Frontiers in Computer Science and Artificial Intelligence
ISSN: 2978-8048
DOI: 10.32996/fcsai
Journal Homepage: www.al-kindipublisher.com/index.php/fcsai

FCSAI
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Use-Case-Driven Model Selection and Configuration for Enterprise AI Workflows

Sathiesh Veera
GenAI Solutions Architect, Phoenix, AZ, USA.
Corresponding Author: Sathiesh Veera, **E-mail**: sathyvsk@gmail.com

| ABSTRACT

Enterprise adoption of generative artificial intelligence (GenAI) has grown rapidly, with 78% of organizations deploying AI in at least one business function as of 2025. Yet studies indicate that 95% if enterprise GenAI pilots fail to deliver measurable business impact. A key contributor to this failure is the ad-hoc, trial-and-error approach that organization use to select and configure large language models (LLMs) for production workflows. Multiple commercial and open-source models are now available, each with different strengths, but enterprise lack a practical methodology for matching models to specific workflow needs and configuring them well. This paper proposes a reference architecture for use-case-driven LLM selection and configuration in enterprise environments. The first layer introduces a Workflow task profiling framework that breaks the workflows into characterized task units along four dimensions: task nature, data profile, output requirements and interaction pattern. The second layer presents a Task-Driven Capability Matching approach that starts from the task profile, identifies the capability categories the task requires, and then identifies candidate models that satisfy those requirements. This task-first direction ensures selection decisions are driven by what the workflow needs, not by model marketing claims or benchmark rankings. The last layer defines a two-dimensional configuration approach where each parameter is driven by the task profile, the selected model, or both. Task-driven parameters such as chunking strategy and corpus structure might remain stable across model changes, while model-specific parameters such as prompt format and message structure change with the model. This enables model transitions while also accounting for relevance of the configurations to the workflow requirements and not just the models. The framework is demonstrated through end-to-end walkthroughs of representative enterprise workflows.

| KEYWORDS

Large Language Models, Model selection, Agentic AI Workflows, GenAI, Use-case-driven, Enterprise AI, Effective Model Selection, Task-Driven capabilities, AI Workflow Optmization

| ARTICLE INFORMATION

ACCEPTED: 01 March 2026 **PUBLISHED:** 25 March 2026 **DOI:** 10.32996/jcsts.2026.5.5.1

1. Introduction

Enterprise adoption of Agentic AI workflows has grown at a pace few anticipated. According to McKinsey, 78% of organizations reported using AI in at least one business function by 2025, up from 55% just one year prior (Singla et al., 2025). Model API spending more than doubled from $3.5 billion to $8.4 billion between late 2024 and mid-2025 (Tully et al., 2025). Organizations are deploying customer-facing workflows such as chatbots, billing assistants, and document processors, alongside internal workflows for knowledge management, code assistance, data analytics etc. Despite this investment, the results have not followed. The Massachusetts Institute of Technology (MIT) reports that 95% of enterprise GenAI pilots fail to deliver measurable profit-and-loss impact (Challapally et al., 2025). The study, based on analysis of 300 public AI deployments and 150 executive interviews, identifies the core issue not as model quality but as a gap in how organizations integrate AI into their workflows. The report notes that generic tools stall in enterprise use because they do not learn from or adapt to the workflows they are meant to support.

One major part of this integration problem is model selection and configuration. The commercial LLM landscape now includes multiple frontier-class models from OpenAI, Anthropic, Google, Meta, Mistral, and DeepSeek, each built differently and strong in different areas (Ong et al., 2025). No single model is best at everything, yet most enterprise teams select models through ad-hoc

experimentation, trying a model, evaluating outputs informally, and proceeding or abandoning without a clear rationale for the choice.

Poor model selection is expensive in ways that are not always obvious. Using an overpowered model for simple classification tasks wastes resources, while using an underpowered model for complex reasoning produces unreliable outputs. When a team switches models, the prompts that worked well on the previous model often produce mediocre results on the new one. RAG parameters may be misaligned, and function-calling patterns may break entirely (Chen et al., 2023). The team then spends weeks in another cycle of trial and error which is exactly the kind of approach the MIT study identifies as a primary failure mode.

This paper addresses that gap by proposing a three-layered reference architecture for workflow driven LLM selection and configuration. The architecture starts from the workflow's tasks, identifies what capabilities each task requires, maps those requirements to candidate models, and then provides configuration guidance driven by both the task profile and the selected model. The contribution is not another model comparison or routing system, but a task-first decision framework that helps engineering teams make informed, structured choices.

2. Literature Review

Researchers and practitioners have approached the problem of LLM selection and optimization from several angles. This section reviews three categories of related work: runtime model routing, model comparison and benchmarking, and LLM optimization techniques.

2.1 Runtime Model Routing

Several systems tackle the problem of routing individual queries to the right model at inference time. RouteLLM (Ong et al., 2025), published at the International Conference on Learning Representations (ICLR) 2025, learns routers from preference data that direct queries to either a strong model (higher quality, higher cost) or a weaker model (lower quality, lower cost) based on query characteristics. FrugalGPT (Chen et al., 2023) takes a cascade approach, sequentially querying models of increasing capability and cost, and achieved up to 98% cost reduction while matching the performance of the most expensive model. OptiRoute (Piskala et al., 2025) extended this idea by factoring in both functional criteria (accuracy, speed, cost) and non-functional criteria (helpfulness, harmlessness). Red Hat's Semantic Router project uses BERT-based embeddings to classify the content of prompts and route them to specialized models based on domain-matched performance benchmarks (Haberman et al., 2025). Commercial platforms including Portkey, OpenRouter, and LiteLLM provide gateway-level routing with policies based on cost, latency, or round-robin distribution.

These routing systems are useful, but they work at the individual query level during runtime. They do not help with the design time question of how to analyze a full enterprise workflow, break it into tasks, and decide which models is optimal for the task.

2.2 Model Comparison and Benchmarking

Researchers have compared LLM capabilities across many standardized benchmarks. Chatbot Arena (Chiang et al., 2024) uses human preference data to create ELO style rankings of model quality. Benchmarks such as MMLU, HumanEval, SWE-bench, and GPQA Diamond test specific capabilities in reasoning, coding, and domain knowledge. Industry reports from Menlo Ventures (2025) and others provide market-level analysis of model adoption and capability trends. But benchmark driven selection has clear limitations in practice. Enterprise data looks fundamentally different from benchmark datasets (SAP Research, 2025), and a model's score on a standardized test does not reliably predict how it will perform on a specific enterprise task. Benchmark scores are also averages, they smooth over how a model actually behaves on particular task types or data profiles.

2.3 LLM Optimization Techniques

Prompt engineering, retrieval-augmented generation (RAG), and fine-tuning are three well-known techniques for improving LLM performance on specific use cases. These are not steps on a ladder, but each one addresses a different kind of problem with its own trade-offs (Kamal, 2026). RAG works best when factual accuracy and current information matter most, fine-tuning when deep behavioral specialization is needed, and prompt engineering when flexibility and fast iteration are the priority. (IBM, 2025) notes that in practice, these methods are often combined. What does not exist today is a single framework that connects these pieces end to end - from analyzing a workflow, to selecting the right model for its tasks, to configuring that model based on how it actually works.

3. The Architectural Transparency Problem

In most areas of software engineering, teams make technology choices with full visibility into how the technology works. When an organization selects a database such as PostgreSQL versus MongoDB versus Cassandra, the internal architecture is well understood. The teams learn about the storage engines, indexing strategies and query execution paths etc., which are all public knowledge. Engineers can reason about why a particular technology fits their use case based on how it works, not just what the vendor claims it can do. LLMs break this pattern. For the first time, enterprises are building production systems on top of

technologies whose internal workings are largely undisclosed. The degree of transparency varies significantly across model families, and this variation has direct implications for how informed a selection decision can be.

Among open-source models, architectural details are fully documented. DeepSeek V3 uses a Mixture-of-Experts (MoE) architecture with 671 billion total parameters but only 37 billion activated per token, combined with Multi-Head Latent Attention (MLA) (DeepSeek-AI et al., 2024). Meta's Llama 4 uses MoE with early fusion multimodality - text, images, and video are processed in a unified architecture from the start of training with 17 billion active parameters out of 400 billion total and an iRoPE architecture that supports its 10-million-token context window (Meta, 2025). Mistral Large 3 uses MoE with 41 billion active parameters out of 675 billion total, with a 256K context window (Mistral AI, 2025). For these models, an engineering team can reason directly about the architecture: MoE means lower inference cost per token, early fusion means more coherent multimodal understanding, MLA means more efficient handling of long contexts. However, commercial models such as OpenAI, Claude or Gemini do not disclose any internals of the architecture that the LLMs are built on. OpenAI's GPT-4 technical report explicitly states that it "contains no further details about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar". The report confirms only that GPT-4 is "a Transformer-based model pre-trained to predict the next token in a document." Anthropic has not published a technical report disclosing Claude's architecture, parameter count, or training methodology. Google published a technical report for the original Gemini 1.0 confirming it as a multimodal model trained from the ground up, but detailed architectural specifications for the current Gemini 2.5 and 3.0 generations are not publicly available.

This opacity matters for model selection in a way that existing approaches do not address. Runtime routing systems like RouteLLM and FrugalGPT delegate the selection decision to another model, often a smaller classifier or scoring model that predicts which LLM will handle a query best. But this means the selection is only as good as the routing model's predictions, and the engineering team still has no informed understanding of why one model was chosen over another for a given task. The decision remains a black box built on top of a black box.

4. Proposed Reference Architecture

This paper proposes the following reference architecture where the model selection is performed based on the specific task profiles instead of benchmark results and ranking models.

4.1 Workflow Task Profiling Framework

The first layer of the reference architecture focuses on understanding the AI workflow before any model is chosen. This is the most crucial part of building any AI driven workflow, and the task profile would be the foundation for everything that follows. The idea is straightforward - an enterprise workflow is made up of distinct tasks, and each task has different requirements. By profiling these tasks in a consistent way, the output of this layer feeds directly into model selection. The profiling covers the following four dimensions:

Task Nature: Each task within a workflow is classified by what it primarily needs to do: generate new text (summaries, responses, creative content), extract information from documents, perform multi-step reasoning or analysis, classify inputs into categories, or process images, audio, or video alongside text etc. A customer service workflow, for example, typically includes a classification task (detecting intent), an extractive task (pulling account information), and a generative task (composing a response). These are different jobs and may call for different model strengths.

Data Profile: The data that is required to perform the task is characterized by its structure (databases, unstructured documents, APIs), volume (a single document versus a large corpus), modality (text-only or multimodal), and sensitivity (public, internal, or regulated). The data profile matters for both model selection and configuration, since models handle different data types with different levels of effectiveness.

Output requirements: Each task has constraints on what its output must look like, that is, how accurate it needs to be, how fast the response must arrive, whether the output needs to be auditable or compliant with specific regulations, does it need to include citations and what the cost ceiling is per transaction or workflow execution.

Interaction Pattern: The interaction characteristics of each task are documented: whether it is single-turn or multi-turn, how much context carries over between turns, whether it requires tool calling or function calling, and whether the task runs independently or as part of a multi-agent orchestration. Models differ considerably in how well they maintain context across turns, how reliably they call tools, and how they perform in agentic setups, making this dimension directly relevant to selection.

The output of this layer is a task profile for each component of the AI workflow. In multi-step agentic workflows, different tasks will often produce different profiles and that is what makes multi-model orchestration possible, as described in second layer of the architecture.

4.2 Task-Driven Capability Matching

The second layer takes the task profiles from the first layer and works outward toward model selection, starting from the task, not from the model. The direction matters: instead of listing what each model is good at and asking the practitioner to match backwards to their task, this layer asks what capability categories the task requires and then identifies which models currently satisfy those requirements. This task-first direction means that when a new model enters the market or an existing model gains a new capability, it naturally becomes a candidate for the task profiles that need that capability, without changing the selection methodology.

4.2.1 Capability Taxonomy

The framework defines six common capability categories for evaluating models against task requirements. This can serve as an initial list, however organizations can develop further, and add taxonomies that are relevant to their workflows.

Reasoning Depth: How well the model handles multi-step logic, mathematical problem-solving, and complex synthesis. Models with dedicated reasoning modes (such as chain-of-thought training or extended thinking) tend to perform better here. This category maps most directly to tasks that were profiled with a reasoning task nature in Layer 1.
Multimodal Processing: How well the model processes and reasons across text, images, audio, and video. Models that were trained from the ground up on multimodal data (such as Google's Gemini family) tend to outperform models where vision was added after initial training. This matters for workflows that involve document scanning, visual inspection, or media analysis.
Instruction Adherence: How reliably the model follows complex, multi-constraint system prompts and user instructions. In enterprise workflows, outputs often need to match specific formats, use required regulatory language, or follow brand guidelines. A model that frequently drifts from instructions creates rework and risk.
Structured Output Fidelity: How reliably the model produces valid JSON, XML, or other structured formats that downstream systems need to parse. Many enterprise workflows depend on LLM outputs being consumed programmatically, so a model that occasionally produces malformed output can break an entire pipeline.
Context Utilization Efficiency: How well the model actually uses a long context window. Models differ not just in their maximum token limit but in how well they retain and reference information spread across a large input. This matters most for RAG-based workflows where large amounts of retrieved content are injected into the prompt.
Tool-Calling Fidelity: How reliably the model calls functions, uses tools, and executes agentic actions. For workflows that involve database queries, API calls, or multi-step orchestration, this is often the factor that makes or breaks the model choice.

The capability taxonomy defined above is designed as a practical approach to the reality of model architecture transparency problem. Where architecture is disclosed as in case of open-source models, the teams can reason from architecture to expected capability. Where architecture is not disclosed, the taxonomy provides an empirical proxy, that is, instead of asking what a model's architecture is, the teams try to understand how a particular model actually perform on tasks that require - reasoning depth, multimodal processing, instruction adherence, and so on. The capability categories are designed to be measurable through evaluation, regardless of whether the underlying architecture is known. While this does not solve the transparency problem, it gives engineering teams a structured, evidence-based way to make model selection decisions.

4.2.2 Task Profile to Model Mapping

Table 1 presents the task-to-model mapping as a practitioner would use it: starting from a task profile (the output of Layer 1), identifying which capability categories are most important for that task, and listing current model families that are strong candidates. The model names in the rightmost column are current examples, not permanent prescriptions and they reflect the state of available models at the time of writing and should be validated against each organization's own evaluation. When a model gains a new capability or a new model enters the market, it simply enters the candidate pool for the task profiles that require that capability. The architectural basis for the open-source models listed here is documented in Section 3.

Table 1. Task-Profile-to-Model Mapping

Task Profile (from Layer 1)	Primary Capability Need	Secondary Need	Current Model Candidates
Customer intent classification	Low latency, cost efficiency	Structured output	Lightweight / open-source tiers (Llama, Mistral)
Policy-compliant response generation	Instruction adherence	Context utilization	Claude, GPT
Document ingestion (scanned / image)	Multimodal processing	Structured output	Gemini, Llama 4
Multi-document analytical synthesis	Reasoning depth	Context utilization	GPT, Claude, DeepSeek R1
Voice agent / real-time conversation	Native audio processing	Tool-calling fidelity	GPT-4o Realtime, Gemini Live
Natural language to SQL / API calls	Tool-calling fidelity	Reasoning depth	GPT, Claude, Mistral
Multilingual customer support	Multilingual fluency	Instruction adherence	Gemini, Llama 4, Mistral
Mathematical / logical analysis	Reasoning depth	Structured output	DeepSeek R1, GPT
Email drafting / brand-consistent content	Instruction adherence	Context utilization	Claude, GPT
Speech-to-text transcription	Native audio processing	Multilingual fluency	GPT-4o Transcribe, Gemini Native Audio

Note: This mapping is a general guideline, not a prescription. Some of the Tasks might need more than the listed capabilities needed as primary and secondary. Model candidates reflect current capabilities at the time of writing and should be validated against each organization's own evaluation data. When a model gains a new capability, it enters the candidate pool for the task profiles that require it.
Sources: (Tully et al., 2025), (Introducing Next-Generation Audio Models in the API, 2025), (Clarifai, 2025), (Ankur Bapna, 2025), (Meta, 2025), (Mistral AI, 2025), (Guo et al., 2025).

As you could notice, the table also includes few additional categories beyond the six core taxonomy dimensions, such as native audio processing and multilingual fluency. The engineering teams can add relevant capabilities, but keeping them minimal is advised to allow categorization and model selection easy.

4.2.3 Multi-Model Orchestration

When Layer 1 produces different task profiles for different steps in the same workflow, Layer 2 supports assigning different models to each step. A document processing pipeline, for example, might use a multimodal-strong model for document ingestion and extraction, a reasoning-strong model for analysis and synthesis, and a cost-efficient model for formatting the final output. Anthropic's research on building effective agents confirms that the most successful production implementations tend to use simple, composable patterns where different models or configurations handle different workflow stages (*Building Effective Agents*, 2024). The orchestration pattern follows the workflow's interaction profile from Layer 1, with defined hand-off points and data contracts between stages.

4.3 Model-Aware Configuration

Once a model is selected, the workflow's configuration needs to be tuned. But configuration is not one-dimensional. It is not simply about adapting everything to the selected model. Each configuration parameter is driven by some combination of the task profile from Layer 1 and the selected model from Layer 2. Some parameters are primarily task-driven and remain stable when models change. Others are primarily model-driven and must change with every model swap. And many are influenced by both. Layer 3 organizes configuration into three tiers based on what drives the parameter, which in turn determines when each tier needs to be revisited. Figure 1 illustrates the complete architecture including the tiered configuration model.

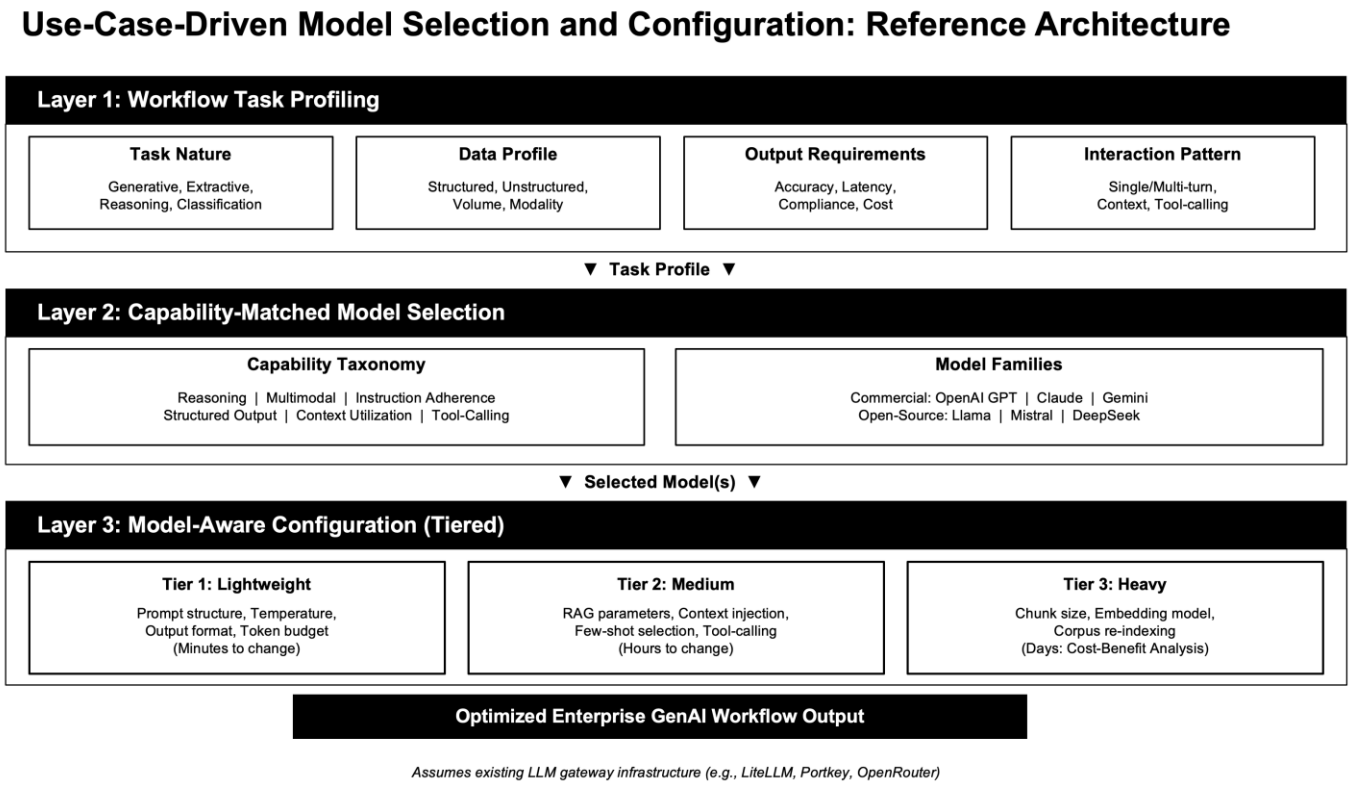

Figure 1: Use-case-Driven Model Selection and Configuration

4.3.1 Tier 1: Light-Weight Configuration

Tier 1 covers primarily model-driven parameters that change with every model swap but are quick to adjust. This includes prompt syntax and system instruction format (each model responds best to a different prompt structure), API-specific settings and function-calling schemas, and response parsing logic. These parameters are shaped almost entirely by how the model works, not by what the task requires. They take minutes to implement and test, and should always be updated when changing models.

4.3.2 Tier 2: Medium-Weight Configuration

Tier 2 covers parameters driven by both the task profile and the model. This includes temperature and sampling settings (a reasoning-heavy task needs low temperature on any model, but the optimal value differs between GPT and Claude), retrieval count and similarity thresholds (how much context to retrieve depends on the task's accuracy requirements, but how to inject it depends on the model's context handling), few-shot example selection (the type of examples follows the task, but the number and format depend on the model), and tool-calling retry logic (the task defines what actions are needed, the model determines how reliably it executes them). Tier 2 changes take hours of testing. They need to be revisited both when models change and when task requirements shift, for example, if a workflow's accuracy requirements are tightened, retrieval thresholds and few-shot strategies may need adjustment even on the same model.

4.3.3 Tier 3: Heavy-Weight Configuration

Tier 3 covers primarily task-driven infrastructure that is designed around the data profile and accuracy requirements from Layer 1, that is chunk size and overlap strategy (determined by document types and how much context each piece needs to carry), embedding model selection and corpus indexing (determined by the data's domain and vocabulary), and knowledge base structure (determined by how the workflow needs to access and retrieve information). These parameters are expensive to change and they should remain stable when only the model changes. If chunking and embedding were designed around the task's data profile, a model swap should not require Tier 3 rework. Tier 3 changes are triggered when the task profile itself changes - when the business adds a new document type, tightens accuracy requirements, changes data sources, or restructures the workflow. A model swap may also trigger Tier 3, but only when the new model's context handling or capabilities are so different from the previous one that the existing infrastructure cannot serve it adequately and that decision should be based on measured performance, not assumed.

This two-dimensional view of configuration has a practical consequence: it makes model transitions cheaper. When Tier 3 infrastructure is designed around the task profile rather than a specific model, swapping models only triggers Tier 1 and Tier 2

work. Teams that design their chunking, embedding, and retrieval infrastructure around their data and accuracy requirements, rather than optimizing for a single model's quirks, will find that model changes require hours of reconfiguration rather than weeks of rebuilding.

5. Performance Feedback and Re-evaluation Loop

Production environments are not static. Models get updated, workflow requirements shift, and new models enter the market. Without a mechanism to detect when a selection or configuration is no longer performing well, teams default to reactive firefighting. The framework addresses this through a cross-cutting performance feedback loop that routes production signals back through the appropriate layer for structured re-evaluation. The contribution is not the monitoring itself that most organizations already track including error rates, latency, and user satisfaction, but connecting those signals to a graduated re-evaluation process tied to the framework's layers.

6. Practical Application: Enterprise Workflow Walkthroughs

This section walks through the framework from end to end, applying it to a sample enterprise AI workflow.

6.1 Customer Service Workflow

Layer 1: Task Profiling. A typical customer service workflow breaks into three tasks:

(a) intent classification, which is categorizing the incoming query into billing, technical support, account inquiry, and so on;

(b) information retrieval, which is pulling relevant account data and knowledge base articles; and

(c) response generation, which is composing a helpful, policy-compliant reply.

The task profiles are different: intent classification is a classification task with tight latency requirements, information retrieval is extractive and works with structured data, and response generation is generative with compliance constraints.

Layer 2: Model Selection. Because the tasks have different profiles, a multi-model approach makes sense. Intent classification is straightforward and high-volume, so it maps to a cost-efficient model - a smaller open-source model or a lightweight API tier can handle it well. Response generation needs stronger instruction adherence and context utilization, so it maps to a more capable commercial model. In practice, many organizations end up using a cheap model for classification and a more expensive one for generating the actual response.

Layer 3: Configuration. At Tier 1, the team optimizes the system prompt for tone, compliance language, and response structure specific to the generation model. At Tier 2, they tune RAG retrieval parameters: how many knowledge base articles to retrieve, what similarity threshold to use, and where in the prompt to inject the retrieved context. Tier 3 considerations are minimal for an initial deployment but will come up if the organization later switches models, at which point they should evaluate whether re-indexing the knowledge base is worth the expected accuracy improvement.

7. Future Work

The framework presented in this paper establishes a foundation for systematic LLM selection and configuration. Several directions for future work would strengthen and extend the approach.

Standardized Enterprise Capability Benchmarks: The six capability categories in Layer 2 are currently mapped to models using published benchmarks, architectural analysis, and practitioner experience. A valuable direction for future work is developing standardized evaluation methods specifically designed to measure each capability category under enterprise-relevant conditions, using domain-specific data, realistic latency constraints, and production-representative workloads rather than academic benchmarks alone. This would make the selection methodology more rigorous and repeatable across organizations.

Automated Task Profiling: Layer 1 currently relies on manual workflow decomposition and task characterization. As tooling for enterprise GenAI matures, it should be possible to partially automate this step, for example, by analyzing workflow logs, API call patterns, and data flow graphs to generate draft task profiles that a practitioner can then review and refine. This would lower the effort required to apply the framework and make it practical for organizations with large numbers of workflows.

Formal Cost-Benefit Models for Tier 3 Decisions: The framework calls for explicit cost-benefit analysis before heavy configuration changes such as corpus re-indexing or embedding model changes, but it does not yet provide a formal quantitative model for this analysis. Future work should develop estimation models that predict the expected accuracy improvement from Tier 3 changes based on factors such as corpus size, task complexity, and the degree of architectural difference between the old and new models. This would give engineering teams a data-driven basis for deciding when the rebuild is worth the investment.

8. Conclusion

The growing number of capable LLMs has made model selection a real problem for enterprise organizations. With 37% of enterprises now running five or more models in production, and model API spending doubling year over year, the cost of getting selection and configuration wrong adds up quickly. The 95% pilot failure rate reported by MIT makes the point clearly: the gap

between model capability and business value is not about the models themselves, but it is about how organizations integrate them into workflows.

This paper has proposed a three-layer reference architecture that addresses that gap through structured, workflow-driven decision-making. Layer 1 gives teams a consistent way to describe what their workflows need from a model. Layer 2 connects those needs to model capabilities without tying decisions to specific vendor names. Layer 3 provides a practical, tiered approach to configuration that respects the reality that some changes are cheap and some are expensive and that the expensive ones should only happen when the expected benefit justifies the cost.

The design rests on a few core ideas: model selection should start with the workflow, not the benchmark; configuration must account for how a specific model works; and switching between models is an engineering cost best managed through tiered investment. By replacing ad-hoc experimentation with structured decision-making, this framework offers a practical path to the business impact that most GenAI initiatives have so far failed to deliver.

Funding: This research received no external funding
Conflicts of Interest: The authors declare no conflict of interest.
ORCID iD: 0009-0000-8207-9493
Publisher's Note:

References

[1]. Ankur Bapna. (2025, June 3). *Advanced audio dialog and generation with Gemini 2.5*. Google. https://blog.google/innovation-and-ai/models-and-research/google-deepmind/gemini-2-5-native-audio/

[2]. *Building effective agents*. (2024). Anthropic.com. https://www.anthropic.com/research/building-effective-agents

[3]. Challapally, A., Pease, C., Raskar, R., & Chari, P. (2025). *The GenAI Divide STATE OF AI IN BUSINESS 2025 MIT NANDA*. https://mlq.ai/media/quarterly_decks/v0.1_State_of_AI_in_Business_2025_Report.pdf

[4]. Chen, L., Zaharia, M., & Zou, J. (2023). *FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance*. https://doi.org/10.48550/arxiv.2305.05176

[5]. Chiang, W.-L., Zheng, L., Sheng, Y., Angelopoulos, A. N., Li, T., Li, D., Zhang, H., Zhu, B., Jordan, M., Gonzalez, J. E., & Stoica, I. (2024). Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. *ArXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2403.04132

[6]. Clarifai. (2025, November 25). *Gemini 3.0 vs GPT-5.1 vs Claude 4.5 vs Grok 4.1: AI Model Comparison*. Clarifai.com; Clarifai. https://www.clarifai.com/blog/gemini-3.0-vs-other-models

[7]. DeepSeek-AI, Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Guo, D., Yang, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., & Luo, F. (2024). DeepSeek-V3 Technical Report. *ArXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2412.19437

[8]. Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., Zhang, X., Yu, X., Wu, Y., F, W. Z., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., & Xue, B. (2025). DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, *645*(8081), 633–638. https://doi.org/10.1038/s41586-025-09422-z

[9]. Haberman, R., Chen, H., Ovadia, Y., Noriega De Soto, R., & Fredette, A. Brewster, D. (2025, November 11). *Intelligent inference request routing for large language models*. Red Hat Emerging Technologies. https://next.redhat.com/2025/11/11/intelligent-inference-request-routing-for-large-language-models/

[10]. IBM. (2025, February 7). *rag vs fine-tuning vs. prompt engineering*. Ibm.com. https://www.ibm.com/think/topics/rag-vs-fine-tuning-vs-prompt-engineering

[11]. *Introducing next-generation audio models in the API*. (2025). Openai.com. https://openai.com/index/introducing-our-next-generation-audio-models/

[12]. Kamal, I. (2026, January 29). *Prompting vs. RAG vs. fine-tuning: Why it's not a ladder*. The New Stack. https://thenewstack.io/prompting-vs-rag-vs-fine-tuning-why-its-not-a-ladder/

[13]. Meta. (2025, April 5). *The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation*. Meta.com. https://ai.meta.com/blog/llama-4-multimodal-intelligence/

[14]. Mistral AI. (2025, December 2). *Introducing Mistral 3*. Mistral.ai. https://mistral.ai/news/mistral-3

[15]. Ong, I., Almahairi, A., Wu, V., Chiang, W.-L., Wu, T., Gonzalez, J. E., Waleed, K. M., & Stoica, I. (2025). *RouteLLM: Learning to Route LLMs with Preference Data*. ArXiv.org. https://arxiv.org/abs/2406.18665v4

[16]. Piskala, D. B., Raajaa, V., Mishra, S., & Bozza, B. (2025). Dynamic LLM Routing and Selection based on User Preferences: Balancing Performance, Cost, and Ethics. *ArXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2502.16696

[17]. Singla, A., Sukharevsky, A., Yee, L., Chui, M., & Hall, B. (2025, March 12). *The state of AI: How organizations are rewiring to capture value*. McKinsey & Company. https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-how-organizations-are-rewiring-to-capture-value

[18]. Tully, T., Redfern, J., Das, D., Xiao, D. (2025, July 31). *2025 Mid-Year LLM Market Update: Foundation Model Landscape + Economics | Menlo Ventures*. Menlo Ventures. https://menlovc.com/perspective/2025-mid-year-llm-market-update/