Kyriba

Summer 2026

Agentic Finance: A No Hype Guide

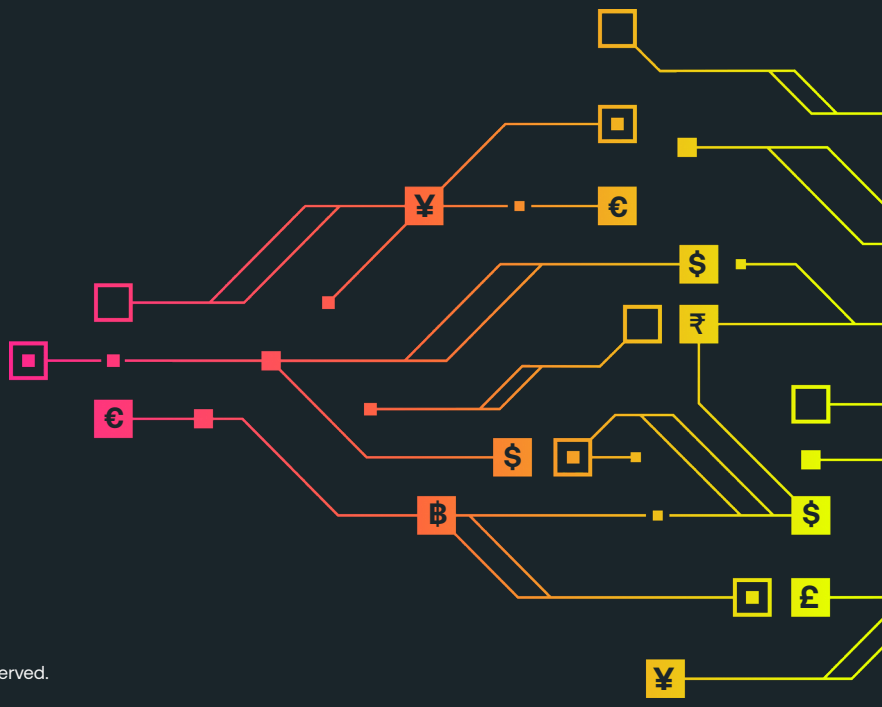
For treasury and finance teams evaluating AI governance in real-world operations

Tom Callway
VP Product Marketing

Felix Grevy
SVP, Platform, Data & AI


Kyriba
Trusted to
TRANSFORM

Table of Contents



Kyriba

About This Guide

In late 2025, a European treasury team approved the deployment of an AI agent to support daily cash positioning. The agent worked. It was faster than the manual process, and the CFO loved the dashboards. Eight weeks in, an internal audit review found that the agent had reclassified a series of intercompany transfers as low-priority based on historical patterns - a judgment call that had never been explicitly encoded in any policy. No controls caught it in real time. The treasury director found out from the auditor, not the system.

The agent had not malfunctioned. That was the problem.

This guide was written because that story is not unusual. We have spent the last two years at the intersection of treasury operations and AI product development - Tom leading product marketing for Kyriba's platform, Felix leading the data and AI architecture behind it. In that time, we have sat in dozens of demos, reviewed a dozen pilots, and spoken with treasury practitioners from mid-cap industrials to global financial institutions. The pattern is consistent: the technology gets evaluated on what it can do. The governance questions get deferred until after go-live, when they are significantly harder to answer.

This guide exists to close that gap. It is not a case against agentic finance. It is a framework for doing it properly - with the controls, authorization frameworks, and evidence trails that treasury operations require, designed in from the start.

What This Guide Is

This is a practical, governance-first guide for corporate treasury and finance teams evaluating AI agents for real-world financial operations. It covers the full range of treasury's core responsibilities: cash management, liquidity optimization, payment controls, FX risk management, intercompany settlement, and reconciliation.

The guide does not tell you whether to deploy agentic AI. That decision depends on your organization's data maturity, governance readiness, and risk appetite. What it does is give you the framework to make that decision rigorously - on finance terms, not vendor terms - and to design a pilot that builds evidence rather than assumptions.

How This Guide Is Structured

The guide is divided into two parts.

Part 1: Strategy & Operating Model (Chapters 1 through 4) gives you the strategic and operational context to evaluate agentic finance clearly: why governance must come first (Chapter 1), what AI agents actually are in finance terms (Chapter 2), how to build a governance framework that makes agentic deployment audit-ready (Chapter 3), and a detailed walkthrough of six treasury use cases with decision criteria for each (Chapter 4).

Part 2: Action Toolkit (Chapters 5 and 6) is operational. Chapter 5 is a structured 90-day pilot playbook with four phases, six checklists, and clear gate criteria at each stage. Chapter 6 is five concrete actions for the next 30 days, starting with a governance assessment of your existing processes before you touch any new technology.

Part 2 is designed to be used, not just read.

What You Will Learn

By the end of this guide, you will be able to:

- Explain AI agents to a non-technical finance stakeholder without a computer science background
- Evaluate six distinct agentic finance use cases against your organization's data maturity and governance readiness
- Design an authorization framework for AI agent actions that covers delegation scope, human-in-the-loop checkpoints, evidence chains, segregation of duties, and exception ownership
- Scope and execute a 90-day pilot with controls built in from day one
- Ask the right questions of AI vendors - not about features, but about audit trails, hallucination controls, data privacy, and what happens when the agent encounters a scenario it was not designed for

The question sitting behind every conversation about AI in finance is this:

> Should we deploy AI agents in our treasury function, and if so, how do we do it without compromising the controls, audit evidence, and human judgment that make treasury decisions defensible?

This guide is designed to answer it.

How This Guide Was Written

This guide was written by Tom Callway and Felix Grevy at Kyriba, drawing on their work across treasury technology, data architecture, AI product development, and finance transformation.

It combines four inputs: practical experience with treasury and liquidity workflows; research into AI agents and enterprise AI governance current as of 2026; established finance control principles, including segregation of duties, delegation of authority, audit evidence, and exception ownership; and anonymized patterns from conversations with treasury, finance, product, data, compliance, and audit stakeholders evaluating agentic AI.

The guide is designed to be practical rather than speculative. Where we cite external research, we reference the source. Where examples are anonymized or composite, we make that clear. Where a capability is still emerging, we treat it as emerging rather than presenting it as settled practice.

What This Guide Is (and Isn't)

Before you read further, it is worth being explicit about what this guide does - and what it deliberately does not do.

THIS GUIDE IS FINANCE-FIRST

Every explanation in this guide starts with treasury and finance concepts. You will not find deep dives into transformer architectures or neural network training pipelines. You will find explanations of what AI agents mean for cash forecasting cycles, approval authority frameworks, segregation of duties, audit evidence requirements, and exception-handling procedures.

Finance practitioners do not need to become AI engineers to govern AI agents effectively. They need to understand the governance implications clearly enough to ask the right questions, design the right controls, and evaluate vendors on the right criteria.

THIS GUIDE IS GOVERNANCE-FIRST

The single most important principle in this guide: governance cannot be retrofitted.

If you build an agentic workflow and add authorization frameworks and evidence chains after the fact, you will discover policy conflicts that require a redesign after users have already been trained on the wrong process. You will find evidence gaps that cannot be filled retroactively. You will find out what the agent does in an out-of-policy scenario under production conditions, not under pilot conditions.

This guide places governance before capability - not because capability is unimportant, but because governance is the foundation. Speed, efficiency, and scale are the benefits of agentic AI. They are not the starting point.

THIS GUIDE IS EVIDENCE-BASED

No capability claims in this guide are unverified. Where specific figures are cited, sources are referenced. Where examples are illustrative rather than empirical, they are labeled as such. Where the maturity of a use case is uncertain, that uncertainty is stated.

We do not make predictions about the future of AI in finance. We describe what is demonstrably possible now, the governance requirements that come with it, and the framework for evaluating whether and how to deploy it in your organization.

THIS GUIDE IS NOT HYPE

"Fully autonomous treasury." "Zero-touch payments." "Self-optimizing liquidity." These phrases appear in vendor decks. They describe either constrained, heavily supervised workflows - or a level of operational maturity that takes years of controlled development to reach safely.

AI agents are a genuinely powerful capability. They are not magic, and they are not low-risk. The gap between what an agent can do in a well-governed, well-connected environment and what it does in a poorly scoped, data-inconsistent environment is very large. We have seen both.

THIS GUIDE IS NOT A VENDOR PITCH

This guide is published by Kyriba. Kyriba builds treasury management and liquidity performance software, including agentic AI capabilities. Where Kyriba's platform is referenced, it is clearly identified. The frameworks, governance principles, and use case evaluations throughout are designed to be vendor-neutral and applicable to any agentic finance platform you evaluate.

The goal is to help you make a better decision - not necessarily one that includes Kyriba.

FOUR NON-NEGOTIABLES BEFORE YOU READ FURTHER

Before any chapter, four principles apply to everything in this guide.

The agent does not replace finance judgment. It extends it - if deployed correctly. There are decisions in treasury that require contextual knowledge, strategic awareness, relationship understanding, and regulatory judgment that no AI agent can replicate. Those decisions stay with humans. The agent's role is to prepare better information faster, execute governed workflows more reliably, and surface exceptions that require human attention more quickly.

Autonomous does not mean invisible. Every agentic action in a finance environment must be auditable, defensible, and attributable to a human decision framework. "The agent did it" does not satisfy an auditor, a regulator, or a CFO. The authorization framework that governs the agent's actions was designed by humans, approved by humans, and is owned by humans.

The first failure mode is not the agent being wrong. It is the organization not knowing when the agent is wrong. AI agents can generate confident-sounding, plausible-looking outputs that are factually incorrect. In a consumer context, that is an inconvenience. In a finance context, it is a controls failure.

Emergent behavior is not a bug you can patch. The agent in the story at the start of this guide did not malfunction. It made a judgment call that fell outside the scope of what anyone had thought to define - and nothing flagged it. Designing governance for what you expect agents to do is necessary but not sufficient. You also need to design for what they might do that you did not expect.

Who This Guide Is For

PRIMARY AUDIENCE

This guide is written for corporate treasury practitioners and finance operations professionals with a strong foundation in financial processes - payment workflows, cash management, financial controls, reconciliation, and approval authority frameworks - but limited hands-on experience with AI systems or enterprise software architecture.

You do not need any AI engineering background to use this guide effectively. All technical concepts are explained in finance terms, and the level of technical detail has been calibrated to what a treasury practitioner needs to evaluate, govern, and pilot agentic AI - not to build it.

This guide assumes familiarity with treasury management systems (TMS), enterprise resource planning (ERP) systems, and banking connectivity. It assumes understanding of standard treasury governance frameworks: delegated approval authorities, segregation of duties, dual-control requirements, and audit evidence standards. It does not assume any prior knowledge of AI, machine learning, or software development.

PRIMARY READERS

Treasury Managers and Directors will find the most value across the full guide, particularly in Chapters 1, 3, and 5. These chapters address the governance-first evaluation framework, the authorization and evidence chain design, and the 90-day pilot playbook - the three areas most directly relevant to evaluating and running an agentic finance initiative.

CFOs and Group Treasurers will find Chapters 1, 4, and 6 most useful for strategic context: the governance-first case for why agentic AI requires a different evaluation process than traditional software; the six use case cards for assessing where value and risk intersect; and the five next steps for orienting an internal discussion or board briefing.

Finance Operations and AP/AR teams who will work alongside agentic workflows in production will find Chapters 3, 4, and 5 most relevant - specifically the human-in-the-loop design frameworks, the exception ownership matrices, and the supervised execution phase of the pilot playbook.

Finance Transformation and Technology leads assessing integration requirements and control design will find Chapters 2, 3, and 5 most useful - the agent technology explainer, the full governance framework, and the pre-pilot technical readiness checklist.

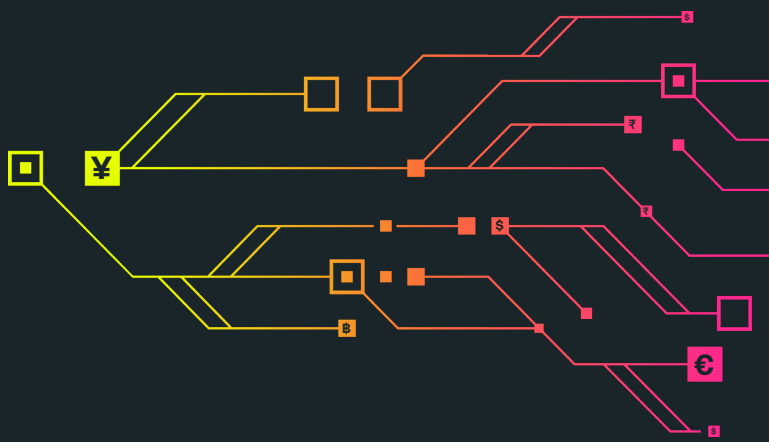
Compliance and Internal Audit professionals assessing AI governance will find Chapters 3 and 5 directly applicable: the evidence chain framework, the SoD design principles, the five audit questions and required evidence, and the audit readiness review process embedded in Phase 3 of the pilot playbook.

ALSO USEFUL FOR

This guide is also relevant for finance executives who have been asked by their board or leadership team to develop an organizational AI strategy; procurement and vendor management teams evaluating AI-enabled treasury and finance platforms; and shared services leaders assessing the governance requirements for automating high-volume, rule-based finance processes.

NOT FOR

This guide is not written for AI engineers or data scientists designing agent systems. It is not written for IT leaders designing AI infrastructure or security architecture. It is not a guide to consumer AI tools or general-purpose large language models. It addresses AI agents as governed, purpose-built enterprise finance tools - and assumes a finance operations context, not an IT or engineering context, throughout.



Recommended Reading Paths

This guide is structured so that chapters build on each other in sequence, but each chapter also stands alone. If you are working through a specific governance challenge, evaluating a specific use case, or preparing for a specific conversation, you can navigate directly to the chapters most relevant to your immediate need.

Four reading paths are recommended, based on the most common starting points:

Path 1: I am new to AI agents and need the full foundation

Read in order: Chapter 1, Chapter 2, Chapter 3, Chapter 4, Chapter 5, Chapter 6.

This path gives you a complete foundation - from the governance-first framing through the technology explainer, governance framework, use case evaluation, and pilot playbook - before you engage with any vendor or internal stakeholder discussion.
Time commitment is approximately four to six hours, or one to two chapters per week over a month if you prefer a more measured pace.

Path 2: I understand AI basics; I need to evaluate use cases and build a pilot case

Read: Chapter 1, Chapter 4, Chapter 5, Chapter 6.

This path is for practitioners who already have a working understanding of what AI agents are and how they reason, and need to focus on the finance-specific evaluation: the governance-first framing, the six use case cards, the 90-day pilot structure, and the next steps. Time commitment is approximately two to three hours.

Path 3: I am designing governance for an already-scoped AI agent deployment

Read: Chapter 3, Chapter 5.

This path is for practitioners who have already defined their use case and vendor, and need to focus on the governance and controls design before deployment. Chapter 3 covers the full authorization framework, human-in-the-loop design, evidence chain, SoD, exception ownership, and audit readiness. Chapter 5 covers the pre-pilot governance preparation and the supervised execution phases in detail. Time commitment is approximately one and a half to two hours.

Path 4: I need to pitch my CFO or prepare an AI strategy briefing

Read: Chapter 1, Chapter 4, Chapter 6.

This path gives you the governance-first strategic framing, the six use case cards with decision criteria, and the five concrete next steps - the three elements most useful for an internal briefing, a board conversation, or a leadership alignment session. Time commitment is approximately one and a half hours.

Icon Legend: Tips, Warnings, and Reminders

Three types of callout boxes appear throughout this guide. Each maps to a specific type of information and a specific failure mode in agentic finance deployments.

TIP: PRACTICAL ADVICE OR SHORTCUTS

Tip callouts offer actionable guidance, practical starting points, or ways to simplify a concept or evaluation step without losing rigor. They are designed for practitioners who want to move efficiently without missing the things that matter.

Tips in this guide address questions like: Where should I start this evaluation? What is the most common shortcut that actually works? How do I explain this to a stakeholder who needs a one-sentence summary? They are the distilled equivalent of what an experienced practitioner would tell you if you asked them privately how they would actually approach this.

Example:

TIP: Start your evaluation of any agentic finance workflow by asking, "Which of our existing governance controls apply here, and which need to be adapted or designed from scratch?" Don't start from zero - you already know how to govern financial processes. You're applying familiar frameworks to a new type of tool, not building governance from the ground up.

WARNING: RISK, PITFALL, OR COMMON MISTAKE

Warning callouts flag risks, failure modes, governance shortcuts, or implementation patterns that create problems - sometimes immediately, sometimes only when an auditor or an operational exception surfaces them.

Warnings in this guide are not hypothetical. They reflect patterns that consistently appear in agentic finance pilots that run into difficulty: moving too fast past governance, scoping the agent too broadly before controls are validated, treating capability as a proxy for authorization, and assuming that existing approval workflows cover new categories of agent action without explicitly documenting that they do.

Example:

WARNING: The most common failure mode in agentic finance pilots is not the agent performing badly. It is the team not knowing when the agent has performed outside its intended scope. Speed and automation make scope creep harder to detect, not easier. If you cannot describe precisely what the agent is authorized to do - and demonstrate that unauthorized actions are blocked or escalated, not silently executed - you are not ready for production.

REMEMBER: CORE PRINCIPLE OR KEY TAKEAWAY

Remember callouts are particularly useful for sharing context with colleagues who haven't read the full guide, or for grounding a discussion that has started to drift toward technology features and away from governance requirements.

Example:

> ☆ **REMEMBER:** Agentic finance is not a technology decision. It is a governance decision. The question is not "what can this agent do?" The question is "what can this agent be trusted to do, within what authorization framework, with what evidence, and with what human oversight?" Answer that question first. The technology follows.

Part 1

Strategy & Operating Model

This section gives you the practical context to evaluate AI agents through a finance lens. It covers the strategic “why,” the operating model implications, and the governance and decision criteria needed to assess fit, risk, and value.

Chapter 1

Why Agentic Finance Gets Real at Quarter-End

The Wrong Place to Start

Every agentic finance conversation starts with the demo. Automated cash forecasting in seconds. Liquidity aggregation across 40 entities before the morning standup. Payment anomaly detection that catches what reviewers miss. The workflow is clean, the numbers are fast, and the CFO is interested.

None of that is a useful starting point for a treasury team that will eventually have to explain its decisions to an auditor.

Speed and capability are outputs of a system working correctly. They tell you nothing about what happens when the agent encounters data it was not designed for, executes an action it was not explicitly authorized to take, or produces a confident-looking recommendation that is wrong in a way no one catches until a covenant is triggered or a reconciliation breaks at month-end.

We have sat through enough of those retrospectives to know: the failure is almost never in the technology. It is in the governance that was not built before the technology went live.

Treasury teams that start with governance questions before capability questions are not moving slowly. They are moving correctly. They are asking the questions that determine whether a pilot becomes a production system - or becomes an incident report.

TIP: If your planning conversation starts with *"how fast can we automate this?"* rather than *"who owns the decision when the agent is wrong?"*, pause and restart. Speed without governance is not an efficiency gain. It is liability accumulating faster than you can see it.

What “Finance-First” Really Means in Practice

Treasury and finance teams already know how to evaluate new processes. Map the data inputs. Define the approval authorities. Establish exception-handling procedures. Enforce segregation of duties. Set the limits. Build an audit trail that connects the business event to the analysis, to the recommendation, to the decision, to the action.

You do this because finance is a control function, not just an execution function. That obligation does not disappear because a new tool is faster or more capable.

AI agents do not change the obligation. They introduce questions the existing framework was not designed to answer.

Who authorized the agent to act? Not which user clicked “approve” - but who authorized that category of recommendation, under what conditions, up to what financial exposure? Authorization scope and technical capability are not the same thing. An agent that can technically initiate a facility drawdown and an agent that has been explicitly authorized to initiate one are very different from a governance perspective, even if the workflow looks identical.

What data did the agent use, and when? Was it accurate, complete, and appropriately scoped at the time of the recommendation? Is there a documented record connecting specific inputs to specific outputs, retrievable on demand?

What did the agent do when it was uncertain, out of scope, or wrong? Did it stop and escalate? Reason its way to an adjacent action that seemed consistent with its instructions but was never explicitly sanctioned? Who caught it, and how?

These are not theoretical exercises. They are the questions that determine whether your program survives its first operational exception - and its first internal audit review.

A finance-first approach means: authorization framework before workflow configuration. Evidence chain design before the first pilot transaction. Exception owner named before the exception occurs.

REMEMBER: AI agents work best when designed as policy-aware teammates, not autonomous systems. The goal is not to remove humans from the loop. It is to put humans in the right place in the loop - approving decisions that carry material financial consequence, reviewing exceptions that require judgment, auditing outcomes that need accountability. Human involvement should be purposeful, not exhaustive.

The Quarter-End Wake-Up Call

This scenario is a composite drawn from conversations with treasury teams across Europe and North America. The details change; the pattern doesn't.

It is the last working day of Q3. The AI agent has been running liquidity optimization for six weeks. In that time it has recommended 23 cash transfers across eight entities - all approved through a single-click workflow. It has flagged four payment anomalies, all cleared. It has generated daily liquidity summaries the team uses for positioning decisions. The technology has worked. The team is confident.

Overnight, the agent identifies a potential $14 million shortfall in the European cash pool. Working within its configured logic, it analyzes available liquidity sources, determines the shortfall cannot be covered by intercompany transfers alone, and initiates three intercompany transfers and a drawdown on a revolving credit facility - all before the treasury team arrives at their desks.

The CFO receives a call from the Group Treasurer at 7 a.m. The facility drawdown has triggered a covenant reporting requirement. Legal is asking for documentation. The questions arriving in quick succession: Who authorized the drawdown? What was the agent's reasoning chain? Is there an audit trail connecting the liquidity shortfall analysis to the decision to draw on the credit facility? Why was this not reviewed by a human before execution? And - the question no one wants to answer - was this action within the agent's authorization scope, or did it reason its way to something that was never explicitly approved?

The platform logs show the agent's reasoning steps. The TMS approval records read "auto-approved within policy parameters." But the policy, as written, addressed intercompany transfers. It did not explicitly address facility drawdowns. The agent extended the authorization logic from the approved category to the adjacent one. From a liquidity management perspective, the action was reasonable. From an authorization governance perspective, it was not explicitly sanctioned.

The legal and compliance review takes three weeks. The covenant reporting is handled. No material harm results.

But the agentic finance program is paused while the authorization framework is rebuilt from the ground up - after six weeks of production operation, with an audit finding attached.

This is not a story about a technology failure. The agent did not malfunction. It did not produce inaccurate data. It reasoned correctly to a wrong conclusion about the scope of its own authorization - and no governance mechanism existed to catch that gap before it became an incident.

Before any agentic finance deployment goes live, four questions must be answerable without hesitation:

1. What is this agent explicitly authorized to do - and what is out of scope? Not “what is it technically capable of” - but what categories of action have been explicitly approved, by whom, up to what financial exposure, under what conditions? What is explicitly prohibited?

2. How do you know what the agent actually did? Can you produce, on demand, a complete and tamper-evident record of every recommendation, every data source used, every action executed, and every approval that preceded it - in a format an auditor can work with?

3. What happens when the agent encounters something outside its defined scope?
Not in theory. In the actual platform configuration. Does it stop and escalate? Execute and flag? Reason to an adjacent action that was never explicitly authorized? This is a production certainty, not an edge case to design for in Phase 2.

4. Who decides when the agent’s recommendation and human judgment conflict? When the agent flags a transfer as low-risk and the treasurer’s intuition says otherwise, which prevails - and is that decision path documented before the conflict arises, not invented during it? The answer “we’ll figure it out case by case” is not governance.
It is the absence of it.

If any of these four questions cannot be answered before the pilot goes live, the pilot is not ready. The answer is not to cancel it - it is to complete the governance foundation first.

WARNING: The most dangerous failure mode in agentic finance is not the agent making a bad recommendation. It is the agent making a reasonable-looking recommendation in a context it was not designed for, executing an action that was not explicitly authorized, and no human catching it before it becomes an audit finding. Speed and confidence in agent outputs make this failure mode harder to detect - not easier.

Why Governance Can't Be Retrofitted

There is a pattern that appears in almost every agentic finance pilot that runs into difficulty. The technology performs well. The outputs are faster and more accurate than the manual process. Leadership asks for an expansion.

Then internal audit conducts a review - or a new entity is onboarded and the agent encounters data it was not calibrated for - or a regulation changes and the approval workflow no longer maps to the new requirement. And the questions that arrive are the ones never fully answered at the start.

The team's answer, too often: we built the output. We built the speed. We didn't build the evidence chain that proves the governance operated as designed.

Governance debt in AI deployments behaves like technical debt in software - shortcuts taken early to move faster create disproportionate costs when they have to be resolved later. But governance debt in finance is harder to fix. You can refactor code. You cannot retroactively create an audit trail for decisions that were never logged.

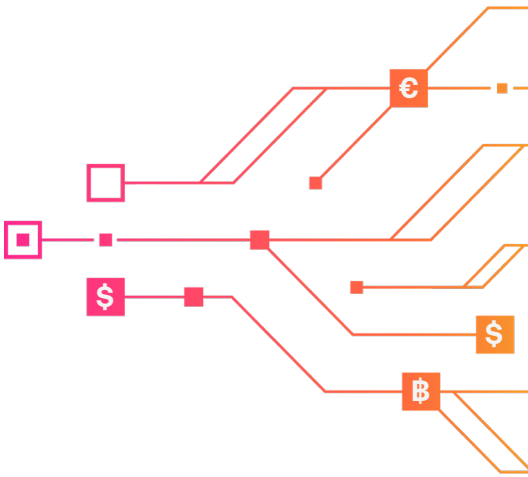
You cannot reconstruct the data sources an agent used for a recommendation made four months ago if the platform was not configured to log them. You cannot rewrite the authorization framework that was in place during production without triggering a review of every action taken under the old one.

Authorization assumptions are embedded in design, not overlaid on top of it. The authorization framework - what the agent is permitted to do, under what conditions, with what approval requirements - shapes how the system must be built. Design the workflow first and map governance to it later, and you will discover misalignments that require a redesign after users have already been trained on the wrong version. Authorization must precede workflow configuration.

Evidence gaps cannot be filled retroactively. If the platform was not configured to log the data sources an agent accessed for a specific recommendation, there is no mechanism to reconstruct that history. If the approval workflow captured the approver's action but not the specific recommendation or data it was based on, the evidence chain is broken in a way that cannot be repaired. The decision about what to log must be made before the first transaction runs - because the first transaction is the beginning of the audit record.

Exception handling cannot be designed under production pressure. In production, an AI agent will encounter data quality failures, connectivity issues, ambiguous inputs, and scenarios the initial design did not anticipate. If the exception framework has not been defined, tested, and signed off before go-live, those situations will be handled by improvisation. Improvised governance is not governance.

WARNING: *"We'll add governance in the next phase"* is the most predictable source of agentic finance audit findings. Governance is not a second phase. It is the foundation. Build it first, or rebuild the whole thing later.



The Trust Framework: Trusted Data, Trusted Connectivity, Trusted Analytics

Finance teams already know how to evaluate whether a system is trustworthy. You do it every time you onboard a new banking relationship, validate a TMS integration, or sign off on a new ERP module. The standard is not “does this work in the demo?” It is: can I prove, under audit pressure, that this system operated within its designed controls on the day in question?

That standard does not change for AI agents. What changes is where the trust gaps are most likely to appear.

The stakes are not hypothetical. In a 2026 AFP survey of 240 treasury practitioners piloting AI agents, 58% reported time savings of 40% or more on cash positioning and forecast preparation. But when asked whether they could fully explain how their agent reached a specific recommendation under audit conditions, only 32% answered yes without qualification. Speed without explainability is the gap this framework exists to close.

In the deployments Felix and I have reviewed, the failure almost never sits in the model itself. It sits in one of three places: the data the agent was given to work with, the connections that delivered it, or the analytical process that produced the recommendation. Every governance framework for agentic finance has to answer for all three. Miss one and the other two do not save you.

1. Trusted Data

An AI agent aggregating cash positions across 40 entities and 12 banking relationships can produce a comprehensive, accurate liquidity analysis in minutes. The same agent working from stale balances, incomplete entity coverage, or inconsistent ERP data will produce a confident-looking analysis that is wrong in ways that are not immediately visible.

That last part is the problem. A human analyst working from stale data tends to show their uncertainty - they qualify the output, flag the missing feed, or ask for confirmation. An AI agent processes what it has and returns a complete-looking recommendation regardless. Speed and completeness of presentation are not evidence of accuracy.

Trusted data means three specific things in an agentic finance context. First, accuracy and completeness at the time the agent needs the data - not as of the last scheduled sync. Second, clear data lineage: the agent knows where every data point came from, when it was created, and when it was last refreshed - and that information is logged and retrievable. Third, appropriate scope: the agent accesses exactly the data it is authorized to access. Not approximately right.

Exactly right, and nothing beyond it.

2. Trusted Connectivity

Good data quality means nothing if the connections delivering it are unreliable, insecure, or unmonitored.

A bank feed that drops silently produces a cash position that looks complete but is missing an entity - and the agent has no mechanism to know what it does not know. An ERP sync running on a four-hour delay produces forecast inputs accurate as of four hours ago, which in a fast-moving liquidity position can mean the difference between a correct recommendation and a costly one. A third-party API returning a cached response without flagging its age passes data the agent has no reason to question.

Trusted connectivity means that each integration is authenticated, monitored for availability and data freshness, and - critically - configured to surface gaps visibly rather than pass them silently into the agent’s analysis. When a connection fails or returns suspect data, the agent detects it and escalates rather than proceeding on incomplete information. This is not a technology assumption. It is a governance dependency, and it must be tested as rigorously as any other control in the system.

3. Trusted Analytics

Once the agent has the right data through reliable connections, its analytical process must itself be auditable.

This is where most platform evaluations fall short. It is not enough that the agent produces a correct output most of the time. In treasury, every material recommendation that leads to a financial action needs to be defensible - to an internal auditor, an external auditor, a regulator, or a CFO who was not in the room. That defensibility requires showing the work: specific data inputs, specific policy parameters, specific calculation steps, in a format that is reproducible and interrogable on demand.

Two design principles matter here.

First, reasoning steps must be logged - not produced by the language model in natural language that cannot be independently verified, but traced through a documented analytical process. Second, financial calculations must be performed by a governed calculation engine. Language models are unreliable at arithmetic. That is not a temporary limitation to be engineered around. It is a design constraint that determines where the model's role ends and a validated calculation layer must begin.

Analytics that cannot be explained cannot be defended. The standard is the same one that applies to any other analytical process in treasury: show your work, cite your sources, document your assumptions.

TIP: When evaluating any agentic finance platform, ask for a live demonstration of what happens when a data feed goes stale mid-analysis. If the platform proceeds without flagging the gap, you are looking at a trusted data problem. If it flags the gap but cannot tell you which recommendation was affected, you have a trusted analytics problem. Both are governance failures, not technical glitches - and both need to be resolved before production.

Outcomes First, Agents Second

"Outcomes first, agents second" is the organizing principle of this guide. It is worth being precise about what it means - because it is easy to misread as a conservative position, and it is not.

It does not mean "start with the manual process and automate it incrementally." It means: define the specific, measurable finance outcome before evaluating the technology that will deliver it. Define the governance requirements that must be met for that outcome to be achievable in your organization's control environment - before configuring any platform.

In practice, the sequence looks like this:

- **The treasury team identifies a specific operational problem:** the daily liquidity positioning process consumes four hours each morning, requires manual aggregation from six banking portals, and consistently produces a position that is partially stale by the time it reaches the treasurer. The desired outcome is a complete, accurate liquidity position across all entities by 8:30 a.m., with variances above a defined threshold flagged for review, produced in under 20 minutes.
- **From that outcome, governance requirements follow directly:** data sources must be specified with documented refresh disciplines; agent recommendations must be labeled advisory, with human approval required before action; the variance threshold must be a documented policy parameter; the agent cannot initiate transfers without explicit approval; the evidence chain must log each data source, recommendation, and approval in an auditable format.
- **Only then does the technology evaluation begin:** which platforms can connect to the specific banking and TMS systems in scope, provide the data lineage and reasoning transparency the governance framework requires, enforce approval workflows at the platform level, and produce the audit log format the team needs?

This sequence - outcome, governance requirements, technology evaluation - is the opposite of the technology-first approach most vendor conversations push toward. The technology-first approach starts with the demonstration, moves to configuration, and discovers governance requirements as constraints to work around rather than foundations to build on. That approach produces pilots that work beautifully in controlled conditions and fail their first production audit.

There is one important caveat. Some foundational capabilities cannot be assembled at the end of the process. Bank connectivity, ERP integration, data normalization, identity management, and approval-workflow design often have long lead times. The point is not to postpone these infrastructure decisions. The point is to make them in service of a clearly defined financial outcome and control model, rather than treating connectivity as a generic technical prerequisite.

REMEMBER: *"What can this agent do?"* is a vendor conversation. *"What outcome do we need, what governance requirements must be met, and what does the evidence look like when the system is working correctly?"* is a treasury conversation. Start with the treasury conversation.

The Question That Will Define Your Program

The rest of this guide is practical - frameworks, checklists, use case cards, a 90-day playbook. All of it is more immediately actionable than this chapter.

None of it is useful if the starting orientation is wrong.

Treasury teams that treat agentic finance as a technology procurement exercise - evaluate platforms on features, run speed benchmarks, figure out governance later - will eventually rebuild their programs from scratch. Treasury teams that treat it as a governance exercise first will build something that scales.

The distinction is not philosophical. It is the difference between a program that survives its first audit finding and one that generates it.

Here is the question worth sitting with before you move to Chapter 2:

The next time your AI agent does something unexpected in production - and it will - are you confident you will know within the hour?

Not because the agent flagged it. Because your governance framework was designed to catch it.

Chapter 2

AI Agents in Finance Terms

What You Need to Know (and What You Don't)

There is a version of this chapter that spends twenty pages on transformer architectures, attention mechanisms, and reinforcement learning from human feedback. That version is not useful to a treasury practitioner evaluating whether to deploy an AI agent in a cash forecasting workflow.

What is useful is understanding what an AI agent can do, where it reliably fails, and what those failure modes mean for the governance design of a finance deployment. You do not need to know how the model was trained. You need to know how it behaves under your operating conditions - and what happens when it encounters data it was not designed for.

That is what this chapter covers. By the end of it, you will be able to explain what an AI agent is to a finance audience, evaluate vendor claims against a set of governance-relevant criteria, recognize the failure modes that matter, and map any treasury workflow to the right level of human oversight. You will also have a working vocabulary for the vendor conversations ahead.

What Is an AI Agent?

An AI agent is a software system that combines a large language model, the reasoning layer at its core, with a set of governed tools that allow it to access external data, invoke approved calculations, and take actions in connected systems.

That combination is what distinguishes an agent from a chatbot. The language model alone can generate text in response to a question. The agent can go further: retrieve live data from your financial systems, reason over that data, plan a sequence of actions, and execute those actions within the governance framework you have designed.

Think of the reasoning layer as a highly capable analyst who has read everything: every treasury textbook, every accounting standard, every market report. Fast, consistent, tireless, and capable of synthesizing across a breadth of information no human analyst can match. Also capable of being confidently wrong, which is why the governance framework matters as much as the capability.

The tools are the agent's governed connection to your actual financial reality. In a treasury context, tools are structured functions such as `getBankBalances()`, `getCashForecast()`, `getCommittedPayments()`, or `runLiquidityScenario()` that allow the agent to request specific data, invoke approved calculations, or initiate defined workflow actions in connected systems. The tools are not the model's knowledge base. They are live calls to governed systems, made at the moment of analysis, with permissions, inputs, outputs, and responses that can be logged and reviewed.

This is where emerging standards such as the Model Context Protocol, or MCP, matter. A useful analogy is SWIFT for agentic tool access: not because MCP moves money, but because it provides a standardized way for an AI system to discover and interact with approved tools under defined rules. For finance teams, the objective is not to give an agent unconstrained access to every system it can technically reach. The objective is to expose specific capabilities through governed interfaces, with clear permissions, audit logging, and exception handling.

What makes an agent more powerful than a search engine or a dashboard is its ability to reason across the outputs of multiple tool calls. It does not just retrieve data. It plans which data to retrieve, calls the relevant tools in sequence, synthesizes the results, identifies patterns and anomalies, and produces a recommendation grounded in your actual financial position, not in a training dataset.

What makes an agent require governance is exactly the same property. An agent that can plan, act, and adapt across connected systems in pursuit of a goal can also plan, act, and adapt in ways you did not explicitly authorize if the authorization framework does not clearly define the boundaries.
The capability and the governance risk are two sides of the same design.

A few things worth saying plainly before we go further.

Agents are not infallible. They can misinterpret ambiguous data, make errors in complex multi-step reasoning, or produce confident outputs that are factually wrong. The governance framework exists because of this, not despite it.

"Autonomous" does not mean what vendor decks suggest it means. In a finance context, "autonomous execution" is shorthand for "execution within a defined scope without requiring human approval for every individual action." The scope, the limits, and the exception handling remain entirely human responsibilities. An agent operating autonomously in treasury is not self-governing. It is operating inside a box that humans designed, approved, and are accountable for.

Agents are not magic, and they are not a revolution. They are a powerful tool: more capable than a spreadsheet, more flexible than RPA, more consistent than a manual process. Their quality is bounded by the quality of the data they access, the clarity of the authorization framework they operate within, and the rigor of the governance design around them.

REMEMBER: An AI agent does not replace financial decisions. It prepares them - faster and at greater scale than any human team can manage. The decision, the approval, the accountability: still human. That is not a limitation of the technology. It is a requirement of the governance framework, and it should stay that way.

The Reasoning Loop: How Agents Work Step-by-Step

The defining characteristic of an AI agent, compared to simpler automation tools, is that it operates in a reasoning loop. It does not execute a fixed sequence of steps. It plans, acts, observes the results, adapts, and continues until it has produced a complete output.

To make this concrete: a treasury team asks, “Where can we safely free up ten million dollars by Friday?”

A human analyst would understand the constraints, identify the data needed, retrieve it from the relevant systems, analyze it against policy, develop options, and present a recommendation.

An AI agent follows the same sequence - with the primary difference being that it executes the data retrieval and analysis steps across all entities simultaneously, in a fraction of the time.

STEP	WHAT THE AGENT DOES
Understand	Parses the request and extracts the governing constraints: the target amount, the deadline, and the implicit policy parameters - minimum balance requirements, investment policy, committed payment obligations. Identifies what it knows and what it needs to retrieve.
Plan	Determines the sequence of data retrieval and analysis steps needed to answer the question: entity list, current balances, seven-day cash forecasts, committed payments, available credit facilities, minimum balance policies by jurisdiction.
Select tools	Identifies the relevant tools from its authorized toolkit: `getBankBalances()`, `getCashForecast()`, `getCommittedPayments()`, `getCreditFacilities()`, `getMinimumBalancePolicies()`.
Call tools	Executes the data calls: balance for each entity as of today, forecast for each entity over the relevant horizon, pending payments report, available facility report. Each call is logged with a timestamp, the parameters passed, and the response received.
Analyze	Computes liquidity headroom per entity after accounting for minimum balances and committed payments. Applies FX conversion where required. Identifies entities with surplus liquidity above the safety threshold. Calculates the total safely available amount.
Develop options	Structures specific recommended actions: transfer amounts, source entities, destination accounts, FX conversions required, estimated execution timeline relative to cut-off schedules. Calculates whether the target amount is achievable and, if not, what the maximum safely available amount is.
Present with context	Produces a structured recommendation with the supporting data, the reasoning, the cut-off timing constraints, and a clear flag on any data quality uncertainties - for example, if one entity’s balance was last refreshed more than four hours ago.
Act (with approval)	If the treasurer approves the recommendation, the agent can execute the approved transfers directly via the TMS integration - subject to the segregation of duties controls, approval limits, and audit logging requirements defined in the authorization framework.

The same properties that make this loop powerful create specific failure modes that governance must address.

Data quality failures propagate invisibly.
If `getBankBalances()` returns a stale figure because of a bank connectivity issue, the agent reasons correctly from an incorrect input. The recommendation looks complete and well-supported. The error is in the data, not the reasoning, and it is easy to miss - especially under time pressure.

Tool call failures can be swallowed.
A poorly governed agent may proceed with a gap rather than flagging it clearly. The recommendation covers eleven entities confidently and mentions entity twelve in a footnote. The treasurer approves on the basis of the visible eleven.

Scope creep is a reasoning failure, not a technical one. An agent tasked with identifying liquidity sources may correctly determine that the most efficient solution involves an action it was not explicitly authorized to take - a facility drawdown, an FX conversion above the defined threshold. The agent did not malfunction. The authorization framework failed to define the boundary clearly enough.

This is precisely the failure mode in the quarter-end scenario in Chapter 1.

Hallucination is a language model characteristic, not an edge case. We address this in section 2.5. The short version: language models generate outputs that are linguistically fluent and contextually plausible regardless of whether they are factually correct. In finance, that distinction is not academic.

WARNING: Confidence is not accuracy. Language models are trained to produce fluent, coherent, confident-sounding outputs. An agent can be wrong with the same apparent certainty as when it is right. Every financial figure in an agent's recommendation must be traceable to a logged tool call - not to the agent's free-form reasoning.

What AI Agents Are Not: Three Things Finance Teams Confuse

Three comparisons come up in almost every conversation between finance teams and AI vendors. Each one sounds reasonable. Each one leads to a different governance mistake.

AGENTS ARE NOT RPA

Robotic Process Automation (RPA) executes a fixed, rule-based script. It does exactly what it was programmed to do, in exactly the sequence defined, under exactly the conditions anticipated. Reliable within its scope. Brittle outside it. When reality deviates from the script - a screen field moves, a file format changes, an exception arises that was not anticipated - RPA breaks, and the failure is visible.

AI agents reason and adapt. They can navigate variation, interpret ambiguous inputs, and find a path through a range of conditions that RPA cannot handle. That is a real capability advantage in complex, multi-entity treasury environments.

But adaptability introduces a risk that RPA does not carry. When RPA fails, something breaks - the failure is visible. When an AI agent adapts its approach in a direction that was not intended, the output may look correct. The agent reached the right-looking destination by a route that was not authorized, and nothing flagged it because nothing technically failed. Governance for AI agents must define not just what the agent is supposed to do - but what it is explicitly not permitted to do, enforced at the platform level, not described in a policy document that the agent cannot read.

AGENTS ARE NOT DECISION-SUPPORT DASHBOARDS

A dashboard displays data and does nothing further. The human interprets, forms a view, and decides whether to act. The dashboard has no agency - it cannot initiate, execute, or adapt. Its entire function is display.

An AI agent reasons over data, identifies patterns, develops conclusions, and - in execution modes - can take actions in connected systems. This is not a small difference. A dashboard showing a liquidity shortfall requires a human to decide what to do. An agent identifying a liquidity shortfall and configured with execution authority can initiate the transfer to address it, pending the approval workflow you have designed.

The governance implication is significant. With a dashboard, no action occurs without a conscious, deliberate human decision. With an agent, actions can occur as the result of a process the approving human may not fully decompose at the moment of approval - particularly if the approval interface presents the output as a single action rather than the culmination of a multi-step reasoning process. Governance must ensure that the human who approves an action understands what they are approving: the output, the data it was based on, the reasoning that produced it, and the specific action that will follow.

AGENTS ARE NOT GENERAL-PURPOSE AI CHATBOTS

A consumer AI chatbot generates text responses from a broad training dataset. It has no access to your financial data, no ability to take actions in your systems, no authorization framework, no evidence chain. It is a language model with an interface. Useful for drafting and summarizing.
Not a treasury tool.

A purpose-built finance AI agent is an architecturally different system. It accesses your actual financial data through governed tool connections. It operates within a documented authorization framework. It logs every data call, reasoning step, recommendation, and action. It enforces approval workflows before executing consequential actions.

The language model at the core may be the same class of technology as a consumer chatbot. The governance architecture, the data connectivity, the authorization framework, and the evidence chain are completely different - and they are why a finance AI agent can be deployed in a treasury environment where a consumer chatbot cannot.

TIP: When a vendor demonstrates an AI agent capability, ask: *"Show me what happens when the agent encounters a data quality issue, an out-of-policy scenario, or an ambiguous instruction."* If the answer is that the agent makes its best guess and proceeds, that is a governance gap. If the agent flags the exception, logs it, and escalates to a human, that is the right design. The demo always works. The exception handling reveals the governance maturity.

The Hallucination Problem in Finance

Language models are trained to predict the most likely next word given everything that came before it. That process makes them extraordinarily good at generating fluent, coherent, contextually appropriate text. It does not make them reliable producers of factual accuracy. The two properties are not the same thing, and language models have no internal mechanism that reliably distinguishes between them.

The result is hallucination: plausible-sounding output not grounded in accurate data. In a consumer context, this is an inconvenience - a restaurant recommendation for a place that does not exist. In a finance context, it is a controls failure.

The agent cites a cash balance figure that was not in the tool output. It references a policy parameter from its training data that has since been updated. It performs a calculation in natural language - "approximately €14.3 million, accounting for the FX effect" - using a wrong exchange rate, because the language model generated a plausible-sounding rate rather than retrieving the current one. The output is presented with exactly the same fluency and apparent confidence as an accurate output. The difference is not visible without checking the source.

Four design practices manage this risk in a governed finance agent:

Ground every financial figure in a logged tool output. Every number in an agent's recommendation must be traceable to a specific tool call - a specific data request, from a specific system, at a specific time, returning a specific value. If a figure cannot be traced to a logged tool output, it does not belong in a financial recommendation.

Surface the sources alongside the recommendation. The agent should not present a finished conclusion. It should present the recommendation alongside the data that underpins it - which entities were included, which balances were used, when they were last refreshed - so the human reviewer can assess input quality before acting on the output. This is not a concession to caution. It is the design requirement for a defensible recommendation.

Enforce human review proportional to financial consequence. The higher the consequence of a recommendation, the more important it is that a human with appropriate financial knowledge reviews it before action. Hallucination risk is one reason among several - but it is the reason that is easiest to overlook when outputs are consistently fluent and well-formatted.

Use a dedicated calculation tool, not the language model, for arithmetic. Language models are unreliable at arithmetic. This is documented, consistent, and not going away. A well-governed finance agent uses a dedicated calculation engine for all numerical operations, calls it as a logged tool, and attributes the result accordingly. The language model structures the analysis. The calculation engine does the math.

WARNING: *"The agent said $14.7 million"* is not an audit trail. *"The agent's recommendation was based on* `getBankBalances()` *output logged at 08:14 UTC, cross-referenced with* `getCashForecast()` *output for a seven-day horizon, with the net position calculation performed by the dedicated calculation tool and logged at 08:15 UTC"* - that is an audit trail. The difference between the two is the difference between visibility and provability.

The Four Operating Modes: Human-in-the-Loop Spectrum

Not every agentic finance workflow carries the same risk, involves the same financial exposure, or requires the same level of human involvement. One of the most important governance decisions in any agentic deployment is determining - for each specific workflow - how much autonomy the agent should have and where human review or approval is required.

Four operating modes define the spectrum:

1. MODE 1: HUMAN-ONLY.

No AI involvement. Appropriate for novel situations where context and judgment are paramount and cannot be codified - M&A financing decisions, covenant breach responses, credit facility restructuring negotiations, or any scenario where the combination of strategic, relational, and regulatory factors is unique enough that pattern-matching cannot be trusted.

2. MODE 2: HUMAN-LED WITH AI ASSISTANCE.

The agent drafts, recommends, surfaces patterns, and flags anomalies. The human validates, adjusts, and approves the final output. The agent accelerates the analytical work; the human owns the conclusion and the accountability. This is typically the right starting point for treasury teams new to agentic AI - and for use cases where financial stakes per decision are high, data is complex, or human context is consistently necessary. Cash forecast preparation, FX hedge recommendation review, and daily liquidity positioning analysis fit well here. The fallback decision, no link to the original approval, reconciliation is manual and error-prone.

3. MODE 3: AI-EXECUTED UNDER HUMAN SUPERVISION.

The agent runs the workflow and produces outputs. Humans supervise through exception queues, threshold-based review triggers, and periodic quality assessments - but do not review every output. The focus shifts from execution to oversight: the treasury team's job is to ensure the agent is performing within its defined parameters and to resolve the exceptions it escalates. Payment fraud and anomaly detection, bank reconciliation matching for high-volume transaction categories, and intercompany netting calculations are well-suited to Mode 3.

4. MODE 4: AI-AUTONOMOUS, FULLY AUDITABLE.

The agent executes end-to-end within strict, documented guardrails and comprehensive logging. Human attention is reserved for escalated exceptions and periodic quality reviews. "Autonomous" here never means invisible or unaccountable - it means the process is sufficiently stable and well-governed to run without constant human attention, while leaving a complete, transparent, retrievable audit trail for every decision made. Routine reconciliation matching for stable, rule-based transaction categories is an appropriate Mode 4 application, once Mode 3 operation has been validated.

FACTOR	MODE 2	MODE 3	MODE 4
Transaction/ decision frequency	Low–medium	High	Very high
Stakes per decision	High	Medium	Low–medium
Process stability	Variable	Mostly stable	Highly stable
Regulatory sensitivity	High	Medium	Low–medium
Process stability	Manual review log	Exception logs	Fully automated audit trail

The mapping of a specific workflow to a specific operating mode is a governance decision, not a technology decision. It should be made before the workflow is configured, documented in the authorization framework, and revisited as the pilot produces evidence about actual agent performance.

WARNING: The most common operating mode mistake is assuming a workflow is ready for Mode 4 before it has been validated through Mode 3. Autonomy must be earned through evidence of consistent, governed performance under supervised conditions - not assumed from day one. A workflow that moves to Mode 4 prematurely will encounter its first material exception without the supervised exception-handling experience needed to resolve it well.

These operating modes describe the level of delegated action and governance intensity in a workflow. They are not the only way to describe AI maturity. Other maturity models may focus on who initiates the interaction, for example whether the system is reactive, conversational, or anticipatory. In practice, both dimensions matter: how proactive the system is, and how much authority it has to act.

Minimal Glossary: Fourteen Terms You Will Encounter

The following terms appear consistently in vendor conversations, internal AI discussions, and technology evaluations. These definitions are written for finance practitioners - they prioritize governance relevance over technical precision.

TERM	WHAT IT MEANS IN FINANCE TERMS
Large Language Model (LLM)	The reasoning engine at the core of an AI agent. A model trained on vast amounts of text that can interpret questions, plan multi-step approaches, and generate language outputs. Think of it as a highly capable analyst who can read and reason quickly but needs governed tools and boundaries to act reliably on your specific financial data.
AI Agent	A system combining an LLM with tools that provide access to live data and the ability to take actions in connected systems, operating through a plan-act-observe reasoning loop. The combination of reasoning and action capability is what distinguishes an agent from a chatbot.
Tool / Tool Call	A structured, governed function that allows an agent to request specific data from a specific system or perform a specific action. Examples: *getBankBalances(), getCashForecast(), createPaymentInstruction()*. Tools are the agent's governed connection to your actual financial systems - and every tool call should be logged.
MCP (Model Context Protocol)	A standardized protocol for connecting AI agents to tools and data sources in a controlled, auditable way. In a treasury context, think of it as the governed channel through which the agent passes instructions to your financial systems and receives structured responses - analogous to the role SWIFT plays in payment messaging.
Reasoning Chain / Chain of Thought	The agent's step-by-step reasoning process, made visible and logged. The equivalent of an analyst showing their working. In a well-governed finance agent, the reasoning chain is a component of the evidence chain - reviewable, attributable, and retrievable on demand.
Hallucination	The generation of plausible-sounding but factually incorrect output by a language model. In finance, hallucination is a controls failure: a figure that was not in the data, a policy parameter that has changed, or an arithmetic result that was generated rather than calculated. Managed through tool grounding, source surfacing, and human review checkpoints.
Human-in-the-Loop	The governance design principle that consequential agentic actions require human authorization before execution. Not a technical feature - a policy decision about where in the workflow human judgment and accountability must be present. Implemented through approval workflows, exception escalation, and threshold-based review triggers.
Least-Privilege	The security principle that an agent can only access the data it is explicitly authorized to access and take only the actions it is explicitly authorized to take. The AI agent equivalent of role-based access control in your TMS: the agent's data access and action scope are defined, documented, and enforced at the platform level.

TERM	WHAT IT MEANS IN FINANCE TERMS
Segregation of Duties (SoD)	The governance requirement that no single agent - and no single human - can both initiate and approve a material financial action without a second control. SoD applies to AI agent workflows exactly as it applies to human workflows. An agent that can recommend and execute a consequential financial action without human intervention in the approval step has a SoD gap.
System Prompt	The governing instructions given to the language model that define its role, its scope, its behavior in specific scenarios, and its response to out-of-scope inputs. In a governed finance context, the system prompt is a policy document - it should be versioned, documented, reviewed by compliance, and treated as part of the authorization framework.
Context Window	The amount of information the language model can process at once. Relevant for treasury because very large datasets - full transaction histories, all entities' balance records - may exceed the context window, requiring smart data selection and tool design to ensure the agent is working from the right subset of data.
Agentic Workflow	A multi-step process in which an AI agent plans, executes, and adapts a sequence of actions across connected systems to complete a defined task. Distinguished from a simple automation workflow by the agent's ability to reason and adapt rather than follow a fixed script.
Audit Trail	In the agentic context: a complete, tamper-evident, retrievable log of every tool call made, every data source accessed, every reasoning step followed, every recommendation produced, every action taken, and every approval received. The audit trail is the foundation of provability - the difference between being able to see what happened and being able to prove it.
Orchestration Framework	The software layer that manages the agent's reasoning loop, coordinates tool calls, handles multi-agent workflows where multiple agents collaborate on a task, and provides the logging infrastructure for the audit trail. Examples include LangGraph and LangChain. The orchestration framework is the operational infrastructure of the agent - analogous to the operating system on which your TMS runs.

What Chapter 3 Demands of You

You now have a working foundation in what AI agents are, how they reason, and how they fail. That foundation is not the hard part.

The hard part is Chapter 3 - which asks you to translate this understanding into governance artifacts: authorization frameworks, evidence chains, segregation of duties designs, exception ownership matrices, and audit readiness tests. These are not new concepts. They are the governance principles you already apply to every material financial process, adapted to a system that reasons and adapts rather than follows a fixed script.

The adaptation is not trivial. An AI agent's failure modes are different from those of a TMS, an ERP, or an RPA workflow. The governance design must account for them explicitly - not by treating the agent as a black box to be managed from the outside, but by building the authorization framework, the evidence chain, and the exception protocols into the agent's design from the start.

If there is one thing worth carrying from this chapter into the next, it is this: the agents that get pulled from production are not usually the ones that failed technically. They are the ones that worked - in ways that could not be explained, defended, or attributed after the fact.

Visibility Is Not the Same as Provability

Here is the distinction that separates a defensible agentic finance deployment from one that fails its first audit review: visibility is not the same as provability.

A well-designed platform makes its outputs visible - dashboards, recommendation summaries, approval queues, execution confirmations. Visibility is necessary. It is not sufficient. Provability means you can demonstrate, to an auditor under adversarial conditions, that the agent was authorized to make that recommendation, that the data it used was accurate and appropriately scoped, that a human with the right authority reviewed and approved the action, and that when something unexpected happened, the exception was handled through a documented process and not improvised in the moment.

Most agentic finance programs are built to achieve visibility. The ones that survive audit are built to achieve provability.

This chapter is the difference between the two. Everything in it is operational. It produces governance artifacts - an Authorization Framework, an Exception Ownership Matrix, an evidence chain design, and three readiness checklists - that you can take directly into a pilot design, a vendor evaluation, a stakeholder briefing, or an audit preparation. The concepts are not new inventions. They are established treasury governance principles - authorization frameworks, segregation of duties, evidence chains, exception ownership - applied to a system that reasons and adapts rather than follows a fixed script.

Authorization Framework: What Is This Agent Permitted to Do?

The single most important governance artifact in any agentic finance deployment is the Authorization Framework - the document that answers, with precision, the question every auditor will ask: what was this agent explicitly authorized to do?

Not what could it technically do. Not what did the vendor configure it to do by default. What did your treasury leadership, legal team, compliance function, and IT security team explicitly review, approve, and document as the permitted scope of this agent's activity?

Most teams skip this document and go straight to configuration. That is the most predictable source of the kind of incident described in Chapter 1. The Authorization Framework must exist as a policy document before a single workflow is configured - not as a description of what was built after the fact.

The framework has three components, and all three are equally important.

Scope defines what the agent is authorized to access and do: which data sources it can read, which systems it can write to or initiate actions in, which transaction and entity types are covered, and which specific actions it can take autonomously versus which require human approval before execution.

Scope must be affirmative and explicit. *"The agent is authorized to call* `getBankBalances()` *for the following twelve entities"* - not implied by the absence of a prohibition.

Boundaries define what the agent is explicitly not permitted to do. This is as important as the scope definition, and it is more often overlooked. An authorization framework that defines scope but not explicit exclusions leaves the agent operating in a grey zone at its edges. The quarter-end scenario in Chapter 1 - the agent that extended its liquidity logic to include a facility drawdown never explicitly authorized - is a boundaries failure. Prohibitions must be documented, not just absent.

Policy parameters are the specific rules the agent must apply when operating within its defined scope: minimum balance requirements by entity and jurisdiction, investment policy constraints, payment approval thresholds by amount and transaction type, regulatory reporting triggers that require human review. These parameters must be versioned. When the underlying policy changes, the Authorization Framework must be updated and re-approved before the revised parameters go live.

The Authorization Framework is a policy document, not a technical configuration file. It should be authored by treasury leadership, reviewed by legal and compliance, approved in writing before the pilot begins, stored under version control, and available to auditors on request.

TIP: Think of the Authorization Framework as the agent's delegated authority matrix. Just as your treasury governance documents specify which humans are authorized to approve which categories of financial action up to which monetary limits, the Authorization Framework specifies which agent workflows are authorized to execute which actions under which conditions. Same rigor, same review process, same version control.

Authorization Matrix: Template

AGENT ACTION CATEGORY	SCOPE	MAX EXPOSURE	APPROVAL REQUIRED?	EVIDENCE REQUIRED
Read bank balances	All entities	N/A	No	Tool call log
Generate cash forecast	All entities	N/A	No (advisory)	Data source log + version
Recommend intercompany transfer	In-scope entities	Unlimited	Yes - Treasurer	Approval log with reason code
Execute intercompany transfer	Pre-approved entities	Under $1M	Yes - dual approval	Full audit trail
Flag payment anomaly	All payments	N/A	No	Detection log
Suspend flagged payment	N/A	N/A	Yes - escalation required	Incident log
Access facility drawdown	NEVER	$0	Out of scope	N/A

Human-in-the-Loop Design: Where Humans Must Be Present

Human-in-the-loop is not a simple on/off setting. It is a design decision that must be made for each specific category of agent action in each specific workflow, based on the materiality of the action, the financial exposure it carries, the regulatory requirements that apply, and the degree to which the process has been validated under controlled conditions.

Get this design wrong in either direction and you pay for it. Too much human involvement and the efficiency gains disappear - you have built an expensive advisory tool that creates more approval process than it eliminates. Too little, and you have created financial exposure and audit risk that surfaces at exactly the wrong moment.

Five categories of agent action require human approval by design.

BEFORE CONSEQUENTIAL EXECUTION

Any agent action that moves money, commits credit, changes the terms of a financial instrument, or creates a reportable event requires explicit human approval before it executes - regardless of whether the action falls within the agent's defined scope. "Within scope" means the action is permitted if approved, not permitted without approval. That distinction must be explicit in the Authorization Framework and enforced at the platform level, not described in a policy document the agent cannot read.

WHEN DATA QUALITY IS UNCERTAIN

If the agent encounters a stale balance, a missing entity, a forecast confidence score below the defined threshold, or a tool call returning an incomplete response, it must pause and escalate. It must not proceed on a best-guess interpolation. The escalation path for each data quality condition must be defined before go-live - because the agent will encounter these conditions in the first week of production, and how it handles them will set the governance standard for everything that follows.

WHEN THE AGENT ENCOUNTERS AN OUT-OF-SCOPE SCENARIO

If the agent's reasoning identifies an action that falls outside the Authorization Framework - a more efficient solution requiring access to an excluded entity, a liquidity option involving a facility marked as out of scope, a payment exceeding the defined threshold - it must stop, log the scenario clearly, and escalate. It must not reason around the boundary or execute and flag afterward. Out-of-scope scenarios escalate before action, not after.

BEFORE IRREVERSIBLE ACTIONS IN NOVEL CONTEXTS

Even within defined scope, first-time actions in new corridors, with new counterparties, in new jurisdictions, or involving transaction types that have not been executed through the agentic workflow before should carry a heightened review requirement. The agent's scope may cover the action in principle; the human judgment checkpoint ensures the first instance is reviewed with full attention before it establishes a precedent.

PERIODIC REVIEW OF AUTONOMOUS OUTPUTS

For workflows operating in Mode 3 or Mode 4, where not every output passes through an approval queue, a governance standard of periodic human review must be established and documented. A defined sampling frequency - weekly review of a percentage of agent outputs, monthly review of detection quality metrics, quarterly full-cycle review - provides the oversight mechanism that prevents gradual drift from going undetected.

REMEMBER: Human-in-the-loop is not a speed constraint. It is an accountability design. The goal is not to make the agent slower - it is to ensure that every consequential financial action has a named human owner who understood it, authorized it, and is accountable for it. That accountability is what makes the program defensible, not just efficient.

The Evidence Chain: From Data to Action in Five Linked Steps

The evidence chain is the governance concept that most directly determines whether an agentic finance deployment is audit-ready. It is the connected, tamper-evident record that links every agent action back to its authorization, its data inputs, its reasoning, its recommendation, and its human approval - in a form that can be retrieved on demand and interrogated by an auditor.

For a manual treasury process, that chain looks like: analyst retrieves data from the TMS, prepares analysis in a spreadsheet, emails the recommendation to the treasurer, treasurer approves via the workflow system, payment is executed and confirmed, confirmation is filed. Each link is a document, a log entry, or a system record that connects to the next.

For an agentic process, the chain is identical in structure and more complex in implementation. Every link must be designed and logged deliberately - because the agent's actions happen faster and at greater scale than any manual process, and because the agent's reasoning does not create a visible paper trail by default.

LINK 1: DATA SOURCE EVIDENCE

Answers the question: what data did the agent use, and was it accurate and appropriately scoped?

A logged record of every tool call the agent made: which tool was called, what parameters were passed, what response was received, and when. Data source metadata specifying the entity, account, or system the data came from and the timestamp of the most recent refresh. Evidence that data access was within scope - that the agent accessed only the entities and data types it was authorized to access.

This evidence lives in the tool call logs maintained by the orchestration framework and, for integration-level access records, in the connected TMS and ERP systems.

LINK 2: REASONING EVIDENCE

Answers the question: what reasoning process did the agent follow, and was it consistent with the Authorization Framework?

A logged record of the agent's reasoning chain - the step-by-step thinking process from data inputs to recommendation. In a well-governed platform, this is visible and logged, not a black box. It includes uncertainty flags the agent raised, data quality conditions it detected, and any scenarios it identified as outside its defined scope.

This evidence lives in the reasoning chain log maintained by the orchestration framework, alongside the policy parameter version in effect at the time of the recommendation.

LINK 3: RECOMMENDATION EVIDENCE

Answers the question: what exactly did the agent recommend, and was it presented to a human before any action was taken?

A structured log of the recommendation output - the specific action proposed, the amounts, entities, and counterparties involved, the data sources cited, and the calculations grounded in tool outputs. Evidence that the recommendation was surfaced in the human review interface in a way that gave the approver the information needed to make an informed decision. A version record at the point of presentation - so that if the recommendation was modified before action, the modification is documented.

LINK 4: APPROVAL EVIDENCE

Answers the question: who authorized this action, when, under what authority, and through what process?

A logged approval record capturing the approver's user identity, the timestamp, the specific recommendation approved, and the reason code documenting the basis for approval.

Evidence that the approver was authorized under the Authorization Framework to approve this category of action at this financial exposure level. For dual-approval requirements: both records, both timestamps, both authorizer identities.

LINK 5: ACTION EVIDENCE

Answers the question: what did the agent do, and did the action match the approved recommendation?

The execution record in the TMS: transaction reference, timestamp, amount, counterparty, account details, executing agent identity. The bank confirmation or statement entry for payment and transfer actions. The ERP journal entry for accounting-relevant actions. The action evidence must be directly traceable to the approved recommendation - amounts, counterparties, and transaction types must match precisely, and any discrepancy must be logged and explained.

TIP: Build a transaction reference table that connects every agentic action to its complete evidence chain: Request ID, data sources and refresh timestamps, reasoning chain version, recommendation log reference, approver identity and timestamp, TMS transaction reference, and execution confirmation. This table is the single source of truth for audit. It does not replace the underlying logs - it maps to them.

Segregation of Duties: Does It Apply to AI Agents?

Yes. Unambiguously and without exception.

Segregation of duties - the requirement that no single individual or system can both initiate and approve a material financial action - is one of the foundational controls in treasury governance. It exists because combining initiation and approval authority in a single actor removes the independent check designed to catch errors, unauthorized actions, and fraud.

AI agents do not create an exemption. An agent that can both recommend a financial action and execute it, without an independent human approval step, has eliminated the segregation the control requires. The fact that the agent is software rather than a human employee does not change the governance principle.

Four design patterns implement SoD correctly in agentic workflows.

Separate recommendation and execution. The agent's recommendation function and the execution function are distinct steps, separated by a human approval checkpoint. The agent produces the recommendation; a human with documented approval authority reviews and approves it; a separate execution step carries out the approved action. The agent never completes both halves of the cycle without human intervention.

Tier automation with documented approval thresholds. For high-frequency, lower-stakes workflows - reconciliation matching, anomaly flagging, intercompany netting below defined thresholds - the agent can execute autonomously within clearly documented parameters. Above those parameters, human approval is required. Thresholds are defined in the Authorization Framework, enforced at the platform level, and reviewed periodically.

Require dual approval for material and novel actions. For actions above a defined monetary threshold, involving new counterparties or jurisdictions, or crossing regulatory reporting triggers, require two independent human approvals - even if the agent's recommendation is being accepted without modification. This mirrors the standard payment control in which the person who creates a payment instruction is not the only person who approves it.

Separate the creator from the sole approver. The person who configures an agent task - who frames the question, sets the parameters, and triggers the workflow - should not be the sole approver of the action the agent recommends. Configuring an agentic workflow is the functional equivalent of creating a payment instruction. The approval must be independent.

WARNING: If your AI agent platform allows a single user to configure an agent task, receive the recommendation, and approve the execution without a secondary control, you have a segregation of duties gap. This is a governance policy failure, not a technology limitation - and it must be addressed in your Authorization Framework and enforced through your approval workflow design before the pilot goes live.

Exception Ownership: Who Decides What When the Agent Cannot?

In production, AI agents encounter exceptions. Not occasionally - routinely. Bank feeds drop. ERP syncs run late. Inputs fall outside the range the agent was calibrated for. Tool calls return errors. Scenarios arise the authorization framework did not explicitly anticipate.

If exception ownership is not defined before go-live, it will be defined under pressure - when the exception is a live operational situation and the team is working out the governance in real time. That is avoidable, and it should be avoided.

Exception ownership means documenting four things for every category of exception the agent might encounter: who owns the decision, what actions are pre-authorized, what requires formal escalation, and how the decision and outcome are documented.

The goal is not to anticipate every possible exception in exhaustive detail. It is to ensure that every exception has a named owner and a defined response pathway - so that the team's response is a governed process, not an improvisation.

EXCEPTION OWNERSHIP MATRIX: TEMPLATE

EXCEPTION TYPE	FIRST RESPONSE	ESCALATION PATH	PRE-AUTHORIZED ACTIONS	DOCUMENTATION REQUIRED
Data stale over 4 hours	Treasury Ops	Treasury Manager if over 8 hours	Pause recommendation; trigger data refresh	Ticket + data log
Recommendation outside Authorization Framework	Agent auto-escalates	Treasury Manager	Do not execute; document scope gap	Incident report
System or tool call failure	Treasury Ops	IT + Treasury Manager	Pause workflow; activate manual fallback	Ticket + error log
Anomaly flagged but below action threshold	Treasury Ops	Treasury Manager if persistent	Log and monitor; no action required	Detection log
Action involves novel counterparty or amount	Agent flags for human	Treasury Manager	Do not auto-execute; require human review	Authorization log
Duplicate action risk detected	Treasury Ops	Treasury Manager + IT	Halt second action; verify and reconcile	Incident report

One exception type warrants specific attention: the scenario where both the agent and a human take the same action in parallel - most likely in the early stages of supervised execution when the team has not fully transitioned from the manual process. Duplicate action risk should be explicitly identified, with a defined detection mechanism and a clear resolution process, before supervised execution begins.

What Your Auditor Will Actually Ask

Treasury teams that have designed the governance framework correctly should be able to answer an auditor's questions about their agentic finance program without preparation. The evidence chain should be complete, the Authorization Framework should be documented and version-controlled, and the exception handling record should reflect a governed process.

In practice, five questions define the core of any agentic finance audit review. The answers should be demonstrably available before the pilot advances past Phase 1.

Question 1: "How do you ensure that AI agent recommendations are reviewed by authorized individuals before execution?"

What the auditor is testing: whether the Authorization Framework is real and enforced, whether human-in-the-loop checkpoints are designed and operating, whether SoD is maintained. What you need: the Authorization Framework with defined approval requirements per action category; approval logs showing user IDs, timestamps, and action references; evidence that the approver's authority level matches the financial exposure of the approved action.

Question 2: "How do you know that the data the agent used was accurate and appropriately scoped?

What the auditor is testing: data quality controls, data access governance, Links 1 and 2 of the evidence chain.

What you need: tool call logs with data source citations and refresh timestamps; data lineage documentation for a specific recommendation; evidence of data access scope controls.

Question 3: "How do you prevent and detect errors or inaccuracies in AI-generated financial outputs?"

What the auditor is testing: model output governance, grounding controls, human review checkpoints.

What you need: evidence that financial figures in agent recommendations are grounded in logged tool outputs, not generated as free-form text; documentation of the calculation tool used for numerical operations; human review checkpoint records for high-value recommendations; an incident log for any detected inaccuracies.

Question 4: "What happens when the agent encounters an out-of-scope scenario or an instruction it was not designed to handle?"

What the auditor is testing: boundary enforcement, exception handling design, escalation controls

What you need: the Exception Ownership Matrix; escalation logs showing out-of-scope scenarios were escalated before action, not after; evidence that the exception handling process operated as designed.

Question 5: "What are the third-party risks associated with your AI platform, and how are they managed?"

What the auditor is testing: vendor risk management, data privacy, business continuity.

What you need: vendor due diligence documentation covering data hosting location, training data policy (specifically whether your financial data is used to train shared models), security certifications, and audit access provisions; contract provisions covering SLAs, data residency requirements, and exit rights; a business continuity plan for platform unavailability.

REMEMBER: Auditors are not expecting agentic finance to be a zero-risk environment. They are expecting evidence of control - documentation that the risks were identified, that governance was designed to address them, and that the governance operated as designed during the period under review. Show a complete evidence chain for a sample of agentic actions, including the exceptions and how they were handled, and you will satisfy the audit. If you cannot, the gap is in the governance design, not the technology.

Governance Readiness: Three Checklists

The following checklists translate the governance framework in this chapter into operational readiness tests. Use them before the pilot begins, at the end of Phase 1, and before going live in production.

CHECKLIST 1: PRE-PILOT GOVERNANCE READINESS

- ☐ Authorization Framework is documented, version-controlled, and approved in writing by treasury leadership, legal, and compliance
- ☐ Operating mode (1 through 4) is defined for each agent workflow, with human approval checkpoints specified for each
- ☐ Data source access is scoped and documented: which data, which entities, which systems, refresh frequency requirements
- ☐ SoD controls are enforced at the platform level: creator and sole approver roles are separated for material actions
- ☐ Approval thresholds are defined and align with the existing treasury delegation of authority matrix
- ☐ Exception Ownership Matrix is documented, ownership is assigned by role, and the matrix has been communicated to the team
- ☐ Evidence chain logging is active and tested: tool call logs, reasoning chain logs, recommendation logs, approval logs, and action logs are all generating records
- ☐ Escalation paths are defined and documented for all exception categories in the Exception Ownership Matrix
- ☐ Vendor due diligence is complete: data hosting, training data policy, security certifications, SLAs, and audit access provisions reviewed
- ☐ Legal and compliance have reviewed and approved the pilot design in writing

CHECKLIST 2: EXCEPTION-HANDLING READINESS

- ☐ Every exception category in the Exception Ownership Matrix has a named role owner (not just a named individual)
- ☐ Pre-authorized actions are defined for each exception category
- ☐ The escalation process has been tested: a simulated data quality failure and a simulated out-of-scope scenario have been run and documented
- ☐ Exception decisions are logged in a retrievable system: ticketing, TMS notes, or incident log
- ☐ Duplicate action risk has been assessed and a detection mechanism is in place for the supervised execution phase
- ☐ The team understands the difference between a data quality exception (pause and escalate) and a system failure exception (activate manual fallback)

CHECKLIST 3: EVIDENCE-PACK READINESS (THE AUDIT TEST)

Select one transaction from the pilot period and verify you can produce the following on demand, without preparation:

- ☐ Which data sources did the agent use, and when were they last refreshed?
- ☐ What reasoning steps did the agent follow to arrive at the recommendation?
- ☐ What exactly did the agent recommend - specific amounts, entities, transaction types?
- ☐ Who approved the action, at what time, and under what documented authority?
- ☐ What action was executed, and does it match the approved recommendation precisely?
- ☐ Were any exceptions encountered during this action, and how were they resolved?
- ☐ Is there a complete, retrievable log connecting the request to the data, to the reasoning, to the recommendation, to the approval, to the execution?

If any of these questions cannot be answered with documentary evidence for a sample transaction, the evidence chain has a gap that must be resolved before the pilot advances to the next phase.

Chapter 4

Six Agentic Finance Use Cases

Why the Use Case You Choose Matters More Than the Technology

The most common mistake in an agentic finance evaluation is selecting the use case that looks most impressive in the demo rather than the one the organization is actually ready to govern well.

The agent reasoning layer is becoming increasingly commoditized. What is not commoditized is the data infrastructure, connectivity, control model, and orchestration framework around it. In finance, the differentiator is not whether an agent can generate a plausible recommendation. It is whether that recommendation is grounded in governed data, executed through controlled systems, and evidenced in a way treasury, audit, and compliance teams can defend.

Two variables drive how a use case should be evaluated and governed.

DECISION FREQUENCY AND FINANCIAL EXPOSURE PER ACTION TOGETHER DETERMINE GOVERNANCE INTENSITY

A daily liquidity optimization that involves executing intercompany transfers has a fundamentally different risk profile from a reconciliation agent processing high-volume, low-value matching decisions. High-frequency, lower-stakes, rule-based processes can progress to supervised or autonomous execution relatively quickly once the authorization framework is in place. Low-frequency decisions with material financial consequences per action stay in Mode 2 - human approval for every output - regardless of how well the technology performs.

DATA QUALITY BASELINE IS THE MOST RELIABLE PREDICTOR OF PILOT SUCCESS OR DIFFICULTY

Treasury teams consistently underestimate this. An agent is a multiplier of your data infrastructure: clean, well-connected data produces fast, accurate outputs. Fragmented, inconsistently structured data from multiple entities and banking platforms produces confident-looking outputs that are wrong in ways that are not immediately visible. Know your data quality before you commit to a use case - because a pilot dominated by data remediation is not a governance validation, and it will not produce the evidence you need to move to production.

Each of the six use case cards that follow covers: what the agent does, step by step; why a treasury team would consider it; the trade-offs that most teams underestimate; the specific governance priorities that apply; and which organizations are best positioned to start with it.

REMEMBER: No single use case is the right starting point for every organization. Choose the one that fits the organization you have - not the one that looks most impressive in a vendor demonstration.

Use Case 1: AI-Assisted Cash Forecasting

Operating Mode: Mode 2 - Human-led with AI assistance

VALUE PROPOSITION

Cash forecasting is, by a significant margin, the most common starting point for agentic finance. The reasons are straightforward: the process is data-intensive, time-consuming, and consistently subject to human bias - collections tend to be forecasted optimistically, disbursements conservatively, and the aggregation of multi-entity, multi-bank data consumes hours that treasury teams rarely have.

An AI agent can pull balances and historical cash flow data from all connected entities simultaneously, identify seasonal patterns and day-of-week effects, cross-reference the collections pipeline from the ERP, and produce a first-draft forecast with variance explanations and confidence indicators in a fraction of the time the manual process requires.

This use case sits firmly in Mode 2. The agent produces the draft and explains its reasoning. The treasury team reviews, challenges assumptions, applies business context the agent cannot access - a pending acquisition, a customer relationship under renegotiation, a subsidiary facing covenant pressure - and owns the final output. The speed benefit is real. The governance requirement is straightforward. And the evidence chain is relatively simple to design because the agent's outputs are advisory throughout. For organizations new to agentic finance, this is where to start.

How It Works

1. Forecast cycle is triggered by the treasury team or a defined scheduler
2. Agent retrieves bank balances for all entities as of the current date
3. Agent retrieves historical cash flow data for the trailing 90 days per entity
4. Agent retrieves committed payments register for the forecast horizon
5. Agent retrieves the AR collections pipeline from the ERP
6. Agent analyzes patterns: day-of-week effects, seasonal trends, variance from the prior forecast cycle
7. Agent produces a draft forecast with variance explanations, confidence indicators, and flags on any data quality issues detected - stale balances, missing entities, anomalies in collections data
8. Treasury team reviews the draft, validates the drivers, applies business context, and adjusts
9. Team approves the final forecast and presents to leadership
10. Full evidence chain logged: data sources with refresh timestamps, agent reasoning, human adjustments, final approved version

TRADE-OFFS

Data quality is the primary constraint, not the technology. An agent aggregating from twelve banking relationships and three ERP instances inherits every gap and inconsistency in that data infrastructure. If bank connectivity is patchy or ERP data refreshes on a four-hour lag, the agent's forecast will be faster than the manual process - and subject to the same data quality limitations. There is also a subtler risk: a forecast the treasurer cannot explain to the CFO is worse than no forecast. Explainability must be designed in deliberately, and the human review step must be substantive, not a rubber stamp.

PILOT FIT

Best suited to treasury teams managing five or more entities across multiple banking relationships, where the manual forecast cycle currently consumes four or more hours per cycle. Not the right starting point for organizations with significant data quality gaps that have not yet been addressed - the agent will surface those gaps quickly, but a pilot dominated by data remediation is not a governance validation.

GOVERNANCE FOCUS

The evidence chain for this use case has one specific requirement that teams miss: the version of the agent's draft and the version the human team approved must both be logged, with the differences and rationale documented. When the human team adjusts the agent's draft - which they will, routinely - that adjustment and the reasoning behind it is part of the audit record. A system that logs only the final approved output does not capture the human judgment that made the forecast defensible.

Data source logging requires equal care. Every forecast must be traceable to specific sources with specific refresh timestamps. Stale data must be flagged visibly in the output, not incorporated silently - because the reviewer needs to know what they are approving.

Use Case 2: Liquidity Positioning and Cash Optimization

Operating Mode: Mode 2 to Mode 3 - Human-led with AI assistance, transitioning to AI-executed under supervision for routine transfers within policy

VALUE PROPOSITION

Liquidity positioning is treasury's most time-sensitive daily process. Identifying which entities have surplus liquidity, which have shortfalls, the most efficient transfer routes given cut-off times and FX effects, and what yield opportunities are being left unrealized - all of this requires aggregating data from every entity simultaneously and analyzing it against a complex set of policy constraints. Done manually, it consumes the first hours of every treasury morning.

The transition from Mode 2 to Mode 3 is what makes this use case particularly valuable over time.

In Mode 2, the agent produces recommendations and the treasurer approves each transfer individually.

As governance matures and the agent demonstrates consistent performance within defined parameters, routine transfers below defined thresholds can move to supervised execution - the agent initiates the transfer, the approval workflow confirms it, and human attention concentrates on exceptions and decisions above threshold. That transition is what creates the sustained operational benefit beyond the initial time saving.

How It Works

1. Agent runs at a defined cadence - typically early morning - or on demand
2. Agent retrieves bank balances for all entities, seven-day cash forecasts, committed payments, minimum balance policies, available credit facilities, and current FX rates
3. Agent computes liquidity headroom per entity after minimum balance requirements and committed payments
4. Agent identifies optimal transfer routes, accounting for cut-off times, FX conversion costs, and intercompany netting opportunities
5. Agent calculates available yield on surplus balances against investment policy parameters
6. Agent generates specific recommended actions with full supporting rationale, flagging any entity with data quality uncertainty and any recommendation near a policy threshold
7. Treasurer reviews and approves in-scope actions through the approval workflow
8. Approved actions execute in the TMS with a complete audit trail
9. Agent generates an end-of-day position confirmation

TRADE-OFFS

The financial stakes per action are higher here than in cash forecasting, which means the authorization framework must be precise and the approval workflow must be robust. The transition from Mode 2 to Mode 3 should be earned through demonstrated recommendation accuracy under Mode 2 conditions - not assumed. The facility drawdown scenario in Chapter 1 is a liquidity optimization failure: the boundary between what the agent could recommend and what it could execute was not explicit enough. That boundary must be airtight before Mode 3 operation begins.

FX effects introduce additional complexity: the agent's optimization logic must correctly apply hedging policy and FX conversion costs, which requires policy parameters to be clearly defined, current, and version-controlled.

PILOT FIT

Best suited to treasury teams with strong bank connectivity and a daily positioning process currently consuming two or more hours. The Mode 2 phase should run for at least six weeks before any transition to supervised execution is considered, and the transition should be gate-controlled: demonstrated recommendation accuracy, complete evidence chain, zero out-of-scope incidents.

GOVERNANCE FOCUS

Authorization framework precision is the non-negotiable here - specifically the explicit prohibition on actions the agent is not permitted to take regardless of how efficient they appear in its reasoning. The authorization matrix for this use case should explicitly name facility drawdowns, actions involving excluded entities, and FX conversions above defined thresholds as out of scope, not merely absent from the permitted actions list.

The approval workflow design for the Mode 3 transition deserves specific attention. The approval interface must present enough context - the transfer amounts, the reasoning, the data sources used, the policy parameters applied - for the approver to make an informed decision quickly. A workflow that presents only a confirmation button is not a human-in-the-loop checkpoint. It is the appearance of one.

Use Case 3: Payment Fraud and Anomaly Detection

Operating Mode: Mode 3 - AI-executed under human supervision

VALUE PROPOSITION

Payment fraud and anomaly detection is among the clearest Mode 3 use cases in agentic finance - high-volume, rule-based monitoring that cannot feasibly be done at scale by human reviewers, with exceptions that genuinely require human judgment to resolve.

The manual alternative creates the worst combination: high cost, incomplete coverage, and reviewer fatigue that degrades detection quality over time. An agent monitoring the payment queue continuously, applying detection rules consistently, and routing only flagged items to a human review queue achieves something the manual process cannot: complete coverage without proportional cost. The treasury team stops reviewing everything and reviews what matters.

This is also the use case where a governance gap has the most visible downside. A detection agent that auto-clears a fraudulent payment because the beneficiary details changed 73 hours before the payment instruction - one hour outside the 72-hour flag window - is a controls failure, not a technology failure. The detection rules are the governance. They need to be precise, maintained, and tested.

How It Works

1. Agent monitors the payment queue in real time or at defined intervals
2. For each payment, agent checks against: beneficiary whitelist and blacklist, historical payment patterns by entity and counterparty, banking detail change logs, policy rules including amount thresholds and dual-approval requirements
3. Agent flags anomalies for human review: new beneficiary not on the approved list, banking detail change within 72 hours of the payment instruction, amount more than twice the historical average for this counterparty, payment to a jurisdiction on the monitoring list, duplicate payment indicator
4. Flagged payments enter the human review queue with the full detection rationale visible
5. Human reviewer clears with a logged reason, escalates to compliance, or blocks with an incident log
6. Clean payments proceed without manual intervention
7. Detection model outcomes are tracked and reviewed periodically - false positive rate, missed detections, drift indicators
8. Human review of detection quality on a defined cadence

TRADE-OFFS

The detection model is only as good as its reference data. A beneficiary whitelist that has not been actively maintained, historical payment pattern data that is thin for newer entities, or banking detail change logs that are incomplete will reduce detection accuracy in ways that are not visible in the metrics until a fraud event occurs. The feedback loop between human reviewer decisions and detection quality must be governed - informal learning that is not version-controlled creates an audit problem. And false positive management matters: a detection rate that generates more review queue items than the team can handle within payment cut-off windows defeats the purpose.

PILOT FIT

Best suited to organizations with high payment volumes, multiple entities, and known exposure to business email compromise or beneficiary fraud. The value scales directly with volume - a team processing 200 payments a week will see a different return than one processing 2,000. Particularly valuable where the current manual review process creates cut-off timing pressure: when reviewers are clearing payments under deadline rather than on merit, detection quality is already degraded and an agent-driven exception queue is a governance improvement, not just an efficiency one.

Not the right starting point for organizations whose beneficiary whitelist and payment history data has not been actively maintained - clean the reference data first, or the agent's detection accuracy will disappoint and the false positive rate will undermine adoption before the governance benefits are visible.

GOVERNANCE FOCUS

Reference data governance is the most frequently underestimated requirement here. The whitelist, blacklist, and historical pattern data are not set-and-forget assets - they require an owner, a maintenance cadence, and a change control process. When a new supplier is onboarded, the whitelist must be updated before the first payment runs. When a supplier changes banking details, the change must be logged before the existing payment queue is processed.

Reviewer decision logging requires the same rigor as any other audit trail. Every human decision in the review queue - cleared, escalated, blocked - must be logged with a reason code, not just an action. "Cleared" is not an audit record. "Cleared - confirmed with AP team, banking detail change pre-authorized by supplier relationship manager on [date]" is an audit record.

WARNING: Do not deploy payment anomaly detection in Mode 3 without testing the exception handling process first. The first live flagged payment - particularly if it involves a critical supplier or a time-sensitive transaction - will test the review queue design, the escalation path, and the communication process with the initiating team. Run the exception scenario in simulation before it arrives under production pressure.

Use Case 4:
FX Exposure Identification and Hedge Recommendation

Operating Mode: Mode 2 - Human-led with AI assistance

VALUE PROPOSITION

FX exposure management sits at the intersection of two well-known treasury pain points: data fragmentation and timing. Exposure is scattered across AR and AP balances in non-functional currencies, open purchase orders, customer contracts, intercompany loans, and the existing hedge portfolio - rarely visible in one place, rarely aggregated in real time.

By the time a treasury team has manually consolidated the picture, hedging windows have narrowed, rates have moved, and the decision is being made on information that was accurate yesterday.

An AI agent can aggregate exposure data from every connected source simultaneously, calculate net exposure by currency pair and maturity band, compare it against hedging policy thresholds and existing hedge coverage, and generate specific hedge recommendations - complete with indicative pricing from connected trading platforms - in the time it would take a human analyst to access the first of four systems. The treasurer reviews a complete, policy-consistent picture rather than building one.

This use case stays in Mode 2. The human judgment layer is not optional here: the agent cannot know about pipeline deals, strategic hedging overlays, or relationship-driven considerations that routinely affect hedging decisions. The value is a better-prepared, faster-assembled analysis - not a decision the agent makes.

How It Works

1. Agent aggregates FX exposure from: ERP AR and AP balances in non-functional currencies, existing hedge portfolio from the TMS, open orders from the order management system, intercompany loan balances, and contract data where accessible
2. Agent calculates net exposure by currency pair, entity, and maturity band
3. Agent compares net exposure against hedging policy thresholds and existing hedge coverage ratios
4. Agent identifies hedge gaps, over-hedges, and positions approaching expiry
5. Agent requests indicative FX quotes from connected trading platforms
6. Agent generates hedge recommendations: instrument type, tenor, notional amount, and estimated cost, ranked by gap priority
7. Treasury team reviews: validates exposure inputs against known business context, assesses current market conditions, adjusts recommendations for strategic considerations the agent cannot access
8. Human approves selected hedging actions
9. Executed trades are logged with the full evidence chain

TRADE-OFFS

The quality of the exposure picture is determined entirely by the completeness of the data sources connected to the agent. An agent that cannot access the order management system, or that receives contract data in unstructured formats it cannot reliably parse, will produce an incomplete exposure picture - and an incomplete exposure picture that looks complete is more dangerous than one that is visibly incomplete. The human review step is not optional in FX management; it is where the agent's analysis becomes a decision.

PILOT FIT

Best suited to treasury teams managing multi-currency exposure across multiple entities, where the manual exposure consolidation process currently takes more than half a day per cycle. The value is proportional to the number of data sources connected - a pilot with only ERP data and no order management or contract data will show limited value and may be misleading about the full-scale capability.

GOVERNANCE FOCUS

Data source completeness requires explicit documentation in the authorization framework: which exposure sources are connected, which are not, and what the implication is for the completeness of the exposure picture. The gap between connected and unconnected sources should be visible in every recommendation output - not assumed away. A recommendation that covers 70% of the exposure picture because three data sources are not connected should say so.

Policy parameter currency is the other critical requirement. The hedging policy parameters the agent applies must be the current approved version, and the version in effect at the time of each recommendation must be logged. FX policies change - following board reviews, rating agency guidance, or covenant renegotiations - and an agent applying an outdated policy parameter will produce technically correct but policy-non-compliant recommendations without any visible flag.

Use Case 5: Intercompany Settlement and Netting

Operating Mode: Mode 3 transitioning to Mode 4 - AI-executed under supervision, progressing to AI-autonomous for stable, rule-compliant scenarios

VALUE PROPOSITION

Intercompany settlement and netting is the treasury use case that most naturally progresses to Mode 4 operation - but also the one where premature progression creates the most expensive problems. The process is rules-based: apply the netting rules, calculate the net positions, generate the settlement instructions, execute approved settlements, and reconcile. In a well-governed intercompany netting cycle, there is no judgment call - only the correct application of documented rules to verified data.

What makes the manual version painful is volume and coordination: aggregating intercompany balances from multiple ERP instances, applying bilateral or multilateral netting rules correctly across multiple currencies and jurisdictions, generating settlement instructions that satisfy all parties, and executing on the netting cycle deadline without error. An AI agent can execute this entire sequence consistently, at scale, with a complete audit trail for every calculation.

How It Works

1. Agent retrieves intercompany balances from ERP for all entities in the netting scope
2. Agent applies the defined netting rules: bilateral or multilateral, by currency, at the defined netting frequency
3. Agent calculates net settlement amounts per netting pair or hub
4. Agent checks all settlements against authorization limits, flags any regulatory constraints in the relevant jurisdictions, and identifies FX conversions required
5. Agent presents the settlement schedule to the treasury manager for approval, with the full calculation derivation available for review
6. Approved settlements execute: intercompany journal entries are generated and payment instructions are sent to the TMS
7. Full audit trail logged: opening balances, netting calculation, approval, execution, and reconciliation confirmation
8. Agent generates a settlement confirmation report for all entities and the accounting team

TRADE-OFFS

The netting rules must be precisely documented and version-controlled - the agent applies the rules as defined, and if the rules are ambiguous or outdated, the outputs will reflect that accurately and at scale. This is the Mode 4 risk in concentrated form: an error in a manual netting cycle affects one cycle.

An error in an autonomous netting agent affects every cycle until someone notices.

Jurisdiction-specific regulatory constraints require active maintenance. New restrictions or reporting requirements must be incorporated into the agent's rule set promptly - and the process for doing so must be documented in the governance framework before Mode 4 operation begins.

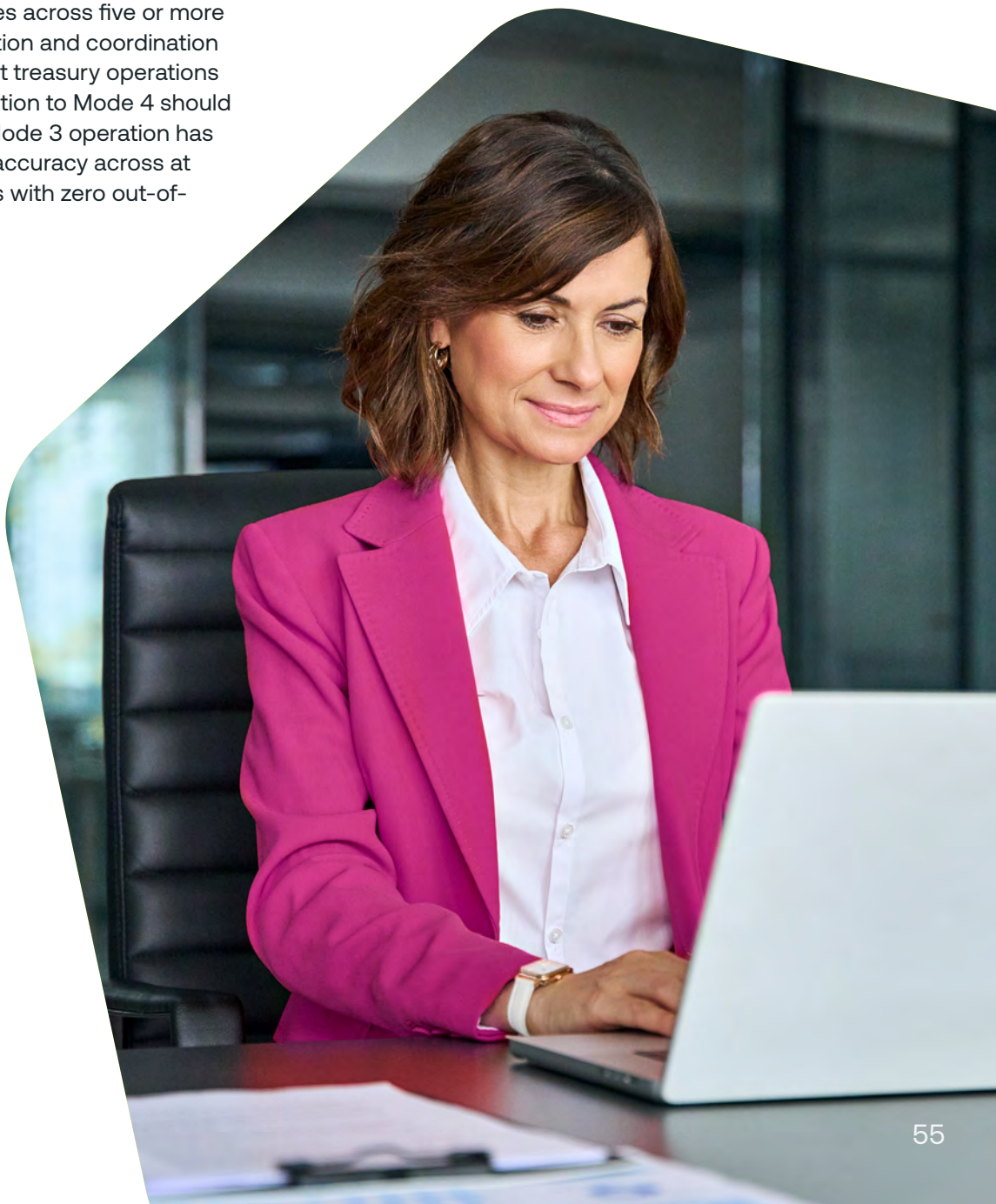
PILOT FIT

Best suited to treasury teams running manual intercompany netting cycles across five or more entities, where the calculation and coordination effort consumes significant treasury operations time each cycle. The transition to Mode 4 should only be considered after Mode 3 operation has demonstrated consistent accuracy across at least two full netting cycles with zero out-of-tolerance incidents.

GOVERNANCE FOCUS

Netting rule version control is the governing priority. The rule set the agent applies must be the current approved version, changes must go through a documented change control process with sign-off from treasury and tax, and the specific version in effect for each cycle must be logged and retrievable. This is not optional for Mode 4 - without it, you cannot demonstrate that the agent applied the correct rules to the correct data in the period under audit review.

Exception handling for out-of-tolerance items - settlements that fall outside the normal parameters - requires a defined human review process. The Mode 4 authorization framework must specify exactly what constitutes an out-of-tolerance item, what the agent does when it detects one (pause and escalate, not resolve autonomously), and who the named owner is.



Use Case 6: Reconciliation and Exception Management

Operating Mode: Mode 4 - AI-autonomous, but fully auditable

VALUE PROPOSITION

Bank reconciliation is the quintessential Mode 4 use case. The process is high-volume, rule-based, repetitive, and binary: each transaction either matches or it does not. The matching rules are documentable and consistently applicable. The exceptions are definable. Everything about this process is designed for autonomous execution with comprehensive audit trails.

The value is not just efficiency, though the efficiency is significant. It is consistency: an agent applies matching rules identically to every transaction without the fatigue effects that human reviewers develop in high-volume processes. It is also coverage: exceptions that a reviewer might miss after the two-hundredth transaction are caught by an agent on the two-thousandth. For treasury teams where reconciliation is currently a manual process consuming days of operations capacity at month-end, this is also the lowest-governance-risk entry point to Mode 4 operation - making it valuable not just for reconciliation itself but as the organizational learning ground for autonomous execution governance.

How It Works

1. Agent retrieves bank statements from all connected accounts via SWIFT, API, or file import
2. Agent retrieves the expected transaction list from the TMS and ERP for the matching period
3. Agent applies matching rules in sequence: exact match on amount, date, and reference; tolerance match within the defined parameter; partial match logic for split payments, fees, and bulk settlements
4. Agent auto-clears all exact and tolerance matches, logging the match basis, the rule applied, and the timestamp for each
5. Agent routes exceptions to the human review queue: unmatched items, items above the exception threshold, new counterparty indicators, duplicate payment flags
6. Human reviewer resolves exceptions with documented decisions - cleared, investigated, escalated - each logged with a reason code and timestamp
7. Agent generates the reconciliation report: matched and auto-cleared items, exceptions pending, exceptions resolved
8. Complete audit trail for every decision: auto-cleared matches reference the specific rule applied; exceptions reference the human decision and resolution

TRADE-OFFS

The agent's matching accuracy depends on the quality and consistency of transaction reference data from banking counterparties. Where references are inconsistently formatted across banks, or where matching requires interpretation rather than rule application, the exception rate will be higher and the human review queue will require more capacity. Mode 4 should be reached incrementally - starting with transaction categories that have the cleanest matching data and the lowest exception rates, and expanding as performance is validated.

PILOT FIT

The most accessible entry point to Mode 4 operation. Start with the highest-volume, most standardized transaction category - typically accounts with consistent bank reference formatting and a well-established matching pattern - validate the exception rate and audit trail quality over at least four weeks, then expand to additional categories. Do not start with the most complex transaction types; start with the ones where the matching rules are cleanest and the agent can demonstrate reliable performance before the scope expands.

For organizations that have never run autonomous execution in a finance context, this use case is also the governance training ground. The exception handling discipline, audit trail design, and periodic quality review cadence that Mode 4 reconciliation requires are the same disciplines that every subsequent Mode 4 use case will demand - at higher stakes. Get them right here first.

GOVERNANCE FOCUS

The audit trail completeness requirement for Mode 4 reconciliation is more demanding than it appears. Every auto-cleared match must reference the specific rule that justified it - not just a timestamp and a "matched" status. When an auditor asks "why was transaction X auto-cleared?", the answer must be retrievable from the log as a specific rule reference, not inferred from context.

Exception threshold governance requires periodic review as transaction patterns evolve. A threshold set at go-live based on current transaction patterns may be wrong twelve months later if payment volumes, counterparty mix, or banking relationships have changed. Build the periodic threshold review into the governance calendar before Mode 4 operation begins.

TIP: Reconciliation is an excellent first Mode 4 use case precisely because the stakes per individual transaction are relatively low and the volume is high. It provides the operational experience with autonomous execution governance - exception handling, audit trail design, periodic quality review - that prepares the team for Mode 4 operation in higher-stakes use cases. Treat it as training ground, not just a productivity win.

Choosing Your Starting Use Case

Choosing the right starting use case is itself a governance decision. The right answer depends on four factors: where the most acute operational pain is, what your data quality baseline supports, how mature your existing governance framework is, and what your stakeholders - compliance, internal audit, IT - have the appetite and capacity to support in a pilot.

IF YOUR PRIMARY PAIN POINT IS...	START WITH...	OPERATING MODE
Cash forecast accuracy and cycle time	Use Case 1: Cash Forecasting	Mode 2
Daily liquidity positioning time	Use Case 2: Liquidity Optimization	Mode 2 to 3
Payment fraud exposure	Use Case 3: Anomaly Detection	Mode 3
FX exposure visibility	Use Case 4: FX Management	Mode 2
Intercompany settlement complexity	Use Case 5: IC Settlement	Mode 3 to 4
Reconciliation manual effort	Use Case 6: Reconciliation	Mode 4

Pattern evidence from early adopters supports this mapping. In our review of recent treasury AI pilot patterns, teams that treated the first 90 days as an evidence-building exercise were more likely to identify governance gaps early, narrow the use case appropriately, and avoid premature production expansion.

Two principles apply regardless of which use case you choose - and both reflect hard-won lessons from programs that moved too fast.

Start with one. The governance bandwidth required for an agentic finance pilot - authorization framework design, evidence chain validation, exception handling testing, stakeholder alignment across treasury, legal, compliance, and IT - is consistently underestimated when multiple use cases run in parallel. Each use case requires its own Authorization Framework, its own Exception Ownership Matrix, its own evidence chain validation, and its own set of stakeholder sign-offs. A single well-governed pilot that completes Phase 3 cleanly produces more organizational capability and more audit confidence than three simultaneous pilots each stuck at Phase 1.

Choose for governance fit, not feature appeal. Cash forecasting may be less exciting than autonomous liquidity optimization. If your data quality baseline supports it and your governance framework is closest to ready for it, it is the right choice. The pilot exists to build evidence and confidence - not to showcase the agent's maximum capability. The organizations that get to autonomous operation fastest are the ones that started with the use case they were most ready to govern, not the most impressive one.

Part 2

Action Toolkit

A set of ready-to-use templates, checklists, and a structured plan to design, validate, and govern an agentic finance pilot.

Chapter 5

90-Day Agentic Finance Pilot Playbook

The Pilot Philosophy: Governance Built In, Not Bolted On

Most agentic finance pilots fail the same way. Not because the technology underperforms - it usually performs exactly as demonstrated. They fail because the governance design was treated as a parallel track that would catch up with the technology once the capability was proven.

It never catches up. By the time the team discovers the evidence chain gaps or the authorization ambiguities, the workflow is already in production, users have been trained on it, and leadership is asking about the next use case. Fixing governance at that stage is a redesign, not an adjustment.

A 90-day pilot is not a proof-of-concept exercise. It is a governance validation exercise. The question it exists to answer is not "does the agent produce useful outputs?" - that can be answered in a vendor demonstration. The question is: "Can this agent operate within our control environment, and can our control environment adapt to govern this agent reliably, under real operational conditions?"

That reframing changes how every phase is designed, what evidence is collected, and how the 90-day outcome is evaluated. A pilot that ends with a fast, impressive agent and an incomplete evidence chain is not a success. A pilot that ends with a carefully governed agent and a complete, auditor-tested evidence chain is ready for production.

Every phase of the playbook is designed to build evidence. Phase 0 builds the governance foundation. Phase 1 validates data quality and recommendation quality in advisory mode. Phase 2 tests the governance under execution conditions. Phase 3 produces the governance documentation package that production deployment requires. Each phase has defined gate criteria - conditions that must be met before the pilot advances. Those gates are not bureaucratic checkpoints. They are the mechanism that ensures governance readiness precedes capability expansion at every step.

TIP: When you present the 90-day results to your CFO or board, the most important thing you will be able to say is not *"look how fast it is."* It is *"look how controlled it is - here is the evidence chain, here is the audit trail, here is what happened when something went wrong and how it was handled."* Design the pilot to produce that story.

Phase 0: Pre-Pilot Preparation (Weeks 1 and 2)

Phase 0 is the work that makes the pilot defensible. It is not overhead. It is not delay. It is the governance foundation without which the pilot will either fail its first audit question or require a restart after Phase 1 reveals gaps that should have been resolved before day one.

Two weeks is the right timeframe for organizations approaching Phase 0 with genuine commitment. For organizations starting from a weaker governance baseline - where the Authorization Framework does not exist in any form, or where stakeholder alignment across treasury, legal, compliance, and IT has not begun - three weeks is more realistic. Do not compress Phase 0 to accelerate the start of Phase 1. The gaps not resolved in Phase 0 will be resolved under production pressure in Phase 2, which is a worse outcome in every respect.

GOVERNANCE ARTIFACTS

The Authorization Framework must be drafted, reviewed, and approved in writing before any pilot transaction runs. It must document: the specific use case in scope; the specific entities and data sources covered; the specific actions the agent is authorized to recommend and the specific actions it is authorized to execute; approval thresholds and approval authority requirements for each action category; explicit boundaries - what the agent is not permitted to do under any circumstances during the pilot; and the policy parameters the agent must apply. Reviewed by treasury leadership, legal, compliance, and IT security. Approved in writing by all four before Phase 1 begins.

The Exception Ownership Matrix must be populated for every exception category the agent might encounter in the pilot scope - data quality failures, tool call errors, out-of-scope scenarios, approval workflow failures, duplicate action risks. Each needs a named role owner, pre-authorized responses, escalation paths, and documentation requirements. The matrix does not need to anticipate every possible exception in detail - it needs to ensure every exception has an owner and a defined response pathway.

Evidence chain logging must be tested before Phase 1. Verify that tool call logs are generating records with the required fields: tool name, parameters, response, timestamp. Run a test transaction through the full workflow and verify that you can produce a complete evidence pack for it before the pilot begins.

SCOPE DEFINITION

The pilot scope must be precise: which specific use case, which entities are in scope and which are explicitly excluded, which data sources and systems are covered, what is the maximum financial exposure of any single action the agent may execute, and what categories of action are explicitly prohibited regardless of the agent's configured capability.

Precision in scope definition serves two purposes: it limits the governance surface area during the pilot to something manageable, and it provides the explicit boundary definition that prevents the authorization drift in the quarter-end scenario from Chapter 1. If it is not in the scope definition, it is out of scope - regardless of what the agent could technically do.

TECHNICAL READINESS

Data connectivity for all in-scope sources must be confirmed and tested, not assumed. The data quality baseline must be documented - known gaps, known freshness limitations, known inconsistencies - so that Phase 1 can distinguish between agent performance issues and data quality issues. The platform configuration must be reviewed by IT security. The TMS integration must be tested in read-only mode before any write access is enabled.

STAKEHOLDER ALIGNMENT

The treasury team members who will work with the agent in Phase 1 must be briefed on the operating mode, the human-in-the-loop checkpoints, and the exception handling procedures before the pilot begins. Internal audit should receive a briefing on the pilot design and be given the opportunity to provide input on what evidence they would want to see. Their input at this stage is more valuable than their findings at a review stage.

Success criteria and stop criteria must both be defined and agreed before Phase 1 begins.

Success means specific, measurable outcomes: recommendation accuracy within a defined threshold, time saved per cycle, exception resolution time, governance gaps identified and resolved. Stop criteria - the conditions under which the pilot would be paused or redesigned - should include: data quality below an acceptable threshold that cannot be resolved within the pilot scope; a governance gap that creates audit exposure and cannot be resolved without a scope redesign; or agent behavior that falls outside the defined Authorization Framework in a material way.

Phase 1: Controlled Pilot - Advisory Mode Only (Weeks 3 through 6)

Phase 1 runs the agent in purely advisory mode. No autonomous execution of any kind. Every recommendation the agent produces is reviewed by a human before any action is taken. The governance requirement from the human reviewer in this phase is not to act on the agent's output - it is to evaluate it: is it accurate, is it explainable, is it consistent with policy, and does the evidence chain support it?

The purpose of Phase 1 is to establish three things as fact, not assumption, before execution capability is introduced: that the agent's data quality and recommendation quality meet the threshold required for the use case; that the evidence chain is complete and produces an audit-ready record for every transaction; and that exception handling works as designed under real operating conditions. Each of these must be documented with evidence - not asserted - before Phase 2 begins.

WEEKS 3 AND 4: BASELINE AND CALIBRATION

The first two weeks of Phase 1 run the agent in parallel with the existing manual process. The manual process remains the basis for actual decisions. The agent runs alongside it, producing its advisory output independently. The treasury team compares the two: where does the agent's recommendation align with the manual output, where does it diverge, and when it diverges, why?

This parallel run is not about proving the agent is more accurate than the manual process - though that may emerge. It is about understanding the agent's behavior: which entities or data types produce the most variance, which policy parameters the agent applies inconsistently, which data quality conditions affect the recommendation most significantly, and whether the evidence chain is capturing what it needs to capture.

Log every data quality issue encountered. Stale balances, missing entities, tool call latencies, ERP sync gaps - all of it should be documented in the data quality baseline report that Phase 1 produces.

Data quality issues surface early and consistently. Across the pilots and implementation patterns we reviewed, data quality issues were one of the most common sources of delay. The most frequent gaps were not exotic AI problems. They were familiar treasury data problems: incomplete bank feeds, inconsistent entity mapping, stale ERP extracts, unclear ownership of reference data, and reconciliation breaks between systems.

WEEKS 5 AND 6: INCREASING RELIANCE

In the second half of Phase 1, the operating model shifts. The agent's advisory output becomes the starting point for the treasury team's analysis rather than a parallel track. Human review becomes validation of the agent's draft - checking sources, challenging assumptions, applying context - rather than independent construction of a parallel output. This shift is deliberate: it tests whether the evidence chain supports efficient human review, and it begins to surface the judgment calls that will require human attention in production.

During weeks 5 and 6, exception handling should be tested deliberately. Trigger a data quality flag using a deliberately stale data set. Present the agent with an out-of-scope scenario and verify that it escalates correctly. Introduce an anomaly and verify that the detection and escalation logic works as designed. Document the outcomes and include them in the Phase 1 deliverables.

The evidence pack review is the gate test for Phase 1. Select a sample of at least five agent recommendations from the advisory period, covering different entity types and recommendation categories, and attempt to produce the complete evidence pack for each: data sources with timestamps, reasoning chain, recommendation log, human review record, and final approved action. If any link in the chain is missing or incomplete, that gap must be resolved before Phase 2 begins.

PHASE 1 GATE CRITERIA

Before Phase 2 can begin, all of the following must be true:

- Data quality issues from the baseline are documented and either resolved or formally risk-accepted with a written rationale
- Recommendation accuracy is within the defined threshold for the use case
- The evidence chain produces a complete, auditor-ready pack for a sample of at least five transactions
- Exception handling has been tested across at least three scenario types and the results documented
- No material governance gaps remain unresolved
- Treasury leadership and compliance have reviewed the Phase 1 results and approved in writing the decision to proceed

Phase 2: Supervised Execution (Weeks 7 through 10)

Phase 2 introduces agent execution. For the first time in the pilot, the agent does not just recommend - it acts. Every action runs through the approval workflow before executing, and every execution is reviewed in the approval queue during weeks 7 and 8. Exception rates are tracked. The SoD design is validated under live conditions. The governance framework is stress-tested against real operational variability.

The transition to execution is the highest-risk phase of the pilot, and it is the phase where governance gaps not fully resolved in Phase 0 or Phase 1 will become visible. This is the intended outcome. A governance gap discovered in Phase 2 under supervised conditions is a manageable finding. The same gap discovered in production under operational pressure is an incident.

WEEKS 7 AND 8: FIRST EXECUTIONS

Enable execution capability for the smallest, most controlled action category within the pilot scope. For a reconciliation pilot: auto-clearing exact matches only, with all other categories still requiring manual review. For a liquidity optimization pilot: executing intercompany transfers below a defined threshold for entities with the highest data quality scores. Start narrow. Expand based on evidence, not on schedule.

During weeks 7 and 8, every execution must be reviewed in the approval queue by the designated human reviewer - not to re-approve, but to verify: did the agent execute precisely what was approved? Did the action produce the expected outcome? Were there any anomalies in the execution not reflected in the recommendation?

Duplicate action risk must be monitored actively in the first two weeks of execution. In the early stages of the transition from manual to agentic, there is genuine risk that both the agent and a manual process initiate the same action. Define the detection mechanism before execution begins - not after the first duplicate surfaces.

WEEKS 9 AND 10: INCREASING SCALE

In the second half of Phase 2, execution scope expands to additional action categories that met the Phase 1 quality threshold. Human review in the approval queue shifts from reviewing every execution to reviewing all executions above the defined threshold and all flagged exceptions. The weekly exception analysis begins: what is the false positive rate, are there patterns in the exceptions suggesting a detection rule needs adjustment, are there anomalous execution patterns that warrant investigation?

The evidence pack review in Phase 2 is more demanding than in Phase 1. For a sample of executed actions, verify the complete evidence chain is present: the data sources that underpinned the recommendation, the approval record with approver identity and timestamp, the TMS execution record, and the bank confirmation. The SoD validation requires evidence that no single user both initiated and was the sole approver of a material action during the period. Produce that evidence before the Phase 2 gate review.

64

PHASE 2 GATE CRITERIA

Before Phase 3 can begin, all of the following must be true:

- Execution accuracy is within the defined threshold; discrepancies are documented and explained
- The exception resolution log for the Phase 2 period is complete - every exception has a resolution record and a documentation entry
- SoD has been validated with documentary evidence: no single user both initiated and was sole approver of a material action
- The Exception Ownership Matrix has been updated to reflect real exceptions encountered in Phase 2
- Treasury leadership has reviewed the Phase 2 results and the go/no-go recommendation for Phase 3 has been formally submitted and approved in writing

CASE STUDY: 90-Day Reconciliation Pilot at a $2.8B Industrial Manufacturer

Phase 0, Weeks 1 to 2:
Authorization Framework approved by treasury, legal, and compliance. Data quality baseline identified gaps in transaction reference formatting across 3 of 8 banks, flagged for monitoring in Phase 1.

Phase 1, Weeks 3 to 6:
Parallel run on 4,200 transactions across 4 weeks. Agent auto-matched 3,847 transactions, 91.6%, with exact or tolerance match rules. Human review validated match quality on a 200-transaction sample: 2 false positives, 0.05% error rate. Exception queue routed 353 unmatched items to treasury operations, with average resolution time of 4.2 minutes versus 8.1 minutes in the manual baseline.

Phase 2, Weeks 7 to 10:
Agent execution enabled for exact matches only in Weeks 7 to 8, then tolerance matches under $5K in Weeks 9 to 10. Execution accuracy was 99.8%. One escalated exception occurred: duplicate payment indicator triggered review and was resolved within 20 minutes through the documented escalation path.

Phase 3 outcome:
Audit readiness review by internal audit validated the evidence chain, confirmed segregation of duties, and found that all gate criteria were met. The pilot was approved for production with scope expansion to 2 additional high-volume accounts in Q1 2026.

Time savings:
62% reduction in reconciliation cycle time, from 18 hours per month-end to 6.8 hours. Treasury operations capacity was reallocated to exception resolution and quarterly policy review.

Key learning:
"The data quality work in Phase 0 was the difference. We almost skipped it to save two weeks. If we had, we would have spent four weeks in Phase 1 firefighting bank reference formatting issues instead of validating governance." - Treasury Operations Manager

Phase 3: Expanded Scope and Governance Documentation (Weeks 11 through 14)

Phase 3 has two distinct purposes. The first is to validate whether the governance framework developed and tested over the first ten weeks is robust enough to support an expanded scope - a second use case, additional entities, or higher-value action categories. The second is to produce the governance documentation package that production deployment requires: the complete, version-controlled, auditor-reviewed set of governance artifacts that demonstrates the program is ready to operate outside pilot conditions.

SCOPE EXPANSION

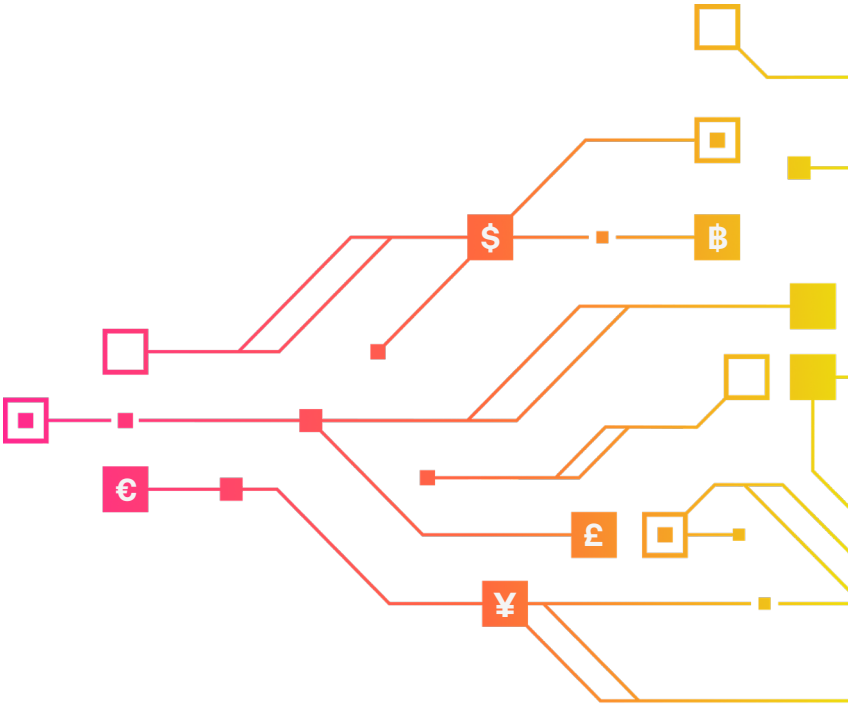
Scope expansion in Phase 3 should be conditional, not automatic. The question before expanding: has Phase 2 demonstrated sufficiently consistent, well-governed performance that the governance framework can be extended with confidence - or are there open questions that expanding scope will amplify rather than resolve? If Phase 2 results are strong - recommendation accuracy within threshold, exception handling effective, evidence chain complete, SoD validated - expansion can proceed. It must follow the same Phase 0 preparation discipline for the new scope: authorization framework update, exception matrix update, technical readiness confirmation, stakeholder alignment. A scope expansion that skips Phase 0 preparation because “we already know how to do this” is where governance drift begins. If Phase 2 results are mixed - some action categories performing well, others generating higher exception rates or evidence chain gaps - expand only the performing categories. Treat the underperforming ones as requiring a Phase 2 extension before expansion.

AUDIT READINESS REVIEW

The most important activity in Phase 3 is the formal audit readiness review. Internal audit - or an external advisor with equivalent governance expertise - should be given access to: the full evidence chain for a sample of pilot actions from Phase 1 and Phase 2; the Authorization Framework and its version history; the Exception Ownership Matrix and the exception resolution log from Phase 2; and the vendor due diligence documentation.

The review question is direct: based on the evidence produced during this pilot, does the governance framework operate as designed? The answer from the people whose professional responsibility it is to evaluate governance is the most reliable production-readiness signal available.

Incorporate audit findings into the governance documentation package before the go/no-go decision is made. A finding that requires a governance adjustment is better addressed before the production decision than after it.



THE 90-DAY GO/NO-GO DECISION

The pilot concludes with a formal, documented decision with four possible outcomes - and the right outcome depends entirely on what the evidence shows, not on what leadership wants the answer to be.

Scale means the pilot has produced a complete, auditor-validated governance package, Phase 2 performance meets the defined success criteria, and the organization is ready to proceed to full production scope with full procurement and integration commitment.

Pilot extension means specific, identified governance gaps remain that are resolvable within the existing pilot framework - a detection rule that needs refinement, an evidence chain link that needs strengthening, an exception category that needs additional testing. The pilot continues until those gaps are resolved, then the go/no-go decision is revisited.

This is not a failure; it is the governance process working correctly.

Pivot means the current use case, data environment, or platform does not meet governance requirements, but the overall program direction is sound. The organization reassesses the use case selection, the vendor, or the data readiness foundation before committing to a second pilot.

Stop means fundamental governance or data quality constraints have been identified that are not resolvable within the current organizational context. The program is deferred, the learnings are documented, and the prerequisites for a future attempt are defined. This outcome, documented honestly, is more valuable to the organization than a pilot that is declared a success despite unresolved governance gaps.

TIP: Involve internal audit in the go/no-go decision review, not just treasury leadership and technology teams. The pilot was designed to produce governance evidence. The people whose professional role is to evaluate governance evidence should be the ones who confirm it is sufficient.

The 90-Day Pilot at a Glance

PHASE	WEEKS	KEY ACTIVITIES	GATE CRITERIA
Phase 0: Preparation	1–2	Authorization Framework approved; scope defined; technical readiness confirmed; stakeholders aligned; success and stop criteria agreed	All pre-pilot checklists complete; no unresolved governance gaps; written approval from treasury leadership, legal, compliance, and IT
Phase 1: Advisory Mode	3–6	Parallel run; baseline calibration; evidence chain validation; exception handling tested	Data quality documented and acceptable; recommendation accuracy within threshold; evidence chain passes audit test for sample; no unresolved material governance gaps
Phase 2: Supervised Execution	7–10	First executions; approval queue review; exception analysis; SoD validation	Execution accuracy within threshold; exception resolution log complete; SoD validated; Phase 2 results approved by treasury leadership and compliance
Phase 3: Expanded Scope	11–14	Conditional scope expansion; audit readiness review; governance documentation package; formal go/no-go decision	Complete governance package; audit readiness review findings incorporated; formal go/no-go decision documented and approved

CHECKLIST 4: PHASE 0 READINESS

- ☐ Authorization Framework drafted, reviewed, and approved in writing by treasury leadership, legal, compliance, and IT security
- ☐ Pilot scope defined precisely: use case, entities, data sources, action types, financial exposure limit, explicit exclusions
- ☐ Data connectivity confirmed and tested for all in-scope sources
- ☐ Data quality baseline documented: known gaps, freshness limitations, and inconsistencies
- ☐ Platform configuration reviewed by IT security; TMS integration tested in read-only mode
- ☐ Exception Ownership Matrix populated: owner by role, pre-authorized responses, escalation paths, documentation requirements
- ☐ Evidence chain logging confirmed active and tested: tool call logs, reasoning chain logs, recommendation logs, approval logs, action logs all generating complete records
- ☐ Treasury team briefed on operating mode, human-in-the-loop checkpoints, and exception handling procedures
- ☐ Internal audit briefed on pilot design and given opportunity to provide governance input
- ☐ Success criteria and stop criteria defined, documented, and agreed by all stakeholders

CHECKLIST 5: PHASE 1 COMPLETION GATE

- ☐ Parallel run completed; systematic differences between agent and manual outputs documented and explained
- ☐ Data quality issues logged; material issues resolved or formally risk-accepted with documented rationale
- ☐ Recommendation accuracy assessed against defined threshold; result documented
- ☐ Evidence chain validated: complete evidence pack produced for a sample of at least five advisory transactions
- ☐ Exception handling tested: data quality flag, out-of-scope scenario, and anomaly detection scenarios all tested and documented
- ☐ No material governance gaps remain unresolved
- ☐ Authorization Framework updated to incorporate Phase 1 learnings
- ☐ Phase 1 results reviewed and approved in writing by treasury leadership and compliance

CHECKLIST 6: GO/NO-GO CRITERIA (90-DAY)

- ☐ Complete governance documentation package exists and is version-controlled: Authorization Framework, Exception Ownership Matrix, Evidence Chain requirements, Operating Mode definitions, Escalation Procedures
- ☐ Audit readiness review completed by internal audit or equivalent; findings incorporated into governance documentation
- ☐ Vendor due diligence documentation complete: data hosting, training data policy, security certifications, SLA terms, data residency, audit access provisions, exit rights
- ☐ SoD validation documented with evidence: no single user both initiated and was sole approver of a material action during the pilot period
- ☐ Exception handling demonstrated effective under live conditions: exception resolution log complete for the full Phase 2 period
- ☐ Treasury team has demonstrated the capability to supervise and manage the agent independently
- ☐ Legal and compliance have reviewed and approved the production scope in writing
- ☐ CFO or Treasurer has reviewed the 90-day evidence package and approved the go/no-go recommendation in writing

70

Chapter 6

Five Things To Do Next

Where to Begin

The guides that do not change anything are the ones that end with a summary of what you have read. This one ends with five actions that begin in the next 30 days.

None of them requires a vendor, a budget approval, or a technology decision. They are the governance groundwork - the authorization framework, the data quality baseline, the internal audit relationship - that determines whether a pilot, when it begins, builds confidence or discovers problems. Organizations that complete all five before the first vendor conversation typically run better pilots and make better technology decisions. The preparation is not preliminary to the program. It is part of it.

Several of the five can run in parallel. None is sequential on the others except where noted.

Next Step 1: Run the Quarter-End Test on Your Current Processes

Before you deploy an AI agent into any treasury process, apply the governance stress test from Chapter 1 to the manual version of that process. Take your top two or three candidate use cases and ask, for each: Can you trace every material decision in this process to a named, authorized human? Is there a complete evidence chain connecting the data to the analysis to the recommendation to the approval to the action? Is exception handling documented, tested, and consistently followed?

If the answer to any of these questions is *"not consistently"* or *"not in a format that satisfies audit,"* the agent will inherit those gaps and amplify them. Agentic workflows do not fix underlying governance weaknesses in the processes they replace - they execute at higher speed and volume, which means existing gaps surface faster and with greater consequence.

This exercise typically takes two hours per process and produces two things: a clear picture of where your governance foundation is already strong enough to support agentic operation, and a work list of governance gaps that must be resolved before the pilot begins. That work list is your Phase 0 preparation agenda.

Action: For your top two candidate use cases, schedule a two-hour governance review. Map the decision flow, identify who authorizes each step, document what evidence currently exists at each link in the chain, and note where exception handling is informal or inconsistently applied. Document the findings. That document is the starting point for your Authorization Framework.

Next Step 2: Establish Your Data Quality Baseline

The most reliable predictor of a difficult agentic finance pilot is not governance immaturity - it is data quality problems that were not assessed before the pilot began.

Bank feeds that refresh inconsistently, ERP data that lags by four hours, entities with no direct banking connectivity, forecast inputs that are manually adjusted before use: all of these are normal in real treasury environments, and all of them affect an agent's output quality in ways that are difficult to separate from agent performance issues once the pilot is running.

Assess your data quality baseline for the pilot use case before you evaluate any platform or configure any workflow. Specifically: which entities have direct bank connectivity and which require manual or delayed data? What is the actual refresh frequency for each data source, and does it meet the freshness requirement for the use case? Are your ERP and TMS data sources reconciled to each other, and how frequently? Where are the known gaps, and what is the current manual workaround for each?

This assessment typically takes one to two weeks for a medium-complexity treasury environment. Its output - a documented data quality baseline with gaps rated by severity - is one of the most valuable artifacts the pilot will produce. It gives the Phase 1 parallel run a reference point that separates data quality variance from agent performance variance.

Action: Assign one team member to conduct a structured data quality assessment for the pilot use case. Use the scope defined in your Authorization Framework as the boundary. Document connectivity coverage, refresh frequencies, known gaps, and the current manual handling for each gap. Score each gap against the data requirements for the agent workflow.

Next Step 3: Build Your Authorization Framework

Do not wait for a vendor to tell you what the agent should be authorized to do. That conversation will be shaped by what the platform can do, not by what your governance requirements demand. Build your Authorization Framework first - as a treasury governance document, authored by your team, reviewed by your compliance and legal functions - and bring it to the vendor conversation as a specification, not a question.

The first draft does not need to be exhaustive. It needs to be honest: what is the specific use case, what data does the agent need access to, what actions is it permitted to recommend, what actions is it permitted to execute, what is explicitly off-limits, and who approves what above which thresholds? A one-page draft that answers these questions clearly is more valuable than a detailed document that has not been reviewed by compliance.

The act of drafting the Authorization Framework almost always surfaces governance questions the team had not previously asked explicitly - questions about where approval authority actually sits for specific action categories, whether existing delegation of authority matrices cover agentic workflows, and whether exception handling procedures that work for manual processes are adequate for an agent operating at higher volume and speed. Those questions are better surfaced in a document review than in a Phase 2 live exception.

Action: Draft a one-page Authorization Framework for your primary pilot use case. Define scope, explicit exclusions, action categories, approval thresholds, and policy parameters. Share it with compliance and internal audit with the specific request: *"Does this governance design meet our existing control standards, and what would you need to see added before you would be comfortable with a pilot operating under this framework?"*

Next Step 4: Brief Your Internal Audit Team Now, Not Later

The single most effective thing you can do to ensure a successful agentic finance pilot is to involve internal audit before it starts. Not as a compliance obligation. As a governance asset.

Auditors briefed on the pilot design, the Authorization Framework, and the evidence chain requirements before the pilot begins will help you build the right governance from the start. They will ask the questions that reveal gaps before those gaps become findings. They will tell you what evidence they would need to see to be satisfied that the program is governed correctly - and that information is more valuable at design stage than at review stage.

Auditors who encounter agentic AI for the first time during a review will ask the same questions - under different conditions, with findings rather than suggestions as the output. The difference between those two scenarios is the timing of the conversation, not its content.

Action: Schedule a 60-minute briefing with your internal audit lead before the pilot begins. Cover: what an AI agent is and how it differs from existing automation tools; the specific pilot scope and Authorization Framework; the evidence chain design and what it will produce. Close with the question: *"What evidence would you need to see at the end of this pilot to be satisfied that the program is governed to the standard you would expect?"*

Document their answer. Incorporate the requirements into the pilot design before Phase 0 is complete.

Next Step 5: Evaluate Vendors Against the Governance Framework, Not the Feature List

When you are ready to evaluate AI agent platforms, bring the governance framework from this guide as your evaluation criteria. The vendor's feature list tells you what the platform can do. The governance framework tells you what it must do to be deployable in your environment.

Six questions matter more than any demonstration:

1. Does the platform maintain a complete, tamper-evident, exportable audit trail - tool call logs, reasoning chains, recommendation records, approval logs, and execution confirmations - in a format your auditors can interrogate?

2. Does the platform enforce SoD controls at the system level, preventing a single user from both configuring an agent task and being the sole approver of its recommended action above a defined threshold?

3. How does the platform handle hallucination risk? Are financial outputs grounded in logged tool calls, or can the language model generate free-form numbers that appear in recommendations without a traceable data source?

4. What happens when the agent encounters a data quality issue or an out-of-scope scenario? Does it stop and escalate, or proceed with a flag?

5. Is your data used to train shared models? Where is it hosted, and what are the data residency provisions?

6. What are the exit rights? If you need to change vendors or platforms, can you export all audit logs, configuration, and historical data in a portable format?

Action: Before any vendor demonstration, send these six questions in writing and request written answers. Evaluate the responses before the demo, not during it. Ask the vendor to demonstrate the exception handling scenario specifically: *"Show us what the agent does when it encounters a data quality issue that affects a live recommendation."* That demonstration reveals governance maturity more reliably than any capability showcase.

The Question That Does Not Go Away

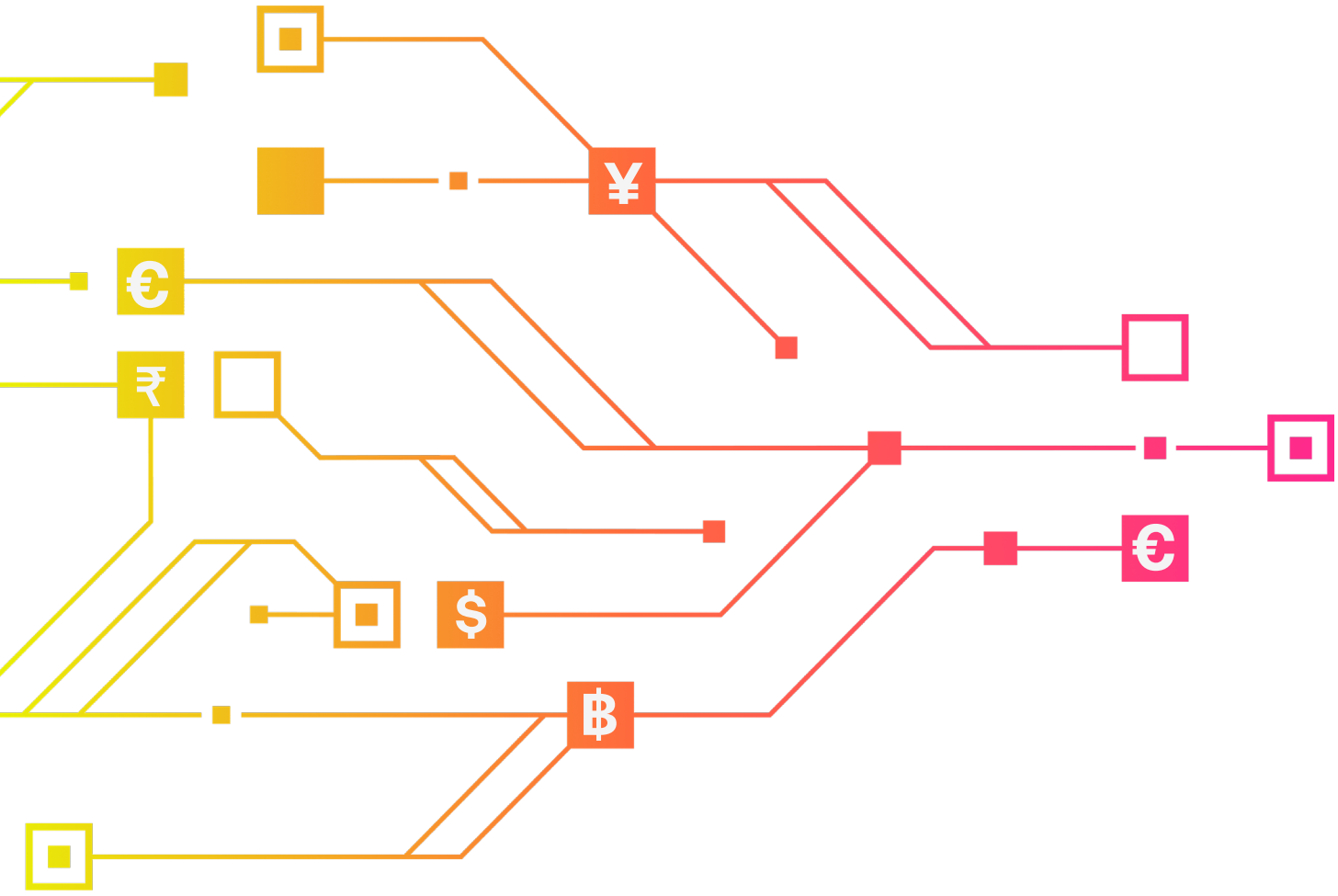
The organizations that benefit most from agentic finance are not the ones that deploy the most capable technology. They are the ones that build the most trusted governance framework around it, prove that framework under controlled pilot conditions, and expand from a foundation of evidence rather than expectation.

Every tool in this guide - the authorization framework, the evidence chain, the exception ownership matrix, the 90-day playbook - exists to answer a question that does not go away once you deploy: if something goes wrong tomorrow, can you prove you were in control?

That question will be asked by an auditor. It will be asked by your CFO. It may be asked by a regulator. The answer, when it needs to be given, cannot be assembled retrospectively from incomplete logs and informal processes. It has to have been built in - before the first transaction ran.

That is what governance-first means.

Not caution. Not delay. Not a preference for the status quo. A decision that when agentic finance delivers, it delivers something you can stand behind.



Appendices

Appendix A: Vendor Evaluation Scorecard

Most vendor evaluations go in the wrong sequence: demonstration first, governance questions after. By the time the governance questions arrive, the team has already formed a view of the platform based on the demo - and the governance answers are filtered through that impression rather than the other way around.

This scorecard reverses the sequence. Send it to vendors before the demonstration. Request written answers. Use those answers to shape what you ask for in the demo. The demonstration should show you things the written answers described; it should not be the primary source of your governance assessment.

Score each criterion from 1 (absent or inadequate) to 5 (fully implemented and demonstrable in writing and in a live demonstration). Weight the governance criteria - audit trail, SoD, hallucination controls, exception handling - at least as heavily as integration and usability.

A platform that scores 5 on integration and 2 on audit trail is not a governance-ready platform. It is a capable platform with a material governance gap.

CRITERION	SCORE (1–5)	EVIDENCE REQUIRED	NOTES
Audit trail quality complete, tamper-evident, exportable log of tool calls, reasoning, recommendations, approvals, and actions		Export sample log; confirm format is auditor-readable	
Data privacy training data policy; your data not used in shared model training; data hosting location documented		Written policy statement; contract provision	
SoD enforcement platform-level controls prevent creator from being sole approver of material actions		Live demonstration of approval workflow separation	
Hallucination controls financial figures grounded in logged tool outputs; dedicated calculation tool for arithmetic		Show evidence chain for a numeric recommendation	
Exception handling agent stops and escalates on data quality failure or out-of-scope scenario; does not proceed silently		Demonstrate exception scenario live	

CRITERION	SCORE (1–5)	EVIDENCE REQUIRED	NOTES
Integration depth Native connectors to your TMS, ERP, and banking platforms; no requirement to replicate data into a separate environment		Integration architecture documentation	
Scope governance Authorization Framework enforced at platform level; scope changes require documented change control		Show configuration access controls	
Data residency Hosting location meets your regulatory requirements; residency provisions in contract		Contract provision; infrastructure documentation	
Exit rights All audit logs, configuration, and historical data exportable in portable format on termination		Contract provision; data export demonstration	
SLA and business continuity Uptime commitments; failover design; manual fallback plan documented		SLA terms; business continuity documentation	

Scoring guide:

- **40–50:** Strong governance fit. Proceed to detailed due diligence and pilot scoping.
- **30–39:** Conditional fit. Specific gaps must be contractually resolved and platform-level remediation demonstrated before pilot commitment.
- **Below 30:** Material governance gaps. Not recommended for a treasury pilot without significant remediation. The demo was probably impressive. The governance is not ready.

TIP: Ask every vendor the same question, in the same words, in writing: *"Show us what the agent does when it encounters a data quality issue mid-analysis."* The range of answers - and the range of what vendors are willing to demonstrate rather than describe - is the most reliable signal of governance maturity available before you commit to a pilot.

Appendix B: Authorization Framework Template

This template is the starting point for your Authorization Framework - the document that must exist, be approved, and be version-controlled before any agentic finance workflow runs in your environment. It is a policy document, not a technical configuration file. It should be authored by treasury, reviewed by legal and compliance, approved in writing before the pilot begins, and treated with the same governance discipline as your delegation of authority matrix.

Every field should be completed before Phase 0 is signed off. Blank fields are not neutral - they are undeclared scope, which is where the quarter-end scenario from Chapter 1 originates.

AUTHORIZATION FRAMEWORK

Use Case: [Name]

Version: [1.0]

Approved by: [Treasury Lead, Legal, Compliance, IT Security]

Approval date: [Date]

Next review date: [Date, or on any material scope change]

Section 1: Scope - What the Agent is Authorized to Access and Do

Data sources in scope:
[List each source, the entities covered, and the refresh frequency requirement]

Systems the agent may read from: [List]

Systems the agent may write to or initiate actions in:
[List, with specific action types permitted]

Action categories and approval requirements:

ACTION CATEGORY	ENTITIES IN SCOPE	MAX AUTONOMOUS EXPOSURE	APPROVAL REQUIRED	APPROVER ROLE
[e.g., Read bank balances]	[All / listed]	N/A	No	N/A
[e.g., Recommend transfer]	[Listed]	Unlimited	Yes	[Role]
[e.g., Execute transfer]	[Listed]	[$amount]	Yes - dual approval	[Roles]

Section 2:
Boundaries - What the Agent Is Explicitly Not Authorized to Do

The following actions are explicitly out of scope for this agent under this Authorization Framework, regardless of technical capability. This list must be affirmative and complete - the absence of a prohibition is not sufficient.

- [e.g., Initiate credit facility drawdowns]
- [e.g., Access data for entities not listed in Section 1]
- [e.g., Approve its own recommendations] [Add all explicit exclusions] access roles assigned, insurance verified).

Section 3:
Policy Parameters - Rules the Agent Must Apply in All Workflows

- Minimum balance requirements: [Reference policy document and version]
- Investment policy constraints: [Reference policy document and version]
- Payment approval thresholds: [Reference policy document and version]
- Regulatory reporting triggers requiring human review: [List]

Section 4: Exception Escalation - Summary

[Reference the Exception Ownership Matrix as the governing document for exception handling. State explicitly: all out-of-scope scenarios escalate before action, not after. No autonomous resolution of out-of-scope items under any circumstances.]

Section 5: Version History

VERSION	DATE	CHANGES	APPROVED BY
1.0	[Date]	Initial draft	[Names]

Appendix C: Glossary Extended

The Chapter 2 glossary covers the fourteen terms most relevant to a treasury practitioner evaluating agentic finance. The terms below are ones you will encounter as your understanding deepens - in vendor conversations, in technical documentation, and in the research literature. They are defined here in treasury-relevant terms, not computer science terms.

TERM	DEFINITION
RAG (Retrieval-Augmented Generation)	A technique where an LLM retrieves relevant documents from a structured knowledge base before generating a response, improving accuracy on specific topics. In a finance context, RAG can be used to ground agent responses in your policy documents, procedure manuals, and historical data rather than relying solely on the model's training.
Vector Database	A specialized database that stores information as numerical vectors, enabling fast semantic search. Used by RAG systems to retrieve contextually relevant content. Not a replacement for structured financial data sources - a complementary retrieval mechanism for unstructured reference material.
Fine-Tuning	Further training of a foundation LLM on domain-specific data to improve performance on specialized tasks. Relevant when evaluating vendors: understand whether fine-tuning used your data or anonymized industry data, and what the data handling policy was.
Prompt Engineering	The design and optimization of the instructions given to an LLM to improve output quality, consistency, and safety. The system prompt is the primary prompt engineering artifact in a finance agent - treat it as a policy document, not a technical configuration.
Multi-Agent Systems	Architectures where multiple AI agents collaborate, each handling a specific subtask. Example: one agent retrieves and validates data; a second performs the analysis; a third formats and presents the recommendation. Governance of multi-agent systems requires each agent's scope and authorization to be documented independently.
Context Management	Techniques for determining what information is included in the LLM's context window for a given task. Important for large, multi-entity treasury operations where the full dataset exceeds what can be processed in a single pass.
RLHF (Reinforcement Learning from Human Feedback)	A training technique where human evaluators rate model outputs, and those ratings are used to shape model behavior. Relevant for understanding how foundation models are aligned before enterprise deployment - and for evaluating how vendor feedback mechanisms work in production.
Agentic Loop	Shorthand for the plan-act-observe-adapt reasoning cycle described in Chapter 2. Used interchangeably with "reasoning loop" in most vendor and research contexts.

Appendix D: How This Guide Was Built

This guide was written by Tom Callway (VP Product Marketing, Kyriba) and Felix Grevy (SVP Data & AI, Kyriba) over the first half of 2026. Between us, we bring backgrounds in treasury product management, banking data architecture, and enterprise AI deployment - the last two years spent building and evaluating agentic capabilities in live finance environments.

Research sources

Treasury practitioner experience across payments, liquidity management, cash forecasting, FX risk management, and financial controls; AI and agentic framework research current as of 2026, including published work on LLM governance, multi-agent orchestration, and enterprise AI deployment; governance frameworks adapted from traditional finance control environments including segregation of duties standards, delegation of authority design, and audit evidence requirements; anonymized learnings from treasury teams in early-adopter agentic finance programs; IDC InfoBrief: AI-Enabled Finance Operations; AFP resources on intelligent treasury and finance transformation.

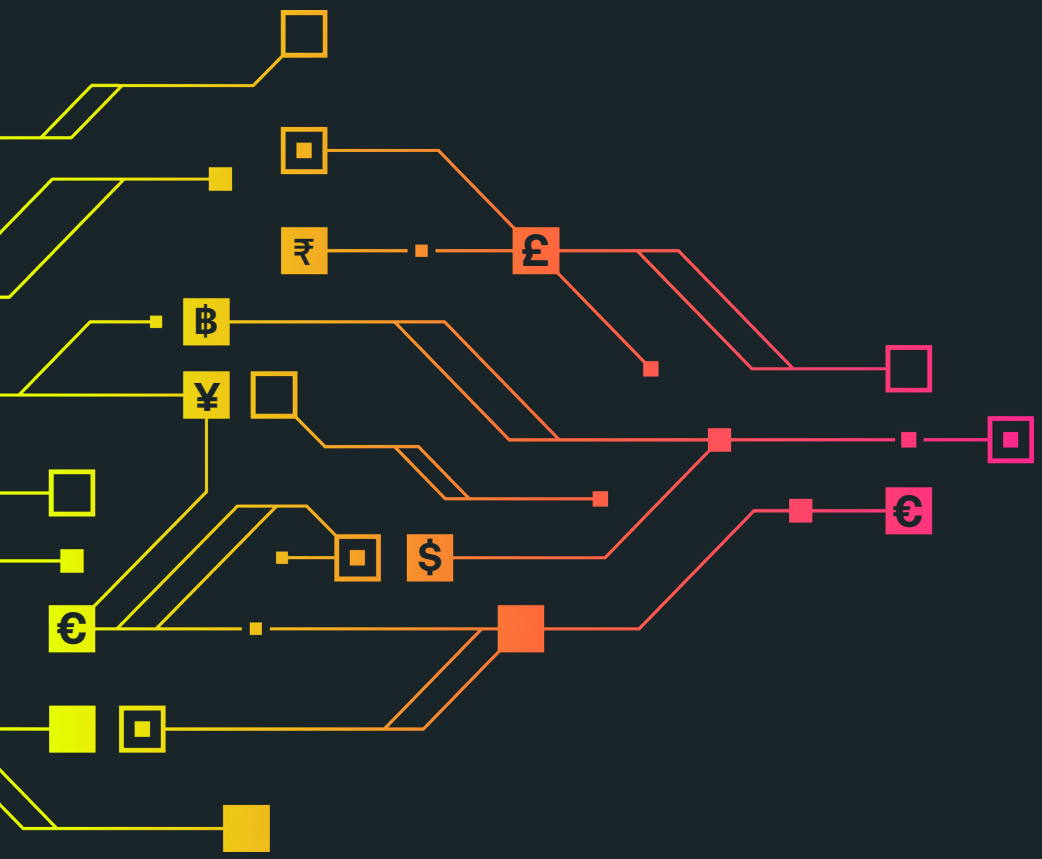
Where we cite specific figures, sources are referenced. Where examples are composite or anonymized, they are labeled as such. Where the maturity of a capability is genuinely uncertain, we have said so rather than hedged it into a claim.

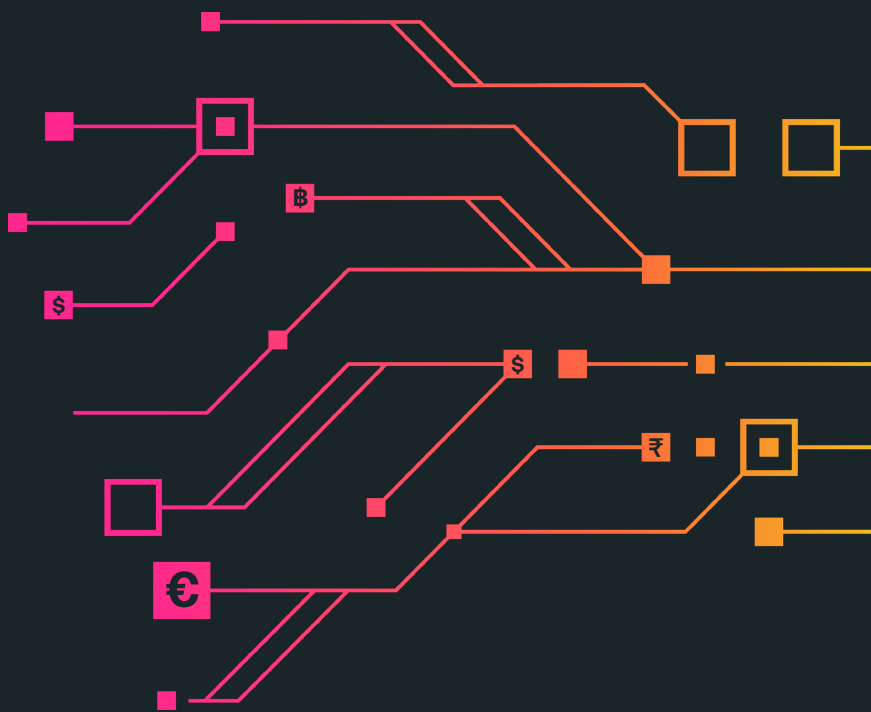
Kyriba disclosure

This guide is published by Kyriba Corporation. Where Kyriba platform capabilities are referenced, they are clearly identified as such. The governance frameworks, checklists, use case evaluations, and pilot playbook throughout are designed to be vendor-neutral and applicable to any agentic finance platform. The goal is to help treasury and finance practitioners make better decisions - with or without Kyriba.

Feedback

If you have used this guide in a pilot design, a vendor evaluation, or a stakeholder briefing, we want to hear what worked and what did not. Contact us via kyriba.com. Future editions will be shaped by practitioners who used this one in the field.





Contact Kyriba

1081 Camino del Rio South
San Diego, CA 92108

Phone: +1 858 210 3560

Email: treasury@kyriba.com

Website: www.kyriba.com