Stop Sprinkling AI

Consider Task Specialized Models That Actually Run Revenue Cycle

Nick Singleton
Principal Architect
Vee Healthtek

Michelle Castillon
Chief Product Transformation Officer
Vee Healthtek

Jonathan Cachat
Director of Data Science & MLOps
Vee Healthtek

The legacy playbook for AI in revenue cycle is still to deploy broad Large Language Models (LLMs) across the workflow. In production, that strategy creates two avoidable risks: dependency on a single model roadmap and rapid obsolescence, especially once you factor in Protected Health Information (PHI) handling and payer-specific rule nuance.

This paper proposes a more durable execution architecture built around Task Specialized Models (TSMs): a portfolio of models designed to do a bounded set of Revenue Cycle Management (RCM) tasks exceptionally well, with structured outputs, explicit evaluation scorecards, and tight feedback loops. TSMs are paired with a control layer that governs minimum necessary data exposure, policy grounding, confidence thresholds, escalation, auditability, and quality oversight - so execution becomes reliable, repeatable, and safe to scale across high volume workflows.

The practical outcome is not "AI as a sidecar." It is a governed execution layer that turns the daily language of revenue cycle work - eligibility text, edits, denials, payer letters, A/R notes - into structured operational action that can be routed, measured, and improved over time.

A production point of view: revenue cycle rewards execution, not general intelligence

In revenue cycle operations, the limiting factor is rarely a lack of insight. Most organizations already know where the friction is. The real constraint is the cost and inconsistency of turning that insight into the next operational step - reliably, repeatedly, at volume.

Every day, revenue cycle teams translate semi structured inputs into work artifacts: eligibility responses become next steps; claim edits become correction plans; denial codes become standardized work instructions; payer letters become evidence checklists and appeal narratives; remittance lines become recovery paths. These are not open-ended reasoning exercises. They are bounded execution tasks that must be carried out under policy constraints, within workflow handoffs, and at industrial scale.

That framing matters because it changes what "good AI" looks like. In production RCM, the right question is not "Can the model answer anything?" It is "Can the system execute the right next action consistently, at predictable cost, with traceable evidence?"

Why “LLM everywhere” becomes fragile in production

The issue with LLMs is not usefulness - LLMs can be valuable for synthesis and exceptions. The fragility comes from an architectural mismatch when a single broad model is used as the default engine for high volume execution.

First, “one model everywhere” increases operational dependence on a single model family and roadmap. When model behavior shifts (because of upgrades, tuning, or vendor-side changes), that behavior change propagates across the workflow. In an environment where consistency is a first order requirement, this is systemic risk, not an inconvenience.

Second, the shape of RCM work is bounded: it expects structured outputs, stable templates, clear gates, and auditable provenance. The more you push a general model into bounded execution, the more you end up rebuilding constraints and control logic around it anyway - without the tight feedback loops that make improvement fast and measurable.

A better approach is to stop treating AI as a single monolith and start treating it like production software: a portfolio of specialized components with explicit contracts (schemas), explicit gates, and explicit measurement.

What a Task Specialized Model is in the operating context

A Task Specialized Model (TSM) is a model designed for a defined operating context and evaluated against the exact shape of the task it must perform. In RCM, that often means one model for eligibility interpretation, another for claim edit explanation, another for denial categorization, another for appeal assembly, and another for A/R next step recommendation. These tasks share a domain, but they do not share identical inputs, outputs, or governance needs - so they should not share identical model behavior.

A simple operational constraint makes TSMs practical: use deterministic logic whenever it can safely decide and invoke a TSM only when the workflow encounters ambiguity that can’t be resolved deterministically. This keeps inference cost low and governance exposure focused on the part of the workflow where interpretation is actually required.

Eligibility is one such example. A 271 eligibility response is structurally standardized, but operational meaning often arrives in payer-specific text. A TSM can convert that response into a structured “next step” determination - coverage confirmed, limitation detected, authorization required, mismatch flagged - only when deterministic rules cannot decide. Promotion to system-executed handling within predefined policy and confidence bounds happens only when the task clears a production scorecard (e.g., ~95%+ accuracy) sustained across payer variation.

Why now: the architecture window has opened

The last two years have made one architectural lesson increasingly visible: capability can be packaged into smaller, deployable execution models through techniques like distillation, enabling bounded-task performance without relying on “largest model everywhere.” Public releases in 2025 (including DeepSeek’s R1 releases and distilled variants) reinforced that distillation and smaller footprints can be commercially practical, not just academic.

For revenue cycle leaders, the point is not benchmarks. It’s deployment math and stability: if bounded execution can run on smaller, specialized models, you can scale workflow automation with more predictable cost and tighter control than a general-model-first strategy.

The execution architecture: execution layer + control layer (working together)

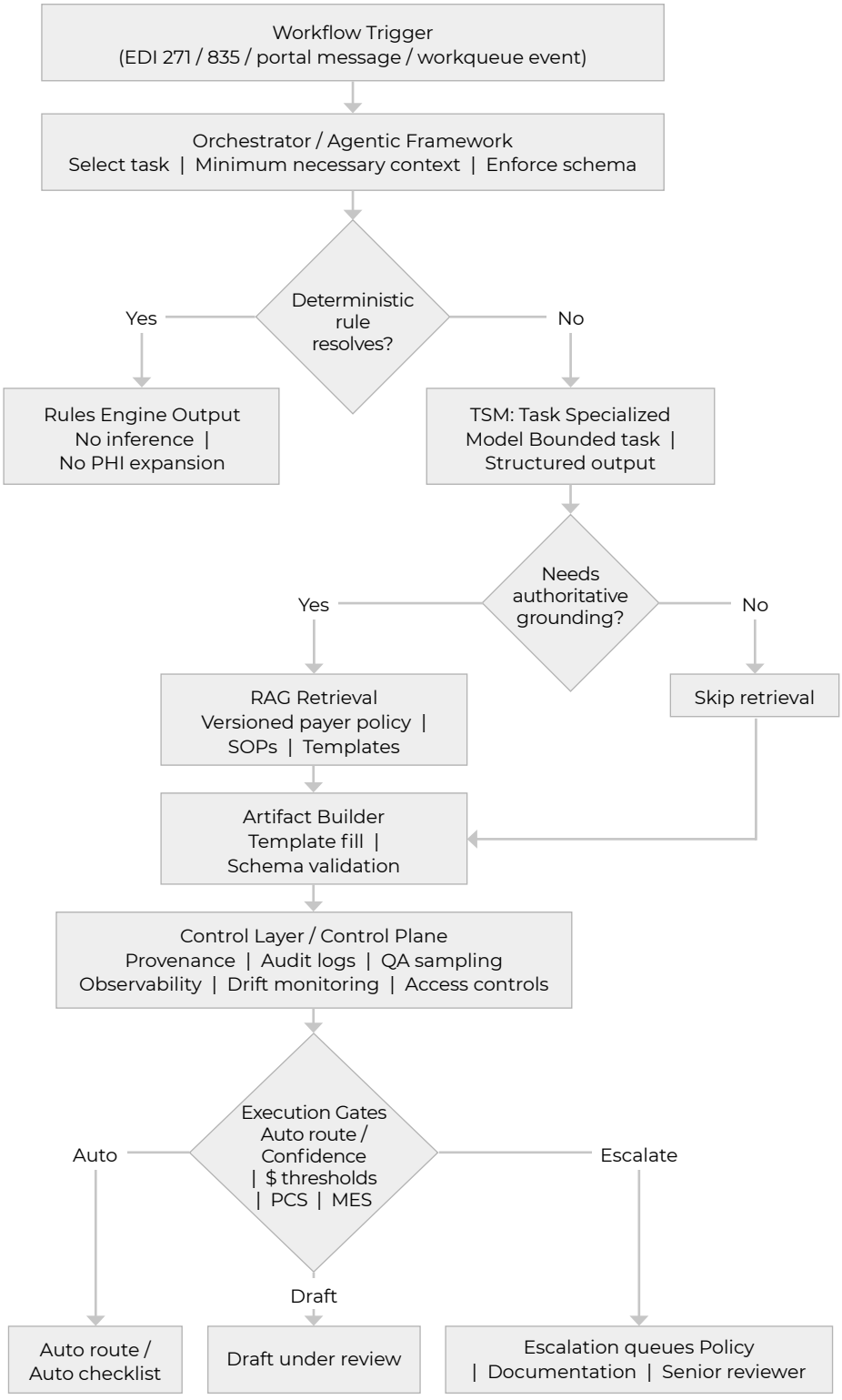
This paper is not arguing that models should be sprinkled across a workflow. It argues that revenue cycle needs an execution layer and a control layer working together.

The **execution layer** performs language-heavy operational work: classification, extraction, normalization, drafting, summarization, checklist creation, and routing. The **control layer** makes that execution enterprise-grade: it governs data exposure, retrieves authoritative sources, enforces template and output constraints, applies confidence thresholds, determines escalation paths, records evidence, and supports quality oversight.

A concrete way to internalize this is simple: without the control layer, AI remains helpful-but-risky. Without the execution layer, humans remain the interpretive middleware between documents and systems. Together, they form a governed execution architecture that is practical for healthcare finance.

TSM-first execution flow: From deterministic rules to governed action

Here is a workflow to demonstrate how RCM work moves through a deterministic-first pipeline, invoking TSMs only when rules cannot resolve the step. A control layer then applies grounding, validation, and execution gates to route each case to policy-bounded system action, draft-under-review, or escalation, with auditability built in.



A working example: Denial intake to appeal draft

A denial arrives through standard remittance channels, typically carrying structured codes plus payer narrative. The first step should be deterministic: extract the fields, attach the record to the right work item, and avoid model inference when extraction can be rules-based.

The TSM enters when interpretation is required. For example, a denial taxonomy classifier can take denial code plus remark text plus minimal claim context and output a bounded label set (category / subcategory), a confidence score, a recommended next action, and a routing target. From there, the control layer determines whether the confidence and risk gates allow auto-routing or require a reviewer.

When the workflow needs payer-policy grounding - especially for evidence requirements - the architecture retrieves the authoritative policy and maps it into a structured checklist. This is where auditability is won: the system is not guessing what the payer wants; it is interpreting retrieved, versioned policy and turning it into required evidence fields.

Finally, drafting should behave like template population under constraints, not free-form prose. A draft-under-review approach allows the system to do the heavy lifting while requiring human review and accountabilityfor external submissions.

Illustration of the gating approach

In production, we treat autonomy as conditional. The control layer applies a small set of explicit gates: it hard-stops and escalates when required evidence is missing, output fails schema validation, policy retrieval lacks a valid / current citation, or confidence falls below the workflow minimum. It also applies dollar-based review thresholds (low-dollar can proceed if other gates pass; mid-dollar demands stricter checks; high-dollar requires human review before external action), and a simple Policy Complexity Score that forces escalation once payer exceptions, multi-document grounding, medical-necessity narratives, or regulatory overlays stack up. Finally, gates vary by workflow: bounded classification can be more autonomous; checklists require stricter completeness; drafting defaults to review at meaningful dollars; and external submission is never fully autonomous at launch.

Where TSM execution delivers measurable lift

Eligibility and authorization

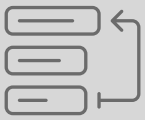
A TSM for eligibility interpretation converts payer language from a 271 response into a structured determination - coverage confirmed, benefit limitation identified, authorization required, demographic mismatch detected, or next action needed. Because outputs are bounded, the control layer can apply confidence gates and route exceptions to human review, reducing preventable downstream touches and improving front-end execution clarity.

Denials and appeals

A TSM for denial taxonomy classification standardizes category and likely root cause, then hands off to policy-grounded checklist generation and constrained drafting under review. Because these outputs are bounded and governable, denial work is especially well suited to task specialization under explicit confidence thresholds, dollar gates, and escalation rules - improving speed and consistency without sacrificing auditability.

Claim edits

A TSM for claim edit explanation interprets the edit, generates precise corrective instructions, flags whether coding or registration review is required, and structures the output for routing.
This reduces billing lag and improves clean-claim performance by translating cryptic edits into operationally usable action without relying on free-form interpretation by every individual biller.



A/R prioritization and follow up

A TSM for A/R next step normalization summarizes account state, normalizes inconsistent notes into a standard action status, and suggests the next operational step. When connected to prioritization signals (e.g., days-to-payment or payment ratio), the execution becomes more disciplined and effort can shift toward the most recoverable dollars rather than the loudest queue.

From prediction to execution: connecting prioritization to action

Healthcare finance already uses predictive signals - days-to-payment, payment ratio, denial propensity - to inform prioritization. Those signals identify where attention is likely to matter; they do not complete the work. The architectural opportunity is to connect prediction to governed execution: prediction sets priority, while TSMs execute bounded tasks (classification, checklist generation, constrained drafting) under the control layer's gates.

What healthcare leaders should do next

The practical path forward is not a vague enterprise AI initiative. Start with one bounded workflow, one measurable operating problem, and a decision-ready pilot. Choose workflows where outputs are constrained and success criteria are clear - then build the control layer from day one so scaling is earned through measured gates, not asserted.

Measure the program with proof metrics that reflect execution lift: touches per case, time-to-first-action, cycle time, defect rate, and overturn yield. When those improve versus baseline, the downstream financial effects tend to follow as operational consequences, not marketing claims.

How can Vee Healthtek help

Vee Healthtek helps healthcare leaders operationalize Task Specialized Models (TSMs) as a production execution layer across the revenue cycle - from patient access through denials, A/R, and payment integrity - so models produce artifacts teams can run with, not side outputs that require manual translation.

We bring practitioner depth to define denial taxonomies aligned to real work, queue logic that matches operational routing, and playbooks calibrated to payer behavior. We implement schema-first TSM execution, policy grounding against versioned payer sources, explicit confidence / dollar / complexity gates, and feedback loops where overrides become structured training signal. The combination is what compounds: each interaction improves the system's execution accuracy and governance posture over time.

Conclusion

The most important strategic shift is not from no AI to AI everywhere. It is from generic AI usage to a governed execution architecture built for the realities of healthcare finance. The more credible path is to place the right kind of intelligence in the right part of the workflow: TSMs for bounded execution tasks, predictive models where prioritization matters, and a control layer that makes outcomes auditable, governable, and enterprise ready.

FAQs

What are Task Specialized Models (TSMs) in healthcare Revenue Cycle Management (RCM)?

TSMs are models built to execute a bounded RCM task with structured outputs - such as eligibility interpretation, claim edit explanation, denial taxonomy classification, or A/R next-step normalization - under explicit evaluation gates and auditability controls.

Are Task Specialized Models (TSMs) the same as Small Language Models (SLMs)?

Not always. TSM describes scope (a model built to do a small number of bounded RCM tasks with structured outputs and tight feedback loops). SLM describes size or deployability (a smaller language model that may be used across many tasks). In practice, many TSMs are implemented using SLM-sized models - but a TSM is defined by task boundedness + schema + gates, not by parameter count.

Where do Large Language Models (LLMs) still fit in a TSM-first RCM architecture?

LLMs can be valuable as exception engines, not default execution engines - e.g., when a case requires cross-document synthesis, unusual payer language, or complex narrative assembly. A TSM-first architecture typically routes routine high-volume work through task-specialized executors and escalates only low-confidence or high-complexity cases to heavier reasoning. This avoids workflow-wide dependence on a single general model and keeps execution behavior stable.

If we already have automation + rules + RPA, why do we need SLMs or TSMs?

Rules and Robotic Process Automation (RPA) work best when inputs and decision paths are stable. Many RCM tasks are stable tasks with unstable phrasing (payer text, denial narratives, portal messages, notes). SLMs / TSMs fill that interpretive gap by converting language-heavy inputs into structured, workflow-ready outputs, while a control layer enforces minimum-necessary context and auditability (a key HIPAA concept).

About Vee Healthtek

Vee Healthtek is a technology-led revenue cycle partner for U.S. health systems and physician groups. We treat revenue cycle as a structural determinant of growth and resilience, not an administrative process. Our modular capabilities span every stage of the revenue cycle, enabling both targeted interventions and full-cycle transformation. By combining engineered workflows, practitioner insight, and a global delivery architecture, we turn revenue friction into flow - from access to A/R - and shift it from a cost center to a revenue performance system.

Granite Park 1, Suite 850, 5800 Granite Parkway
Plano, TX 75024, United States

VEE HEALTHTEK