willow

WHITE PAPER · Willow Basecamp

Deploying Autonomous Agents in the Enterprise

A practical framework for identifying, designing, and operationalizing autonomous workflows in modern organizations.

WHAT THIS PAPER COVERS

- A working definition of background agents and how they differ from chat assistants and RPA.
- A shared vocabulary: triggers, workflow logic, tools, actions, guardrails, and evaluations.
- A four-step deployment framework: Define, Design, Choose, Iterate.
- Department-level examples across sales, support, engineering, operations, HR, and finance.
- Governance, risk, and control patterns for running agents safely in production.

Published June 2026 | withwillow.ai

00 EXECUTIVE SUMMARY

Executive Summary

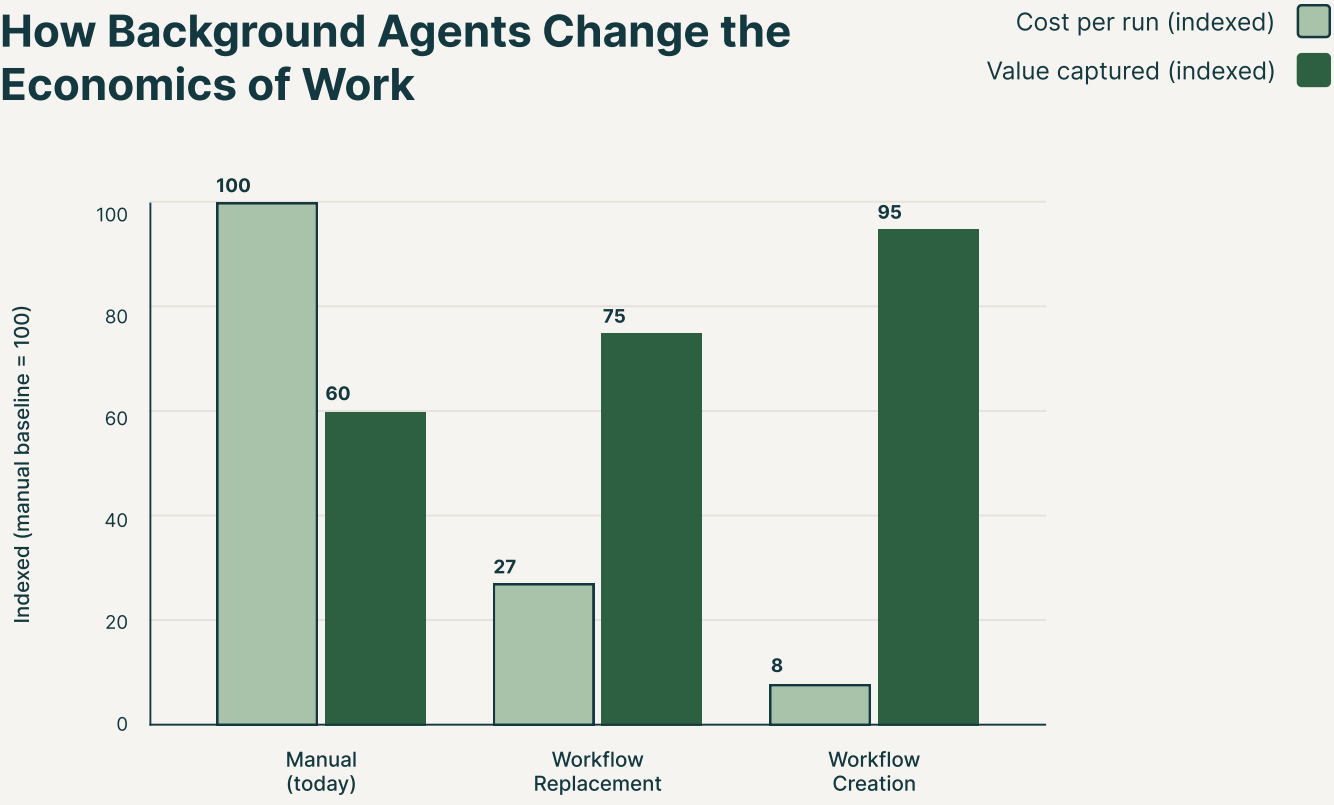
Background agents are a new category of AI system that execute recurring or event-driven work without requiring a user to initiate each task manually. Unlike chat assistants, which support a person during a task, background agents independently carry out multi-step workflows such as generating reports, preparing meeting briefs, monitoring systems, and routing follow-up actions.

For mid-market organizations, background agents offer a practical way to reduce manual effort, improve consistency across teams, and unlock tasks that were previously too costly to perform regularly. They are especially well suited to companies that already have structured workflows and integrated business systems, but can still move faster than large enterprises.

Successful deployment depends on four steps. Organizations must identify the right use cases, design the agent workflow with clear triggers and data access, select the right implementation approach for their governance and integration needs, and iterate based on real-world output. Treated as a disciplined engineering and operating practice rather than an experiment, background agents can meaningfully change the economics of back-office and cross-functional work.

Traditional assistants help people do the work. Background agents do the work for people, or do the work that otherwise would not get done at all.

How Background Agents Change the Economics of Work



willow

01 KEY TERMINOLOGY

Key Terminology

The field of AI agents has accumulated a vocabulary that is used inconsistently across vendors, researchers, and practitioners. The following definitions reflect the usage in this paper.

Term	Definition
Background Agent	An AI system that runs autonomously in response to a schedule or event — and produces an output or takes an action without requiring a human to initiate or supervise the run.
Chat Assistant	An AI system that responds to human prompts in a conversational interface. The human is always in the loop, initiating each exchange.
Agentic Workflow	A sequence of steps executed by an AI agent to complete a task. May include planning, tool calls, iteration, and conditional logic.
Trigger	The condition that causes an agent to start a run. Types: scheduled (time-based), event-driven (responding to a system event), threshold-based (metric crosses a value), or manual (human-initiated).
Workflow Logic	The set of instructions that governs what the agent does between receiving a trigger and producing an output. The most important thing to specify precisely.
Tool / Tool Call	A function the agent can invoke to interact with an external system — reading from a database, calling an API, sending a message, writing a file.
Action	Any real-world effect produced by agent tool calls: sending an email, updating a CRM record, posting a message, creating a document.
Context	The information available to the agent at the start of a run: the system prompt, trigger payload, and outputs of tool calls made during the run.

01 KEY TERMINOLOGY (CONT.)

Term	Definition
System Prompt	The instructions given to an agent at the start of every run. Defines the agent's role, behaviour, output format, and constraints.
Orchestrator	The code or system that manages execution of an agentic workflow: calling the model, routing tool calls, handling errors, and logging results.
Guardrail	A constraint that limits agent behaviour. May be implemented in the system prompt (instructional), in the orchestrator (programmatic), or externally (firewall).
Evaluation	The process of measuring how well an agent performs against a defined success criterion. Critical for knowing when to deploy and how to improve.
Human-in-the-Loop	A checkpoint in an agentic workflow where a human must review and approve an agent output or action before the workflow continues.
Observability	The ability to inspect what an agent is doing in real time or after the fact: inputs, outputs, tool calls, errors, and latency.
Drift	The gradual degradation of agent performance over time as the real-world environment changes but the agent's instructions and tools do not.
RPA	Robotic Process Automation. Automates rule-based, deterministic workflows by mimicking human interaction with software interfaces. Distinguished from AI agents by its deterministic, rule-based execution model.

Two framing terms are worth highlighting up front. Workflow Replacement refers to recurring tasks already performed manually that an agent can take over. Workflow Creation refers to tasks that were never performed because the manual cost was too high, but which become feasible once an agent can run them continuously. Both categories appear throughout this paper, and most organizations eventually deploy a mix of the two.

02 WHERE BACKGROUND AGENTS SIT IN THE AI LANDSCAPE

Where Background Agents Sit in the AI Landscape

AI deployments in the enterprise currently fall into three broad categories: chat assistants, agentic workflows, and background agents. Understanding where background agents fit — and how they differ from the other two — is a prerequisite for designing them well.

Chat assistants are synchronous: a human sends a message, the model responds. The human is always in the loop, directing the model at every turn. This model is well suited to tasks that require human judgment at every step — drafting, brainstorming, analysis, and Q&A.

Agentic workflows are multi-step processes in which the model takes a sequence of actions — calling tools, reading data, making decisions, and synthesising outputs — to complete a longer-horizon task. Background agents are a specific class of agentic workflow: asynchronous, scheduled or event-driven, and designed to run without human initiation.

Background agents overlap with RPA in the class of tasks they target, but differ in their ability to handle variability, ambiguity, and unstructured data. They are flexible and well suited to tasks that require judgment, synthesis, or interpretation. RPA is rigid and best suited to deterministic, rule-based execution.

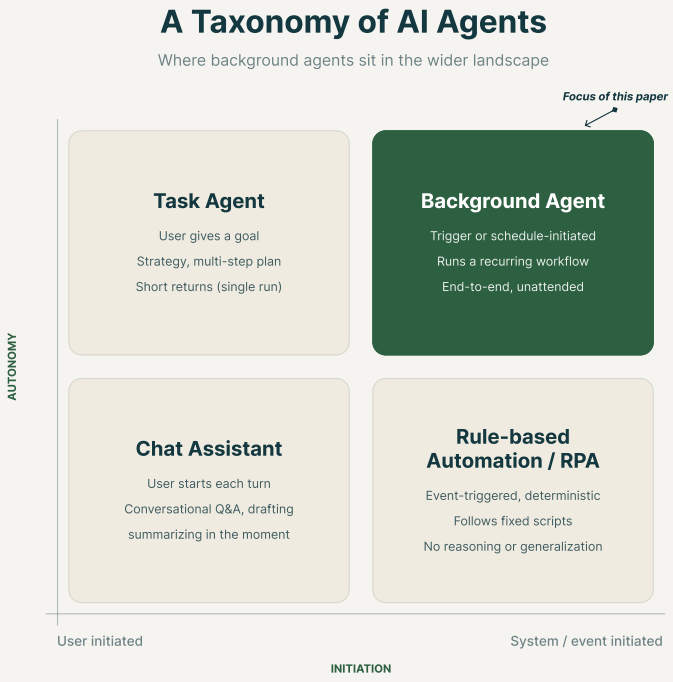


Figure 2. A taxonomy of AI agents. Chat assistants and rule-based automations address narrower problems; task agents and background agents are both autonomous, but differ in how they are initiated.

INTRODUCTION

The premise of this paper is straightforward: there is a large class of valuable enterprise workflows that are currently performed manually by skilled employees, that meet the criteria for automation by background agents, and that most organisations have not yet automated — not because the technology is unavailable, but because they lack a structured approach to identifying, designing, and deploying these agents.

The four-step framework in the rest of this paper is designed to change that. It moves organisations from the question of whether to deploy background agents to the question of how.

03 WHAT BACKGROUND AGENTS ARE

What Background Agents Are

A background agent is an AI system that runs autonomously in response to a trigger — a scheduled time, an event in a connected system, or a threshold being crossed — and produces an output or takes an action without requiring a human to initiate or supervise the run.

The defining characteristic of a background agent is the execution model: the agent runs when the trigger fires, not when a human asks it to. This seemingly simple difference has large practical implications for how the agent is designed, tested, deployed, and governed.

Chat Assistants vs. Background Agents

The table below captures the operational contrast. Most organizations will continue to use both; the design choices, governance needs, and change-management implications are different.

Dimension	Chat Assistant	Background Agent
Initiation	Human sends a message	Trigger fires (schedule, event, threshold)
Execution	Synchronous — the human waits for a response	Asynchronous — runs without human supervision
Human involvement	Every interaction is human-directed	Human reviews output; may not be involved in run
Scope	Single turn or multi-turn conversation	Multi-step workflow with tool calls and actions
Output destination	Conversation interface	Email, Slack, CRM, document store, or other system
Failure mode	Bad response the human sees immediately	Bad output that may not be reviewed until later
Latency expectation	Seconds	Minutes to hours (acceptable for background work)

03 CONCRETE EXAMPLES

Concrete Examples

- A daily 7:30 AM agent that reads Slack, email, and the ticketing system, then writes a prioritised briefing and sends it to the on-call engineer before the day begins.
- A meeting prep agent that runs whenever a calendar invite lands, pulls the attendees LinkedIn profiles, recent emails, and CRM notes, and drafts a one-page brief.
- A CI/CD agent that runs on every pull request, checks for style violations and test coverage, and posts a structured comment before a human reviewer opens the diff.
- A finance reconciliation agent that runs nightly, reconciles accounts against invoices, flags discrepancies, and drafts the CFR report.
- A legal monitoring agent that monitors contract expiry dates, pulls usage data and support-ticket history, and drafts a renewal recommendation summary with each vendor's contact details.
- A customer follow-up agent that runs every 14 days, identifies applications with no activity, and either sends a follow-up email or escalates to an account executive.
- An incident response agent that monitors applications, scores them against a rubric, sends an acknowledgement email to every applicant, and notifies the on-call engineer only if the automated fix fails.

04 WHY THIS MATTERS NOW

Why This Matters Now

Three structural shifts have converged to make background agents viable at enterprise scale.

First, frontier models now reason reliably enough to handle multi-step tasks without constant human correction. Earlier generations of AI required tight scripting and frequent fallbacks; current models can navigate ambiguity, recover from partial failures, and produce outputs that meet a professional bar most of the time.

Second, tool ecosystems have matured. Calendar APIs, CRM webhooks, code-review hooks, and document stores now expose clean interfaces that agents can call without custom integration work. The plumbing that would have taken months to build in 2022 is available off the shelf.

Third, organisations have accumulated handoffs. Teams that once required a human to touch every output are now comfortable with asynchronous, automated handoffs that enhance metrics, triage queues, and AI-drafted first passes.

Two Categories of Value

Category	What it is	Typical examples
Workflow Replacement	The agent takes over a task a human was doing	Daily briefing, pipeline hygiene, invoice reconciliation
Workflow Creation	The agent enables a workflow that was previously impractical because it required too much human time	Candidate screening, PR review, renewal watch

05 WHO THIS IS FOR

Who This Is For

This paper is written for three overlapping audiences. Technology leaders — CTOs, VPs of Engineering, and AI Platform leads — who are evaluating where background agents fit in their organisation's roadmap. Functional leaders — heads of Sales, Marketing, Finance, Legal, HR, and Operations — who want concrete use cases and a realistic sense of investment required. AI practitioners — engineers, architects, and product managers — who are tasked with designing and deploying these systems and need a framework that translates strategy into execution.

You do not need to be an AI researcher or have a background in machine learning. The framework assumes familiarity with enterprise software and business operations, not model internals.

A Four-Step Framework for Deploying Background Agents

The rest of this paper is organised around four steps. Each step has a clear purpose, a set of decisions to make, and common failure modes to avoid.

Step	Name	What you do
Step 1	Define	Identify which workflows are worth automating and what success looks like.
Step 2	Design	Specify the trigger, the workflow logic, the tools, and the output.
Step 3	Choose	Select an implementation approach that matches your team and timeline.
Step 4	Iterate	Run the agent in a limited scope, measure it, and improve it systematically.

STEP 1 DEFINE

Step 1: Define — Which Workflows Are Worth Automating?

Most organisations have more potential agent use cases than they have capacity to build. The best candidates combine high frequency, low variability, and measurable output. High frequency means the workflow runs often enough to create meaningful cumulative value. Low variability means the inputs and expected outputs are consistent enough that the agent can handle them reliably. Measurable output means there is a clear criterion for success.

Screening questions

- How often does this workflow run today? How often should it run?
- What is the quality cost when it is skipped, delayed, or rushed?
- Which systems of record does it depend on, and are they already integrated?
- Is the output structured (a report, a record update) or unstructured (a conversation)?
- What is the cost of a bad output, and how quickly can it be detected and reversed?

05 WHO THIS IS FOR

Use Cases by Category

Category	Example use case	Frequency	Output type
Sales	Pre-call research brief	Per meeting	Document
Sales	Pipeline hygiene update	Daily	CRM action + report
Marketing	Content performance digest	Weekly	Slack summary
Marketing	Lead scoring refresh	Daily	CRM score update
Finance	Month-end reconciliation	Monthly	Spreadsheet + memo
Finance	Invoice exception report	Weekly	Flagged records
Engineering	PR review and coverage check	Per PR	Code comment
Engineering	Incident triage	Per alert	Runbook action + page
HR	Candidate screening	Per application	Scored profile
HR	Onboarding task tracker	Per hire	Task checklist
Legal	Contract renewal watch	Daily	Risk score + action
Operations	Supplier risk monitor	Weekly	Risk report

STEP 2 DESIGN

Step 2: Design — Specifying the Agent

Once you have identified a workflow, the Design step translates it into a specification an engineering team can build against. A background agent has four components: a trigger, workflow logic, tools, and an output. Getting each one right — before writing a single line of code — is the single most valuable thing a team can do to avoid expensive rework.

Trigger
The trigger defines what starts the workflow. Triggers are either event-driven or schedule-driven. Mature deployments typically combine both — an event-driven primary path, with a scheduled safety net that re-scans missed cases.

Trigger type	When it fires	Example agents
Event - system	A record changes, a webhook arrives	PR opened; opportunity closed-lost; ticket escalated
Event - human	A person takes an action	Meeting created; Slack message in a channel; form submitted
Schedule	A cron or recurring interval	Every weekday 8:00 AM; every 15 minutes; end of quarter
Threshold	A metric crosses a boundary	Error rate > 2%; pipeline coverage < 3x; cash runway < 9 months
External signal	Market, product, or customer event	Competitor funding announcement; product usage drop; status-page incident

Workflow Logic

Workflow logic is more than a system prompt. It is the explicit sequence of tasks, decisions, data retrieval, transformations, and outputs the agent must perform. A well-designed workflow is clear about how information moves through the agent and what the final deliverable looks like.
A typical meeting-prep agent, for example, performs these steps:

- 1. Read the calendar event, attendees, and meeting description.
- 2. Look up each external attendee in the CRM and enrich with role, title, and recent interactions.
- 3. Pull the last 90 days of email and call notes tagged to the account.
- 4. Check for open support tickets and product usage trends for that customer.
- 5. Summarize into a 1-page brief with relationship map, open items, and suggested agenda.
- 6. Deliver to the meeting organizer via email or calendar attachment.

STEP 2 DESIGN

Data Access and Actions Matrix

The final design dimension is the set of systems the agent must read from and the actions it is allowed to take. Being deliberate here is essential for both quality and governance. The table below is a useful template that teams can copy per agent.

Data source / Action	Read	Write	Auth method	Notes
Calendar (Google / Outlook)	Yes	No	OAuth 2.0	Read-only; do not allow agent to create or delete events
CRM (Salesforce / HubSpot)	Yes	Yes	API key or OAuth	Scope write access to specific objects only
Email (Gmail / Outlook)	Yes	Draft only	OAuth 2.0	Agent drafts; human sends. Never grant send permissions
Slack / Teams	Yes	Designated	Bot token	Restrict to specific channels; avoid DM permissions
Code repository	Yes	Comment only	Personal access token	Do not grant push or merge permissions
Document store	Yes	Designated	OAuth 2.0 or service account	Scope to project folders; avoid root access
Ticketing (Jira / Linear)	Yes	Yes	API key	Log all status changes and comment for audit trail
Financial systems (ERP)	Yes	No	Service account + IP allowlist	Read-only strongly recommended for all agent access

STEP 3 CHOOSE

Step 3: Choose — Selecting an Implementation Approach

There is no single right way to implement a background agent. The best approach depends on your team composition, your existing infrastructure, and how much flexibility you need. The three main options are: using a no-code or low-code platform, building on an agent framework, and building from scratch on the API.

The choice matters less than the execution. Teams that pick the wrong tool but execute well will outperform teams that pick the right tool but invest in evaluation and iteration.

Implementation Approaches

Approach	Best for	Trade-offs	Time to first agent
No-code / low-code	Non-technical teams; fast prototyping; workflows with clear steps	Limited flexibility; harder to customise; vendor lock-in; may hit limits as complexity grows	Hours to days
Agent framework (e.g. LangChain, CrewAI, Autogen)	Engineering teams; complex workflows; teams that want built-in abstractions	Requires Python/JS skills; framework opinionation can constrain design; version fragility	Days to weeks
Direct API (Anthropic Messages API + custom orchestration)	Teams needing maximum control; complex branching logic; fine-grained observability	Highest engineering investment; no built-in memory; team must build scaffolding from scratch	Weeks to months

Regardless of implementation approach, every production agent should have: a retry and error-handling layer, structured logging, a human escalation path, and a rollback mechanism if the agent's output is used to drive downstream actions.

STEP 4 ITERATE

Step 4: Iterate — Running, Measuring, and Improving the Agent

The most dangerous moment in an agent deployment is the first week after launch. Without a structured iteration process, teams make one of two mistakes: they declare success too early based on anecdotal positive feedback, or they abandon the agent after the first failure without diagnosing the root cause.

The Iterate step has three phases: shadow mode, limited production, and full production. Shadow mode is where the agent runs on real inputs but its outputs are not delivered to end users. Shadow mode reveals failures that would not show up in unit tests — edge cases in real data, formatting preferences, tone issues, and tool call failures under production conditions. The exit criterion is a defined accuracy threshold.

Limited production is where the agent delivers outputs to a small group of real users. The team collects structured feedback for every run. The exit criterion is a sustained satisfaction rate above a defined threshold across a minimum number of runs. Full production is where the agent runs at scale, and a production agent needs a monitoring process to detect when the distribution of inputs drifts from what the prompt was designed for.

Common Iteration Issues and Fixes

Issue	Likely cause	Fix
Output quality is poor	Prompt is under-specified; instructions are ambiguous	Add explicit instructions for each step; use few-shot examples; add output format constraints
Agent fails on a subset of inputs	Edge cases not covered in the prompt	Add conditional handling in the workflow logic; expand shadow mode test set
Tool calls are slow or failing	API rate limits or network latency	Add caching for repeated lookups; restructure workflow to batch calls; add retry logic
Users are not reading the output	Format is wrong for the delivery channel	Shorten output; change delivery format; test different structures with users
Agent is producing hallucinations	Model is hallucinating facts it should retrieve	Add verification steps; instruct agent to cite sources; add retrieval step
Drift: quality has degraded over time	Upstream data or API has changed; input distribution has shifted	Re-run shadow mode on recent inputs; update prompt to handle new patterns; add regression tests

06 DEPARTMENT-LEVEL EXAMPLES

Department-Level Examples

The following examples illustrate what agent deployments look like across the most common enterprise functions.

Department	Agent Name	Trigger	Actions	Output	Human Review
Sales	Pre-Call Brief Agent	Meeting added to calendar	Pull CRM data, news, LinkedIn	One-page brief	AE reviews before call
Sales	Deal Risk Monitor	Opportunity stage change	Analyze deal signals, compare benchmarks	Risk score + recommendation	Sales manager
Marketing	Campaign Brief Agent	Brief doc uploaded	Research audience, draft messaging	Campaign brief	Marketing lead
Marketing	Content Repurpose Agent	Blog published	Reformat for email, social, ads	Multi-channel assets	Content team
Finance	Invoice Reconciliation	Invoice received	Match PO, flag discrepancies	Reconciliation report	AP team for exceptions
Finance	Budget Variance Agent	Monthly close	Pull actuals vs. budget, explain variances	Variance commentary	CFO / Controller
HR	Job Description Agent	Role approved	Research benchmarks, draft JD	Draft JD + comp range	Recruiter + HRBP
HR	Onboarding Coordinator	Offer accepted	Create tasks, send docs, schedule meetings	Onboarding checklist	HR partner
Legal	Contract Review Agent	Contract uploaded	Flag non-standard clauses, compare to playbook	Redline + risk summary	Associate GC
Legal	NDA Drafting Agent	Request submitted	Draft NDA from template + context	Draft NDA	Legal ops or counsel
IT	Incident Summary Agent	Ticket resolved	Summarize issue, root cause, resolution	Post-incident report	IT manager
IT	Access Request Agent	Request submitted	Verify approvals, provision access	Access granted + audit log	IT security review

07 WORKED EXAMPLE

Worked Example: Sales Pre-Call Brief Agent

This section walks through a complete deployment of the Sales Pre-Call Brief Agent — one of the highest-impact, lowest-risk starting points for enterprise agent adoption.

- **Trigger**. A new external meeting is created in the CRM-linked calendar for an account-owning rep, at least 12 hours out.
- **Workflow logic**. The agent enriches each attendee, pulls recent CRM activity, checks open tickets and usage, and assembles a one-page brief with relationship map, open items, suggested agenda, and two likely objections.
- **Data access.** Reads from CRM, calendar, email, ticketing, and product telemetry. Writes a note on the opportunity record and emails the brief to the meeting organizer.
- **Guardrails**. Does not send anything outside the company. All outbound communication remains with the rep. Logs every tool call with inputs and outputs.
- **Evals**. Weekly, reps rate the brief 1–5 on a one-click survey. Scores below 3 on a given account trigger a design review.

A deployment like this typically takes a small mid-market team two to four weeks to get into production, and usually displaces 20–40 minutes of manual prep per external meeting.

Risks, Controls, and Governance

Because background agents can operate autonomously across business systems, organizations should treat governance as a design requirement rather than an afterthought. The more valuable the workflow, the more important it becomes to define permissions, review mechanisms, monitoring, and clear operational boundaries.

Control area	What it means	Example mechanism
Data permissions	Agent operates with the narrowest read/write scopes needed	Per-agent service accounts with scoped OAuth tokens
Auditability	Every run is logged and reviewable after the fact	Structured logs of inputs, tool calls, outputs; retention policy
Human review	Sensitive actions require explicit approval	Approval queue in Slack or the task tracker before send/commit
Reliability	Agent quality and uptime are monitored	Run-level metrics, alerts on failure rate and latency, eval dashboards
Security	Credentials and data are handled to standard	Secret manager integration, no keys in prompts, encrypted storage
Failure handling	Failures degrade gracefully instead of silently	Clear error states, idempotent retries, circuit breakers
Escalation paths	Every agent has an owner and a defined off-ramp	Named product owner, runbook, clear paging rules

08 GOVERNANCE

Governance Framework

As agents proliferate across the enterprise, governance becomes critical. Without it, you risk data leaks, compliance failures, rogue automation, and loss of employee trust. A lightweight but consistent governance framework should cover five areas:

1. Ownership and Accountability

Every agent must have a named owner — a human who is accountable for its behavior, performance, and compliance. Ownership should be assigned at deployment and reviewed quarterly. The owner is responsible for approving changes, reviewing audit logs, and escalating issues.

2. Access Controls

Agents should operate on the principle of least privilege — they should have access only to the systems and data they need to complete their task, and nothing more. Access should be granted through service accounts with auditable permissions, not individual user credentials. Agents should never have admin-level access unless explicitly required and separately approved.

3. Audit Logging

All agent actions should be logged: what data was read, what actions were taken, what outputs were generated, and when. Logs should be stored in a tamper-evident system and retained according to your company's data retention policy. Audit logs are essential for debugging, compliance, and incident response.

4. Human-in-the-Loop (HITL) Gates

Define in advance which actions require human approval before execution. As a rule of thumb: any action that is irreversible, involves sensitive data, or has external impact should require a human sign-off. HITL gates should be built into the agent workflow, not bolted on after the fact.

5. Incident Response

Define a clear process for what happens when an agent makes a mistake or behaves unexpectedly. This should include: how to pause or kill the agent, how to notify affected stakeholders, how to assess impact, and how to remediate. Every agent deployment should have a runbook that covers common failure modes.

CONCLUSION

Conclusion

Autonomous agents represent a fundamental shift in how enterprises operate — not just a new category of software, but a new type of workforce. They work continuously, don't forget, don't get tired, and can operate across systems that have never been connected before.

The enterprises that move thoughtfully and deliberately today will have a significant advantage in 12–18 months. Not because they deployed the most agents, but because they built the foundations — the data infrastructure, the governance frameworks, the change management muscle — that allow agents to operate safely and effectively at scale.

> **"The question is no longer whether to deploy agents, but how to deploy them well."**

The playbook is straightforward: start with high-value, low-risk tasks. Nail the fundamentals. Build trust with employees and leadership. Expand deliberately. Measure everything.

The organizations that get this right won't just be more efficient — they'll be structurally different, capable of operating at a speed and scale that their competitors cannot match.

About This Document

This white paper was produced to help enterprise leaders navigate the practical challenges of deploying autonomous AI agents. It draws on real deployment patterns and is intended as a practical guide, not a theoretical framework.