records. The relevant services should evaluate the approved transaction version, financial limits, and business unit scope before changing state. The conversational layer can explain the decision but should not override the enforcement result.

The system should make delegation understandable to business owners. A configuration screen should state which actions are permitted, the conditions that apply, and who reviews exceptions. A list of technical tool names alone is insufficient for someone deciding how much business responsibility to delegate.

Governance also needs change management. Updating a policy should identify affected agents, queued requests, and active approvals. A preview using representative cases can help the policy owner understand the consequences. Production activation should preserve the previous version for investigation and authorised rollback.

These are design requirements proposed for evaluation. The paper does not establish that every requirement is implemented in MatruOS. Its relevance lies in providing a specific basis for architecture review and deployment decisions.

8. Proposed evaluation and falsifiable questions

A test set should include permitted requests, clear violations, ambiguous instructions, changed approvals, expired delegations, cumulative purchases, and revoked roles. Each case needs an expected outcome and a responsible business reviewer who can explain the applicable rule.

The principal control measure is the rate of actions executed outside their effective authority. It should be reported separately from task completion. A system that completes many requests while occasionally bypassing a financial limit cannot conceal those violations inside an aggregate success score.

Measure valid actions rejected, requests unnecessarily escalated, and time required to resolve exceptions. These reveal the operating cost of the controls. Record whether reviewers can correctly identify the amount, scope, and consequences they are approving through a comprehension check in a controlled study.

The first research proposition is that binding approvals to transaction versions reduces execution under stale approvals. The second is that cumulative exposure controls prevent circumvention through related requests. The third is that review workload constrains safe delegated scope. These are propositions to test, not established findings of this paper.

A study of human review requires appropriate consent and careful handling of participant information. Synthetic procurement cases can be used initially. Results should separate policy defects, enforcement defects, ambiguous reference outcomes, and reviewer errors. A single combined failure category would obscure which part of the system needs correction.

9. Limitations and conclusion

The framework depends on explicit business ownership of policy. Software can enforce a rule, but disagreement about who may approve an exception requires organisational resolution. Controls also need maintenance as roles, suppliers, and operational circumstances change.

No universal approval threshold is proposed. Consequences depend on the business, transaction type, and ability to recover. A small recurring action can create substantial cumulative exposure, while a large reversible draft may create little immediate obligation. Authority should be assigned with those distinctions in view.

Delegated authority becomes accountable when the business can identify who permitted an action, what conditions applied, what evidence was reviewed, and how execution was controlled. Human oversight requires sufficient information, time, and practical means to intervene.

An AI operating system should therefore treat delegation as a maintained operational record. Its success should be judged through valid work completed within authority, the handling of exceptions, and the ability to reconstruct and correct consequential decisions.

Author affiliation and disclosure

The author is affiliated with Code Colonies Private Limited, Ahmedabad, and leads the MatruOS team. MatruOS is an architectural application context. No independent product evaluation is reported.

References

[1] Hu, V. C., et al. (2014, updated 2019). Guide to Attribute-Based Access Control (ABAC) Definition and Considerations. NIST Special Publication 800-162.

https://csrc.nist.gov/pubs/sp/800/162/upd2/final

[2] National Institute of Standards and Technology (2023). Artificial Intelligence Risk Management Framework, version 1.0. AI RMF Core.

https://airc.nist.gov/airmf-resources/airmf/5-sec-core/

[3] Rose, S., Borchert, O., Mitchell, S., and Connelly, S. (2020). Zero Trust Architecture. NIST Special Publication 800-207.

https://csrc.nist.gov/pubs/sp/800/207/final