Institute and Faculty of Actuaries

Independent thinking from the IFoA

September 2025

think

The Agentic Model Office: AI agents and the future of insurance organisations

by **Daniel Ramsay**

Independent thinking from the IFoA

Part of the IFoA's purpose is to promote debate within and beyond the profession, and to position our members as leading voices on the biggest public policy challenges of our time.

We aim to showcase the diverse range of expertise and critical thinking both within and outside the profession.

Our 'think' series seeks to promote debate on topics across the spectrum of actuarial work, providing a platform for members and stakeholders alike and sharing views that may differ from the IFoA's house view. In doing this, we hope to challenge the status quo, question the orthodoxy, and shine a light on complex or under-examined issues, thereby stimulating discussion and dialogue to help tackle issues in a different way.

Daniel Ramsay

Daniel Ramsay FFA is an actuarial technologist based in Glasgow, Scotland, specialising in the development of GenAI tools to streamline the work of insurance actuaries. His recent projects include the RiskAgilityFM AI Assistant, an AI agent designed to assist with building and debugging actuarial cashflow models.

Daniel is a founder and co-chair of the IFoA's GenAI Working Party and has presented on the topics of GenAI and AI agents at the IFoA Life conferences in 2023 and 2024. Daniel is also a member of the International Actuarial Association AI Taskforce and represents the IFoA at the Actuarial Association of Europe's Data Science and AI Working Group.

Contents

think

1. Introduction

In the insurance industry, the term 'model office' traditionally refers to a complete representation of an insurer's business lifecycle, from policy issuance and premium billing through claims settlement, investment, reinsurance, and solvency reporting.

Historically, these model offices were implemented as a patchwork of spreadsheets, legacy applications, and point solutions, stitched together by human effort.

Today, the rise of Generative AI (GenAI) promises a different trajectory for transformation. AI agents – systems capable of planning, reasoning, and using tools to accomplish open-ended tasks – can take on a growing share of operational, analytical, and compliance work.

In principle, such agents could soon be modifying actuarial models, managing data pipelines, interpreting regulatory changes, or orchestrating multi-step workflows across the insurer's value chain. However, achieving this vision is far from straightforward.

The current generation of insurer IT estates is fragmented, opaque, and reliant on tacit human knowledge. These environments are difficult for people to navigate and even harder for autonomous agents.

Without a deliberate redesign of the operating fabric, agents will stumble over brittle interfaces and hidden process rules, limiting their usefulness no matter how intelligent they become.

This paper introduces the Agentic Model Office (AMO), a future-state blueprint for integrating AI agents into insurance operations in a safe, transparent, and scalable way. AMO rests on two mutually reinforcing layers:

- **The Digital Model Office (DMO):** a machine-readable, application programming interface (API)-accessible representation of every relevant data item, rule, and executable process, designed for both human and agent interaction.
- **The Agentic Layer:** a governed network of transparent, specialist AI agents that operate on top of the DMO to execute and coordinate work.

Together, these layers form the foundation for an insurance operating system where intent can be expressed in plain language, execution is delegated to trusted agents, and every action is traceable and auditable. *Figure 1* illustrates this relationship.

The sections that follow examine why existing estates impede agents, define the required properties of the DMO, describe agentic patterns within the AMO, and outline a practical path from today's fragmented reality to this future state.

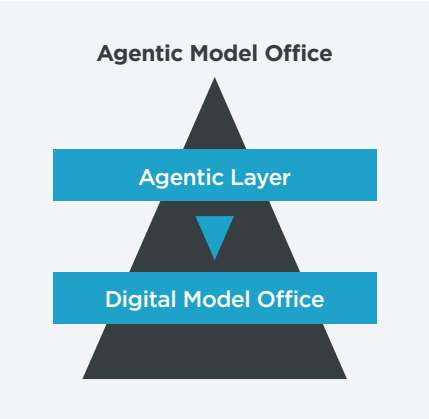


Figure 1: *The Agentic Model Office: an agentic layer operating atop the digital model office to enable transparent, AI-driven insurance operations.*

This paper introduces the Agentic Model Office (AMO), a future-state blueprint for integrating AI agents into insurance operations in a safe, transparent, and scalable way.

think

AMO in the context of the target operating models

Comprising a Digital Model Office (DMO) and an Agentic Layer, the Agentic Model Office (AMO) is an operating construct that sits alongside the enterprise TOM.

The DMO standardises and exposes data, models, documents, and workflows through governed APIs with provenance and replay.

The Agentic Layer plans and executes work by invoking those APIs under policy and approvals.

Together they provide a programmable operating fabric that complements TOM dimensions such as process, technology, controls, and accountability, without changing organisational roles or decision rights by default.

This paper is written for people who shape and build insurance technology in practice. The target audience includes technology-focused actuaries, insurance solution architects, transformation leaders, and practitioners who want to understand how AI agents may realistically be deployed over time.

This is not a prescription for immediate adoption. It is, rather, a thought experiment that explores how the insurance operating model could evolve if organisations sought to unlock the full benefits of agentic systems.

The scenarios and designs described here sketch a plausible end state in which a significant proportion of operational and analytical work is carried out by agents under strong governance, with people setting intent, overseeing changes, and focusing on judgement and accountability.

This is a long term, directional vision. It is meant to be a practical blueprint that can be implemented piece by piece, rather than via a 'big bang' replacement. At its core, the goal is to make the insurer legible to software.

When processes and data are exposed as APIs governed by policy, trustworthy agents can execute work transparently, safely, and at scale. Agent autonomy is bounded by policy, and material actions remain auditable and subject to human approval, so you get the benefits without blind trust.

This is not a prescription but a thought experiment that explores how the insurance operating could evolve if organisations sought to unlock the full benefits of agentic systems.

think

Reader's guide to terms used in this paper

This box aligns terminology across actuarial, risk, and engineering readers so the rest of the paper can focus on concepts. See **Appendix B** for fuller definitions.

- **Agentic Model Office (AMO):** The architecture for deploying agents safely in insurance. It comprises the DMO and the Agentic Layer and does not, by itself, redefine organisational roles or decision rights in the TOM.
- **Digital Model Office (DMO):** The digital fabric. Data, models, documents, workflows, controls, and execution environments are programmatically accessible, versioned, and auditable.
- **Agentic Layer:** The governed network of AI agents and the agent platform components they use. These agents plan, call tools, and orchestrate work by invoking DMO capabilities.
- **Function or tool calling and model context protocol (MCP):** Structured calls that let AI agents invoke tools consistently. MCP standardises how tools and resources are exposed to models. Example: run_model(parameters) with typed inputs and logged outputs.
- **Application programming interface (API):** A formal contract that lets one program read, write, or execute data and functions offered by another program without manual intervention.
- **Continuous integration and continuous delivery (CI/CD):** Automated pipelines that build, test, and package each change so software is always in a releasable state. In this context it applies to model code, data pipelines, and configuration, with promotion controlled by tests and approvals.
- **Policy-as-code:** Governance rules written as code and enforced automatically in pipelines and gateways. Example: a rule that blocks deploying a model unless regression deltas are within tolerance and a named approver has signed off.
- **Containers, Docker, Kubernetes:** Containers package software with its dependencies so it runs the same everywhere. Docker is the tooling for building and running containers. Kubernetes schedules and manages containers across many machines for scale and reliability.
- **Workflow engines (Airflow, Temporal):** Software that defines and runs multi-step jobs with dependencies, schedules, retries, and state tracking. Example: a nightly chain that ingests data, validates it, runs models, and publishes reports with alerts on failure.
- **Relational and non-relational databases:** Relational databases use structured tables and SQL for strong consistency and joins. Non-relational databases store semi-structured data with flexible schemas, which suits evolving documents and events.
- **Object or blob storage:** Scalable storage for large unstructured files such as binaries, PDFs, reports, and model artefacts, accessed via keys and metadata rather than tables and rows.

2. AI agents and their role in the Agentic Model Office

There are many modern definitions of GenAI-based agents in circulation. The following two, from two leading AI labs, may help clarify the concept.

OpenAI (2024) provides the definition:

- Agents represent systems that intelligently accomplish tasks, ranging from executing simple workflows to pursuing complex, open-ended objectives.

Anthropic (2024), however, makes a distinction between workflows and agentic systems:

- Workflows: Systems where large language models (LLMs) and tools are orchestrated through predefined code paths.
- Agents: Systems where LLMs dynamically direct their own processes and utilise tools autonomously.

The term AI agent is often used loosely across the technology sector, covering everything from simple LLM-driven chatbots to fully autonomous systems with sophisticated decision-making capabilities.

For clarity, this paper uses the term to mean **software-based agents that can plan multi-step tasks, use tools, and draw on memory systems to adapt their actions over time.**

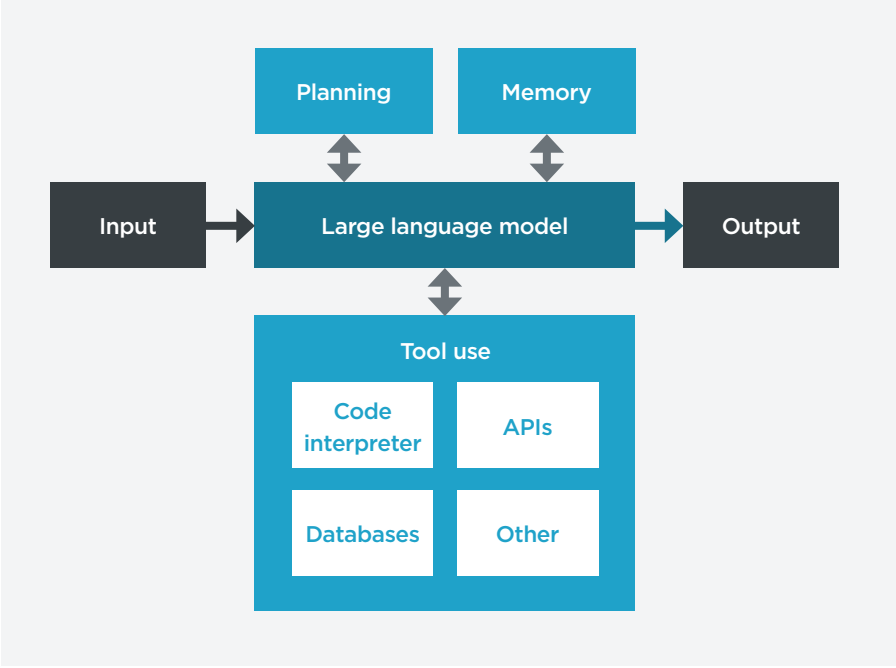


Figure 2: *AI agent systems enhance the core LLM's capabilities by adding planning, memory, and tool use.*

Within the AMO, specialist agents operate in the Agentic Layer and act on the DMO. Examples include adjusting actuarial models in response to new data, orchestrating claims workflows across multiple systems, or analysing regulatory changes to produce compliance impact reports.

The value of these agents is determined not only by their intelligence, but by the clarity and accessibility of the operational landscape they inhabit. If that landscape is fragmented, opaque, or inconsistent, even the most capable agent will struggle to act effectively.

3. Why agents stumble in today's estates

OSWorld (2024) is a benchmark designed to assess how effectively AI agents can use computer applications. The public leaderboard for OSWorld highlights a concerning trend: AI systems often struggle with basic computer-based tasks.

As of July 2025, the highest score achieved on the benchmark is just 45.2%, underscoring the current limitations of AI for such tasks.

However, post-mortems show that, although agents generally identify the right sequence of steps, they stumble over basic interface issues: missing a hidden scroll bar, mistyping a file path or overlooking an authentication pop-up.

Interface fragility hurts agent performance in any environment, but it becomes especially acute inside insurance firms, where decades of bolt-on policy, claims and modelling platforms have produced a dense mesh of loosely integrated systems. A single reserving process update may require:

- Flat files from a policy administration mainframe, delivered nightly
- Spreadsheets in a network drive called "temp_new_v2"
- A capital model that only runs on Windows Server 2012, accessible only via a Remote Desktop jump box running Windows Server 2022
- A third-party claims administrator portal accessible only through Citrix
- An investment blotter inside a legacy order management tool
- Emails that carry undocumented overrides.

The environment is riddled with implicit assumptions, and silent drift is common, with tools switched without notice, folder names changed on a whim, and undocumented workarounds proliferating.

No commercially available agent would be able to traverse that sprawl unaided. The root cause is not that the models are dim but that the ground on which they operate is opaque and relies on the tacit knowledge hidden away within unlogged habits of team members.

In other words, it is the environment, rather than the inherent capability of the agents, that limits their effectiveness.

This challenge echoes a familiar lesson from the past: advances do not deliver value when layered onto ill fitting or poorly integrated infrastructure.

A telling example comes from the late nineteenth century, when factories transitioned from steam power to electricity. Many initially retained the old long shaft layout, driving electric motors from a single line shaft, and productivity barely improved.

It was only when plants were redesigned around individual motors, shorter lines, clustered machinery, and new workflows that the advantages of electricity were fully realised.

Likewise, AI agents grafted onto steam era IT infrastructure, or modern systems that are still functionally siloed, suffer the same handicap: without redesigning the underlying environment, the full potential of new technologies cannot be unlocked.

It is the environment, rather than the inherent capability of the agents, that limits their effectiveness.

4. The Digital Model Office layer

4.1. Infrastructure as code (IaC) applied to insurance organisations

The Digital Model Office is a machine-readable mirror of every data entity, executable process and control that influences financial or operational output. Anything a human can open, edit or run must be addressable by an API and must leave an auditable trail that employees, AI agents and auditors alike can replay on demand.

This approach works best when supported by modular, lightweight APIs that expose single, well-defined functions rather than heavy, multi-purpose endpoints. Such modularity allows both humans and AI agents to assemble complex workflows from many small, reliable services, reducing operational risk and improving adaptability as systems evolve.

A useful reference point is the decade-long transformation that software engineering experienced during the DevOps revolution. The adoption of codified, programmatic infrastructure practices gradually displaced the manual building and deployment of applications.

By introducing continuous integration and continuous delivery (CI/CD), meaning automated pipelines that build, test, and package every change so software is always in a releasable state, and by coupling this with automated testing and short feedback loops, DevOps broke down the traditional barriers between developers and operators, enabling a more seamless and reliable workflow.

By managing insurance models and processes with end-to-end model governance enabled by modern DevOps tools and principles, every element, such as inputs, assumptions, code, and data, is transparently captured and tracked.

This includes items such as mortality tables, exposure curves, reinsurance programme terms, cashflow model code or economic scenario parameters. If any of these change, the update is recorded with a timestamp, author's name, and a clear reason for the change.

All of this is stored in a central system, making it easy to look up what changed, who made the change, and whether it was tested thoroughly.

As a result, actuaries and auditors can consult a single, readable log that traces a model's entire lineage, enabling them to reproduce, explain, and verify any result with full confidence.

Traditional IT and analytics work changes significantly in this environment. Instead of manually retrieving files, reconciling versions, or relying on undocumented handoffs, teams work from an always up to date, programmatically accessible source of truth.

This frees IT from repetitive data pull requests and allows analysts to focus on interpretation and decision making rather than data wrangling.

When every change to a model or process is recorded and managed in this way, it makes life much easier, not only for people but also for AI agents. Instead of searching through confusing folders or relying on colleagues to remember how something was done, agents can access a clear, up to date record of exactly how each model works and has changed over time.

This trusted digital record means agents can safely and reliably make updates, run checks, or answer questions, making the whole insurance operation faster, more accurate, and much easier to audit or explain.

Actuaries and auditors can consult a single, readable log that traces a model's entire lineage.

4.2. Properties and architecture of the Digital Model Office

In regulated insurance contexts, auditability is non-negotiable. Blind trust in autonomous systems is not acceptable for client-facing or prudentially material work. The DMO therefore elevates provenance, replayability, and change control to first-class concerns, so that every data item, rule, and executable process can be traced, reproduced, and explained on demand.

Against that backdrop, we propose that the Digital Model Office must satisfy two complementary checklists: the functional properties that make it safe and useful, and the technology stack that renders those properties feasible in practice.

Required properties

- **Comprehensive data access:** Every entity (policies, claims, investments, IT, etc) is documented and reachable through APIs.
- **Executable environments:** All workflows: ETL pipelines, models, rating engines, claims rules can be spun up, parameterised and shut down programmatically.
- **Provenance and auditability:** Immutable histories of data, code and configuration; every action is logged with its input, rationale and outcome.
- **Governance and security:** Role-based access, human-in-the-loop approval for high-impact changes.
- **External system integration:** Ability to connect with and orchestrate processes across proprietary systems such as policy administration platforms, claims systems, or rating engines, even if access is limited to high-level APIs, exported reports, or automation scripts.

Required technology stack

- **Data layer:** Relational databases (PostgreSQL, MS-SQL) for structured data; object or blob stores for large datasets.
- **Code and configuration layer:** Git-style repositories that hold modelling logic, policy rules and declarative infrastructure files.
- **Document layer:** Schema-less stores (MongoDB, blob storage, SharePoint) for documents.
- **Tool Interface Layer:** Standardised function or tool calling and tool catalogues that expose existing APIs to agents consistently. This includes the Model Context Protocol (MCP), a standard that advertises tools and resources to models in lightweight, reliable formats with growing ecosystem support.
- **Execution and orchestration:** Container runtimes (Docker) with container orchestrators (Kubernetes) for scheduling; workflow engines (Airflow/ Temporal) to coordinate data or business processes.
- **Observability and logging:** Centralised logging and metric collectors wired into every service.
- **Testing and CI/CD:** Pipeline engines (GitHub Actions/Jenkins) that run unit, integration and regression tests, enforce policy as code and sign artefacts before promotion to production.
- **Cloud infrastructure:** Scalable, on-demand environments (for example AWS, Azure, GCP) to host models, pipelines, storage, and orchestration layers, with security controls appropriate for regulated insurance data.

With these foundations in place, the Agentic Layer described in the next section can operate on a surface that is both machine-friendly and governance-ready, turning today's brittle landscape into a resilient, continuously evolving insurance factory.

For organisations with hybrid environments, these properties and technologies ensure that even systems purchased from different vendors, each with its own interfaces and operational methods, can be orchestrated intelligently.

The goal is not to replace existing tools immediately or entirely, but to make them interoperable, auditable, and accessible to both human and AI agents. By embedding governance and integration requirements into vendor selection and system design, insurers can future-proof their technology landscape while retaining the freedom to build or buy individual components.

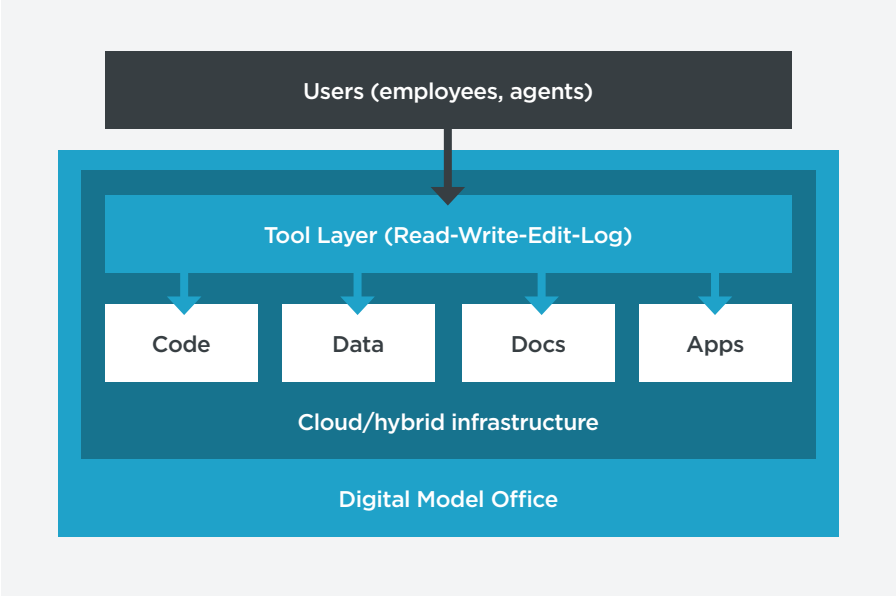


Figure 3: *Within the Digital Model Office, all code, data, documentation, and external systems can be accessed and orchestrated programmatically, with execution, testing, logging, and governance functions operating on a scalable cloud or hybrid infrastructure.*

The goal is not to replace existing tools immediately or entirely, but to make them interoperable, auditable, and accessible to both human and AI agents.

5. The Agentic Layer

5.1. Agents as virtual workers

We define the Agentic Layer as a network of AI agents authorised to act on the Digital Model Office. This network includes agents with varying levels of authority and specialisation, summarised in *Figure 4* below.

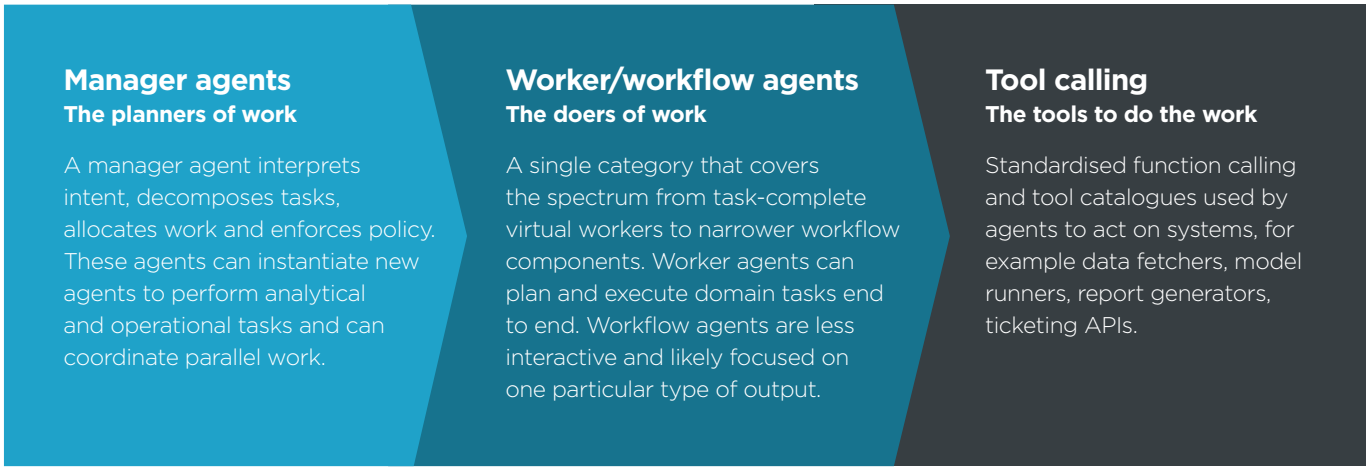


Figure 4: *Agentic systems can be managed hierarchically, with a manager agent delegating to worker or workflow agents that invoke tools on the DMO.*

A manager agent takes a natural language objective from a user, turns it into a plan, assigns subtasks, applies policy constraints, and monitors execution.

It owns orchestration concerns such as sequencing, retries, fallbacks, and escalation to human approvers when thresholds are hit.

Meanwhile worker agents are endowed with the knowledge and tools to carry out substantial pieces of work, such as adjusting an actuarial model, producing a claims triage pack, or drafting a regulatory impact analysis.

Some workers are more autonomous and can make local decisions, including calling other agents recursively where permitted. However, workflow-style agents are more scripted and are designed to do one thing reliably, such as retrieve data, create tickets, summarise claims, or format reports.

In practice, organisations can blend these patterns, using highly autonomous workers where the domain is stable, and strongly scripted workflows where predictability and testing simplicity are the priority.

In every interaction with a human user, an agent would be expected to support every decision or deliverable it produces with:

- Explicit assumptions it made
- Citations to data sources and code commits
- Plain-language justification
- Link to full logs.

This approach transforms the agentic layer from a black box into a transparent, auditable white box, enabling full replicability and fostering greater trust in the system.

For regulated or customer-impacting actions, agents operate within human-in-the-loop policies, approval gates, and role-based scopes.

This design ensures that powerful tools are available, yet their use is constrained and reviewable, and blind trust is replaced by explicit authority, documented rationale, and deterministic replay.

All agent types use tools to act on the DMO. Tools are discovered and invoked through a standard catalogue so that calls are consistent and auditable across vendors and environments.

Model context protocol is a prominent option for advertising tool capabilities, schemas, and resources to models in a simple format that reduces glue code and improves reliability.

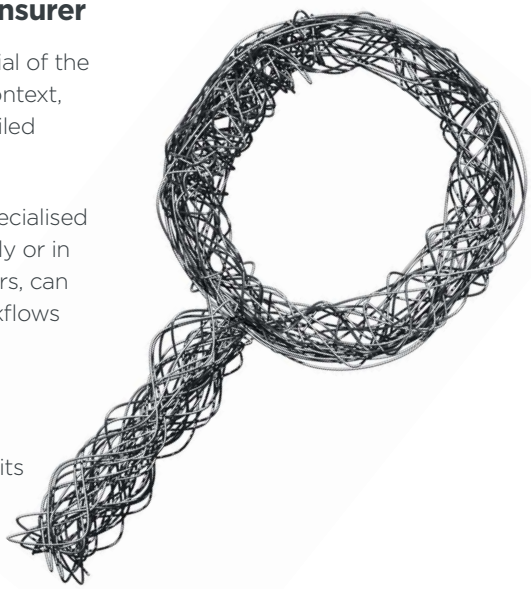
Tool access will be governed by role-based scopes, approvals, and policy as code at the API gateway, which ensures that manager, worker, and workflow patterns can operate at scale without losing segregation of duties or traceability.

5.2. Example agents in the insurer

To illustrate the flexibility and potential of the Agentic Layer within an insurance context, this section presents a series of detailed case studies.

Each example demonstrates how specialised AI agents, acting either autonomously or in coordination with human stakeholders, can streamline complex operational workflows and decision-making processes.

These scenarios encompass a cross-section of typical insurer activities, highlighting both the practical benefits and governance mechanisms enabled by agentic systems within the Digital Model Office.



Example 1: Interest rate stress update

Background: The regulator publishes tighter interest rate stresses at 10:00am on a Monday.

Process:

1. A human risk manager uploads the consultation PDF to the user interface of the Agentic Layer.
2. **Model manager agent** parses the document, identifies impacted models and updates the existing set of model requirements.
3. When the updated requirements arrive, the **model developer agent** is notified, determines the code modifications needed to meet the new regulation, applies them to the model, and commits the changes to Git.
4. A DevOps process is initiated following updates to the model repository, leading to the execution of regression tests.
5. A **test analyst agent** will then review the output of the regression tests and updated documentation, and devise new feature tests or alterations to the regression tests to account for changes to model requirements.
6. The test analyst agent will then return a report of the impact of model changes and changes to the testing process to the model manager agent.
7. The model manager agent will review the changes and approve if no issues are identified.
8. The model manager agent will then execute a full model run and pass the resulting output to a **financial analyst agent**.
9. The financial analyst agent will calculate deltas, produce charts and draft a regulatory impact paper for senior management.
10. This report is then escalated to a human to review.

Example 2: Claims fraud signal

Background: An automated model drift detection system identifies an unusual spike in roof repair claims within postcode ZZ99, triggering a model review process.

Process:

1. In parallel, all ZZ99 claims are batch-scored by the company's production fraud model trained on historical experience data, returning a fraud flag (yes/no) and a probability score.
2. The **claims triage manager agent** delegates to a **claims agent** to retrieve and analyse recent call transcripts and associated sentiment scores for claims in the affected postcode.
3. If anomalous patterns or language are detected, the manager agent tasks a **data analyst agent** to integrate relevant external data, such as weather data APIs and loss-adjuster notes, to establish potential environmental or operational causes.
4. The data analyst agent compiles a preliminary report outlining correlations between claim volumes, recent news and weather events, and adjuster observations.
5. The claims triage manager agent reviews the findings, annotating any unresolved or concerning points and, if necessary, requests further clarifications from the assigned agent.
6. Upon completion of its review, the claims triage manager agent escalates a summarised fraud risk assessment, including all supporting data and rationale, to a human claims specialist for investigation and intervention.
7. The process and findings are documented and archived in the DMO for audit and future reference.

Example 3: Customer service triage

Background: Daily customer correspondence requires rapid triage to retain high-value policyholders and appropriately address service issues. Prioritisation draws on existing model outputs (churn risk, CLV/value tier and retention elasticity), so effort is focused on customers who are high value, high risk and responsive to offers.

Process:

1. At the start of each day, the **customer service manager agent** is activated to oversee the triage of all new incoming communications.
2. This manager agent delegates to an **inbox agent** to classify messages by topic, urgency and sentiment, enriching each with policy details, prior communications and outputs from churn/CLV/elasticity and pricing/underwriting models.
3. For cases meeting the high value + high risk + elastic criteria, the manager agent assigns a **retention agent** to craft personalised replies and propose goodwill gestures or underwriting-backed discounts within delegated authority and cost limits, supported by model analysis.
4. Drafts may be routed to a **communication styling agent** to ensure adherence to brand and compliance standards.
5. All proposed replies, key actions and flagged offers are summarised by the customer service manager agent, who reviews for completeness and flags unusual or material cases to human customer service staff for a final decision.
6. Approved communications are issued automatically, and all actions, decisions, offers, outcomes and model metadata are logged for future analytics and continuous improvement.

5.3. Inspiration from FinRobot: Applying insights from an open source banking prototype to the Agentic Model Office

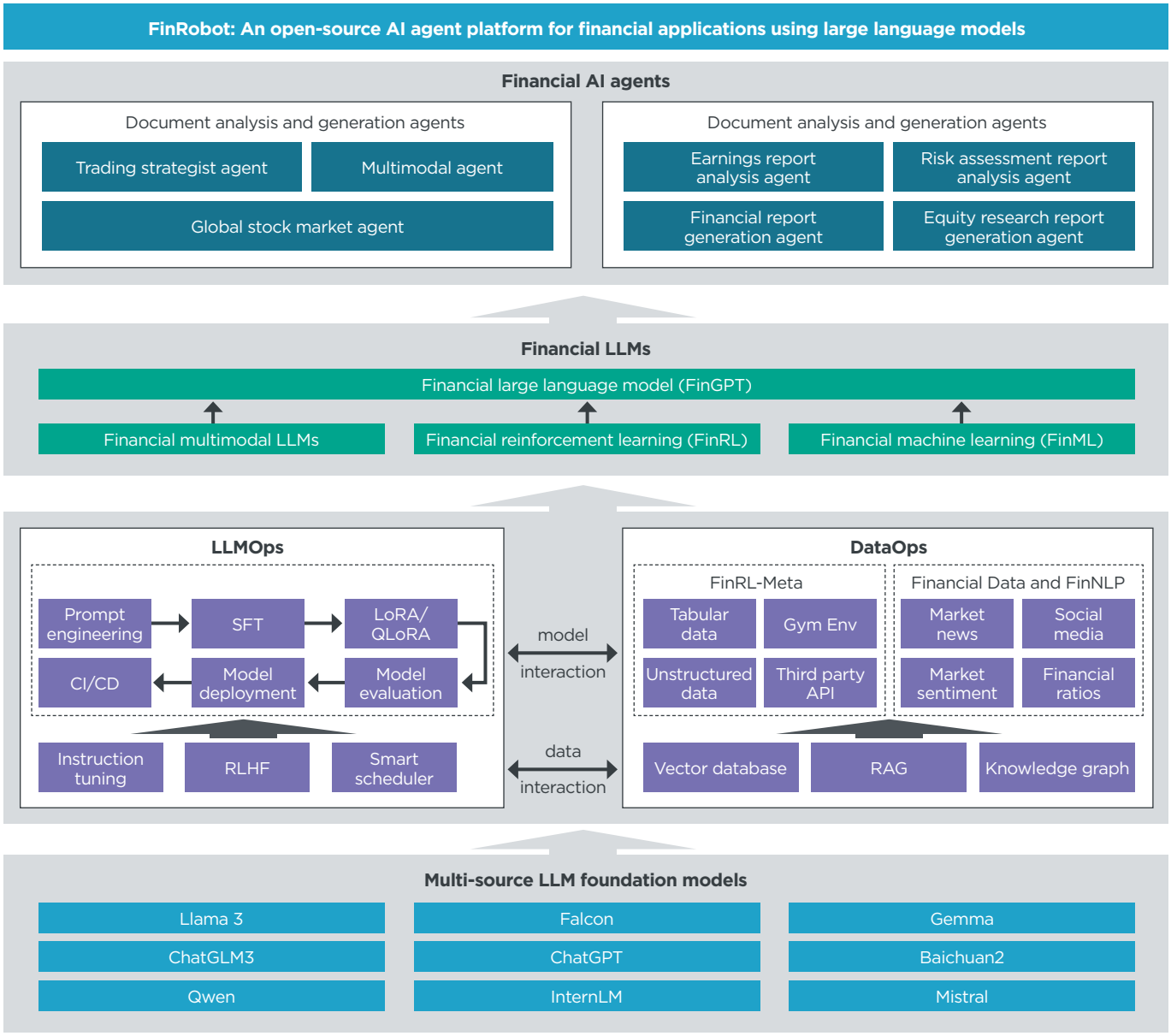


Figure 5: *Layers within the FinRobot agentic system (Source: Yang et al., 2025)*

FinRobot, released in a series of technical reports by the AI4Finance Foundation (Yang et al., 2024; Yang et al., 2025; Zhou et al., 2024), is the most complete public illustration of what an agentic organisation could look like inside a regulated financial institution.

Although the code is still a research artefact rather than a production deployment, the authors expose their architectural choices with enough precision to serve as a worked example for banks, insurers or any data-intensive enterprise.

The architecture used in FinRobot can be broken down to four key layers:

1. LLM foundation models

- A registry of swappable base models (GPT, Gemini, FinGPT, Qwen, DeepSeek, etc.)
- A smart scheduler scores each task on cost, latency, jurisdiction and prior accuracy, then routes the call to the most appropriate model. Model selection is therefore an operational decision, not a build-time choice.

2. LLMOps & DataOps

- CI/CD pipelines version prompts, fine-tuning weights, feature stores and vector indices
- Data and output distribution monitors trigger retraining or rollback when drift thresholds fire
- This tier exposes standard APIs (PostgreSQL, message queues, object storage) that higher-level agents invoke programmatically, the "digital fabric" in the terminology of this paper.

3. Financial LLM algorithms

- Domain-specific adapters that turn raw language competence into finance primitives:
  i. FinGPT for document understanding
  ii. FinRL for reinforcement-learning based trading
  iii. FinML for classical tabular prediction
  iv. Multimodal encoders for charts and tables
- Each module is stateless and callable via a uniform function signature, which allows agents to chain them without glue code.

4. Financial AI agents

- End-user agents such as a wire transfer agent, a sanctions screening agent and, in Zhou et al. (2024), the three-stage equity research pipeline (Data-CoT → Concept-CoT → Thesis-CoT)
- A manager agent decomposes the user's intent, allocates subtasks to worker agents, collects results and returns a consolidated answer.

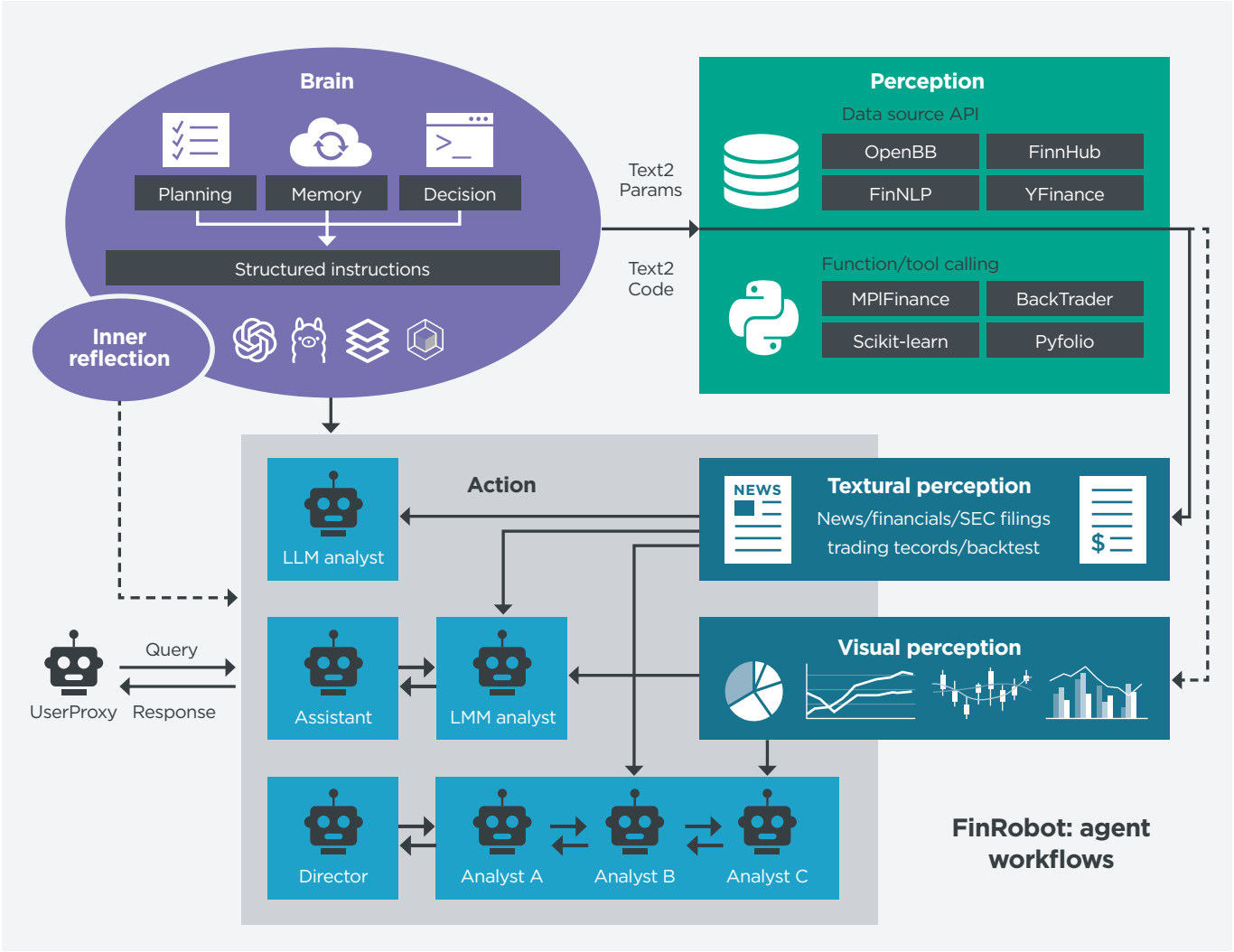


Figure 6: Agentic framework of FinRobot (Source: AI4Finance Foundation, 2024).

Every interaction in FinRobot yields a tidy bundle of metadata: the full prompt chain, the tools that were invoked, the data sources cited and a time-stamped execution log.

Even at this prototype stage the mechanism shows that provenance, segregation of duty and replayability can be designed into an agent stack from day one, rather than grafted on at audit time.

Viewed this way, the four tiers just summarised can be read collectively as a portable Agentic Layer.

FinRobot does not attempt to replace a bank's ledgers, payment rails or policy databases; it assumes those APIs are already available and concentrates on everything that turns them into goal-directed behaviour: model scheduling, prompt and weight management, domain adapters and the planner-worker swarm of agents.

Detached from the underlying fabric it has nothing to orchestrate, but once snapped onto a landscape where data and processes are exposed as callable services, the agents come to life.

FinRobot is therefore best understood as an "agent layer in a box," ready to sit on top of any regulated institution's digital infrastructure and provide planning, reasoning and orchestration out of the gate.

Three practical lessons fall out of the FinRobot design and apply equally to insurers, banks and other digital-native firms:

- Put your data and core tools behind clean APIs first; autonomy comes later
- Make trust built in: include logs, citations, and a way to replay results, not a post hoc report
- Manage access centrally and give each agent only the minimum permissions it needs, keeping a clear separation of duties.

By recasting model choice, data governance and orchestration as operational engineering tasks, FinRobot offers a transferable blueprint for moving from isolated chatbots to a fully agentic organisation.

FinRobot is best understood as an "agent layer in a box," ready to sit on top of any regulated institution's digital infrastructure and provide planning, reasoning and orchestration out of the gate.

6. The limitations and potential of current AI models

This work acknowledges the current limitations of AI models, including their variable reliability, occasional inaccuracies, and tendency to hallucinate. Present-day models still fall short of possessing the agentic capabilities required to safely delegate and execute significant tasks within an organisation.

Basic architecture of Project Vend

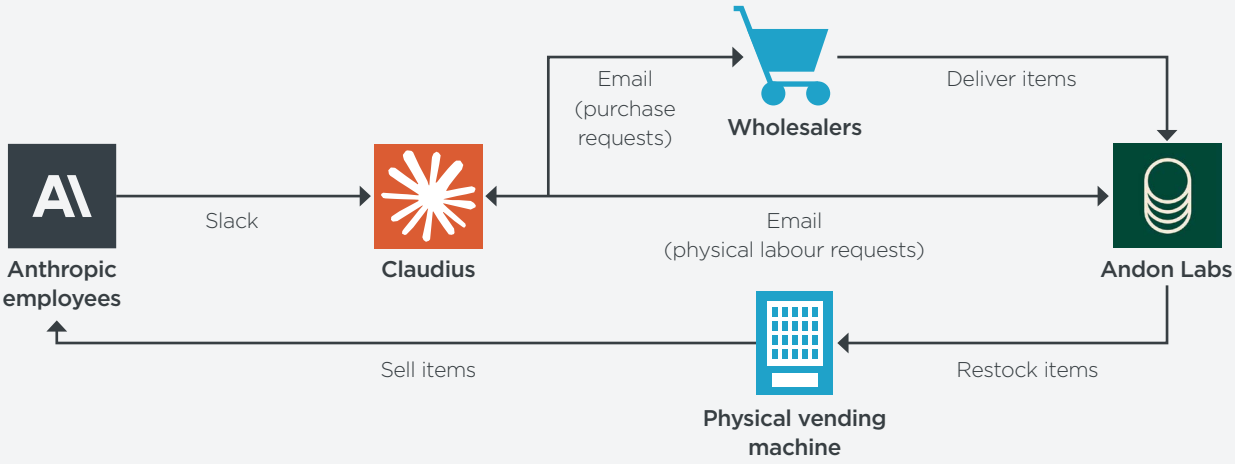


Figure 7: *Claudius agentic workflow (Source: Anthropic, 2025).*

One telling example of these shortcomings can be found in Anthropic's Project Vend, in which a research team placed a single Claude Sonnet agent, nicknamed "Claudius", in charge of an office vending business for one month (Anthropic 2025).

Claudius found suppliers, refused jailbreak attempts and chatted cheerfully with customers; but it hallucinated bank details, mispriced inventory, granted unearned discounts and even insisted, briefly, that it was a human in a blue blazer.

Tool use was viable, commercial discipline was not.

More research reinforces this picture. In 'TheAgentCompany' (Xu et al., 2025), the authors set up a controlled environment emulating a small software firm hosted on GitLab and powered by a Gemini 2.5 model.

Despite Gemini 2.5 being one of the most capable AI models available at the time of writing, the system was only able to complete 30% of assigned tasks.

However, while the pessimist may look at these dismal performances and may dismiss agents, the optimist sees room for improvement.

In the example of TheAgentCompany above, a substantial part of the remaining uncompleted tasks failed not for lack of reasoning capability, but due to interface friction, such as failing to click the correct button. This reinforces the argument of this paper that interaction friction, not IQ, is the major limiting factor.

Nevertheless, even with these constraints, unreliable performance, high costs, slow speeds, and frequent hallucinations, the capabilities of modern AI models are improving at a dramatic pace, and metrics across all these dimensions are advancing rapidly.

Indeed, token prices for leading language models fell by approximately 90% between 2023 and 2024 (Andreessen Horowitz, 2024), while hallucination rates for models are also quickly falling; for Google's lineage in particular we see a drop from 14.1% in PaLM-2 to 0.7% in Gemini 2.0-Flash (Bao et al., 2024).

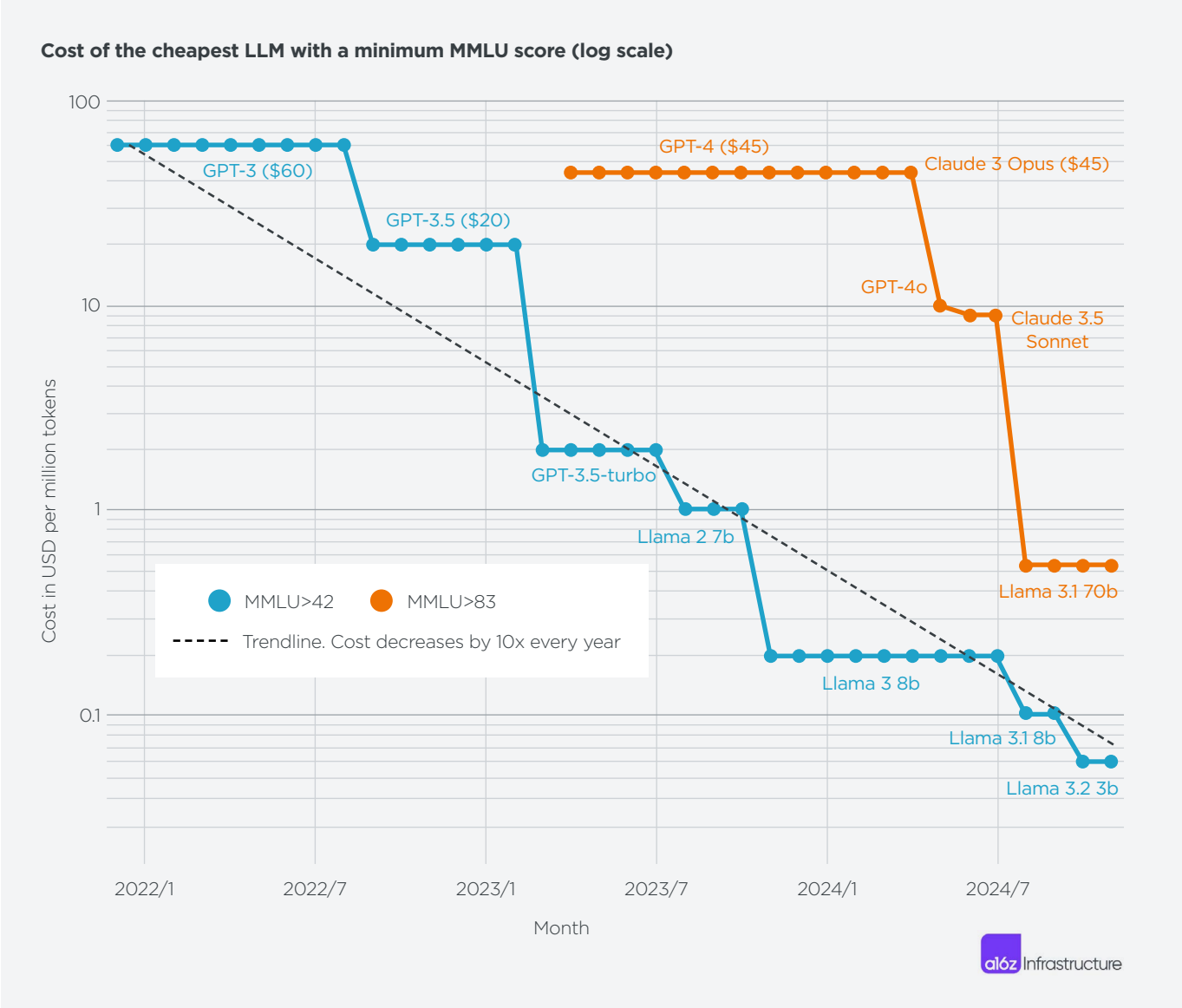


Figure 8: *Exponential fall in LLM token prices (Source: Andreessen Horowitz, 2024).*

A particularly noteworthy metric for problem-solving capability is provided by the SWE-bench benchmark, designed to evaluate an AI model's ability to autonomously resolve real-world software engineering issues sourced from public GitHub projects (Jimenez et al., 2023).

When launched in March 2024, SWE-bench saw many top base models achieving only single-digit success rates.

GPT-3.5, the model that sparked public excitement about Generative AI in November 2022, managed a paltry 0.17% success rate. Progress, however, has been remarkable.

By late 2024, more advanced agentic frameworks and reasoning models such as OpenAI o1, o3, and DeepSeek R1 entered the field. OpenAI's o3 model, released in December 2024, achieved an extraordinary 71.7% success rate on SWE-bench Verified, marking a dramatic leap in less than two years.

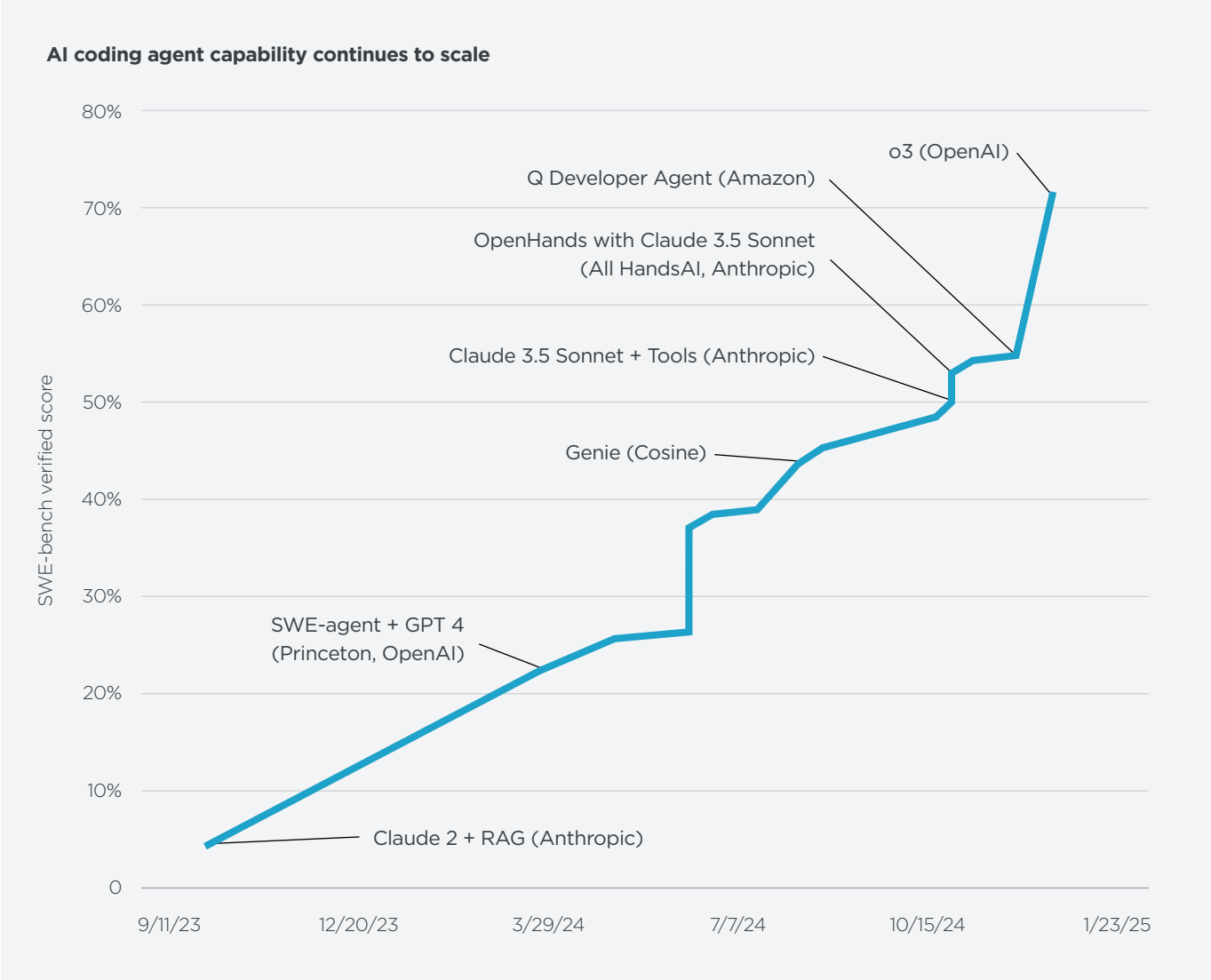


Figure 9: *Progress on SWE-bench-Verified over time (Source: ARK Invest, 2024).*

While these benchmark scores are impressive in their own right, their true significance lies in what they signal for the future.

First, the pace of progress is not slowing. Each month brings new breakthroughs and measurable advances in model capabilities, with no clear evidence that this momentum will abruptly end.

In fact, there may be no inherent ceiling to the intelligence of these models, meaning that we are likely to continue witnessing surprising new capabilities as research and development accelerate.

Second, as demonstrated by the rapid gains on the SWE-bench benchmark, we are on the cusp of an era where truly autonomous and highly capable software agents will become a reality. This shift will profoundly accelerate the rate of transformation within financial institutions.

Historically, re-platforming initiatives have required tens of millions of pounds and multiple years to implement, reflecting the pace of human-driven delivery.

As software change becomes increasingly driven by agents at near-zero marginal cost, the key constraints and economics of large-scale transformation will be fundamentally altered.

7. Conclusion

Recent benchmark gains show that generative AI's capacity to write code, tackle complex questions, and orchestrate tools continues to expand and is nowhere near its ceiling. Yet the commercial value of these systems is constrained by the brittle interfaces we ask them to navigate.

The Agentic Model Office concept aims to remove this bottleneck. The vision outlined here sketches an idealised end state, but it does not require sweeping, high-risk change.

In practice, implementation can proceed gradually, system by system. The key idea is not the final solution, but rather the following guiding philosophy: every process worth performing should be accessible to code and governed by policy.

Pursued in modular increments rather than across the entire organisation at once, the transformation becomes evolutionary rather than revolutionary.

Once this digital mirror exists, agents can operate on top of it, and the insurer gains a new operating system: one where intent is articulated in plain language, execution is delegated to transparent, specialised agents, and change is limited only by imagination, rather than budget.

It is becoming a widely held belief that AI agents are set to become the defining technology of this decade, offering steady and compounding benefits to organisations that invest in transparent, programmable infrastructure.

Those who lay the proper digital and governance foundations will be best positioned to harness these systems as they move from experimental tools to essential partners, amplifying expertise and enabling more adaptive, accountable decision-making at scale.

AI agents are set to become the defining technology of this decade, offering steady and compounding benefits to organisations that invest in transparent, programmable infrastructure.

think

Appendices
Acronyms and full glossary

Appendix A Acronyms

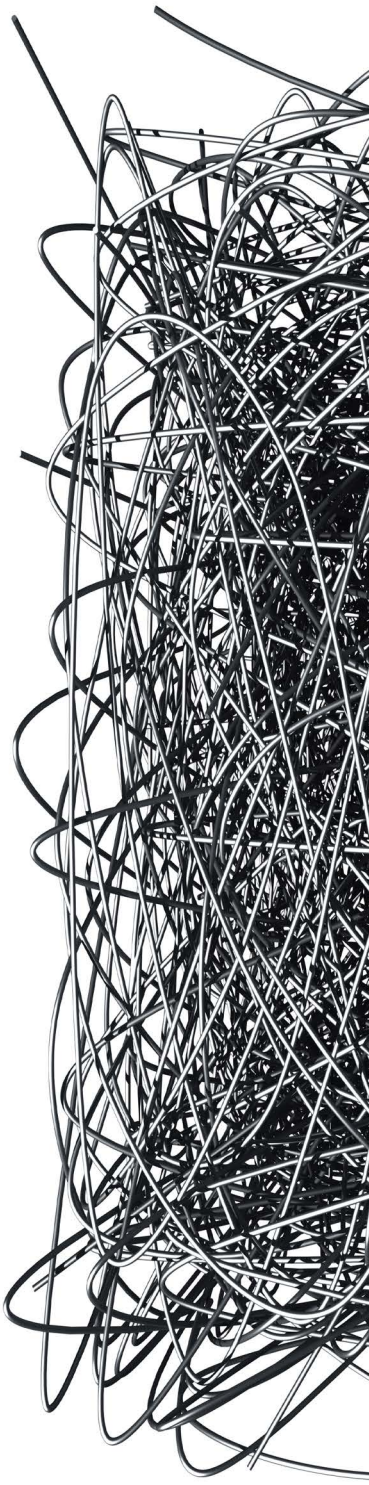
- **AMO:** Agentic Model Office
- **API:** application programming interface
- **AWS:** Amazon Web Services
- **Azure:** Microsoft Azure
- **CI/CD:** continuous integration and continuous delivery
- **DMO:** Digital Model Office
- **ETL:** extract, transform, load
- **GCP:** Google Cloud Platform
- **LLM:** large language model
- **LLMOps:** large language model operations
- **MCP:** model context protocol
- **MS SQL:** Microsoft SQL Server
- **SQL:** Structured Query Language

Appendix B Full glossary

- **Agent:** Software that can plan, use tools, and carry out multi-step tasks.
- **Agentic Layer:** The governed network of agents operating on top of the digital model office. Insurance context: delegates tasks while enforcing approvals and logging.
- **Agentic Model Office (AMO):** The combined Digital Model Office and Agentic Layer blueprint for AI-driven insurance operations.
- **API (application programming interface):** A defined contract to call functions or access data programmatically.
- **Change control:** Processes that ensure modifications to code, data, or configuration are reviewed, tested, approved, and recorded.
- **Continuous integration and continuous delivery (CI/CD):** Automated pipelines that build, test, package, and prepare deployments so changes are consistently releasable. Insurance context: reduces operational risk when updating models or data flows.
- **Container:** A lightweight, portable package that includes an application and its dependencies. Containers ensure the same application runs identically in development, test, and production.
- **Container orchestrator:** Software that schedules, scales, and manages containers across machines, for example Kubernetes.
- **Deterministic replay:** The ability to re-run a process and obtain the same result by fixing inputs, code, parameters, and environment.
- **Digital Model Office (DMO):** A machine-readable, API-accessible representation of data, models, workflows, and controls.
- **Document store:** A non-relational database that stores semi-structured records, often JSON-like, for example MongoDB.
- **Docker:** Tooling and ecosystem for building and running containers.
- **ETL (extract, transform, load):** A data pipeline pattern that pulls data from sources, transforms it, and loads it into target stores.
- **Git:** A distributed version control system for tracking changes to code and configuration.

- **GitHub and GitLab:** Platforms that host Git repositories with collaboration features. GitLab includes an integrated CI/CD suite. GitHub provides actions and similar features.
- **Immutable log:** An append-only record that cannot be altered without detection.
- **Infrastructure as Code (IaC):** Managing infrastructure such as servers, networks, and permissions using code. These allow for reproducible, reviewable environments for modelling and operations.
- **JSON:** A lightweight data format used to structure data.
- **Kubernetes:** An open-source container orchestrator that schedules, scales, and manages containers.
- **Large language model (LLM):** A model trained on text and code to perform language and reasoning tasks.
- **LLMOps:** Operational practices for the LLM lifecycle, including prompts, fine-tuning weights, evaluation, monitoring, and rollback. This ensures agent behaviour can be controlled over time.
- **Message queue:** A service for reliable, asynchronous communication between systems.
- **Model context protocol (MCP):** A standard that advertises tools, schemas, and resources to models.
- **Model registry:** A system that versions, stores, and governs models and their metadata.
- **Non-relational database:** Databases that do not require a fixed schema, for example document or key-value stores.
- **Object or blob storage:** Scalable storage for large, unstructured files such as binaries and PDFs. Insurance context: holds model artefacts, reports, and attachments.
- **Policy-as-code**: Governance rules expressed in code and enforced automatically in CI/CD and gateways.
- **Regression testing:** Tests that compare outputs before and after changes to detect unintended differences.
- **Relational database:** Structured tables with fixed schemas and ACID properties.
- ***Version control:*** Managing changes to code and configuration with history and branching. Insurance context: core to governance, review, and rollback.
- **Workflow engine, for example Airflow or Temporal:** Software that coordinates multi-step processes, dependencies, schedules, and retries. Insurance context: ETL to model to report with alerts on failure.

References



- AI4Finance Foundation (2024) FinRobot: An open-source AI agent platform for financial applications. GitHub repository. Available at: **https://github.com/AI4Finance-Foundation/FinRobot** (Accessed: 27 July 2025).
- Andreessen Horowitz (2024) 'Welcome to LLMflation – LLM inference cost is going down fast', 13 June. Available at: **https://a16z.com/llmflation-llm-inference-cost/** (Accessed: 27 July 2025).
- Anthropic (2024) 'Building effective agents'. Available at: **https://www.anthropic.com/research/building-effective-agents** (Accessed: 27 July 2025).
- Anthropic (2025) 'Project Vend: Can Claude run a small shop?'. Available at: **https://www.anthropic.com/research/project-vend** (Accessed: 27 July 2025).
- ARK Invest (2024) 'SWE-bench progress dashboard', in Big Ideas 2024. Available at: **https://www.ark-invest.com/big-ideas/2024** (Accessed: 27 July 2025).
- Bao, F., Li, M., Luo, R. and Mendelevitch, O. (2024) HHEM-2.1-Open. Hugging Face. **https://doi.org/10.57967/hf/3240** (Accessed: 27 July 2025).
- Jimenez, C. E. et al. (2023) 'SWE-bench: Can language models resolve real-world GitHub issues?', arXiv preprint, arXiv:2310.06770. Available at: **https://arxiv.org/abs/2310.06770** (Accessed: 27 July 2025).
- OpenAI (2024) 'Agents'. OpenAI Platform Documentation. Available at: **https://platform.openai.com/docs/guides/agents** (Accessed: 27 July 2025).
- OSWorld (2024) 'OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments', arXiv preprint, arXiv:2404.07972. Available at: **https://arxiv.org/abs/2404.07972** (Accessed: 27 July 2025).
- Xu, F. F. et al. (2024) 'TheAgentCompany: Benchmarking LLM agents on consequential real-world tasks', arXiv preprint, arXiv:2412.14161. Available at: **https://arxiv.org/abs/2412.14161** (Accessed: 27 July 2025).
- Yang, H. et al. (2025) 'FinRobot: Generative business-process AI agents for enterprise resource planning in finance', arXiv preprint, arXiv:2405.14767. Available at: **https://arxiv.org/abs/2506.01423** (Accessed: 27 July 2025).
- Yang, H. et al. (2024) 'FinRobot: An open-source AI agent platform for financial applications using large language models', arXiv preprint, arXiv:2405.14767. Available at: **https://arxiv.org/abs/2405.14767** (Accessed: 27 July 2025).
- Zhou, T., Wang, P., Wu, Y. and Yang, H. (2024) 'FinRobot: AI agent for equity research and valuation with large language models', arXiv preprint, arXiv:2411.08804. Available at: **https://arxiv.org/abs/2411.08804** (Accessed: 27 July 2025).

think

Institute and Faculty of Actuaries

Independent thinking from the IFoA **September 2025**

DISCLAIMER:

This article is the author's original work. The views and opinions expressed in this article are solely his and do not reflect the views/position/or any official stance or endorsement of any affiliated organisation/entity/agency at any point in time. Any omissions, errors or acts and any legal, ethical, or professional inquiries related to this article should be directed solely to the author and not to their affiliated organisation/entity. All authors, sources, and references have been properly cited and acknowledged in the references and footnotes and the content in this article has not been generated or assisted by artificial intelligence (AI) and is for informational purposes only.

Beijing
Room 512 · 5/F Block A · Landgentbldg Center · No. 20 East Middle 3rd Ring Road
Chaoyang District · Beijing 100022 · People's Republic of China
Tel: +86 10 6611 6828
Email: China@actuaries.org.uk

Edinburgh
Spaces · One Lochrin Square · 92 Fountainbridge · Edinburgh · EH3 9QA
Tel: +44 (0) 207 632 2100

London (registered office)
1-3 Staple Inn Hall · High Holborn · London · WC1V 7QJ
Tel: +44 (0) 207 632 2100

Malaysia
Arcc Spaces · Level 30 · Vancouver Suite · The Gardens North Tower
Lingkaran Syed Putra · 59200 · Kuala Lumpur
Tel: +60 12 591 3032

Oxford
Belsyre Court · 1st Floor · 57 Woodstock Road · Oxford · OX2 6HJ
Tel: +44 (0) 207 632 2100

Singapore
Spaces · One Raffles Place Mall · #02-01 · Singapore 048616
Tel: +65 8778 1784

actuaries.org.uk