AGENTIC ORCHESTRATION IN THE ENTERPRISE: WHAT IT IS AND HOW IT RELATES TO WORKLOAD AUTOMATION

AI agents are entering production environments — taking actions, making decisions, coordinating with humans and automated systems in ways that didn't exist two years ago. The question for enterprises is how the autonomous work these agents do fits into the execution discipline that already runs the business. Agentic orchestration is the answer many enterprises are moving toward: a coordination layer that governs how autonomous agents operate within business processes, alongside the automated systems and human operators those processes already involve.

What is agentic orchestration?

Agentic orchestration is the coordination of AI agents, automated systems, and human reviewers within governed business processes. Unlike traditional workload automation, which executes predefined sequences of tasks, agentic orchestration coordinates work that combines rules-based execution with the judgment of AI agents, with consistent policy enforcement and visibility across both.

The orchestration layer doesn't dictate every step. It provides the goals, defines what agents are allowed and not allowed to do, and gives them the tools and information they need to do their work. But it's the agents themselves that figure out how to reach the goal within those limits. The result is a system where individual agents have room to make judgments — but the rules they operate under, the visibility into what they're doing, and the accountability for what results stay under enterprise control. This combination is what makes agentic systems durable enough for production environments, where business workflows have to run reliably, prove auditable, and meet enterprise

standards.

Most enterprises don't introduce agentic orchestration as a replacement for what they already run. They add it to the orchestration layer they've been using for workload automation across applications, data pipelines, and cloud services — extending that layer to govern AI agents alongside the work it already coordinates.

How does agentic orchestration work?

Agentic orchestration begins with a defined goal — process a support ticket, execute a financial close, investigate a security alert — and a set of agents, automated systems, and human reviewers available to do the work. Different agents typically take different roles: A triage agent might categorize an incoming request; a research agent might pull information from internal systems or external sources; an executor agent might call an API to take a specific action. Each agent uses a defined set of tools that the orchestration layer makes available to it. The orchestration layer routes work to the right agent, gives the agent the context it needs, and tracks the state of the overall effort.

When an agent receives a piece of the work — investigate this anomaly, resolve this ticket — it typically breaks down the goal into a sequence of steps and works through them, adjusting the plan as it encounters new information or unexpected conditions along the way. The agent retains what's happened across those steps — what it's tried, what's worked, what data it's gathered — so that each action builds on what came before rather than starting from scratch. In more complex processes, agents also collaborate directly: one agent may delegate a subtask to another with the right specialization, or multiple agents may work different parts of a problem in parallel. The orchestration layer manages these interactions, tracking which agent is responsible for what and ensuring the overall process stays coherent as work moves among them.

When one stage of work is complete, the next stage usually involves a handoff — to another agent, to an automated system, or to a human for review or approval. The orchestration layer manages these transitions, carrying the relevant context forward, enforcing the policies that apply to the work in question, and recording every decision, tool call, and outcome at each step.

Some systems use this history to refine agent behavior over time. But whether or not learning is in scope, the audit trail is the foundation for accountability — it's what makes governed autonomy possible at the scale production requires.

How does agentic orchestration relate to AI orchestration and AI agents?

Agentic orchestration is sometimes confused with related concepts. Here are the two distinctions that matter most.

How does agentic orchestration differ from AI orchestration?

AI orchestration is sometimes used as a catch-all term, but in practice it usually refers to something specific: coordinating the steps required to run AI models. A typical example is a fraud-detection system: A transaction comes in, the system enriches it with relevant history, and then an AI model scores the transaction for risk and a second model checks the customer's recent behavior. The two scores get combined into a single assessment, and the result is sent to a downstream system. AI

orchestration is what choreographs that sequence — what runs when, what data flows where, what triggers each model. The platforms that do this are built around the lifecycle of the AI models themselves: when they're invoked, how their inputs and outputs connect to surrounding systems, and how their performance is tracked.

Agentic orchestration coordinates something different. The actors aren't models running predefined steps, they're autonomous agents pursuing goals — making judgments, choosing actions, working alongside automated systems and human reviewers inside a business process. Where AI orchestration manages a sequence of model calls, agentic orchestration coordinates and governs the actors themselves — agents whose paths aren't fully defined in advance, because some of what they do gets decided as they go.

The two often coexist. The fraud-detection pipeline might be coordinated by AI orchestration, while the agent that detects a fraud pattern and initiates an investigation — pulling case history, contacting the customer, escalating to a human reviewer — operates within agentic orchestration. Both layers may run inside the same enterprise control plane, but they're solving different problems: how an AI model should run as part of a larger flow of work, versus how autonomous agents should be governed as they do their work.

How does agentic orchestration relate to individual AI agents**?**

An AI agent is an autonomous actor — typically a large language model connected to a set of tools, given a defined goal to pursue, and able to take actions in the world to make progress toward that goal. Agents perceive their situation, decide what to do, and act through the tools they've been given. Each agent is typically specialized — built for a defined scope of work, with a defined set of tools, and a defined role within a larger process.

Agentic orchestration is the layer above the agents, managing the system the agents operate inside. The orchestration layer provides the goal to be achieved, allocates tasks among the available agents, determines which tools each agent can use, enforces the policies the work has to follow, manages handoffs between agents and other systems, and keeps track of what's happened so the work stays coherent as it moves from step to step. It also records actions and decisions for visibility and accountability.

A single agent can complete a task. Multiple agents can complete related tasks in sequence or in parallel. Agentic orchestration is what coordinates those tasks — and the agents performing them — into a coherent, governed business process. It's the layer that helps ensure the work as a whole completes reliably, transparently, and within the limits the enterprise has set.

How does agentic orchestration relate to traditional workload automation?

Agentic orchestration is best understood as an extension of the decades-old practice of workload automation — the next stage in the evolution of enterprise execution control.

The lineage from batch workload automation to agentic orchestration

That evolution has a recognizable shape. Enterprise execution began as batch workload automation

— scheduled jobs running on predictable cycles, with dependencies managed across multiple systems. It expanded into application and data pipeline orchestration, coordinating workflows across the mix of systems most enterprises now run, including on-premises servers and multiple cloud environments. It grew to handle event-driven systems and cloud-native services, where work was triggered by signals rather than schedules. More recently, it has come to cover AI-powered workflows — where machine-learning models and AI-driven automation participate in business processes alongside everything else. Agentic orchestration is the next step: extending the execution layer to also coordinate autonomous agents that reason, plan, and act within the same business processes.

What carries forward is not just terminology but specific practices. The capabilities that mature workload automation platforms have developed over decades — dependency management across heterogeneous systems, SLA enforcement, retry and recovery logic, audit traceability, exception handling, observability across long-running workflows — are exactly the capabilities agent-driven workflows require to run reliably in production. The disciplines that make a financial close reliable when it consists of scheduled batch jobs are the same disciplines that make it reliable when it consists of a mix of scheduled jobs and AI-driven exception handling. The underlying execution layer doesn't need to be rebuilt for agentic workflows; it needs to be extended to them.

What changes with agent-driven execution

What is genuinely new is the kind of work being coordinated. Traditional workload automation governs largely deterministic execution: predefined steps that follow defined rules and produce predictable outcomes. In general, the same job, run with the same inputs under the same conditions, should produce the same result. Agentic systems introduce non-deterministic execution: agents make judgments based on context, and similar situations can lead to different actions depending on the conditions the agent observes. Agents also introduce challenges that have no direct workload-automation precedent: the quality of the plans agents construct, the reliability of the reasoning behind their decisions, and the management of the memory and awareness that they carry across steps all require forms of oversight that traditional execution platforms were never designed to provide. Many production business processes now combine both. A financial close has dozens of deterministic steps (data extraction, scheduled reporting, file transfers) and a growing number of non-deterministic ones (anomaly investigation, AI-assisted exception handling, reprioritization when something upstream is late). Supply chain operations, customer service workflows, and IT remediation increasingly look the same way.

Mixed-execution processes are why agentic orchestration is best understood as an extension of the existing execution layer rather than as a separate AI-specific one. When agent-driven work is governed in one place and the deterministic work it coordinates with is governed somewhere else, the business process as a whole can end up with fragmented governance. Even if both systems enforced similar policies, the process crossing between them would have two separate audit trails to reconcile, two separate observability surfaces to integrate, and two separate state-management systems to keep in sync.

Coherent enterprise execution requires consistent governance across the whole process, for both operational and architectural reasons. A consistent governance layer reduces the cost of keeping connected systems in sync, helps policy changes propagate predictably, and produces a more coherent audit trail when the result has to be explained to an auditor, a regulator, a board, or an

internal investigator.

How is agentic orchestration governed at enterprise scale?

Governance is what makes agentic orchestration different from agent experimentation. An AI agent operating in isolation can be unreliable in ways that are tolerable for a prototype; an agent operating inside a business process has to meet the same standards of reliability, accountability, and auditability as anything else running in production. Governance in this context isn't a single capability. It's a set of practices that together make autonomous execution safe enough to scale.

Five capabilities form the practical core:

Dependency control: Agent-driven work, like every other kind of enterprise work, has dependencies — on data being available, on upstream processes completing, on downstream systems being ready to receive results. The orchestration layer manages these dependencies across both deterministic and agent-driven steps, ensuring that an agent doesn't act on stale data, that a downstream system isn't asked to process incomplete inputs, and that the sequence of work — wherever the rules permit flexibility — still completes in a coherent order.

Policy enforcement: Every agent operates within defined constraints: what data it can access, what actions it can take, what thresholds require human review, what kinds of decisions it can make autonomously and which it must escalate. These policies are defined when agents are authored and enforced at runtime, not left to be implemented inconsistently by individual agents or development teams. Policy enforcement is what allows the enterprise to extend autonomy to agents without ceding control over what they do with it.

Observability: Production-grade agentic orchestration requires visibility into what's happening across the full process — agent work and deterministic work alike. For agent work, that means tracking not just whether an agent completed its task but what actions it took, what tools it called, what AI model invocations it made, what data it accessed, and what outcomes resulted. For deterministic work, the requirements are the ones workload automation has long handled: job completion, dependency resolution, SLA performance, exception handling. Across mixed-execution processes, the orchestration layer has to bring these two kinds of observability together, because the signals each generates are different and the operations teams running the process need a single coherent view. The orchestration layer integrates those signals so operations teams can see the whole process at once, not just its individual parts.

Exception handling: Real business processes encounter conditions their designers didn't anticipate — data that's late, systems that are unavailable, situations that fall outside an agent's defined scope. Exception handling in agentic orchestration combines retry and recovery logic familiar from workload automation with judgment-driven response from agents themselves, governed by clear escalation paths to human reviewers when the situation requires it. The result is a process that can absorb variability without losing its coherence.

Auditability: Every action taken by every agent — and every decision the orchestration layer makes about how to coordinate that action — is recorded with the context required to reconstruct what happened and why. Auditability is what makes governed autonomy demonstrable to regulators, auditors, and internal stakeholders. It's also what makes systematic improvement possible: the audit trail is the data from which patterns of agent behavior, process performance, and recurring exceptions can be understood and addressed.

These five capabilities have to operate consistently across both deterministic and agent-driven execution — and across the points where the two interact. Coherent governance has to span the boundary between the two, not just hold on either side.

What are the core components of an agentic orchestration system?

A working agentic orchestration system typically combines several components, each with a defined role. The specifics vary by platform, but the underlying architecture is increasingly consistent across mature implementations.

Agents: The autonomous actors that perform individual units of work. Each agent has a defined scope — what kinds of tasks it handles, what tools it uses, what models it relies on, what decisions it can make on its own. Specialization is the norm; a system might include a triage agent, a research agent, an investigation agent, and an executor agent, each tuned for the work it's meant to do.

Planning and reasoning: The capability that allows an agent to break down a goal into steps, decide what to do next based on what it knows, and adjust its approach as conditions change. Planning and reasoning are what distinguish an agent from a script — they're what make it possible for the agent to pursue a goal rather than just execute a sequence.

Multi-agent coordination and delegation: The mechanisms by which work gets allocated and handed off among agents within a process. One agent may delegate a subtask to another agent with the right specialization, or multiple agents may work different parts of a problem in parallel. This is distinct from the control plane's broader routing role — it's the coordination among agents themselves, within the structure the control plane provides.

Control plane: The layer that routes work across the system, manages state (what's happened so far, where each piece of work currently sits, what context needs to be carried forward), enforces policies, and coordinates handoffs between agents, automated systems, and human reviewers. This is the orchestration layer proper. It's where the rules of engagement are enforced and where the audit trail is generated.

Tool registry: The central reference that defines what tools are available to which agents, with the appropriate access controls. Agents can access enterprise systems, APIs, data sources, and other resources through this registry, rather than through direct integrations that have to be configured for each agent. The registry is what makes governance manageable at scale: changes to access policies, new tools, or revoked permissions propagate consistently across the system rather than having to be applied agent by agent. The Model Context Protocol (MCP) — now backed by major AI vendors and governed by the Linux Foundation — enables agents to discover and connect to tools across systems, giving the registry a common interface rather than requiring custom integrations for each tool an agent needs to use.

Shared state and memory: The context preserved across the multiple steps that make up a coordinated process. An agent that takes the second action in a sequence needs to know what happened in the first; a downstream reviewer needs to see what the agent has already done. Shared state is what makes coordination across agents possible at all.

Human-in-the-loop mechanisms: Defined points at which humans participate in agent-driven processes — for review, for approval of consequential decisions, for handling exceptions the system can't resolve on its own. These mechanisms aren't a fallback for failure; they're a structural component of the system, designed in from the start.

Audit and observability layer: The layer that captures the actions, decisions, and outcomes of every agent and every coordination event, in a form that's query-able and retainable for compliance and operational analysis.

Together, these components form a system that can coordinate agent-driven work alongside everything else an enterprise runs, under coherent governance, with the visibility production requires.

What are common enterprise use cases?

Enterprise work increasingly combines deterministic, rules-based execution with adaptive, judgment-driven response. Below are three common examples — processes where governance has to span the boundary between scheduled work and AI-driven response.

Financial close and reconciliation: A financial close consists of dozens of scheduled, rules-based steps — data extractions, reconciliations, validation routines, reports. It also requires judgment when something doesn't reconcile, when an anomaly appears, when an exception has to be investigated and resolved before the close can complete. Agentic orchestration coordinates AI agents that handle the exception-investigation work alongside the scheduled jobs that handle the predictable work, with full auditability across both.

Supply chain and operational continuity: Supply chain operations depend on scheduled coordination across procurement, inventory, logistics, and fulfillment systems. They also depend on adaptive response when something goes wrong — a delayed shipment, an inventory shortage, a routing change that affects downstream commitments. Agentic orchestration handles the coordination across the systems agents need to work with and gives agents the latitude to assess disruption signals, propose responses, and coordinate adjustments across the affected systems, all within policy boundaries set by the operations team.

IT operations and incident response: Modern IT environments combine deterministic automation (scheduled maintenance, automated patching, defined remediation playbooks) with situations that require judgment (incident triage, anomaly investigation, novel failure modes). Agentic orchestration coordinates AI-driven triage and investigation with the deterministic remediation playbooks already in place, escalating to human operators when the system reaches the limits of what it can resolve on its own. The result is faster response without loss of operational control.

What challenges does agentic orchestration introduce?

The challenges enterprises encounter when adopting agentic orchestration tend to be as structural as they are technological. They're not only about whether the technology works; they're about whether the organization has the architecture and discipline to deploy it responsibly.

Multi-agent coordination: Coordinating multiple agents across a complex process is harder than coordinating a single agent. Keeping track of what's happened across multiple agents working in parallel gets more involved. Handoffs have to be defined clearly. The temptation to let agents communicate freely with each other has to be balanced against the need for the orchestration layer to maintain coherent visibility into what's happening.

Observability across mixed execution: Producing a single coherent view across deterministic and agent-driven work requires the orchestration layer to make sense of two very different kinds of

operational data: the predictable status updates that scheduled jobs generate, and the more varied signals that agent-driven work produces. Operations teams used to monitoring scheduled jobs have to extend their mental model to include those new signals — confidence levels, alternative paths the agent considered, decisions the agent escalated for human review.

Integration with existing systems: Agents typically need access to enterprise data and tools that were designed for human users or for system-to-system automation, not for autonomous agents. Integrating agents into existing environments — without bypassing the governance those environments enforce — requires careful work, particularly around identity, access control, and audit logging.

Operational discipline: The biggest challenge is often cultural. Operations teams used to deterministic systems sometimes treat agent-driven work as inherently less trustworthy, and either over-constrain it (eliminating the value of autonomy) or under-constrain it (creating governance gaps). Building the right practices for mixed-execution processes is mostly a matter of clear policy and experience — but it takes time, and it benefits from platforms that make those practices easier to enforce.

None of these challenges is unique to agentic orchestration as a category; they're variants of challenges that enterprise execution has always involved. What's new is the need to apply them to a kind of work that wasn't prevalent in production environments a few years ago.

Control-M, BMC's enterprise orchestration platform, the workload automation and workflow orchestration disciplines it has run for decades to coordinating agentic execution in enterprise environments. It holds agents to

In BMC's view, the future of orchestration is not about AI agents or any other new technology. It's about orchestrating the work — all the work, of agents, applications, data pipelines, and people — under coherent governance and through a single execution layer.

More on Control-M's agentic orchestration capabilities is available here.

Control-M is named a Leader in the 2025 Gartner® Magic Quadrant™ for Service Orchestration and Automation Platforms. Eighty percent of the Forbes Global 100 — across financial services, manufacturing, healthcare, retail, telecommunications, and the public sector — use BMC to coordinate the systems their businesses depend on.