Kyriba

Spring 2026

Stablecoins & On-Chain Liquidity: A No Hype Guide

For treasury practitioners evaluating on-chain payments rails

Tom Callway
VP Product Marketing

Jean-Baptiste Gaudemet
SVP Strategic Innovation Lab


Kyriba
Trusted to
TRANSFORM

Table of Contents

About This Guide

This is a practical, jargon-free guide for corporate treasury teams who need to assess stablecoins for real-world payments - cross-border payouts, supplier/partner payments, and settlement - and evaluate when (and when not) to use on-chain payment rails, starting with treasury outcomes, not technology.

How this Guide Is Structured

This guide is split into two parts. Use Part 1 to get clarity and alignment; use Part 2 to turn that alignment into a concrete pilot plan.

Part 1: Strategy & Operating Model helps you build a shared understanding of what stablecoins change (and what they don't), how to evaluate the options available, and what "good" looks like from a treasury governance, controls, and operating model perspective. The goal is to help you reach an informed point of view and align stakeholders on the right questions, constraints, and decision criteria.

Part 2: Action Toolkit is a set of ready-to-use templates, checklists, and a structured 90-day plan you can adapt internally to design, validate, and govern a stablecoin initiative-ending with a clear go/no-go decision and next steps.

What This Guide Is

- **Treasury-first.** Written for practitioners who understand payments, approvals, reconciliation, and controls - but have little to no background in blockchain or digital assets.
- **Governance-first.** Starts with the principle that governance can't be added later. If you can't prove approvals, execution, settlement, and reconciliation under audit, speed and cost don't matter.
- **Evidence-based.** No speculation, no unverified market claims, no investment advice. Where examples or data are uncertain, they're labeled as illustrative or framed as questions to validate.
- **Decision-focused.** Every chapter is designed to help you make informed decisions: Should we explore this? Which payment model fits our use case? How do we pilot without creating governance debt? What do we do next?

Ground Rules

This guide is:

- **Not a whitepaper.** No academic deep dives into blockchain consensus mechanisms or cryptographic theory. You'll learn just enough to make treasury decisions.
- **Not hype.** Stablecoins are a payment rail, not a revolution. They have real benefits (speed, cost, transparency) and real risks (custody, peg stability, regulatory uncertainty). We'll cover both.
- **Not a vendor pitch.** Vendor and partner examples are illustrative, not endorsements. This guide doesn't sell you a solution - it helps you evaluate solutions.
- **Not investment or legal advice.** This guide addresses stablecoins as treasury instruments and payment rails, not as investments. Always consult your legal, tax, and compliance advisors before piloting.

Kyriba disclosure: This guide is published by Kyriba, a treasury management system provider that offers stablecoin payment capabilities as part of its platform. Where Kyriba is mentioned, it is clearly labeled to maintain transparency. Our goal is to help you make informed decisions - whether or not you use Kyriba's platform.

Who This Guide Is For

Primary audience: Corporate treasury practitioners and treasury operations professionals with ≥3 years of experience who understand payment workflows (approvals, execution, settlement, reconciliation), bank connectivity and payment rails (SWIFT, ACH, wires, payment APIs), treasury controls (segregation of duties, approval authorities, audit trails), and liquidity management - but have little to no blockchain or crypto experience.

This guide assumes strong treasury fundamentals and zero crypto knowledge. We'll explain blockchain concepts in treasury terms, not the other way around. Also useful for:

- **Treasury managers and directors** evaluating whether to explore stablecoin pilots.
- **AP/AR operations teams** who will execute or reconcile stablecoin payments (focus on Chapters 3, 5, and 6).
- **CFOs and Treasurers** who need strategic context and risk frameworks (read Chapters 1, 4, and 6).
- **Compliance and internal audit** professionals assessing controls for stablecoin payments (focus on Chapters 3 and 5).

Not for: Crypto traders or speculators, blockchain developers, or treasury teams exploring cryptocurrency holdings as investments.

How To Use This Guide

This guide is structured in six chapters plus front matter. Chapters build on each other, but each chapter stands alone - you can read Chapter 4 (use cases) without reading Chapter 2 (stablecoins 101) if you're already familiar with the basics.

Recommended Reading Paths

Path 1: I'm new to stablecoins and need the full foundation

Read in order: Chapter 1 → Chapter 2 → Chapter 3 → Chapter 4 → Chapter 5 → Chapter 6
Time commitment: 4-6 hours (or read 1-2 chapters per week over a month).

Path 2: I understand stablecoins; I need to evaluate use cases and pilots

Read: Chapter 1 (governance framing) → Chapter 4 (six payment models) → Chapter 5 (pilot playbook) → Chapter 6 (next steps)

Time commitment: 2-3 hours.

Path 3: I'm designing governance for an already-scoped pilot

Read: Chapter 3 (governance & evidence) → Chapter 5 (pilot playbook)

Time commitment: 1.5-2 hours.

Path 4: I need to pitch my CFO or prepare next steps

Read: Chapter 1 (why governance first) → Chapter 4 (use cases) → Chapter 6 (five things to do next)

Time commitment: 1.5 hours.

Icon Legend: Tips, Warnings, and Reminders

Throughout the guide, you'll see three types of callouts:

TIP: PRACTICAL ADVICE OR SHORTCUTS

Tips offer actionable guidance, common starting points, or ways to simplify a complex topic.

Example:

TIP: Start your evaluation by asking, "Which of our existing governance controls apply here, and which need to be adapted?" Don't start from zero. You already know how to govern payments.

WARNING: RISK, PITFALL, OR COMMON MISTAKE

Warnings flag risks, failure modes, or governance shortcuts that create problems later.

Example:

WARNING: "Governance debt" is like technical debt, but worse. You can refactor code. You can't refactor an audit finding that says you moved $5 million without documented approvals.

REMEMBER: CORE PRINCIPLE OR KEY TAKEAWAY

Remember callouts reinforce principles or concepts you should internalize.

Example:

REMEMBER: Visibility is not the same as provability. You can see transaction hashes on a blockchain explorer. That doesn't mean you can prove the payment was authorized, executed within policy, and reconciled correctly.

What You'll Learn

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

- Explain stablecoins in treasury terms (instrument vs. rail, how the peg works, failure modes that matter)
- Evaluate six distinct stablecoin payment models and choose the one that fits your use case and readiness
- Design governance for stablecoin payments (dependency mapping, controlled fallbacks, evidence chains, exception handling)
- Scope and execute a 90-day pilot with governance built in from day one, not bolted on later
- Ask the right questions of vendors and partners (custody, approvals, reconciliation, SLAs, insurance)
- Make a confident decision (scale, pivot, or stop, based on evidence, not hope or hype)

Most importantly, you'll be able to answer this question:

```
Should my treasury function explore stablecoin payments and, if so,
how do we do it without compromising governance or creating
operational risk?
```

That's the question this guide exists to answer.

How This Guide Was Written

This guide draws on treasury practitioner experience in payments, liquidity management, and controls; stablecoin market research (as of early 2026); governance frameworks from traditional payment systems adapted for on-chain rails; anonymized pilot learnings from early adopters; and AFP resources on digital assets for corporate treasury.

Let's get started.

Part 1

Strategy & Operating Model

This section gives you the practical context to evaluate stablecoins through a treasury 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 On-Chain Payments Get Real at 3 a.m.

The Wrong Place to Start

Most treasury conversations about stablecoins start with a pitch deck full of promises: faster settlement, lower fees, 24/7 availability, programmable money, elimination of intermediaries.

All of those things can be true. None of them is the right starting point.

Because if you can't prove who approved the payment, when it executed, what happened when something went wrong, and how you'd do it differently next time, then speed and cost savings don't matter.

You've just built a compliance liability with a blockchain wrapper.

Outcomes first. Rails second.

That's not a tagline. It's the difference between a pilot that scales into production and a pilot that gets shut down the first time an auditor asks a question you can't answer - or the first time an exception happens outside business hours and no one knows who owns the decision.

This chapter explains why governance and operational readiness must come before you evaluate speed, cost, or technology.

It introduces the trust framework that underpins any treasury payment system - on-chain or off.

And it gives you a simple, repeatable filter for stress-testing any stablecoin payment model: the 3 a.m. test.

If your approach passes that test, it will work Tuesday at 2 p.m. too.

What "Treasury-First" Really Means in Practice

Treasury teams already know how to evaluate payment rails. You map the parties, understand the dependencies, define approval workflows, establish segregation of duties, set limits, document exceptions, and build an audit trail that connects the invoice to the approval to the payment instruction to the settlement confirmation.

You do take these steps because treasury is a control function, not just an execution function. On-chain stablecoin payments don't change that. But they introduce new dependencies - wallet custody providers, liquidity partners, blockchain infrastructure, on/off-ramp providers, settlement providers - and most of those dependencies behave differently than banks.

A treasury-first approach means asking governance questions before technology questions:

- **Who approves?** (Not just in the ERP, but at every handoff point.)
- **Who executes?** (And how do you enforce segregation of duties when a private key can move $10 million?)
- **Who reconciles?** (Can your AP team tie a blockchain transaction hash back to an invoice?)
- **What happens when a dependency fails?** (And who decides whether to retry, cancel, or escalate?)

These aren't theoretical questions. They're the questions that determine whether your stablecoin payment program survives its first operational exception.

TIP: If your planning starts with "How fast can we move money?" instead of "Who owns the decision when something breaks?", pause and restart. Speed without governance is just fast failure.

The 3 a.m. Wake-Up Call

Here's the scenario every treasury team should use to stress-test a stablecoin payment model:

It's 3 a.m. on a Saturday. A $2 million USDC payment to a supplier in Singapore has been sitting in "pending" status for six hours. Your ERP shows the payment approved. Your wallet dashboard shows a transaction hash. The supplier says they see nothing in their account.

Your bank's treasury services team doesn't support stablecoin inquiries. The vendor who sold you the wallet integration platform is headquartered in Europe; their support line opens in five hours. Your internal payment operations team doesn't have access to the wallet custody provider's admin console. No one on call tonight was trained on blockchain explorers.

Now what?

This isn't a nightmare scenario. It's a normal operational exception playing out on an unfamiliar rail at an inconvenient time.

Every payment system has exceptions. Payments get stuck, counterparty details turn out to be wrong, a provider has an outage, a transaction times out. In traditional rails, you have established escalation paths: you know who to call, what information they'll ask for, and what your fallback options are.

On-chain rails can be faster and cheaper, but if you haven't designed for exceptions before you go live, you're not ready for production. You're just hoping nothing breaks during business hours.

The 3 a.m. Test: Three Questions

Every stablecoin payment model - whether it's fiat-to-fiat with stablecoin in the middle, direct USDC to suppliers, or intercompany settlement - should be able to answer three questions confidently:

1. Who can investigate right now?

Do you have the access, the evidence trail, and the escalation path you need to diagnose and resolve the issue. Or, are you waiting for business hours in multiple time zones?

Being prepared means:

- Someone on your team (or on-call coverage) has access to wallet dashboards, transaction logs, and custody provider support.
- Your evidence trail connects the ERP approval, the payment instruction, the on-chain transaction, and the settlement outcome in one auditable chain.
- You have documented escalation contacts for every dependency in your payment flow: custody provider, liquidity provider, on/off-ramp partner, and internal stakeholders (AP, treasury ops, compliance).

If your answer is "We'd have to wait until Monday and email three vendors," your model isn't ready.

2. What's the controlled fallback?

If the on-chain payment is stuck, delayed, or fails, can you execute the payment through an alternate rail - without re-keying data, breaking segregation of duties, and creating reconciliation chaos?

"Controlled fallback" means:

- You've pre-defined the conditions where you would switch rails (e.g., payment pending for >2 hours, counterparty reports non-receipt, custody provider outage).
- The fallback path uses the same approval and the same data; you're just changing the execution rail.
- You explicitly design for the rare but high-impact dual-settlement outcome, where the on-chain payment later settles after a SWIFT fallback has already been executed (yes, this can happen).

If your fallback plan is "We'll figure it out," you don't have a fallback plan.

WARNING: "Fast and cheap" becomes "slow and expensive" the moment you can't resolve an exception. If your 3 a.m. fallback is "wait and hope," your operational risk is higher than your cost savings.

3. What does the auditor see on Monday?

When your auditor (internal or external) asks, "Walk me through what happened with payment #45234," can you produce a complete, tamper-evident record that shows:

- Who approved the payment, and when?
- What instruction was sent to the wallet/custody provider, and when?
- What transaction executed on-chain (transaction hash, timestamp, sending and receiving addresses)?
- What confirmation was received, and how did it flow back to your ERP/TMS?
- If there was an exception: who made what decision, when, and what was the outcome?

Or do you have fragments of evidence in three different systems, a spreadsheet someone reconstructed from wallet logs, and a Slack thread where the decision to retry was made?

Visibility is not the same as provability.
You can see transaction hashes on a blockchain explorer. That doesn't mean you can prove the payment was authorized, executed within policy, and reconciled correctly.

If the answer is "We'd have to pull logs from four places and build a timeline manually," your governance model isn't done.

Why Governance Can't Be Added Later

Here's a pattern we see often:

A treasury team runs a 60- or 90-day stablecoin pilot. It goes well. Payments settle quickly, fees are low, stakeholders are happy. Leadership says, "Great, let's scale this."

Then someone from compliance or internal audit asks, "How do we know this payment was approved?" or "What happens if the wallet provider goes offline?" or "How do we reconcile this in the ERP?"

And the team realizes: they built the payment flow, but they didn't build the evidence chain.

They didn't document who owns exceptions. They didn't map dependencies or define controlled fallbacks.

They assumed they could retrofit governance after proving the concept worked.

You can't retrofit accountability into a system that wasn't designed to produce evidence.

Here's why:

1. **Workflow assumptions were wrong.** Approvals, limits, and segregation of duties aren't add-ons; they're constraints that shape how the payment flow must work. If you design the flow first and try to overlay governance later, you'll discover your flow violates policy - and now you've trained users on a process you have to redesign.
2. **Evidence gaps can't be filled retroactively.** If your custody provider doesn't log who initiated a wallet transaction, you can't go back in time and create that log. If your integration didn't capture approval metadata from the ERP, you can't reconstruct it from a blockchain transaction hash.
3. **Exception handling is not an edge case.** Exceptions are normal operations. If you haven't pre-defined who decides what when a payment fails, you'll make it up under pressure, and your auditor will correctly point out that "we figured it out" isn't a control.

WARNING: "Governance debt" is like technical debt, but worse. You can refactor code. You can't refactor an audit finding that says you moved $5 million without documented approvals.

The Trust Framework: Three Pillars

Treasury teams are used to evaluating trust in payment systems. You trust that your bank will execute your wire instructions accurately. You trust that SWIFT messages will route correctly. You trust that your TMS will enforce your approval policies.

On-chain stablecoin payments require the same trust framework, just with different dependencies.

Every treasury payment system - whether it's SWIFT, ACH, or USDC - rests on three pillars:

1. Trusted Connectivity

Can you reliably connect to the parties and systems you need in order to instruct, execute, monitor, and confirm payments?

Traditional rails: Your bank connectivity (SWIFT, host-to-host file transfer, API) is the trusted channel.

On-chain rails: Your wallet custodian and your on/off partners and blockchain infrastructure are the trusted channels. You need to know: Who operates them? What are their SLAs? What happens if they go offline? Do you have redundant access?

2. Trusted Data

Can you trust that the payment instructions, status updates, and settlement confirmations you see are accurate, complete, and tamper-proof?

Traditional rails: Your bank sends you MT940 statements and payment confirmations. You trust the bank's systems and audit controls.

On-chain rails: Transaction data is recorded on a public or permissioned blockchain. You can verify it independently (that's a feature). But you also need to trust your wallet provider is showing you accurate balances, your integration is pulling the right transaction hashes with the associated metadata, and your reconciliation process is matching on-chain settlement to ERP invoices correctly.

3. Strong Governance

Can you prove - under audit - that every payment was approved by the right people, executed within policy, and reconciled correctly? And can you show that when exceptions happened, they were handled according to documented procedures?

Traditional rails: Your TMS, ERP, and bank reporting provide the evidence chain. You have documented approval workflows, limits, and exception-handling procedures.

On-chain rails: You need the same governance, but now you also need to map new dependencies (custody provider, wallet access management, smart contract logic if applicable) and new evidence sources (blockchain transaction hashes, wallet logs, on/off-ramp settlement confirmations).

The rails change. The governance requirements don't.

If your stablecoin payment model can't deliver trusted connectivity, trusted data, and strong governance, it's not ready for treasury.

TIP: Start your evaluation by asking, "Which of our existing governance controls apply here, and which need to be adapted?" Don't start from zero. You already know how to govern payments.

Outcomes First, Rails Second (What It Actually Means)

"Outcomes first, rails second" is a principle, not a slogan. It means:

- Define the **outcome** you need: pay a supplier on time, cut cross-border payment fees, settle intercompany balances, answer business requests to accept payment in stablecoins with customers and suppliers, provide payment proof to a counterparty.
- Define the **governance requirements**: approval authorities, limits, segregation of duties, evidence trail, exception handling, fallback options.
- Then evaluate which **rail** (SWIFT, ACH, stablecoin sandwich, stablecoin issuer, bank deposit token) best delivers that outcome within those governance requirements.

It does not mean "ignore the technology." It means don't let the technology dictate your outcomes or compromise your governance.

Here's an example scenario:

You want to pay a supplier in the Philippines faster and cheaper than SWIFT.

Outcomes-first approach:

1. Outcome: Supplier receives USD equivalent in PHP in their local bank account within 4 hours, with payment proof you can tie to the invoice.
2. Governance: Standard approval workflow in ERP, segregation of duties (treasury approves, ops executes), audit trail connecting invoice → approval → payment → settlement.
3. Rails evaluation: Could use correspondent banking, a payment API provider, or a stablecoin on-ramp partner who converts fiat → USDC → PHP and deposits in the supplier's bank. Evaluate each on speed, cost, and governance fit.

Rails-first approach (wrong):

1. "Stablecoins are fast and cheap - let's use USDC."
2. Build the integration.
3. Discover the supplier doesn't have a wallet, so you need an off-ramp provider.
4. Find out the off-ramp provider's KYC process takes 5 days.
5. Then realize your ERP can't automatically reconcile a blockchain transaction hash to an invoice.
6. Retrofit governance after you've already made workflow decisions that don't fit your approval policies.

Outcomes-first means starting with the problem, not the solution.

If stablecoins are the right rail for that outcome under your governance requirements, great. If they're not, that's also useful information. Either way, you made a decision based on treasury judgment, not vendor hype.

Why This Chapter Comes First

The rest of this guide will give you the building blocks to evaluate stablecoin payment models with confidence:

- **Chapter 2** will demystify stablecoins in treasury terms: what they are, how they work, how they fail - so you can assess them as instruments and rails.
- **Chapter 3** will show you how to map dependencies, design controlled fallbacks, and build evidence chains that prove governance under audit.
- **Chapter 4** will walk you through six distinct stablecoin payment models - from fiat-in/fiat-out "sandwiches" to direct payments - with trade-offs, governance focus areas, and pilot fit.
- **Chapter 5** will give you a 90-day pilot playbook that builds governance in from day one.
- **Chapter 6** will give you five concrete next steps.

But none of that matters if you start in the wrong place.

Start with governance. Start with the 3 a.m. test. Start with outcomes.

The rails will follow.

> ☆ **REMEMBER:** Speed, cost, and innovation are real benefits. But they're not the foundation. The foundation is trust: trusted connectivity, trusted data, and strong governance. If you build on that foundation, everything else gets easier.

Chapter 2 Stablecoins in Treasury Terms

What This Chapter Is (and Isn't)

This chapter gives you just enough technical foundation to evaluate stablecoins as treasury instruments and payment rails. It's not a deep dive into blockchain architecture or cryptography. It's a practical explainer written for treasury practitioners who understand payments, foreign exchange, credit risk, and liquidity - but have limited or no background in digital assets. You'll learn:

- What stablecoins are in treasury terms (and what they're not).
- How they differ from both fiat currency and cryptocurrency.
- How the "peg" to fiat works - and how it can break.
- The failure modes that matter to treasury operations.
- A minimal glossary of terms you'll encounter.

By the end of this chapter, you'll be able to read a stablecoin payment proposal and ask the right questions about credit risk, liquidity risk, settlement finality, and counterparty dependencies.

What Is a Stablecoin?

A stablecoin is a digital token designed to maintain a stable value relative to a reference asset - almost always the US dollar, though euro- and other fiat-pegged stablecoins exist. Think of it as a digital asset backed in dollar (or euro, or yen) that can be transferred on a blockchain network instead of through traditional banking rails. Key characteristics:

1. **Digital and blockchain-native.** Stablecoins exist as tokens on public blockchain networks (Ethereum, Solana, and others). They don't live in a bank account. They live in a digital wallet controlled by cryptographic private keys.
2. **Designed for price stability.** Unlike Bitcoin or Ether, which fluctuate in value, stablecoins aim to hold a 1:1 peg with fiat, e.g. one USDC should always be worth approximately one US dollar.
3. **Transferable 24/7/365.** Because they run on blockchain infrastructure, stablecoin transfers can happen any time - weekends, holidays, outside banking hours - as long as the network is operational.
4. **Settlement finality is fast.** Depending on the blockchain, a stablecoin transaction can achieve final, irreversible settlement in seconds or minutes (not days).

What Stablecoins Are Not

- **Not bank deposits.** Stablecoins are not FDIC-insured. They are not held in your name at a regulated depository institution. Depending on the stablecoin, they may be backed by reserves (cash, Treasuries, other assets), but the legal and regulatory treatment is different from a demand deposit.
- **Not tokenized deposits.** A tokenized deposit is a bank deposit represented on-chain; your claim is on a bank deposit account under that bank's framework. A stablecoin is an issuer-backed token with a different legal claim, governance model, and redemption path.
- **Not tokenized funds.** Tokenized funds are on-chain representations of fund shares (often money market-like) with their own dealing, valuation, and liquidity terms. They can be useful for cash management, but they are not designed primarily as a payments settlement token.
- **Not "cryptocurrency" in the speculative sense.** While stablecoins use blockchain technology, they're designed not to fluctuate in value. They're a medium of exchange and settlement, not an investment thesis. Regulations (US GENIUS Act and EU MiCA) typically prohibit offering yield on stablecoins.
- **Not risk-free.** The peg can break. Despite tight regulatory constraints on the quality of assets held to keep the value of the stablecoin, the issuer can fail. The blockchain network can experience downtime. Wallet custody arrangements matter. These are manageable risks, but they're real.

☆ **REMEMBER:** A stablecoin is a *digital bearer instrument*. Whoever controls the private keys to the wallet controls the funds. There's no "forgot password" recovery process, and there's no customer service line that can reverse a transaction sent to the wrong address.

Asset vs. Rail: Why the Distinction Matters

One of the most important concepts for treasury practitioners is separating (a) the asset you hold from (b) the network that moves it.

STABLECOIN AS THE ASSET (WHAT YOU HOLD)

When you hold USDC in a wallet, you're holding a cash-like digital asset issued by a regulated financial infrastructure company (e.g., Circle) and redeemable (subject to terms) against its cash reserves. For treasury, it should typically be a transient asset - held only as long as needed to complete settlement. In most treasury use cases, that means holding USDC for minutes to hours, not days or weeks - similar to how you'd minimize float in a zero-balance account structure or sweep excess balances daily. Even held briefly, stablecoins still introduce issuer, liquidity, and legal/regulatory considerations.

Treasury questions:

- Who is the issuer, and what is their credit/operational standing?
- What assets back the coin, and how liquid are they?
- What are the redemption terms and legal rights in stress?
- What regulatory regime applies to the issuer and reserves?

RAIL AS THE NETWORK (HOW VALUE MOVES)

When you transfer stablecoin, the rail is the blockchain ledger/network (validators/nodes, consensus, network uptime, and the wallet/custody stack interacting with it). Stablecoin is the asset being transferred over that rail.

A key nuance: the same stablecoin can exist on multiple rails (Ethereum, Solana, etc.), so your rail assessment is really an assessment of the specific blockchain and wallet/custody stack you're using.

Treasury questions:

- How fast is settlement on this chain when the network is busy?
- What are the transaction costs/fee dynamics?
- What dependencies must work (chain uptime, validators, wallet/custody provider, on/off-ramps)?
- What finality do you get, and can transactions be reversed or recalled?
- How do you evidence payment (on-chain proofs + operational records)?

Transaction fees (a rail attribute).

Stablecoin transfers aren't "free." Most blockchains charge a network transaction fee ("gas") to process and record a transaction. Fees are paid to the network and can vary based on network demand and congestion.

Don't compare the blockchain fee alone to a wire fee. Compare the end-to-end cost to complete the payment, including custody/wallet, on/off-ramp fees or spreads, FX spreads (if any), and operational overhead.

Treasury implications:

- Fees are variable. A rail that is cheap today may be more expensive during peak periods, which matters for high-volume flows.
- Design for fee spikes. Set fee thresholds, approval rules for high-fee periods, and a fallback rail for time-critical payments.

WHY THIS DISTINCTION MATTERS

Many proposals blur these layers. A vendor might pitch "instant, low-cost payments" (rail performance) without fully addressing the risk of holding the asset, even briefly. Or they might emphasize "1:1 reserves" (asset quality) without addressing network outages, congestion, or custody-stack failures.

TIP: Evaluate separately: **(1)** "Am I comfortable holding this asset, even briefly?" and **(2)** "Am I comfortable relying on this blockchain rail for high value payments?"

How the Peg Holds

The defining feature of a stablecoin is its peg to fiat, usually 1 token = 1 USD. But how does that peg hold? There are several mechanisms, but the two that matter most to corporate treasury are fiat-backed stablecoins and algorithmic stablecoins (treasury should generally avoid the latter for operational payments).

FIAT-BACKED STABLECOINS (RESERVE MODEL)

Fiat-backed stablecoins are the most common model and the most relevant to treasury operations. The issuer holds reserves (cash, Treasury bills, short-term government debt, sometimes other high-quality liquid assets) equal to or greater than the total stablecoins in circulation.

How it works:

1. You (or a partner on your behalf) deposit $1,000 USD with the stablecoin issuer.
2. The issuer mints 1,000 stablecoin tokens and credits them to your wallet.
3. The issuer holds your $1,000 in reserve.
4. You can transfer the 1,000 tokens to anyone, anytime.
5. When you (or whoever ends up holding those tokens) want to redeem them for fiat, you send the tokens back to the issuer, and they burn (destroy) the tokens and return $1,000 USD to your bank account.

Examples: USDC (Circle), USDP (Paxos), and several others operate on this model.

Why the peg holds:

As long as:

- The reserves exist and are liquid,
- The issuer honors redemptions promptly,
- The market believes both of the above are true,

... then the stablecoin will trade at or very close to $1.00.

If you can always redeem 1 token for $1, no rational actor will pay significantly more or less than $1 per token.

Algorithmic Stablecoins (Avoid for Treasury)

Some stablecoins attempt to maintain a peg without holding fiat reserves. Instead, they use algorithms, incentives, and secondary tokens to expand or contract supply in response to demand.

Why treasury should avoid them:

While algorithmic stablecoins have the advantage of operating fully on-chain, they have a history of catastrophic failures (TerraUSD / Luna, 2022). When confidence breaks, the algorithm can't defend the peg, and the value can collapse to near zero in hours.

Bottom line: If a stablecoin isn't backed by auditable reserves, it's not appropriate for corporate treasury payments.

WARNING: Not all stablecoins are created equal. Before using any stablecoin for treasury operations, verify: (1) What reserves back it? (2) Who holds and audits those reserves? (3) What is the redemption process and timeline? If you can't get clear answers, walk away.

How the Peg Breaks (Failure Modes Treasury Needs to Understand)

Even fiat-backed stablecoins can "de-peg," meaning the market price diverges from $1.00. Understanding why and how this happens is critical for risk management.

FAILURE MODE 1: LIQUIDITY CRUNCH (TEMPORARY DE-PEG)

What happens: Demand to redeem stablecoins for fiat spikes suddenly. The issuer holds reserves, but they're not instantly liquid (e.g., some reserves are in Treasury bills that mature in 30 days).

Result: Redemptions slow down. Holders panic and try to sell stablecoins on secondary markets. Price drops to $0.95 or $0.90 as sellers accept a discount to exit quickly.

Is the stablecoin "broken"? Not necessarily. If reserves are sound and redemptions eventually process, the peg recovers. But if you needed to move funds during the de-peg window, you took a loss.

Treasury implication: Even "safe" stablecoins can experience short-term price volatility during stress events. Don't hold stablecoins longer than operationally necessary.

FAILURE MODE 2: RESERVE QUALITY CONCERNS (CONFIDENCE CRISIS)

What happens: The market learns (or suspects) that the issuer's reserves are not what they claimed, e.g., backed by risky commercial paper instead of cash, or reserves were commingled with the issuer's operating funds.

Result: Holders lose confidence and rush to redeem. Even if reserves technically exist, the perception of risk causes a run.

Is the stablecoin "broken"? Depends. If reserves are genuinely at risk, yes. If it's just a perception issue that gets resolved with transparency (audited attestations, proof of reserves), the peg can stabilize.

FAILURE MODE 3: ISSUER FAILURE (CATASTROPHIC)

What happens: The stablecoin issuer becomes insolvent, faces regulatory action, or ceases operations.

Result: Redemptions halt. Token holders become unsecured creditors in a bankruptcy or wind-down process. Recovery could take months or years, and might be partial.

Is the stablecoin "broken"? Yes.

Treasury implication: Treat stablecoin issuer risk like you would treat any financial counterparty. Understand their regulatory status, capital structure, and operational controls. Diversify if you're moving large amounts. Don't hold balances you can't afford to lose.

FAILURE MODE 4: BLOCKCHAIN NETWORK ISSUES

What happens: The underlying blockchain network experiences downtime, congestion, or a security incident. You can't move your stablecoins, even though they're still worth $1.00.

Result: Your funds are temporarily stuck. Not a peg break, but an operational failure.

Treasury implication: Stablecoins depend on blockchain infrastructure. If Ethereum (or Solana, or any other network) goes down or becomes prohibitively expensive to use, your payment rail is unavailable. Have fallback options.

REMEMBER: The peg is a function of reserves, redemption processes, and market confidence. All three must hold for the peg to remain stable. As a treasury practitioner, your job is to assess whether you're comfortable with the issuer's credit quality, reserve composition, and liquidity - and to limit exposure accordingly.

Counterparty Risk vs. Protocol Risk

Two types of risk underpin stablecoin operations: **counterparty risk** and **protocol risk**. Understanding the difference helps you ask the right due diligence questions.

COUNTERPARTY RISK

Counterparty risk is familiar territory for treasury: the risk that a party you're depending on fails to perform.

In stablecoin payments, your counterparties might include:

- The stablecoin issuer (credit risk: will they honor redemptions?)
- Your wallet custody provider (operational risk: will they execute your instructions correctly and securely?)
- On-ramp and off-ramp partners (performance risk: will they convert fiat ↔ stablecoin reliably and on time?)

How to manage it: Use the same frameworks you'd use for any financial counterparty - due diligence, credit analysis, contractual protections, exposure limits, and diversification.

PROTOCOL RISK

Protocol risk is less familiar: the risk that the underlying technology (the blockchain, the smart contract, the wallet software) has a flaw, gets exploited, or behaves in an unexpected way.

Examples:

- A smart contract bug that allows unauthorized withdrawals (has happened).
- A blockchain network fork that creates two competing versions of the ledger (rare but possible).
- Wallet software that exposes your private keys through a security vulnerability (has happened).

How to manage it: Choose mature, widely used blockchain infrastructure and stablecoin protocols with strong security track records. Use reputable custody providers with insurance and robust operational controls. Limit exposure during the pilot phase.

Protocol risk is additional risk, not a replacement for counterparty risk. You face both.

TIP: When evaluating a stablecoin payment partner, ask: "What happens if there's a smart contract exploit on the blockchain we're using?" If the answer is vague or dismissive ("That's extremely unlikely"), probe deeper. Unlikely ≠ impossible, and treasury needs a plan for low-probability, high-impact scenarios.

Settlement Finality: What "Irreversible" Really Means

One of the most-touted benefits of blockchain-based payments is **settlement finality** - the idea that once a transaction is confirmed on-chain, it's final and irreversible.

That's *mostly* true, but treasury needs to understand the nuances.

HOW SETTLEMENT WORKS ON A PUBLIC BLOCKCHAIN

When you send a stablecoin payment:

1. Your wallet creates a transaction and signs it with your private key.
2. The transaction is broadcast to the blockchain network.
3. Network validators (miners, in some systems) include your transaction in a new block.
4. That block is added to the blockchain.
5. After a certain number of additional blocks are added on top of yours (called "confirmations"), the transaction is considered final and secured by proof-of-work or proof-of-stake depending on the type of blockchain.

How long does this take?

- **Ethereum:** 12 seconds per block; typically considered final after 12-20 confirmations (3-5 minutes).
- **Solana:** ~400 milliseconds per block; finality in seconds.
- Other blockchains vary.

Once final, the transaction cannot be reversed, charged back, or recalled.

WHY "IRREVERSIBLE" IS BOTH A FEATURE AND A RISK

Feature: No chargeback risk. If you're a merchant or supplier receiving stablecoin payments, you don't have to worry about the buyer reversing the payment 60 days later (unlike credit cards). Settlement is final.

Risk: If you send a stablecoin payment to the wrong wallet address, or to a counterparty who turns out to be fraudulent, you can't reverse it. There's no dispute process, no bank to call, no regulatory mechanism to recover the funds.

Treasury implication: Final settlement is powerful, but it means your pre-payment controls (approval workflows, counterparty validation, wallet address verification) are even more critical. Once you send the payment, it's gone.

WARNING: "Immutable" and "irreversible" sound secure, but they also mean mistakes are permanent. If treasury accidentally sends $500,000 to the wrong wallet address, there's no "undo." Your governance model must prevent errors, not rely on fixing them after the fact.

Minimal Glossary: 12 Terms You'll Encounter

You don't need to become a blockchain expert, but you will encounter certain terms in stablecoin payment conversations. Here's a quick-reference glossary with treasury-relevant definitions.

TERM	WHAT IT MEANS (IN TREASURY TERMS)
Blockchain	A distributed digital ledger that records transactions across many computers. Think of it as a shared, tamper-evident database that no single party controls.
Wallet	A software application or hardware device that stores your private keys and lets you send/receive stablecoins. Similar to a bank account, but you control the keys (and the associated risk).
Private Key	A cryptographic code that proves you control a wallet. Like a password + signing authority combined. If someone gets your private key, they control your funds.
Public Key / Address	The "account number" others use to send you stablecoins. Safe to share publicly (like an IBAN or account number).
Gas Fee / Transaction Fee	The network fee to process an on-chain transaction. Variable, can spike during congestion, and affects the per-payment economics.
On-Ramp	The process/service of converting fiat currency → stablecoin (e.g., deposit USD, receive USDC).
Off-Ramp	The process/service of converting stablecoin → fiat currency (e.g., send USDC, receive USD in your bank account).
Mint / Burn	Minting = creating new stablecoin tokens (when reserves are deposited). Burning = destroying tokens (when stablecoins are redeemed for fiat).
Smart Contract	Self-executing code on a blockchain that automatically enforces rules. Stablecoin is a smart contract (ERC 20). It can also be used for conditional payments, escrow, etc. (Not always involved in simple stablecoin transfers.)
Custody Provider	A third party that holds and safeguards your private keys and executes transactions on your behalf. Think of it as outsourced wallet management with institutional controls.
Transaction Hash (TxHash)	A unique identifier for a blockchain transaction. Like a SWIFT reference number, you can use it to look up and verify the transaction on a blockchain explorer.
Blockchain Explorer	A website that lets you search and view transaction history on a blockchain (e.g., Etherscan for Ethereum). Public and transparent - anyone can look up any transaction.

REMEMBER: You don't need to memorize blockchain jargon to evaluate stablecoins for treasury. You need to understand the treasury implications: credit risk, liquidity risk, custody risk, settlement finality, and governance requirements.

One Decision Table: Stablecoin Risk Factors and Mitigations

Here's a practical summary of the key risks treasury must evaluate when considering stablecoin payments, along with mitigations.

RISK FACTOR	WHAT COULD GO WRONG	HOW TO MITIGATE
Issuer Credit Risk	Stablecoin issuer becomes insolvent; reserves are insufficient or illiquid; redemptions halt.	Choose issuers with strong credit profiles, transparent reserve composition, regular attestations. Limit exposure. Treat like any financial counterparty.
Peg Stability Risk	Stablecoin de-pegs during market stress; value drops below $1.00.	Minimize hold time. Use fiat-backed (not algorithmic) stablecoins. Monitor secondary market pricing during pilot. Have fiat fallback ready.
Custody Risk	Private keys are lost, stolen, or misused; funds are irretrievable.	Use institutional custody providers with insurance, multi-signature controls, and robust operational security. With MPC (multi-party computation protocol), the private key should never be formed in one place.
Blockchain Network Risk	Network downtime or congestion prevents you from sending/receiving payments when needed.	Choose mature, high-uptime networks. Have fallback rails (SWIFT, ACH) ready. Set transaction fees high enough to ensure timely processing during congestion.
Counterparty Performance Risk	On-ramp or off-ramp provider delays or fails to convert fiat ↔ stablecoin; payment doesn't reach recipient's bank account on time.	Due diligence on partners. Define SLAs. Test end-to-end in pilot. Build in buffer time for critical payments.
Irreversibility Risk	Payment sent to wrong address, or to a fraudulent counterparty; no way to reverse or recover funds.	Strengthen pre-payment controls: wallet address verification (use registries or test transactions), counterparty KYC, approval workflows, payment limits.
Regulatory / Legal Risk	Regulatory status of stablecoins changes; issuer faces enforcement action; legal claim on reserves is unclear.	Stay informed on regulatory developments. Choose issuers with clear legal structure and regulatory engagement. Consult legal counsel before scaling.
FX / Conversion Risk	If using non-USD stablecoins or converting between stablecoin and local currency, exchange rate risk during the conversion window.	Lock in rates where possible. Use fiat-to-fiat stablecoin models (issuer takes FX risk). Monitor conversion spreads.

TIP: Use this table as a checklist during partner evaluation and pilot design. For each risk, ask: "Do we have a mitigation in place? What's our residual risk tolerance? What evidence will we need to show we managed this under audit?"

What You've Learned (and What's Next)

You now have a working understanding of stablecoins in treasury terms:

- They're digital tokens designed to hold a 1:1 peg with fiat - an asset you may hold transiently (with issuer/liquidity/legal risk) that moves over a blockchain rail (with operational dependencies).
- The peg holds when reserves are sound, redemptions work, and confidence remains high. It breaks when any of those conditions fail.
- Settlement is fast and final - a feature for certainty, a risk for errors.
- Risk management requires evaluating issuer credit quality, reserve transparency, custody arrangements, and blockchain network reliability.

What this chapter did NOT cover:

- How to build governance and evidence chains around stablecoin payments (that's Chapter 3).
- Which stablecoin payment models make sense for which use cases (that's Chapter 4).
- How to run a pilot without creating governance debt (that's Chapter 5).

Next, we'll move from "what stablecoins are" to "how to govern them."

Understanding the stablecoin asset is necessary, but it's not sufficient. Treasury's job is to ensure that every payment - no matter what blockchain rail it uses - is approved, executed, and reconciled within policy, with evidence that survives audit.

That's where governance comes in.

Chapter 3 **Governance & Evidence**

Why This Chapter Matters

You understand what stablecoins are (Chapter 2). You understand why governance must come first (Chapter 1). Now it's time to build the governance framework that makes stablecoin payments operationally sound and audit-ready.

This chapter is practical, not theoretical. You'll learn how to:

- **Map your dependency chain:** Identify every party and system that must work for a payment to succeed
- **Design controlled fallbacks:** Define what happens when a dependency fails, and who owns the decision.
- **Build an evidence chain:** Connect approvals, execution, settlement, and reconciliation in a way that proves governance under audit.
- **Establish exception ownership:** Ensure someone can act at 3 a.m. (or escalate appropriately).
- **Pass the audit test:** Know what your auditor will ask, and have answers ready.

By the end of this chapter, you'll have three practical checklists you can use to assess whether a stablecoin payment model is governance-ready.

One core principle runs through this entire chapter:

Visibility is not the same as provability.

You can see a transaction hash on a blockchain explorer. You can see a wallet balance. You can see a payment confirmation. That's *visibility.*

But can you prove - to an auditor, a regulator, or your CFO - that the payment was approved by the right person, executed within policy, reconciled correctly, and handled appropriately when an exception occurred?

That's *provability.* And it requires governance and evidence, not just data.

Dependency-Chain Mapping: Who and What Must Work

Every payment system - traditional or on-chain - depends on multiple parties and systems working correctly. When treasury teams evaluate SWIFT or ACH, they intuitively understand the dependency chain: the ERP must generate the payment file, the TMS must enforce approvals, the bank must execute the wire, SWIFT must route the message, the beneficiary bank must credit the account.

Stablecoin payments introduce new dependencies - and most teams underestimate how many there are.

1. ERP / AP system generates the payment instruction.
2. TMS or approval workflow enforces policy and captures approvals.
3. On-ramp provider converts your fiat (USD) into stablecoin (USDC) and credits your wallet.
4. Wallet custody provider holds your private keys and executes the on-chain transaction on your instruction.
5. Blockchain network (e.g., Ethereum, Solana) processes and confirms the transaction.
6. Off-ramp provider (on the recipient's side) converts the stablecoin back to fiat and deposits it in the recipient's bank account.
7. Recipient's bank credits their account.
8. Your reconciliation process matches the outgoing payment in your ERP to the settled transaction (using transaction hashes, confirmations, and bank statements).

That's at least eight dependencies. If any one of them fails, delays, or behaves unexpectedly, your payment doesn't complete as planned.

ALTERNATIVE OPERATING MODEL: PAYMENT SERVICE PROVIDER

Some treasury teams simplify this dependency chain by working with a stablecoin Payment Service Provider (PSP) that handles on-ramp, custody, execution, off-ramp, and reconciliation data on your behalf. From treasury's perspective, the PSP acts like a specialized bank: you send payment instructions (via API or portal), the PSP executes the stablecoin payment end-to-end, and you receive settlement confirmations.

This consolidates dependencies but doesn't eliminate governance requirements. You're now dependent on one provider instead of five - which simplifies operations but concentrates risk. Your governance model must still cover: approvals before instructions are sent to the PSP, SoD within your team, evidence capture (the PSP must provide audit-grade logs), exception ownership (what happens if the PSP has an outage?), and exit rights (can you recover funds and switch providers if needed?).

The rest of this chapter applies whether you're managing the stack directly or working through a PSP. The dependency map gets simpler, but the governance questions remain the same.

Examples of stablecoin PSPs include providers like Brale, Bridge, Fipto, and Monerium (EUR-focused). These platforms differ in their supported currencies, corridors, custody models, and integration options - but all follow the general pattern of consolidating on-ramp, custody, execution, and off-ramp into a single service. Some PSPs also support traditional payment rails alongside stablecoins, letting you route payments through the most cost-effective or time-sensitive option without changing your approval workflow.

WHY MAPPING DEPENDENCIES MATTERS

Most pilots focus on the "happy path" - when everything works.

But treasury operations are defined by how you handle the exceptions: the payment that gets stuck, the provider that goes offline, the wallet that shows a pending transaction for six hours.

If you haven't mapped your dependencies, you can't:

- Identify single points of failure.
- Design fallback options.
- Assign exception ownership.
- Set realistic SLAs for payment completion.

TIP: Start your dependency map by asking, "What has to happen, in order, for this payment to go from approved in our ERP to settled in the recipient's bank account?" Write down every step. Then ask, "Who operates each step, and what could cause it to fail?"

Drawing Your Dependency Map (Step-by-Step)

Here's a practical method for mapping dependencies for any stablecoin payment model.

STEP 1: IDENTIFY THE PARTIES

List every entity involved in the payment flow, from initiation to final settlement.

Example (fiat → stablecoin → fiat model):

- Your company (treasury / AP)
- Your ERP vendor
- Your TMS or workflow tool
- Your on-ramp provider (converts fiat → stablecoin)
- Your wallet custody provider
- The blockchain network
- Recipient's wallet custody provider (if applicable)
- Recipient's off-ramp provider (converts stablecoin → fiat)
- Recipient's bank
- Your bank (for reconciliation, if you're tracking fiat outflows)

STEP 2: MAP THE HANDOFFS

Draw a flow diagram (or just a numbered list) showing where value and data move between parties.

Example:

1. AP clerk enters invoice and payment details in ERP.
2. Treasury manager approves payment in TMS.
3. Payment instruction is sent to the on-ramp provider (API call, file upload, or manual portal entry).
4. On-ramp provider debits your bank account and credits your stablecoin wallet.
5. Treasury ops instructs wallet custody provider to send stablecoin to recipient's wallet address.
6. Wallet custody provider signs and broadcasts the transaction to the blockchain.
7. Blockchain confirms the transaction (finality achieved).
8. Recipient's off-ramp provider detects the incoming stablecoin, converts to fiat, and initiates bank deposit.
9. Recipient's bank credits their account.
10. Your AP team matches the payment to the invoice using transaction hash and confirmation data.

STEP 3: IDENTIFY WHAT COULD FAIL AT EACH STEP

For each handoff, ask: "What could prevent this step from completing successfully?"

Example:

- **Step 3 (send instruction to on-ramp):** API goes down; authentication fails; rate limit hit.
- **Step 4 (on-ramp converts fiat → stablecoin):** Insufficient liquidity at on-ramp; your bank account has insufficient funds; on-ramp's bank has an outage.
- **Step 6 (wallet custody provider broadcasts transaction):** Custody provider's systems are offline for maintenance; transaction fee (gas) set too low, transaction doesn't confirm; private key access issue.
- **Step 7 (blockchain confirms):** Network congestion delays confirmation; blockchain hard fork creates ambiguity; smart contract bug (if applicable).
- **Step 8 (off-ramp converts stablecoin → fiat):** Off-ramp provider doesn't recognize the incoming transaction (wrong memo field, wrong wallet address); off-ramp has KYC hold on recipient; off-ramp has liquidity constraints.

STEP 4: ASSIGN OWNERSHIP FOR EACH DEPENDENCY

For every dependency, document:

- **Who operates it?** (Your team, a vendor, a third party, decentralized infrastructure)
- **Who is accountable if it fails?** (Who does your team call or escalate to?)
- **What's the SLA or expected resolution time?**

DEPENDENCY	OPERATED BY	ACCOUNTABLE PARTY	SLA / RESPONSE TIME
On-ramp API	On-ramp provider	Provider's support team	4-hour response (business hours)
Blockchain network	Decentralized (Ethereum validators)	No single party	No SLA (monitor status via block explorers)
Wallet custody provider	Custody vendor	Vendor support + your admin	1-hour response (24/7 for Tier 1 issues)

If you can't assign clear ownership and accountability for a dependency, you've found a governance gap.

WARNING: Decentralized infrastructure (like blockchain networks) has no customer support. You can't call Ethereum's help desk. If the network is congested or down, you wait - or you switch to a fallback rail. Plan accordingly.

Controlled Fallbacks: What Happens When a Dependency Fails

A “controlled fallback” is a pre-defined alternative path you take when a primary dependency fails. The goal is to complete the payment obligation (or safely abort and retry) without breaking segregation of duties, losing evidence, or creating reconciliation chaos.

Controlled fallbacks are not “Plan B.” They’re part of Plan A.

DESIGNING FALLBACKS: THREE PRINCIPLES

1. Pre-define the trigger conditions.

Don’t wait until an exception happens to decide whether to retry, switch rails, or escalate. Document the conditions under which you’ll invoke the fallback.

Examples:

- If on-chain transaction is pending for >2 hours, escalate to custody provider.
- If custody provider confirms system outage, switch to SWIFT fallback and notify AP.
- If recipient reports non-receipt after 4 hours and blockchain explorer shows confirmed transaction, escalate to off-ramp provider.

2. Preserve governance and evidence.

Your fallback path must use the same approval, the same data, and the same audit trail as the primary path. You’re changing the execution rail, not bypassing controls.

Example (good fallback):

- Payment was approved in TMS for $50,000 to Supplier X.
- On-ramp provider experiences an outage.
- Treasury ops triggers pre-approved SWIFT fallback using the same payment data and approval reference.
- AP reconciliation process notes: “Payment #12345 executed via SWIFT fallback due to on-ramp outage (ticket #6789).”

Example (bad fallback):

- Payment was approved in TMS for $50,000 to Supplier X.
- On-ramp provider experiences an outage.
- Treasury manager manually wires $50,000 via online banking portal “to get it done.”
- No documentation of who approved the fallback decision, no link to the original approval, reconciliation is manual and error-prone.

3. Plan for “double execution” scenarios.

Because stablecoin transactions are irreversible, and because some failures are delays rather than true failures, you can end up in a scenario where:

- The on-chain payment eventually succeeds (after a delay).
- But you’ve already executed the SWIFT fallback.
- Now the supplier has been paid twice.

Your fallback plan must address this:

- Can you cancel or recall the on-chain transaction before executing the fallback? (Usually no - once broadcast, it’s in flight.)
- Can you detect duplicate payments in your reconciliation process? (Yes - this is where transaction hash logging is critical.)
- Do you have a process for recovering duplicate payments from suppliers? (You should.)

REMEMBER: ”Controlled” means you’ve thought through the failure mode, documented the decision rules, assigned ownership, and preserved evidence. If your fallback plan is “figure it out in the moment,” it’s not controlled.

The Evidence Chain: Approvals → Execution → Settlement → Reconciliation

An auditor doesn't care that your payment was fast or cheap. They care that you can prove it was:

1. **Authorized** (approved by someone with delegated authority, within limits).
2. **Executed correctly** (instruction matched approval; sent to the right party).
3. **Settled as intended** (value transferred; recipient received funds).
4. **Reconciled accurately** (your books reflect what actually happened; no unexplained discrepancies).

This is the evidence chain. Every link must be documented, tamper-evident, and auditable.

Stablecoin payments can actually strengthen evidence chains (blockchain records are immutable and independently verifiable), but only if you design the chain intentionally.

LINK 1: APPROVAL EVIDENCE

What you need to prove:

- Who approved the payment?
- When?
- What were the approved terms (amount, recipient, purpose)?
- Was the approver authorized under your policy?

Where this evidence lives:

- Your ERP or TMS approval workflow logs.
- Digitally signed approval records (if your system supports it).
- Email confirmations, if policy requires (less ideal but sometimes necessary).

On-chain connection: Some custody providers let you attach metadata (approval reference numbers, invoice IDs) to on-chain transactions. If available, use it - it creates a direct link between your internal approval and the blockchain transaction.

LINK 2: EXECUTION EVIDENCE

What you need to prove:

- The payment instruction matched the approval (amount, recipient, timing).
- The instruction was sent to the custody provider or wallet securely (no unauthorized changes).
- The transaction was signed and broadcast to the blockchain.

Where this evidence lives:

- API logs showing payment instruction sent to custody provider.
- Custody provider's transaction log (who initiated, who approved if multi-sig, timestamp).
- Blockchain transaction hash (unique identifier for the on-chain transaction).

Key point: The transaction hash is your anchor. It's a tamper-proof reference that anyone can verify on a public blockchain explorer. Log it in your ERP/TMS as soon as it's available.

LINK 3: SETTLEMENT EVIDENCE

What you need to prove:

- The transaction achieved finality (was confirmed on-chain).
- The recipient's wallet or off-ramp provider received the funds.
- The recipient's bank account was credited (if the model involves fiat conversion).

Where this evidence lives:

- Blockchain explorer confirmation (number of confirmations, block number, timestamp).
- Custody provider's settlement report.
- Off-ramp provider's confirmation (if applicable): "USDC received, USD deposited to recipient's bank account #123."
- Recipient's confirmation (email, portal acknowledgment, or invoice marked paid).

Where this evidence lives:

- Blockchain explorer confirmation (number of confirmations, block number, timestamp).

Best Practice: Automate the capture of transaction hashes and settlement confirmations into your TMS or reconciliation tool. Manual copy-paste from blockchain explorers doesn't scale and introduces error risk.

LINK 4: RECONCILIATION EVIDENCE

What you need to prove:

- The payment in your ERP/AP system matches the on-chain transaction and the final settlement.
- Any discrepancies (timing differences, FX variances, fee deductions) are explained and documented.
- Your cash position and ledger balances are accurate.

Where this evidence lives:

- Your ERP's reconciliation module (matching invoice → payment → settlement).
- Bank statements (if fiat on-ramp or off-ramp steps involved).
- Blockchain transaction records cross-referenced to ERP payment IDs.

Common reconciliation challenge with stablecoins: Traditional bank reconciliation matches your ledger to a bank statement. With stablecoins, you may need to reconcile:

- Fiat out (to on-ramp provider).
- Stablecoin transaction (on-chain).
- Fiat in (at off-ramp provider, to recipient's bank).

Each step might happen at a different time, on a different date, and involve different entities. Your reconciliation process must be able to tie all three together using transaction hashes, timestamps, and reference IDs.

TIP: Build a "transaction reference table" that logs: **ERP payment ID | TMS approval ID | On-ramp reference | Transaction hash | Off-ramp confirmation | Settlement date**. This reference table becomes your single source of truth for reconciliation and audit.

Segregation of Duties On-Chain (Yes, It's Possible)

One of the most common concerns treasury teams raise about stablecoin payments: "If a private key can move millions of dollars instantly, how do we enforce segregation of duties?"

Great question. And the answer is: the same way you do with traditional payments - by separating approval from execution and by using technology controls to enforce the separation.

HOW SOD WORKS IN TRADITIONAL PAYMENTS

In a typical corporate payment workflow:

- AP clerk enters the invoice and payment details (initiation).
- Treasury manager reviews and approves (approval).
- Treasury operations or an automated process sends the payment file to the bank (execution).
- The bank validates the authorized signers and executes the wire (settlement).

No single person can initiate, approve, and execute alone.

HOW SOD WORKS WITH STABLECOIN PAYMENTS

The same separation applies, but the tools and controls look different.

Option 1: Multi-Signature Wallets

A multi-signature (multi-sig) wallet requires multiple private keys to authorize a transaction. For example, a 2-of-3 multi-sig wallet requires any 2 out of 3 designated keyholders to sign a transaction before it's broadcast to the blockchain.

Treasury application:

- Treasury manager holds key 1 (approval authority).
- Treasury ops holds key 2 (execution authority).
- CFO or compliance holds key 3 (escalation/override authority).
- Any transaction requires 2 signatures - enforcing SoD at the wallet level.

Trade-off: Multi-sig adds operational complexity (coordinating signatures, key management). It's appropriate for high-value or high-risk payment flows, less necessary for low-value, high-frequency payments.

Option 2: Custody Provider with Workflow Controls

Institutional custody providers (e.g., Coinbase Custody, BitGo, Anchorage) offer policy engines that let you define approval workflows, spending limits, and role-based access controls.

Example:

- **Payments under $10,000:** Single approver (treasury ops).
- **Payments $10,000-$100,000:** Two approvers (treasury ops + treasury manager).
- **Payments over $100,000:** Three approvers (treasury ops + treasury manager + CFO).
- All transactions logged with approver identity, timestamp, and IP address.

This mirrors the approval matrix you'd configure in a TMS - but applied to wallet transactions.

Option 3: Integration Layer with ERP/TMS Approval

Some integration platforms let you keep your existing ERP or TMS approval workflow and only send approved payment instructions to the wallet custody provider.

Example:

- Payment is entered and approved in your TMS (your existing SoD controls apply).
- Once approved, TMS sends a signed API call to the custody provider with payment details and approval metadata.
- Custody provider executes the on-chain transaction (no additional human approval required, because the TMS approval satisfies your policy).

This approach is often the cleanest for treasury teams: Keep the governance layer you already trust (your TMS), and treat the custody provider as an execution agent (like you treat your bank)

Option 4: PSP with Embedded Approval Workflow

If you're working with a stablecoin PSP, the provider may offer its own approval workflow engine (similar to what banks offer for wire approvals).

You configure your approval matrix in the PSP's platform, and your team approves payments there before execution.

Treasury application: This scenario can work - but make sure the PSP's controls match your policy (role-based access, approval limits, audit logs, no single-user overrides).

And ensure you can export approval evidence for audit (user ID, timestamp, approved terms).

If the PSP can't provide this, fall back to Option 3: keep approvals in your TMS and send only approved instructions to the PSP.

WARNING: If your custody provider gives a single admin user unrestricted access to move funds, you have a segregation-of-duties gap. Make sure your custody arrangement supports role-based permissions, approval workflows, and audit logs before you fund the wallet.

Exception Ownership: Who Decides What at 3 a.m.

Let's return to the scenario from Chapter 1: a $2 million payment is stuck in "pending" status at 3 a.m. on a Saturday. Who can act? Who is authorized to decide whether to retry, escalate, switch to a fallback, or wait?

If the answer is "I don't know," your governance model isn't done.

Exception ownership means documenting:

1. What qualifies as an exception (payment pending >X hours, counterparty reports non-receipt, provider outage, etc.).
2. Who owns the decision for each type of exception (treasury ops, treasury manager, CFO, custody provider support).
3. What actions are pre-authorized (retry with higher gas fee, escalate to provider, switch to SWIFT fallback).
4. What requires escalation (amounts above certain thresholds, exceptions involving suspected fraud, provider failures with no ETA).
5. How decisions are documented (ticket system, email log, TMS notes, incident report).

BUILDING AN EXCEPTION OWNERSHIP MATRIX

Here's a simple framework:

EXCEPTION TYPE	OWNER (FIRST RESPONSE)	ESCALATION PATH	PRE-AUTHORIZED ACTIONS	PRE-DOCUMENTATION REQUIRED
Payment pending >2 hours	Treasury ops	Treasury manager (if >4 hours)	Check blockchain explorer; contact custody provider; increase gas fee and retry	Log ticket # and outcome in TMS
Custody provider system outage	Treasury ops	Treasury manager	Switch to pre-approved SWIFT fallback if payment is time-sensitive	Log fallback execution; note original approval reference
Recipient reports non-receipt (but blockchain shows "confirmed")	Treasury ops	Off-ramp provider + treasury manager	Provide transaction hash to recipient; escalate to off-ramp provider if no resolution in 2 hours	Email trail + ticket #
Suspected fraudulent recipient wallet address	Treasury manager	CFO + compliance	Do not execute payment; freeze wallet if possible; investigate	Incident report required
Payment >$500K stuck or delayed	Treasury manager	CFO	Escalate immediately; no retry without CFO approval	Executive summary + incident timeline

If you're using a PSP: Most of your operational exceptions (pending payments, custody issues, blockchain congestion) are handled by the PSP's support team. But you still own the governance exceptions: Who authorized the payment? What do we do if the PSP has a multi-hour outage? When do we switch to a fallback rail? The PSP can't answer those questions for you - those decisions stay with treasury.

The goal: Anyone on call (or anyone who gets pulled into a 3 a.m. issue) can look at this matrix and know what they're authorized to do, when to escalate, and how to document the decision.

TIP: Test your exception ownership matrix during the pilot. Simulate a stuck payment, a provider outage, a wallet address typo. Does your team know what to do? Can they execute the fallback without breaking SoD or losing the audit trail? If not, refine the matrix before you scale.

Escalation Paths and Communication Protocols

Exception ownership tells you who decides. Escalation paths tell you how to get the right person involved, fast.

KEY ELEMENTS OF AN EFFECTIVE ESCALATION PATH

1. **Contact list with roles, not just names.** People change jobs; roles persist. ("Escalate to Treasury Manager" is better than "Escalate to Jane.")
2. **Primary and backup contacts.** If the treasury manager is unreachable, who's next?
3. **Support contacts for every dependency**. Custody provider support line, on-ramp provider account manager, off-ramp provider escalation email, blockchain network status page URL.
4. **Communication channels.** Phone, email, Slack/Teams, SMS for urgent escalations.
5. **Expected response times.** Set SLAs with vendors in advance: "Critical payment issues will receive a response within 1 hour, 24/7."

EXAMPLE ESCALATION PATH (PAYMENT PENDING >4 HOURS)

1. Treasury ops confirms exception (checks blockchain explorer, custody provider dashboard).
2. Treasury ops opens ticket with custody provider support; logs ticket # in TMS.
3. If no response in 30 minutes, treasury ops escalates to treasury manager via phone + Slack.
4. Treasury manager evaluates: Can we retry? Switch to fallback? Wait for provider?
5. If decision is "switch to fallback," treasury manager approves fallback execution in TMS; treasury ops executes SWIFT payment.
6. Treasury ops documents outcome: original payment eventually confirmed (note duplicate risk) or original payment canceled.)

The point: No one is improvising at 3 a.m. The playbook is written in advance.

Audit-Readiness: What Your Auditor Will Ask

Internal and external auditors evaluate payment systems by testing whether controls are designed effectively and operating as intended. Here's what they'll ask about stablecoin payments - and what you need to have ready.

Question 1: "How do you ensure payments are approved by authorized individuals?"

What they're testing: Segregation of duties; approval workflows; authorization limits.

What you need to show:

- Your approval policy (who can approve what, up to what limits).
- TMS or ERP logs showing approvals, with user IDs and timestamps.
- Evidence custody provider enforces your approval rules (multi-sig config, policy engine settings, API authentication).

Red flag for auditors: A single person can approve and execute a $1 million payment with no secondary control.

Question 2: "How do you know the payment instruction matched the approved amount and recipient?"

What they're testing: Accuracy; prevention of unauthorized changes between approval and execution.

What you need to show:

- API logs or file transfer logs showing payment instruction sent to custody provider.
- Transaction hash linking the on-chain transaction to the approved payment in your ERP.
- No manual re-keying of payment details (data flows automatically from ERP → TMS → custody provider).

Red flag for auditors: Payment details are manually entered into a wallet interface after approval (opportunity for error or fraud).

Question 3: "How do you reconcile stablecoin payments to your general ledger?"

What they're testing: Completeness and accuracy of recorded transactions; cash position integrity.

What you need to show:

- Reconciliation process that ties ERP payments → blockchain transactions → bank statements (if fiat conversions involved).
- Transaction reference table (ERP payment ID, transaction hash, settlement date, amounts).
- Variance analysis for any discrepancies (FX differences, fees, timing).

Red flag for auditors: "We reconcile stablecoin payments manually in a spreadsheet once a month."

Question 4: "What happens when a payment fails or gets stuck?"

What they're testing: Exception handling; residual risk from operational failures.

What you need to show:

- Exception ownership matrix.
- Documented escalation paths.
- Incident logs showing how past exceptions were resolved.
- Evidence that fallback controls preserve SoD and audit trail.

Red flag for auditors: "It hasn't happened yet, so we don't have a documented process."

Question 5: "What are the risks associated with your custody provider or PSP, and how do you mitigate them?"

What they're testing: Third-party risk management; business continuity.

What you need to show:

- Vendor due diligence documentation (financial stability, insurance, security certifications).
- Contract provisions (SLAs, liability, data access for audit).
- Contingency plan if custody provider fails or exits the business (how do you recover access to funds?).

Red flag for auditors: "We trust the vendor; we haven't documented the risks."

If you're using a stablecoin PSP, the auditor will treat this as concentrated third-party risk: the PSP handles custody, execution, and possibly reconciliation data. You'll need to show deeper due diligence (financial stability, insurance, disaster recovery, data access for audit, exit plan).

REMEMBER: Auditors don't expect perfection. They expect evidence of control. If you've documented your governance model, tested it during the pilot, and can show that controls operated as designed, you'll pass audit - even if you had exceptions along the way.

Three Practical Checklists

Use these checklists to assess whether your stablecoin payment model is governance-ready.

CHECKLIST 1: PRE-PILOT GOVERNANCE READINESS

- ☐ **Dependency chain is fully mapped** (all parties, all handoffs, all failure modes identified).
- ☐ **Exception ownership matrix is documented and communicated to the team.**
- ☐ **Escalation paths are defined, with primary and backup contacts for every dependency.**
- ☐ **Controlled fallback options are designed and tested** (trigger conditions, decision rules, evidence preservation).
- ☐ **Approval workflows enforce segregation of duties** (no single person can approve and execute without a secondary control).
- ☐ Custody provider supports role-based access, approval policies, and audit logs.
- ☐ **Transaction reference logging is automated** (ERP payment ID → transaction hash → settlement confirmation).
- ☐ **Reconciliation process can tie ERP payments to on-chain transactions and bank statements.**
- ☐ **Vendor due diligence is complete for all critical dependencies** (custody provider, on-ramp, off-ramp).
- ☐ **Legal and compliance have reviewed and approved the pilot design.**

If you can't check all the boxes, you're not ready to go live - even in a pilot.

CHECKLIST 2: EXCEPTION-HANDLING READINESS

- ☐ **We have documented what qualifies as an exception**
  (payment pending >X hours, provider outage, non-receipt report, suspected fraud).
- ☐ **We know who owns the decision for each exception type** (role, not just a name).
- ☐ **We have pre-authorized actions for common exceptions** (retry, escalate, switch to fallback).
- ☐ **We have tested our escalation process**
  (simulated a stuck payment, confirmed we can reach the right people).
- ☐ **We have a communication protocol for exceptions** (phone, email, ticketing system).
- ☐ **We have a fallback rail that can be activated without breaking SoD or losing audit trail.**
- ☐ **We have a process for documenting exception decisions**
  (ticket #, TMS notes, incident report).
- ☐ **We have a process for reconciling payments that were delayed or re-routed**
  (handling timing differences, detecting duplicates).
- ☐ **We have tested our exception process during the pilot**
  (forced a failure, validated our response).

If you haven't tested exception handling, you don't know if your governance model works under pressure.

CHECKLIST 3: EVIDENCE-PACK READINESS (AUDIT TEST)

For a sample stablecoin payment, can you produce the following evidence on demand?

- ☐ Who approved the payment, when, and under what authority? (ERP/TMS log)
- ☐ What were the approved payment terms? (Amount, recipient, purpose, timing)
- ☐ What instruction was sent to the custody provider? (API log, file transfer log)
- ☐ What transaction was executed on-chain? (Transaction hash, block number, timestamp)
- ☐ Was the transaction confirmed and finalized?
  (Blockchain explorer confirmation, custody provider report)
- ☐ Did the recipient receive the funds? (Off-ramp confirmation, recipient acknowledgment)
- ☐ How was the payment reconciled? (ERP payment ID matched to transaction hash and settlement date)
- ☐ Were there any exceptions, and how were they resolved?
  (Incident log, escalation record, fallback execution)
- ☐ What fees were incurred, and how were they accounted for?
  (On-ramp fee, gas fee, off-ramp fee, FX spread, stablecoin liquidity spread)

Note on liquidity spreads: In emerging markets or less-liquid corridors, off-ramp providers may charge wider spreads when converting stablecoins to local fiat - especially if stablecoin liquidity in that currency is thin. Track this separately from FX spread so you can compare true all-in costs across payment rails and identify corridors where stablecoin rails are less competitive.

If you can produce this evidence pack for every payment, you're audit-ready.

What You've Learned (And Why It Matters)

You now have the tools to design governance into stablecoin payments from day one:

- **Dependency mapping** ensures you know what can fail and who owns the fix.
- **Controlled fallbacks** give you options when things go wrong, without breaking controls
- **Evidence chains** connect approval → execution → settlement → reconciliation in a way that proves governance under audit.
- **Exception ownership** means someone can act at 3 a.m. (or escalate appropriately).
- **Audit-readiness** means you can answer the hard questions with documentation, not hope.

The goal isn't perfection. The goal is control.

You will have exceptions. Payments will get stuck. Providers will have outages. The blockchain network will get congested. That's normal. What's not normal - and not acceptable - is improvising your response under pressure, losing the audit trail, or breaking segregation of duties because "it was an emergency."

Governance isn't a compliance checkbox. It's the design constraint that makes stablecoin payments operationally sound.

And now that you have the governance foundation, you're ready to evaluate specific payment models - which is where we go next.

Chapter 4 Six Stablecoin Payment Models

Why Payment Models Differ

Not all stablecoin payments work the same way. The differences aren't just technical details. They determine who takes what risk, where governance controls must be applied, and which use cases make sense for treasury.

Three variables define the payment model:

1. **Conversion points:** Where does fiat become stablecoin, and where does stablecoin become fiat? At your end, at the recipient's end, or both?
2. **Custody:** Who holds the stablecoin wallet: your treasury team, a third-party provider, the recipient, or no one (if the stablecoin never actually sits in a wallet for more than seconds)?
3. **Dependencies:** How many parties and systems must work for the payment to succeed? More dependencies = more operational complexity and more potential failure points.

These three variables create distinct payment models - each with different governance requirements, different pilot-readiness profiles, and different risk/reward trade-offs.

This chapter walks through six models, structured as "model cards" for easy comparison.

Each card answers:

- **How does it work (step-by-step)?**
- **What's the value proposition?**
- **What are the trade-offs?**
- **What are the top 3 governance focus areas?**
- **Who is this model a good pilot fit for?**
- **What partners enable this model (illustrative examples, not endorsements)?**

By the end of this chapter, you'll know which model(s) to explore based on your priorities - and you'll have a decision tree to guide your next steps.

How to Read the Model Cards

Each of the six models is presented with the same structure, so you can compare apples to apples.

How It Works: A step-by-step flow from payment initiation to settlement.

Value Proposition: Why would treasury consider this model? What problems does it solve?

Trade-offs: What are the downsides, risks, or constraints?

Governance Focus (Top 3): Where should governance and control efforts concentrate?

Pilot Fit: What types of companies, use cases, or corridors is this model best suited for?

Illustrative Partners: Example vendors or service providers who enable this model (not exhaustive; not an endorsement; for awareness only).

REMEMBER: No single model is "best." The right model depends on your priorities (speed, cost, counterparty preferences, governance complexity), your risk tolerance, and your operational readiness. Pilots often test 1-2 models in a narrow corridor before scaling.

Model 1: Fiat → Stablecoin → Fiat (The "Sandwich")

HOW IT WORKS

The "stablecoin sandwich" is the most common introductory model for corporate treasury. Stablecoin is used only as the settlement rail. Both the payer (you) and the payee (supplier, contractor, or beneficiary) interact in fiat currency. The stablecoin conversion happens behind the scenes.

Step-by-step:

1. You initiate a payment in your ERP/TMS: "Pay Supplier X $50,000 USD."
2. Your payment is approved per your standard workflow (treasury manager, dual approval, etc.).
3. Payment instruction is sent to an on-ramp provider (or your bank, if they offer this service).
4. On-ramp provider debits your bank account for $50,000 USD (plus fees).
5. On-ramp provider converts $50,000 USD → 50,000 USD in stablecoin (e.g., USDC, USDT, or others) and holds it briefly in a custodial wallet.
6. On-ramp provider (or a connected partner) transfers the stablecoin on-chain to an off-ramp provider in the recipient's country.
7. Off-ramp provider receives the stablecoin, converts it to the recipient's local currency (or USD), and deposits the fiat into Supplier X's bank account.
8. Supplier X receives $50,000 (or local currency equivalent) in their bank account - never sees or holds stablecoin.
9. You reconcile the payment using a transaction reference or hash provided by the on-ramp provider.

Key characteristic: Neither you nor the recipient holds a stablecoin wallet or takes custody of the asset. It's fiat in, stablecoin in the middle, fiat out - hence "sandwich."

VALUE PROPOSITION

- **No wallet required:** Your treasury team doesn't need to manage private keys, custody arrangements, or stablecoin balances. The provider handles it.
- **No recipient friction:** Suppliers don't need wallets or crypto knowledge. They receive fiat in their bank account, just like a traditional wire or ACH payment.
- **Faster settlement (in some corridors):** Particularly effective for cross-border payments where traditional correspondent banking is slow (e.g., US → Philippines, US → Latin America).
- **Lower fees (in some corridors):** Depending on the route, total cost (your fee + recipient fee) may be lower than SWIFT + correspondent bank charges.
- **Pilot-friendly:** Because it resembles traditional payment flows (you send fiat, recipient receives fiat), it's easier to pilot without requiring organizational change.

TRADE-OFFS

- **Opaque conversion:** You may not see the on-chain transaction or control the conversion timing/rate. You're relying on the provider's liquidity and pricing.
- **Dependency on two providers:** Both on-ramp and off-ramp must work correctly. If the off-ramp provider in the recipient's country has KYC delays or liquidity issues, the payment doesn't complete - even though you've already debited your account.
- **Limited transparency:** Unlike direct USDC payments (Model 3), you may not have real-time visibility into where the payment is in the flow. You're waiting for the off-ramp provider to confirm final settlement.
- **FX and liquidity risk (if cross-currency):** If the recipient receives local currency, the FX conversion happens at the off-ramp provider's rate - and you may not lock in the rate at payment initiation. More importantly, in emerging markets or less-liquid corridors, the off-ramp provider may face thin stablecoin liquidity in the local currency market, resulting in wider spreads, higher costs, or conversion delays. This liquidity risk is separate from FX volatility and should be assessed corridor-by-corridor during vendor selection.
- **Still feels like a black box to treasury:** You don't control the rail; the provider does. If something goes wrong, you're escalating to vendor support, not troubleshooting internally.

GOVERNANCE FOCUS (TOP 3)

1. **Vendor due diligence (on-ramp and off-ramp providers):** Vet their financial stability, regulatory standing, insurance coverage, and SLAs. Both providers are critical dependencies.
2. **Evidence chain from fiat out to fiat in:** Ensure you can tie your bank debit → provider reference → on-chain transaction (if visible) → recipient bank credit. Reconciliation must work even if you don't see the blockchain transaction directly.
3. **Exception handling for incomplete payments:** Define what happens if the off-ramp provider delays or fails. Who owns the escalation? Can you get the funds back if the recipient never receives them? Document this in your vendor contract.

PILOT FIT

Best for:

- Treasury teams who want the benefits of stablecoin rails (speed, cost) but aren't ready to manage wallets or custody internally.
- Cross-border supplier payments in corridors where traditional rails are slow or expensive (e.g., US to Southeast Asia, Latin America, Africa).
- Companies with low transaction volumes (dozens per month, not thousands) who want a managed service approach.

Not ideal for:

- High-frequency payment flows (fee per transaction adds up; Model 4 may be better).
- Payments where you need real-time, second-by-second visibility into settlement status (you're relying on provider reporting).
- Situations where the recipient is already set up to receive USDC directly (Model 3 is simpler and cheaper).

ILLUSTRATIVE PARTNERS (NOT ENDORSEMENTS)

- **On-ramp/off-ramp providers:** Bitso, Orbital - offer fiat-to-stablecoin and stablecoin-to-fiat conversion services.
- **Integrated stablecoin PSPs:** Bridge, Fipto - handle end-to-end payment execution (on-ramp, custody, settlement, off-ramp) via treasury API.
- **Banks piloting stablecoin rails:** J.P. Morgan (JPM Coin for institutional clients) - "fiat in, tokenized rail, fiat out" models.
- **Payment API platforms:** Stripe (USDC payment support announced), Nium, Thunes - may enable stablecoin-backed cross-border payments.

TIP (Model 1): Start here if your priority is "test the concept with minimal internal change." You can pilot Model 1 without training your team on wallets, keys, or blockchain explorers. Once you prove value, you can explore more direct models (3 or 4) that give you more control.

Model 2: Fiat → Stablecoin (Payee Holds)

HOW IT WORKS

In this model, you (the payer) convert fiat to stablecoin and send it directly to the recipient's wallet. The recipient holds the stablecoin and decides when (or whether) to convert it back to fiat.

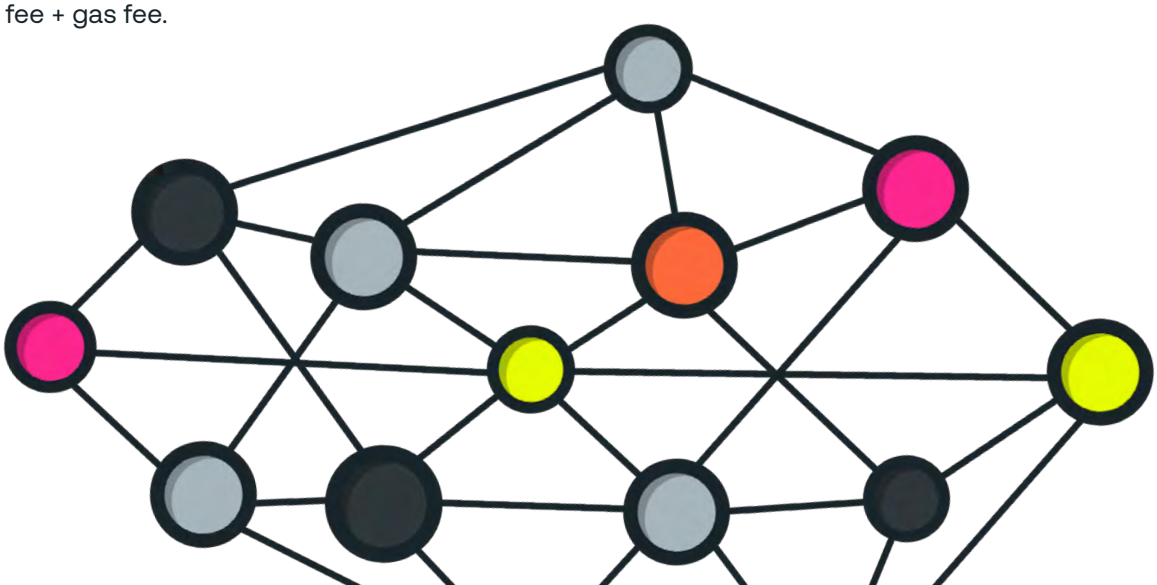
Step-by-step:

1. You initiate a payment: "Pay Contractor Y 10,000 USDC."
2. Payment is approved per your workflow.
3. Your on-ramp provider (or your wallet, if you hold a stablecoin balance) converts $10,000 USD → 10,000 USDC.
4. You send 10,000 USDC to Contractor Y's wallet address (provided by the contractor).
5. On-chain transaction confirms (seconds to minutes).
6. Contractor Y's wallet now holds 10,000 USDC. They can:
   - Hold it (if they want stablecoin exposure or plan to use it for other payments).
   - Convert it to fiat immediately via their own off-ramp provider.
   - Use it to pay their own suppliers or invoices (if those parties accept USDC).

Key characteristic: The recipient must have a wallet and must be comfortable holding stablecoin, at least briefly.

VALUE PROPOSITION

- **Fast, final settlement:** Once the on-chain transaction confirms, payment is done. No waiting for the off-ramp provider to deposit fiat.
- **Recipient flexibility:** The payee decides what to do with the stablecoin. If they have use for it (paying their own suppliers, holding it as liquidity), they don't pay an off-ramp fee. Recipients may request payment in their preferred stablecoin (USD, EUR, or other fiat-pegged options).
- **Lower cost (if recipient doesn't convert immediately):** You avoid the off-ramp provider's fee. Total cost is just your on-ramp fee + gas fee.
- **Transparency and independent verification:** You see the on-chain transaction in real time and can verify settlement on a public blockchain explorer without waiting for recipient confirmation. This eliminates "did they receive it?" uncertainty - you have independent, tamper-proof proof of delivery the moment the transaction confirms.
- **Good for contractors/vendors who prefer crypto:** Some freelancers, agencies, or international contractors actively prefer to be paid in stablecoin (to avoid their local banking frictions, local inflation or FX volatility).



TRADE-OFFS

- **Recipient must be onboarded:** The recipient needs a wallet, needs to understand how to receive stablecoin (e.g., USDC, EURC), and must be comfortable with the (small) risk of holding stablecoin.
- **Recipient takes conversion risk:** If they want fiat, they pay the off-ramp fee and take FX/ timing risk. Some recipients may resist this (they want fiat, not stablecoin).
- **Wallet address errors are catastrophic:** If you send USDC to the wrong wallet address (a typo, a malicious fake address), the funds are gone. There's no "recall payment" button. Mitigations: use verified address registries, test with small amounts first, implement wallet address whitelists.

GOVERNANCE FOCUS (TOP 3)

1. **Wallet address verification and whitelisting:** Establish a process to verify recipient wallet addresses (test transactions, address registries, recipient-confirmed address change process). Do not allow ad hoc address entry.
2. **Recipient onboarding and education:** Create a simple guide: "How to receive USDC payments." Include how to set up a wallet, how to confirm receipt, and how to convert to fiat if desired. Provide support contact.
3. **Irreversibility controls:** Because you can't reverse a stablecoin transaction, your pre-payment controls (amount verification, address verification, dual approval for high-value payments) must be airtight.

PILOT FIT

Best for:

- Payments to contractors, freelancers, or agencies who are already comfortable with crypto/stablecoins.
- Cross-border payments to recipients in countries with difficult banking systems (where receiving USDC is easier than receiving a wire).
- Companies whose suppliers or service providers want to be paid in stablecoin (some software vendors, design agencies, remote workers actively request this).

Not ideal for:

- Large, traditional suppliers who have no interest in learning about stablecoins (they'll push back).
- High-volume AP flows with hundreds of suppliers (onboarding friction is too high).
- Payments to recipients in jurisdictions where crypto is legally ambiguous or restricted (compliance risk).

ILLUSTRATIVE PARTNERS (NOT ENDORSEMENTS)

- **Stablecoin payment platforms (managed service):** Fipto - handles fiat-to-stablecoin conversion and payment execution without requiring you to manage a wallet directly.
- **Circle (USDC issuer):** Offers Circle Account for businesses to hold and send USDC.
- **Coinbase Commerce, BitPay (merchant payment processors):** Allow businesses to accept and hold USDC.
- **Wallet providers for recipients:** MetaMask, Coinbase Wallet, Trust Wallet (consumer wallets); institutional wallets from Anchorage, BitGo (for larger recipients).

TIP (Model 2): Use this model when the recipient wants to be paid in stablecoin. Don't force it on suppliers who aren't ready. Pilot with 5-10 willing recipients, learn the onboarding and address-management process, then expand if it works.

Model 3: Direct Stablecoin (No Intermediaries)

HOW IT WORKS

Direct stablecoin is the purest form of stablecoin payment. Both you and the recipient hold stablecoin wallets. You send stablecoin directly from your wallet to theirs, on-chain, with no fiat conversion on either end.

Step-by-step:

1. Your treasury team maintains a USDC balance in a custody wallet (funded via on-ramp when needed,
   or held as liquidity).
2. You initiate a payment: "Pay Third party 25,000 USDC."
3. Payment is approved per your workflow (multi-sig wallet, custody provider approval policy, or TMS approval).
4. Your wallet custody provider signs and broadcasts the transaction on-chain.
5. Transaction confirms (seconds to minutes).
6. Third party's wallet receives 25,000 USDC.
7. Third party's Z holds the USDC (or converts to fiat on their own schedule).

Key characteristic: No on-ramp or off-ramp in the transaction flow. It's pure wallet-to-wallet stablecoin transfer. Fiat conversion, if any, happens separately before (when you fund your wallet) or after (when recipient cashes out).

VALUE PROPOSITION

- **Fastest settlement:** On-chain confirmation is final in minutes (or seconds on fast blockchains). No waiting for bank processing.
- **Lowest cost:** Gas fee is typically under $1 on Ethereum during normal network conditions (or fractions of a cent on Polygon, Solana). No on-ramp or off-ramp fees per transaction (you pay those separately, when you fund or withdraw from the wallet).
- **Maximum transparency:** Both parties see the on-chain transaction in real time. Blockchain explorer provides independent verification.
- **No intermediaries:** You control the wallet, the private keys (either in self-custody or via custody provider), and the timing. No dependency on a payment processor's liquidity or uptime (just the blockchain network itself).
- **Ideal for programmatic or recurring payments:** If you're making hundreds of payments per month to recipients who also operate in stablecoin, this is the most efficient model.

TRADE-OFFS

- **Both parties must hold stablecoin:** You need treasury-grade custody. The recipient needs a wallet. Not viable if either party isn't ready for that.
- **Liquidity management:** Your treasury team must decide how much stablecoin to hold, when to on-ramp (convert fiat → stablecoin), and when to off-ramp (convert stablecoin → fiat). You're taking instrument risk (brief exposure to the stablecoin issuer's creditworthiness).
- **Custody responsibility:** In self custody, your team is responsible for securing private keys. If keys are lost or stolen, funds are gone. Mitigation: use institutional custody providers with insurance and controls.
- **Reconciliation complexity:** Your ERP ledger is in USD; your wallet holds stablecoin. You need to reconcile on-chain stablecoin movements back to fiat-equivalent GL entries. If the stablecoin briefly de-pegs, you have a variance to explain.

GOVERNANCE FOCUS (TOP 3)

1. **Custody and key management:** Choose an institutional custody provider with multi-sig, role-based access, insurance, and SOC 2 or equivalent certification. Document key recovery procedures.
2. **Liquidity and exposure limits:** Define how much stablecoin your treasury will hold at any time (e.g., "maximum 7 days of expected outflows"). Document when and how you'll on-ramp or off-ramp to stay within limits.
3. **On-chain transaction approval and SoD:** Ensure wallet transactions require dual approval (multi-sig or custody provider policy engine). Log every transaction with approver identity, timestamp, and transaction hash.

PILOT FIT

Best for:

- Treasury teams ready to hold stablecoin balances and manage custody directly.
- Payments to counterparties who are already operating in stablecoin (crypto-native vendors, other treasury teams piloting stablecoins, liquidity pools, DeFi protocols if applicable).
- High-frequency, low-friction payment flows (intercompany settlements, daily vendor payments, platform payouts).

Not ideal for:

- First pilots (too much operational change at once - start with Model 1 or 2).
- Treasury teams without strong operational controls and tech integration (custody management is non-trivial).
- Payments to traditional suppliers who don't want to touch crypto.

ILLUSTRATIVE PARTNERS (NOT ENDORSEMENTS)

- **Custody providers:** BitGo, Coinbase Custody, Anchorage Digital, Copper (institutional-grade custody with policy engines and insurance).
- **Treasury-integrated platforms:** Kyriba (treasury management system with stablecoin payment integration), other TMS vendors exploring stablecoin modules.
- **Circle Business Account:** Direct USDC issuance and wallet for corporate treasury use.

TIP (Model 3): Don't pilot this model unless you've already built the governance foundation (Chapters 1-3). Managing liquidity in stablecoins is not fully equivalent to fiat. But for treasury teams ready to operate in stablecoin, this is the most efficient, transparent, and scalable model long-term.

Model 4: Intercompany through Stablecoin

HOW IT WORKS

This model applies stablecoins to one of treasury's most persistent pain points: **intercompany settlements** across subsidiaries in different countries. Instead of moving fiat through multiple banking relationships and dealing with correspondent bank delays and fees, the parent company and subsidiaries settle balances in stablecoin.

Step-by-step:

1. Your ERP calculates intercompany balances
   (e.g., US Parent owes $500,000 to Singapore Subsidiary for shared service charges).
2. Settlement is approved per intercompany policy.
3. US Parent converts $500,000 USD → 500,000 USD stablecoin
   (via on-ramp or from existing USDC liquidity).
4. US Parent sends 500,000 stablecoin to Singapore Subsidiary's wallet
   (on-chain, confirmed in minutes).
5. Singapore Subsidiary receives 500,000 stablecoin. Options:
   - Hold it (if Singapore Sub has other stablecoin payment needs).
   - Convert it to SGD via local off-ramp (same day or next day).
   - Transfer it onward to another subsidiary if needed.
6. Intercompany ledger is updated: US Parent debits, Singapore Sub credits, using USDC transaction hash as proof of settlement.

Note: This example uses a USD-pegged stablecoin (e.g., USDC, USDT) for a US-to-Singapore settlement. The model works for any currency pairing: EUR parent → EURC subsidiary, GBP → GBPS, or mixed-currency netting using multiple stablecoins. The key is that both entities agree on which stablecoin(s) to use for settlement.

Key characteristic: Both entities are under the same corporate umbrella, so you can standardize custody, governance, and liquidity management across the group.
No third-party recipient friction.

VALUE PROPOSITION

- **Eliminates correspondent banking delays:** Traditional intercompany settlements through banks can take 2-5 days (or longer in difficult corridors). Stablecoin settles in minutes.
- **Significantly lower fees:** No correspondent bank fees, no SWIFT charges. Just on-ramp/off-ramp fees (if converting to/from fiat) and gas fees.
- **Netting becomes real-time:** If your subsidiaries owe each other, you can net in stablecoin and settle instantly, rather than waiting for monthly/quarterly settlement cycles.
- **FX flexibility:** Subsidiaries can hold stablecoin as a common liquidity buffer, converting to local currency only when needed. Reduces FX conversion frequency.
- **Proof of settlement is instant and independent:** Blockchain transaction hash is irrefutable proof that the intercompany settlement occurred. No disputes about "did the wire go through?"

TRADE-OFFS

- **Requires corporate-wide custody infrastructure:** Each subsidiary (or at least the key ones) needs a wallet and custody arrangement. That means governance, key management, and training across multiple entities.
- **Tax and accounting implications:** Holding stablecoin on the balance sheet may require new accounting treatment. Settlements in stablecoin (rather than fiat) may have tax reporting implications depending on jurisdiction. Consult tax and accounting advisors.
- **Regulatory complexity across jurisdictions:** Some countries have restrictions or ambiguous rules about corporate entities holding or transacting in stablecoins. Legal and compliance review required for each subsidiary jurisdiction.
- **Liquidity management:** Each entity needs to decide how much liquidity to keep in stablecoins. Adds a layer of treasury planning.

GOVERNANCE FOCUS (TOP 3)

1. **Standardized custody and key management across entities:** Use the same custody provider and governance model (multi-sig, approval policies, audit logging) for all subsidiaries. Don't create 10 different custody arrangements.
2. **Intercompany policy and documentation:** Update your intercompany agreement to include stablecoin settlements. Define how FX is calculated (at time of settlement, using what rate), how transaction hashes are logged, and how disputes (if any) are resolved.
3. **Tax and regulatory compliance per jurisdiction:** Validate that stablecoin holdings and settlements are legally and tax-compliant in each subsidiary's country. Document the treatment for audit and tax filings.

PILOT FIT

Best for:

- Multinational companies with frequent intercompany settlements (monthly or more often).
- Corridors where traditional banking is slow or expensive (e.g., US ↔ Latin America, Europe ↔ Southeast Asia, intra-Asia).
- Treasury teams that already have strong intercompany governance and are looking to optimize settlement speed and cost.

Not ideal for:

- Companies with only 1-2 subsidiaries and infrequent settlements (complexity outweighs benefit).
- Jurisdictions where holding stablecoins is legally unclear or restricted (e.g., China, some Middle Eastern countries - check local rules).
- Treasury teams not ready to manage custody and liquidity in multiple entities.

ILLUSTRATIVE PARTNERS (NOT ENDORSEMENTS)

- **Custody providers with multi-entity support:** Fireblocks (custody infrastructure with self-custody model; supports organizational hierarchy and entity-level wallets), Anchorage Digital, Copper (qualified custodians holding keys on your behalf).
- **Treasury management systems:** Kyriba (intercompany module + stablecoin integration), SAP Treasury (if integrating stablecoin custody).
- **Accounting/ERP integration platforms:** Platforms that bridge on-chain settlement data to ERP ledgers (still emerging; may require custom integration).

TIP (Model 4): This is a high-value use case if you have the operational maturity to execute it. Pilot with 2 entities in a single corridor first (e.g., US ↔ Singapore or US ↔ Mexico). Prove you can settle, reconcile, and maintain governance. Then expand entity by entity.

Model 5: Stablecoin → Fiat (Conversion/Redemption)

HOW IT WORKS

This model is the reverse of Model 2. You receive stablecoin (as payment for goods/services, or as settlement from a counterparty) and convert it to fiat in your bank account.

Step-by-step:

1. A customer, partner, or counterparty sends you stablecoin (to your wallet address).
2. Your wallet receives the stablecoin; on-chain transaction confirms.
3. Your custody provider or off-ramp partner converts the stablecoin → USD and deposits the fiat into your bank account (same day or next day, depending on provider).
4. Your AR or treasury ops team reconciles: on-chain stablecoin receipt → fiat deposit → customer invoice.

Key characteristic: You're receiving stablecoin, not sending it. Common for companies who sell into crypto-native markets, who serve as payment service providers or who accept stablecoin as payment from customers.

VALUE PROPOSITION

- **Access to crypto-native customers:** If you sell software, services, or digital goods to customers who prefer to pay in stablecoin (common in Web3, crypto, international tech communities), you can accept payment without forcing them to use traditional banking.
- **Faster settlement from international customers:** Customers in countries with difficult banking systems can pay you in stablecoin, and you convert to USD - faster and cheaper than international wires for them.
- **Lower payment processing fees (vs. credit cards):** Stablecoin payments have no chargeback risk and lower fees than credit card processing (typically 2-3%). Gas fee + off-ramp fee is often under 1%.
- **Transparent, instant confirmation:** You see the on-chain payment in real time. No waiting days to confirm an international wire arrived.

TRADE-OFFS

- **You hold stablecoin (briefly):** Even if you convert to fiat immediately, there's a window where you hold stablecoin. You're exposed to issuer risk and peg risk during that window (usually hours, not days).
- **Custody required:** You need a wallet and custody arrangement to receive stablecoin. Smaller companies may use a payment processor (like BitPay or Coinbase Commerce) to handle this; larger companies may custody directly and payment specialist may be comfortable with self custody (e.g. Fireblocks technology).
- **Customer must be willing/able to pay in stablecoin:** Not all customers will. This works for certain markets (crypto-native, tech, international freelancers) but not mainstream B2B or consumer.
- **Tax/accounting implications:** Receiving stablecoin and converting to fiat may have tax reporting nuances (consult your accountant). In the US, it's generally treated as fiat-equivalent income, but rules vary by jurisdiction.

GOVERNANCE FOCUS (TOP 3)

1. **Custody and security for incoming payments:** Ensure your wallet is secure (institutional custody if volumes are significant). Monitor wallet balance regularly; don't let large stablecoin balances accumulate unconverted (increases exposure).
2. **Conversion policy and FX risk:** Define how quickly you convert stablecoin → fiat (e.g., "within 24 hours of receipt"). Document the FX rate used (spot rate at time of conversion). Track conversion fees for accounting.
3. **AR reconciliation:** Ensure your AR team can match on-chain stablecoin receipts → fiat deposits → customer invoices. Log transaction hashes and customer wallet addresses for audit trail.

PILOT FIT

Best for:

- Companies selling digital goods, SaaS, or services to international or crypto-native customers.
- Businesses who want to expand payment options to attract customers in underserved banking markets.
- Marketplaces, platforms, service providers or gig economy companies that already handle multi-currency payments.

Not ideal for:

- Traditional B2B companies with established customer payment terms (customers won't adopt).
- Low-margin businesses where even small conversion fees matter (may not save enough vs. traditional payments).
- Companies not ready to manage inbound stablecoin custody.

ILLUSTRATIVE PARTNERS (NOT ENDORSEMENTS)

- **Payment processors:** Coinbase Commerce, BitPay, Circle Payments API (handle inbound stablecoin, convert to fiat, deposit to your bank).
- **Custody + off-ramp:** Fireblocks (self custody), Alchemy Pay, MoonPay (off-ramp services).
- **Accounting integrations:** QuickBooks, Xero (some payment processors offer direct integration to push fiat-equivalent transactions to your accounting system).

TIP (Model 5): If you're piloting this model, start by offering stablecoin as a payment option to a small segment of customers (international, crypto-native, or customers who've requested it). Track adoption rate and customer feedback. If it's low, don't force it - save your governance effort for higher-value models.

Model 6: Multi-Token Acceptance via Clearer

HOW IT WORKS

This model is appropriate for companies operating globally who want to accept payments in any stablecoin or tokenized deposit without managing multi-currency stablecoin liquidity or custody complexity themselves. A specialized clearer acts as an intermediary, ensuring instant conversion to fiat and managing relationships with multiple issuers and banks.

Step-by-step:

1. Your company publishes payment instructions allowing customers, partners, or subsidiaries to pay you in any supported stablecoin or tokenized deposit (USDC, EURC, tokenized bank deposits, or others).
2. A customer or counterparty sends payment in their stablecoin of choice (e.g., 50,000 USDC or 45,000 EURC) to an address provided by your clearer partner.
3. The clearer receives the stablecoin on-chain and immediately converts it to your target fiat currency (e.g., USD) using pre-established margining agreements with issuers that guarantee 1:1 peg and off-ramp SLAs.
4. The clearer coordinates the off-ramp process with the stablecoin issuer and deposits fiat into your designated bank account (same day or next day, depending on SLA).
5. Your AR or treasury team reconciles: customer payment (on-chain stablecoin receipt captured by clearer) → fiat deposit → invoice

Key characteristic: You never hold stablecoin or manage multi-token custody. The clearer provides "singleness of money" across different stablecoins and tokenized deposits; your treasury team sees only fiat inflows. The clearer absorbs conversion risk, timing risk, and issuer coordination complexity via margining agreements.

VALUE PROPOSITION

- **Universal payment acceptance without custody complexity:** Accept payments in any major stablecoin or tokenized deposit (USDC, EURC, JPM Coin, tokenized bank deposits) without managing wallets, keys, or multi-chain custody. The clearer handles allstablecoin infrastructure.
- **Guaranteed conversion and settlement:** Margining agreements between the clearer and issuers ensure 1:1 peg and predictable off-ramp SLAs. You receive fiat in your bank account with known timing and cost - no exposure to stablecoin peg risk or liquidity spreads.
- **Faster, cheaper cross-border inflows:** Customers in any jurisdiction can pay you in their preferred stablecoin or tokenized deposit, often faster and cheaper than international wires. You receive fiat without dealing with correspondent banking delays.
- **Treasury sees only fiat:** From treasury's perspective, this operates like a multi-currency payment processor. Customers pay in stablecoin; you receive fiat. Reconciliation and accounting remain fiat-denominated.

TRADE-OFFS

- **Dependency on clearer as single point of failure:** You're reliant on the clearer's operational resilience, margining arrangements with issuers, and relationship with your bank. If the clearer has an outage or solvency issue, inbound payments may be delayed or interrupted.
- **Less control over conversion timing and pricing:** The clearer manages conversion. You don't control the exact timing or see the on-chain transaction directly (unless the clearer provides visibility). Conversion rates and fees are set by the clearer's agreements with issuers.
- **Regulatory and contractual complexity:** The clearer sits between you, your customers, and multiple issuers/banks. Legal review is required to understand liability, data access for audit, and compliance obligations (AML/KYC, jurisdictional rules).

GOVERNANCE FOCUS (TOP 3)

1. **Clearer due diligence and contract terms:** Vet the clearer's financial stability, operational resilience, insurance coverage, and margining arrangements with issuers. Ensure contract specifies SLAs, liability for failed conversions, fee structure, and data access for audit and reconciliation.
2. **Customer payment instruction governance:** Define how payment instructions (wallet addresses, identifiers) are issued and updated. Ensure customers can't send payments to incorrect addresses or without proper invoice/reference identifiers. Build error-handling procedures for mispayments.
3. **Reconciliation and AR integration:** Ensure the clearer provides transaction-level data (customer identifier, stablecoin received, conversion rate, fiat deposited, fees) that can be matched to invoices and integrated into your AR system. Automate this if volumes are significant; manual reconciliation won't scale.

PILOT FIT

Best for:

- Companies with significant cross-border receivables from customers in multiple countries who want to offer universal stablecoin payment acceptance without building custody infrastructure.
- Businesses selling digital goods, SaaS, services, or operating marketplaces where customers prefer stablecoin payments.
- Treasury teams that want the benefits of stablecoin acceptance (speed, reach, lower cost) but prefer a managed service model with fiat settlement (no stablecoin balance sheet exposure).

Not ideal for:

- Companies whose customers have no interest in paying via stablecoin (traditional B2B with established wire/ACH payment terms).
- Treasury teams that want direct control over stablecoin custody, conversion timing, and on-chain transactions (Model 3 or 5 is better).
- First pilots (start with simpler models like 1 or 2; Model 6 requires mature vendor management

ILLUSTRATIVE PARTNERS (NOT ENDORSEMENTS)

- **Stablecoin clearers:** Ubyx (provides universal stablecoin acceptance with clearer-based conversion and fiat settlement via margining agreements with issuers).
- **Payment processors with multi-token support:** Coinbase Commerce, BitPay (may offer similar conversion-to-fiat models for inbound stablecoin payments; capabilities vary).
- **Treasury and AR integration platforms:** Platforms that connect clearer data feeds to ERP or AR systems for automated reconciliation (emerging category; may require custom integration).

TIP (Model 6): This model is ideal for receivables-heavy businesses that want to accept stablecoin payments globally without operational complexity. The clearer acts like a payment processor: customers pay in stablecoin, you receive fiat, and the clearer manages the middle. Pilot with a small segment of international customers (5-10 willing participants) to test customer experience, reconciliation workflows, and clearer SLA performance before scaling. Ensure your AR team can handle multi-token inbound payments and clearer data feeds before go-live.

Decision Tree: Your Priority → Recommended Model(s) → Next Step

Use this table to identify which model(s) align with your priorities.

YOUR PRIORITY	RECOMMENDED MODEL(S)	WHY	NEXT STEP
Pilot stablecoins with minimal internal change	Model 1 (Fiat → Stablecoin → Fiat)	No wallet management; recipients get fiat; you get faster/cheaper cross-border payments without operational overhaul.	Identify 2-3 cross-border corridors; vet on-ramp/off-ramp providers; pilot with 10-20 payments.
Pay contractors or suppliers who want stablecoin	Model 2 (Fiat → Stablecoin, payee holds)	Recipients get stablecoin (which they want); you avoid off-ramp fees; settlement is fast and transparent.	Survey 5-10 suppliers/ contractors; onboard willing participants; pilot with small payments first.
Optimize high-frequency or intercompany payments	Model 3 (Direct Stablecoin) or Model 4 (Intercompany)	Lowest cost, fastest settlement, maximum control; ideal for frequent, recurring, or internal payments.	Build custody and governance foundation (Chapters 1-3); select custody provider; pilot with one corridor or entity pair.
Accept payments from crypto-native customers	Model 5 (Stablecoin → Fiat)	Expands payment options; lower fees than credit cards; fast settlement from international customers.	Evaluate payment processors (Coinbase Commerce, BitPay); integrate to AR; pilot with opt-in customers.
Accept payments in any stablecoin without custody complexity	Model 6 (Multi-token via clearer)	Universal stablecoin acceptance; clearer converts to fiat instantly via margining agreements; no custody or multi-chain complexity; guaranteed settlement SLAs.	Vet clearer providers (e.g., Ubyx); pilot with 5-10 international customers; integrate clearer data feed to AR system; test reconciliation workflows.
Prove concept with low risk before committing	Model 1 or Model 2	Both allow small-scale testing without building internal custody; you can evaluate speed, cost, and counterparty acceptance before scaling.	Choose one corridor, one supplier, or one use case; document learnings after 10-20 transactions; decide whether to scale or stop.

REMEMBER: You don't have to choose one model forever. Many companies pilot Model 1, learn the ecosystem, then graduate to Model 3 or 4 as their governance and custody capabilities mature. Start with the model that matches your current readiness, not your aspirational state.

Governance Mapping: Which Models Emphasize Which Controls

Not all models stress the same governance areas. Use this mapping to understand where to focus your control design for each model.

GOVERNANCE FOCUS AREA	HIGH PRIORITY FOR THESE MODELS	WHY
Vendor due diligence (on-ramp/off-ramp)	Models 1, 2, 5	You depend on third parties to convert fiat ↔ stablecoin reliably. Their failure is your operational failure.
Custody and key management	Models 3, 4, 6	You hold stablecoin balances directly. Losing keys = losing funds. Institutional custody with insurance and controls is essential.
Wallet address verification	Models 2, 3, 4	Sending to the wrong address is irreversible. Whitelisting, test transactions, and verification processes are critical.
Liquidity and exposure limits	Models 3, 4, 6	You hold stablecoin balances over time. Define max exposure, rebalancing rules, and triggers for converting back to fiat.
Evidence chain (fiat → stablecoin → fiat)	Models 1, 2, 5	Multiple conversion steps; reconciliation must tie payer bank → on-ramp → on-chain → off-ramp → payee bank.
Intercompany policy and tax treatment	Model 4	Settlements between entities in different jurisdictions; tax and legal review required for each entity
Multi-chain custody and security	Model 6	Managing wallets and keys across multiple blockchains adds complexity and risk. Standardization and monitoring are critical.
Exception handling and fallbacks	All models	Every model can experience stuck payments, provider outages, or recipient issues. All require pre-defined exception ownership and fallback plans.

Pratical use: When designing governance for your pilot (Chapter 5), use this table to prioritize where you invest control effort. If you're piloting Model 1, spend more time on vendor due diligence and less on custody key management. If you're piloting Model 3, flip that priority.

What You've Learned (And How to Choose)

You now understand the six distinct stablecoin payment models, each with different conversion points, custody arrangements, dependencies, and governance requirements.

TO CHOOSE YOUR STARTING MODEL, ASK:

1. **What problem are we solving?**
   (Speed? Cost? Counterparty preference? Liquidity optimization?)
2. **What's our operational readiness?** (Can we manage custody?
   Do we have strong vendor management? Can we train recipients?)
3. **What's our risk tolerance?**
   (Comfortable holding stablecoin balances, or prefer managed service?)
4. **What does success look like in 90 days?**
   (Proof of concept? Cost savings? Faster settlement? Supplier satisfaction?)

Most treasury teams start with Model 1 or Model 2 because they're lower-complexity, lower-risk ways to learn the ecosystem. Once you've proven governance and operational capability, you can move to Model 3 or 4 for higher efficiency and control.

Model 6 is the endgame for sophisticated, global treasury operations - not the starting point.

And remember: **no model is "best."** The right model is the one that matches your priorities, your readiness, and your counterparties' capabilities. Now that you know which model to pilot, the next chapter will show you how to pilot it - with a 90-day playbook that builds governance in from day one.

Part 2

Action Toolkit

You've got the context - now it's time to act. This section is a toolkit you can copy, adapt, and run: templates, checklists, and a 90-day action plan to take stablecoins from concept to a controlled pilot and a decision.

Chapter 5 90-Day Pilot Playbook

WHY 90 DAYS (AND WHY GOVERNANCE FROM DAY ONE)

Most pilots fail in one of two ways:

Way 1: They run forever, never reaching a clear "go/no-go" decision. After six months, the pilot is still "in progress," stakeholders lose interest, and it quietly dies.

Way 2: They move fast, prove the technology works, then discover they can't scale because they skipped governance. Now they're retrofitting approvals, evidence chains, and exception handling - creating "governance debt" that's expensive and risky to pay down.

This playbook prevents both failures.

Ninety days is long enough to:

- Design the governance model (not bolt it on later).
- Execute real transactions in a controlled environment.
- Hit operational exceptions and test your response.
- Evaluate results against success criteria.
- Make a confident decision: scale, pivot, or stop.

Ninety days is short enough to:

- Maintain executive attention and stakeholder momentum.
- Limit risk exposure (time-bounded, volume-limited).
- Prevent scope creep ("let's just try one more thing…").

The playbook is structured in five phases:

- **Phase 0 (Weeks 1-2):** Strategy & model selection
- **Phase 1 (Weeks 3-4):** Design & dependency mapping
- **Phase 2 (Weeks 5-8):** Build & integrate
- **Phase 3 (Weeks 9-11):** Controlled pilot execution
- **Phase 4 (Week 12):** Evaluate & decide

Each phase has clear inputs, outputs, decisions, and governance checkpoints. By the end of Phase 4, you'll have evidence-based answers to:
Should we scale? What would it take?
Or should we stop?

One rule governs this entire playbook: No governance shortcuts. If you can't prove a payment was approved, executed correctly, and reconciled within policy, don't call it a success - even if it settled in 30 seconds.

Phase 0: Strategy & Model Selection (Weeks 1-2)

OBJECTIVE

Define *why* you're piloting stablecoins, which payment model fits your priorities, and what success looks like. Exit this phase with executive alignment, a scoped pilot, and a clear decision framework.

KEY DECISIONS

1. What problem are we solving?

Don't start with "let's try stablecoins."
Start with a business problem:

- Cross-border payments to Country X are slow and expensive (3-5 days, 3-5% in fees).
- Intercompany settlements between Entity A and Entity B create working capital drag.
- Contractors in Region Y prefer stablecoin payments; we're losing talent to competitors who offer it.
- We want to accept payments from crypto-native customers but don't have infrastructure.

Document the problem in treasury terms: cost, speed, counterparty friction, working capital impact.

2. Which payment model (from Chapter 4) fits this problem?

Use the decision tree from Chapter 4. Map your problem to a model:

- Cross-border supplier payments, minimal internal change? → Model 1 (Fiat → Stablecoin → Fiat).
- Intercompany settlements? → Model 4.
- Contractors who want stablecoin? → Model 2.
- High-frequency, direct control? → Model 3.

Choose one model for the pilot. Don't try to test multiple models at once (scope creep kills pilots).

3. What does success look like?

Define measurable criteria:

- **Speed:** Average settlement time < X hours (vs. current Y days).
- **Cost:** Total cost per payment < $Z (vs. current $W).
- **Governance:** 100% of payments have documented approvals, transaction hashes logged, and reconciled within 48 hours.
- **Operational:** Zero loss events; all exceptions resolved within documented procedures; team confidence rating ≥ 4/5.

Also define stop conditions (see end of chapter): What would cause you to kill the pilot early?

4. What corridor and counterparties will we pilot with?

Start narrow:

- **One corridor** (e.g., US → Singapore, US → Mexico, US entity → UK entity).
- **5-10 counterparties** (suppliers, contractors, or subsidiaries who are willing participants and understand this is a pilot).
- **Low to moderate value** ($1,000-$50,000 per payment; not your largest or most critical payments yet).
- **Frequency:** Aim for 20-50 transactions over the pilot period (enough to test operational rhythm, not so many that exceptions overwhelm you).

Selection criteria for pilot counterparties:

- Willing to participate (don't force it on resistant suppliers).
- Low criticality (if something goes wrong, there's time to fix it without business impact).
- Clear communication (they'll give you feedback on their experience).
- Existing relationship (you trust them; they trust you).

REQUIRED CONTROLS (PHASE 0)

Even in the strategy phase, governance starts:

- **Executive sponsor identified** (VP Treasury, CFO, or Treasurer who will make the scale/stop decision).
- **Pilot scope documented** (model, corridor, counterparties, transaction limits, duration).
- **Success criteria and stop conditions documented** (measurable outcomes; triggers for early termination).
- **Legal and compliance review initiated** (are stablecoins permissible in the jurisdictions involved? Any regulatory approvals needed?).
- **Initial risk assessment** (what could go wrong? What's the max loss exposure?).

OUTPUTS

- **One-page pilot charter:** Problem statement, model choice, success criteria, scope, timeline, decision-makers.
- **Risk assessment summary:** Key risks identified; initial mitigations planned.
- **Stakeholder alignment:** Finance, treasury, AP/AR (depending on model), legal, compliance, IT/security all informed and aligned.

COMMON FAILURE MODES (PHASE 0)

- **Scope creep:** "Let's also test Model 3 while we're at it." → No. One model, one corridor.
- **Vague success criteria:** "We'll know it when we see it." → No. Define metrics now.
- **Skipping legal/compliance review:** "We'll deal with that later." → No. If it's not legal, the pilot stops before it starts.

3 A.M. TEST CHECKPOINT

Ask: "If this pilot encounters a critical exception at 3 a.m., who decides what to do - and is that person ready to make the call?" If the answer is unclear, your governance isn't ready.

TIP (Phase 0): Get your CFO or Treasurer to sign off on the pilot charter before you move to Phase 1. This ensures you have air cover and decision authority when trade-offs arise later. Without executive sponsorship, pilots get stuck in "wait and see" limbo.

Phase 1: Design & Dependency Mapping (Weeks 3-4)

OBJECTIVE

Design the end-to-end payment flow, map all dependencies, define approval workflows, establish custody/provider arrangements, and document exception-handling procedures. Exit this phase with a fully documented governance model - before you move any money.

KEY DECISIONS

1. **Who are our dependencies, and what are the SLAs?**

Using the dependency mapping method from Chapter 3:

- List every party: ERP, TMS, on-ramp provider, custody provider, off-ramp provider (if applicable), blockchain network, counterparties, your bank.
- For each dependency, document: Who operates it? What's their SLA? What's the escalation contact?
- Identify single points of failure (e.g., "If custody provider goes offline, we have no fallback").

2. **What's our approval workflow and segregation of duties?**

- Who initiates payments? (AP clerk, treasury ops)
- Who approves? (Treasury manager, CFO for amounts >$X)
- Who executes? (Treasury ops, custody provider on your instruction)
- How is SoD enforced? (Multi-sig wallet, TMS approval policy, custody provider policy engine)

Document this in your TMS or ERP configuration, or in the custody provider's policy settings. Don't rely on "we'll just be careful."

3. **What's our custody model?**

If your model requires holding stablecoin (Models 3, 4, 6):

- **Who is our custody provider?** (Fireblocks, BitGo, Coinbase Custody, etc.)
- **What insurance and security do they offer?** (self custody with MPC storage of keys, Cold storage, multi-sig, SOC 2, insurance coverage up to $X)
- **Who on our team has access?** (Treasury manager, treasury ops; document roles and permissions)
- **How do we recover if keys are lost?** (Key recovery process documented and tested)

If your model uses a managed service (Models 1, 2, 5), custody is handled by the provider - but you still need to vet their custody arrangements.

4. **What are our controlled fallbacks?**

For each potential failure mode identified in dependency mapping:

- **Trigger condition:** Payment pending >2 hours.
- **Owner:** Treasury ops.
- **Action:** Escalate to custody provider; if no resolution in 1 hour, switch to SWIFT fallback.
- **Evidence:** Log ticket #, document fallback execution in TMS, note original approval reference.

Document 3-5 common failure scenarios and their fallback procedures before you encounter them live.

5. **How will we reconcile?**

- **What data flows from ERP → custody provider (or on-ramp) → on-chain → off-ramp (if applicable) → bank?**
- **How do we tie it all together?** Build a "transaction reference table" (Chapter 3): ERP payment ID | TMS approval ID | Transaction hash | Settlement date | Recipient confirmation.
- **Who owns reconciliation?** (Treasury ops, AP team, joint process?)
- **What's the timeline?** (Daily? Weekly? Real-time?)

REQUIRED CONTROLS (PHASE 1)

- **Dependency map completed** (all parties, all handoffs, all failure modes documented).
- **Approval workflow designed and configured** (in TMS, ERP, or custody provider; SoD enforced).
- **Custody arrangement finalized** (provider selected, accounts created, access roles assigned, insurance verified).
- **Exception ownership matrix documented** (who decides what for each failure type).
- **Fallback procedures documented** (SWIFT as backup; trigger conditions and decision rules defined).
- **Reconciliation process designed** (data flows mapped; reference table structure defined).
- **Vendor contracts signed** (custody provider, on-ramp/off-ramp if applicable; SLAs and liability terms reviewed by legal).

OUTPUTS

- **Governance playbook document** (15-25 pages): Dependency map, approval workflow, custody model, exception matrix, fallback procedures, reconciliation process.
- **Custody configuration** (multi-sig settings, policy engine rules, user access logs).
- **Vendor onboarding complete** (accounts created, test credentials validated, support contacts confirmed).

COMMON FAILURE MODES

- **"We'll figure out exceptions when they happen."** → No. Document procedures now.
- **"The vendor says their platform is secure; that's good enough."** → No. Verify insurance, certifications, and contract liability terms
- **"Reconciliation can be manual for the pilot."** → Maybe, but only if you document the manual process and assign clear ownership. Don't let it be ad hoc.

3 A.M. TEST CHECKPOINT

Simulate a stuck payment on paper: "It's Saturday, 3 a.m. Payment #1 has been pending for 4 hours. Walk me through what happens next." Can your team answer this with reference to documented procedures? If not, refine Phase 1 outputs.

TIP (Phase 1): Spend 80% of your time in Phases 0-1 on governance design, not on tech integration. The technology is the easy part. Getting approvals, exception handling, and reconciliation right is what separates pilots that scale from pilots that create governance debt.

Phase 2: Build & Integrate (Weeks 5-8)

OBJECTIVE

Build the technical integrations, configure systems, train the team, and execute test transactions in a sandbox or low-risk environment. Exit this phase with a working end-to-end flow that's been tested under controlled conditions.

KEY DECISIONS

1. **What's our integration approach?**

- **API integration** (TMS or ERP sends payment instructions to custody provider via API)? → Best for automation and scale, but requires dev resources.
- **File upload** (generate payment file, upload to custody provider portal)? → Lower tech lift, but more manual and error-prone.
- **Manual portal entry** (treasury ops logs into custody provider dashboard and initiates each payment)? → Easiest to pilot, doesn't scale.

Choose the approach that matches your pilot volume and your IT capacity. For 20-50 transactions over 12 weeks, manual or file-based may be sufficient. For 100+, invest in API integration.

2. **How will we train the team?**

Identify who needs training:

- **Treasury ops:** How to initiate payments, monitor status, escalate exceptions.
- **AP team (if applicable)**: How to enter payment details, what wallet address means, how to verify recipient.
- **Approvers (treasury manager, CFO)**: How to review and approve (may be unchanged if approval happens in TMS).
- **Reconciliation team:** How to match transaction hashes to payments, how to pull data from custody dashboard or blockchain explorer.

Create short training guides (1-2 pages each) with screenshots. Hold a 30-minute training session. Test comprehension with a simulated payment.

3. **What test transactions will we run?**

Before executing real payments, run 3-5 test transactions in a sandbox or with test counterparties (internal wallets, friendly vendors):

- **Test 1:** Small payment ($10-$100), happy path, no issues. Validates end-to-end flow.
- **Test 2:** Payment with incorrect wallet address (caught by validation step). Validates error handling.
- **Test 3:** Payment that requires fallback (simulate on-ramp outage by pausing process). Validates exception procedures.
- **Test 4:** Payment that needs escalation (simulate delay). Validates escalation path.
- **Test 5:** Reconciliation test (match test payment to ERP, log transaction hash). Validates reconciliation process.

All tests must be documented: what happened, what worked, what didn't, what we changed.

REQUIRED CONTROLS (PHASE 2)

- **System integration complete and tested** (API, file upload, or portal access functional; connectivity validated).
- **Approval workflow tested** (test payment goes through approval process; multi-sig or policy engine enforces SoD).
- **Custody access tested** (authorized users can access dashboards; private keys secured; test transaction executed successfully).
- **Reconciliation tested** (test payment matched to ERP; transaction hash logged; no data gaps).
- **Team trained** (all participants completed training; demonstrated understanding via test transaction).
- **Exception procedures tested** (simulated stuck payment; team followed escalation path; fallback executed correctly).
- **Audit trail validated** (test payment has complete evidence chain: approval → instruction → transaction hash → settlement → reconciliation).

OUTPUTS

- **Integration documentation** (how to initiate payments, API endpoints or file formats, error codes and meanings).
- **Training materials** (guides for treasury ops, AP, approvers, reconciliation).
- **Test transaction log** (5 tests executed; results documented; issues resolved).
- **Go-live readiness checklist** (all systems functional, all team members trained, all test scenarios passed).

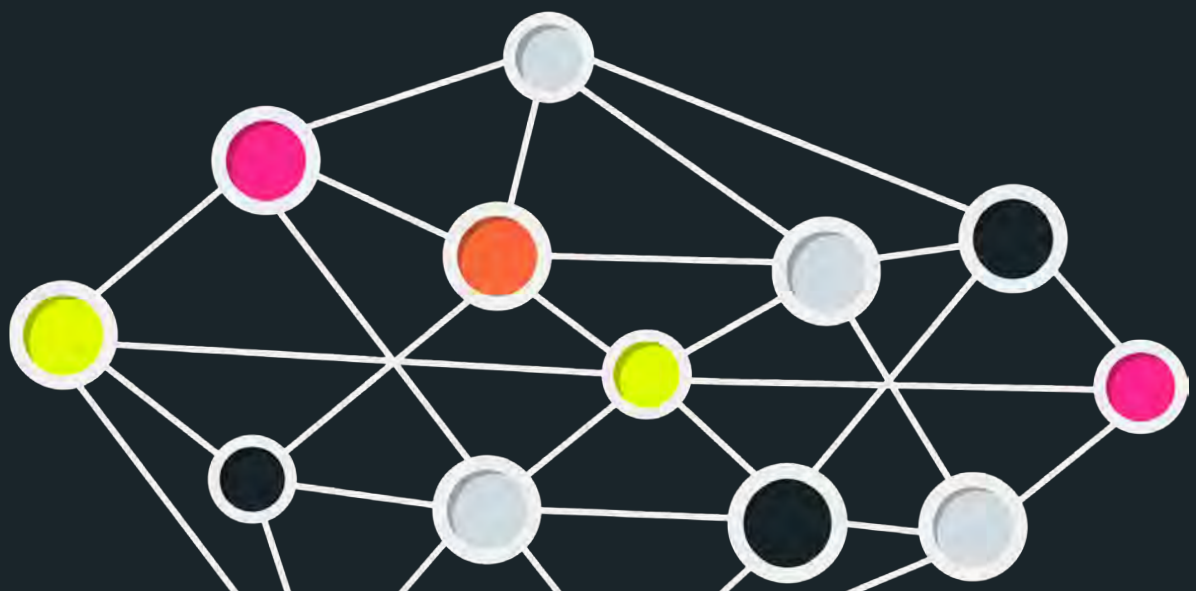
COMMON FAILURE MODES

- **Skipping test transactions:** "We'll just start with real payments and see what happens." → No. Test first.
- **Training only one person:** "Sarah knows how to do it; she'll handle everything." → No. Cross-train. Sarah might be on vacation when an exception happens.
- **Ignoring test failures:** "The test payment didn't reconcile correctly, but we'll fix it later." → No. Fix it now, before real payments start.

3 A.M. TEST CHECKPOINT

Run a simulated 3 a.m. exception during the test phase: pull one team member into a (planned) "urgent" scenario on a weekend evening. Can they follow the escalation procedures without guidance? If not, improve training and documentation.

TIP (Phase 2): Don't rush through testing to "go live faster." The time you invest in Phase 2 prevents chaos in Phase 3. A well-tested pilot runs smoothly; a poorly tested pilot creates emergencies that erode stakeholder confidence.



Phase 3: Controlled Pilot Execution (Weeks 9-11)

OBJECTIVE

Execute real payments in the scoped corridor with pilot counterparties. Monitor closely, capture data, handle exceptions using documented procedures, and gather qualitative feedback. Exit this phase with evidence of operational performance and lessons learned.

KEY DECISIONS

1. How do we ramp volume?

Don't start with 50 payments in Week 9. Ramp gradually:

- **Week 9:** 5-10 payments (low-value, non-critical suppliers).
- **Week 10:** 10-20 payments (slightly higher value, broader counterparty mix).
- **Week 11:** 20-30 payments (full pilot volume, including a few time-sensitive payments to test operational rhythm).

Monitor closely each week. If exceptions pile up or governance breaks down, pause and fix before ramping further.

2. How do we capture data?

For every payment, log:

- **ERP payment ID**
- **Approval date/time and approver**
- **Instruction sent date/time**
- **Transaction hash (on-chain**
- **Settlement confirmation date/time**
- **Total cost** (on-ramp fee, gas fee, off-ramp fee if applicable, FX spread if applicable)
- **Settlement time** (hours from approval to recipient confirmation)
- **Exceptions** (if any: what happened, how resolved, time to resolution)

Build this into a tracking spreadsheet or dashboard. At the end of Phase 3, you'll use this data to evaluate success criteria.

3. How do we handle exceptions?

When (not if) exceptions occur:

- **Follow the exception ownership matrix from Phase 1.**
- **Document every exception:** Ticket #, timeline, decision made, resolution, outcome.
- **Debrief after each exception:** What went wrong? Did our procedures work? What would we do differently?

Treat exceptions as learning opportunities, not failures. A pilot that encounters zero exceptions didn't test operational readiness - it got lucky.

4. How do we gather feedback?

At the end of Phase 3 (or mid-way through if volume is high):

- **Survey pilot counterparties:** Did they receive payments on time? Was the process clear? Any issues?
- **Debrief with treasury ops, AP, approvers:** What worked well? What was painful? What would they change?
- **Review with executive sponsor:** Are we on track to meet success criteria? Any concerns?

Capture qualitative feedback, not just quantitative data. "Payments were fast, but our team is stressed because reconciliation is taking 2 hours per payment" is important information.

REQUIRED CONTROLS (PHASE 3)

- **All payments follow a documented approval workflow** (no shortcuts, no exceptions to SoD).
- **Transaction hashes logged for every payment** (no "we'll capture that later").
- **Exceptions documented and resolved per procedures** (escalation paths followed, decisions logged).
- **Weekly check-ins with executive sponsor** (status update, exception summary, any concerns flagged).
- **Stop condition monitoring** (if any stop condition is triggered, pilot pauses immediately for review).
- **Reconciliation completed within 48 hours of settlement** (no backlog).

OUTPUTS

- **Pilot transaction log** (20-50 payments executed; data captured for all).
- **Exception log** (all exceptions documented: type, resolution, time to resolution).
- **Performance data** (average settlement time, average cost, governance compliance rate).
- **Qualitative feedback** (survey results, team debrief notes, counterparty feedback).

COMMON FAILURE MODES

- **"It's just a pilot, so we'll skip the approval this once."** → No. Every shortcut creates governance debt.
- **"We're too busy executing payments to document exceptions."** → No. Documentation is part of execution. If you can't document and execute, slow down.
- **"This one payment is urgent; we'll use the fallback even though it's not a true exception."** → No. Fallbacks are for failures, not convenience. Stick to the process.

3 A.M. TEST CHECKPOINT

If you encounter a real exception during Phase 3 (and you will), ask afterward: "Did we handle it the way we would at 3 a.m.?" If the answer is "we improvised," update your procedures based on what you learned.

TIP (Phase 3): Celebrate the exceptions, not just the successes. A pilot that encounters a stuck payment, follows the escalation procedure, executes the fallback cleanly, and documents the outcome has proven governance works. That's more valuable than 50 payments with zero exceptions (which might just mean you didn't stress-test the system).

Phase 4: Evaluate & Decide (Week 12)

OBJECTIVE

Analyze pilot results against success criteria, assess operational readiness for scale, identify gaps, and make a clear decision: scale, pivot (test a different model or corridor), or stop.

KEY DECISIONS

1. Did we meet success criteria?

Compare actual results to the criteria you set in Phase 0:

CRITERION	TARGET	ACTUAL RESULT	MET?
Average settlement time	< 4 hours	2.3 hours	Yes
Average cost per payment	< $15	$12 (incl. all fees)	Yes
Governance compliance	100% documented approvals	98% (1 payment missing approval signature due to system glitch)	Mostly (fixable)
Operational confidence	Team rating ≥ 4/5	3.8/5 (reconciliation pain point)	No (needs improvement)

If you met most criteria but have 1-2 gaps, those are your scale blockers. Don't ignore them. Fix them or accept the residual risk.

2. What were the top pain points?

Review exception logs, team feedback, and counterparty feedback. Common pain points:

- Reconciliation is manual and time-consuming.
- Wallet address verification is nerve-wracking (fear of typos).
- Custody provider's dashboard is clunky; hard to pull data.
- One counterparty had trouble receiving off-ramp deposits (off-ramp provider KYC issue).

For each pain point, ask: Is this fixable? What would it take? If we scale without fixing it, what's the risk?

3. What's the business case for scaling?

Quantify the benefits observed in the pilot:

- **Cost savings:** $X per payment × Y payments per month = $Z annual savings.
- **Speed improvement:** T hours faster settlement → working capital benefit or supplier satisfaction.
- **Operational efficiency:** Reduced manual work in Z process (quantify hours saved).

Compare to the cost of scaling:

- Custody provider monthly fees or per-transaction costs at scale.
- IT resources to automate reconciliation or build API integration (if not done yet).
- Training and change management for broader team.

Does the ROI justify scale? If not, is there strategic value (access to new markets, supplier preference) that outweighs the cost?

4. Scale, pivot, or stop?

Three possible outcomes:

SCALE: Pilot met success criteria (or gaps are minor and fixable). Business case is strong. Team is confident. Executive sponsor approves moving to production with defined scope (e.g., expand to 100 suppliers in the same corridor, or add a second corridor).

PIVOT: Pilot revealed that this model/corridor isn't optimal, but a different approach might work. Example: "Model 1 worked, but reconciliation is too manual. Let's test Model 3 (direct stablecoin) with our tech team's help, which would simplify reconciliation."

STOP: Pilot didn't meet success criteria, pain points are too costly to fix, business case is weak, or team/counterparties aren't ready. **Stopping is a legitimate, valuable outcome.** You learned what doesn't work, and you didn't scale a broken process.

REQUIRED CONTROLS (PHASE 4)

- **Results report completed** (data analysis, success criteria comparison, pain points identified).
- **Business case model completed** (cost/benefit at scale; ROI calculation; assumptions documented).
- **Scale plan drafted (if scaling):** Scope, timeline, resources required, governance updates needed.
- **Executive decision documented:** Scale / Pivot / Stop, with rationale and next steps.
- **Lessons learned documented:** What worked, what didn't, what we'd do differently next time.

OUTPUTS

- **Pilot results report** (10-15 pages): Executive summary, data analysis, success criteria assessment, pain points, business case, recommendation.
- **Decision memo** (1-2 pages): Executive sponsor's decision and rationale.
- **Lessons learned document** (5 pages): Governance insights, operational insights, vendor performance, counterparty feedback, recommendations for future pilots.

COMMON FAILURE MODES

- **"Let's just keep the pilot running while we decide."** → No. Make a decision in Week 12. Perpetual pilots don't add value.
- **"We mostly met criteria; let's scale and fix the gaps later."** → Depends. Small, fixable gaps? Okay. Governance or operational gaps? Fix first.
- **"We can't stop now; we've invested too much."** → Sunk cost fallacy. If the pilot showed it doesn't work, stopping is the right decision.

3 A.M. TEST CHECKPOINT

Final question: "If we scale this, and 6 months from now we have a critical exception at 3 a.m., are we confident our team can handle it?" If yes, scale. If no, identify what's missing and address it before scaling.

TIP (Phase 4): A "stop" decision is not a failure. You proved governance, learned the ecosystem, validated (or invalidated) assumptions, and avoided scaling a process that wasn't ready. That's a win. Treasure the data and lessons learned - they'll inform your next move (maybe a different model, a different corridor, or a decision to wait until the market matures).

How to Choose Your First Corridor and Counterparties

Corridor selection criteria:

1. **Pain is real:** Current rails are slow, expensive, or unreliable (not just "let's try something new").
2. **Volume is moderate:** Enough transactions to test operational rhythm (20-50 in 90 days), but not so many that you're overwhelmed.
3. **Regulatory clarity:** Both jurisdictions allow corporate use of stablecoins (or at least don't explicitly prohibit it).
4. **Partner ecosystem exists:** On-ramp and off-ramp providers operate in both countries (if using Model 1 or 2).
5. **Counterparties are willing:** Suppliers or subsidiaries in that corridor are open to participating.

Good starting corridors (as of 2026):

- US → Mexico, US → Philippines, US → Singapore, US → UK, US ↔ EU (for intercompany).

Avoid (for first pilots):

- Countries with unclear crypto regulations (China, India in some contexts).
- Corridors with low payment volume (hard to get meaningful data).
- Corridors where current rails already work well (pilot won't demonstrate value).

Counterparty selection criteria:

1. **Willing participants:** They understand it's a pilot and are patient with occasional hiccups.
2. **Low criticality:** Not your biggest supplier or most time-sensitive payments.
3. **Good communication:** They'll give you feedback and work with you if issues arise.
4. **Existing relationship:** You trust them; they trust you.
5. **Tech-savvy (if using Model 2 or 3):** They can set up a wallet and confirm receipt of stablecoin.

Recruit 5-10 counterparties. Start conversations early: "We're exploring faster, cheaper cross-border payments. Would you be interested in testing this with us for a few months?"

Stop Conditions: When to Pause or Kill the Pilot (And Why That's a Win)

Define these in Phase 0; monitor throughout Phases 2-4.

Stop conditions (pause pilot immediately for review):

- **Loss event**: Funds are lost, stolen, or sent to wrong address irreversibly (investigate, assess root cause, fix controls before resuming).
- **Governance failure:** A payment was executed without documented approval, or SoD was bypassed (even once).
- **Vendor failure:** Custody provider, on-ramp, or off-ramp experiences extended outage (>24 hours) or solvency concerns.
- **Regulatory issue:** New regulation or guidance prohibits or restricts the pilot activity in one of the jurisdictions.
- **Team overwhelm:** Exception volume is too high, team is burned out, operational quality is degrading.
- **Business case collapse:** Cost or speed targets missed by >30%; counterparties refuse to participate; reconciliation overhead makes scaling impossible.

If a stop condition is triggered, pause new payments, conduct a root-cause review, and decide with the executive sponsor whether to fix and resume, or terminate the pilot.

Don't continue a pilot that's broken just because "we started it."

Stopping with Dignity: Why Termination Is a Success

A pilot that stops based on evidence is not a failure - it's exactly what pilots are for.

If you reach Week 12 (or Week 6) and the data says "don't scale," stopping cleanly is a treasury win. You:

- Tested assumptions with real transactions.
- Protected governance and avoided scaling a broken process.
- Learned what doesn't work in your environment (use case, corridor, vendor, or operational readiness).

How to execute a dignified stop:

1. **Write a 2-page Pilot Closure Report for your executive sponsor:**
   - Why we stopped (link to stop condition).
   - What we learned (3-5 key findings).
   - Recommendation: Stop now; revisit in [12-18 months] if [conditions change].
2. **Communicate internally and to counterparties:**
   - Stakeholders: "We completed the pilot. Based on [data/governance/cost], we're not scaling. Here's what we learned."
   - Pilot suppliers: "Thank you for participating. We're reverting to [standard rail]. No action needed."
3. **Archive everything:** Transaction logs, governance playbook, lessons learned. If you revisit stablecoins later, this is your head start

☆ **REMEMBER:** Stopping is leadership. You made an evidence-based decision and protected the organization from governance debt or operational risk. Document why you stopped - it's one of the most valuable outputs of the pilot.

Scale Criteria: What "Success" Looks Like

Define these in Phase 0; assess in Phase 4.

Scale criteria (must meet most/all to approve scaling):

- **Met success criteria:** Speed, cost, governance compliance targets achieved (or gaps are minor and fixable).
- **Positive ROI:** Business case shows cost savings or strategic value that justifies investment to scale.
- **Operational confidence:** Team rates their confidence at ≥ 4/5; willing to scale with support.
- **Exception handling proven:** Pilot encountered at least 2-3 exceptions, and all were resolved using documented procedures.
- **Counterparty satisfaction:** Pilot participants (suppliers, subsidiaries) report positive or neutral experience (no major complaints).
- **No unresolved governance gaps:** All payments have complete evidence chains; no audit concerns.
- **Executive sponsor approval:** CFO/Treasurer reviews pilot results and approves scale plan.

If you meet these criteria, you're ready to scale - gradually. Don't go from 50 pilot payments to 500 production payments overnight. Scale in phases (expand corridor, add counterparties, increase limits) with continued monitoring.

What You've Learned (And What Happens Next)

You now have a structured 90-day playbook that builds governance from day one, tests operational readiness under controlled conditions, and produces a clear decision at the end.

Key takeaways:

- **Governance first, always.** No shortcuts, even in a pilot. If you can't prove approvals, execution, and reconciliation, the pilot didn't prove readiness.
- **Test exceptions, not just happy paths.** A pilot that never breaks isn't a stress test.
- **Decide in 90 days.** Scale, pivot, or stop - but decide. Perpetual pilots waste resources and erode confidence.
- **Stopping is okay.** If the pilot shows it doesn't work, you learned something valuable. Don't scale broken processes.

What happens after Phase 4?

If you **scale:**

- Execute the scale plan (expand scope, automate integration, train broader team).
- Continue monitoring (monthly reviews, quarterly governance audits).
- Iterate and improve (refine reconciliation, optimize costs, add corridors).

If you **pivot:**

- Document lessons learned.
- Select a different model or corridor based on what you learned.
- Run a new 90-day pilot with adjusted scope.

If you **stop**:

- Document why (what didn't work, what assumptions were wrong).
- Share findings with stakeholders (so the organization learns).
- Revisit in 12-18 months if market conditions or your operational readiness changes.

Either way, you've run a disciplined, governance-first pilot. That alone is a success - and it positions you to make the right strategic decisions for treasury.

Now, let's move to the final chapter:
Five Things to Do Next.

Chapter 6 Five Things to Do Next

Two Paths Forward

You've read the guide. You understand stablecoins in treasury terms. You know the governance requirements. You've seen the payment models and the 90-day pilot playbook.

Now what?

There are two paths forward, and they're not mutually exclusive:

Path A: Build Your Knowledge and Network

Deepen your understanding of stablecoins and digital assets in treasury through structured learning (AFP's course and certification), industry conferences, and peer networks. This path is about becoming fluent in the language, the regulatory landscape, and the strategic implications - so when your CFO asks, "Should we explore this?" you can answer confidently.

Path B: Start a Strategy and Pilot

If you've identified a real business problem that stablecoins could solve (slow cross-border payments, expensive intercompany settlements, supplier friction), start designing a governance-first pilot.

This path is about moving from concept to controlled execution - proving (or disproving) that stablecoins deliver value in your specific context.

Most treasury practitioners should do both. Path A gives you the strategic context and confidence. Path B gives you operational evidence and experience.

The five actions below support both paths. Some are foundational (do them regardless). Some branch depending on whether you're focusing on learning first or piloting first.

By the end of these five actions, you'll either:

- Be enrolled in (or have completed) the AFP course and connected to a peer network, with a clear view of where stablecoins fit in your treasury strategy, OR
- Have a scoped, governance-ready pilot plan with executive sponsorship-ready to execute a 90-day controlled pilot, OR
- Both.

Let's get started.

Thing 1: Assess Where You Are (The Treasury Readiness Self-Assessment)

WHY IT MATTERS

Before you commit time and resources to learning or piloting, you need to understand your starting point. Are you exploring from curiosity, or do you have a pressing business problem? Is your treasury operation mature enough to pilot new rails, or do you need to build foundational governance first?

A 15-minute self-assessment prevents you from pursuing the wrong path or moving too fast.

WHAT TO DO THIS WEEK

Answer these six questions honestly:

1. Do we have a real business problem that stablecoins might solve?

- Cross-border payments are slow or expensive (>3 days, >2% in fees)?
- Intercompany settlements create working capital drag?
- Suppliers or contractors are asking for crypto payment options?
- We want to accept payments from crypto-native customers?

If yes to any: Path B (pilot) becomes relevant.

If no: Path A (learning) is your priority - build knowledge before chasing solutions.

2. Is our current treasury governance strong?

- We have documented approval workflows and segregation of duties?
- We can produce a complete audit trail for any payment (approval → execution → settlement → reconciliation)?
- We have vendor due diligence processes and exception-handling procedures?

If yes: You're ready to pilot (after more scoping).

If no: Fix foundational governance first. Don't add stablecoin complexity to weak controls.

3. Do we have executive sponsorship or interest?

- Our CFO, Treasurer, or VP of Treasury has asked about stablecoins or digital assets?
- We have budget/time to explore new payment rails?

If yes: Path B is viable (you have air cover).

If no: Path A is safer (build your case and knowledge base before asking for resources).

4. Do we have technical capability?

- We have IT/dev resources who could integrate APIs or build data flows (if needed)?
- We use a modern TMS or ERP with integration capabilities?

If yes: Direct custody models (Models 3, 4) are feasible

If no: Start with managed-service models (Model 1) that require less technical lift.

5. Are we in jurisdictions where stablecoins are legally clear (or at least not prohibited)?

- Our HQ and key subsidiaries are in the US, EU, UK, Singapore, or other crypto-friendly jurisdictions?

If yes: Regulatory risk is manageable.

If no: Proceed cautiously; legal review is mandatory before piloting.

6. What's our risk tolerance?

- We're comfortable piloting new rails with limited transaction volumes and tight controls?
- We can absorb a small loss ($10K-$50K) if something goes wrong (though controls should prevent this)?

If yes: Pilot is appropriate.

If no: Stick with Path A (learning) until risk tolerance or controls improve.

WHAT "DONE" LOOKS LIKE

You've answered all six questions. You've identified whether you're currently a fit for Path A (learning), Path B (piloting), or both. You have a written summary (1 page) of your assessment to share with your manager or executive sponsor.

TIP: Don't skip this step. Treasury teams that jump straight to "let's pilot stablecoins" without assessing readiness often waste months on initiatives that were never viable. Start with self-awareness.

Thing 2: Build Your Strategic Knowledge

WHY IT MATTERS

While this guide provides a treasury-focused foundation, continuing education from credible, non-vendor sources will sharpen your evaluation and help you engage more effectively with partners and internal stakeholders.

WHAT TO DO THIS WEEK

Step 1: Start with AFP's existing resources

- Download the AFP Payments Guide: Cryptocurrency and Non-fungible Tokens from the AFP website
- Read the AFP Payments Guide: Seismic Shifts Are Coming for Cross-Border Payments
- Review AFP's stablecoin glossary entry and recent articles on digital assets

Step 2: Watch for AFP's upcoming certificate course

- AFP is developing "Stablecoin in Treasury by AFP in conjunction with Kyriba," expected in June 2026
- Consider it as a structured next step if you want deeper education on stablecoins, payments, governance, and treasury readiness

Step 3: Join relevant professional communities

- Join AFP's payments community (free for members) to access webinars and discussion forums
- Connect with treasury peers exploring digital assets on LinkedIn
- Consider joining other relevant treasury or digital asset communities

Step 4: Attend industry events

- Register for AFP's annual conference and attend digital assets / stablecoin sessions
- Look for webinars from AFP, Strategic Treasurer, PYMNTs, or industry working groups
- Attend at least one webinar or event in the next 30 days

Step 5: Monitor regulatory developments

- Subscribe to treasury-focused newsletters that cover digital asset regulation
- Bookmark relevant policy sources for updates on stablecoin and payments regulation
- Set a monthly reminder to check for major developments

WHAT "DONE" LOOKS LIKE

- You've read at least two AFP resources related to digital payments
- You're aware of the upcoming certificate course and can assess whether it is relevant once available
- You've joined at least one treasury community focused on digital assets
- You've attended one webinar or event on stablecoins/digital assets in treasury
- You have a system for staying current on regulatory developments

For Path A: This is your primary focus for the first 30 days, build foundational knowledge from credible sources.

For Path B: Do this in parallel with piloting, it will make you a better pilot designer and evaluator.

Timeframe: 30-60 days for initial knowledge building; ongoing for regulatory monitoring.

Thing 3: Define Your Business Problem (And Quantify It)

WHY IT MATTERS

Stablecoins are a solution. But what's the problem?

If you can't articulate a specific, measurable business problem - with cost, time, or friction you want to reduce - you don't have a pilot. You have an expensive science experiment.

Treasury doesn't pilot technology. Treasury solves problems.

WHAT TO DO THIS WEEK

Pick one (and only one) of these problems that's real for your treasury function:

Problem 1: Cross-border payments are slow and expensive.

- Current state: Payments to [Country X] take 3-5 days and cost $50-$150 per transaction in bank fees, FX spreads, and correspondent charges.
- Impact: We process ~100 such payments per month; annual cost ~$120K; supplier complaints about delays.

Problem 2: Intercompany settlements create working capital drag.

- Current state: Settling balances between [Entity A] and [Entity B] takes 2-3 days; ties up $500K in float; monthly cost ~$2K in bank fees.
- Impact: Slows month-end close; creates FX exposure during settlement window.

Problem 3: Suppliers or contractors prefer stablecoin payments.

- Current state: 5-10 contractors (freelancers, agencies in [regions]) have requested USDC payment; we currently decline and risk losing them to competitors.
- Impact: Talent/vendor access; losing ~$200K/year in services we'd like to procure.

Problem 4: We want to accept stablecoin payments from customers.

- Current state: We sell to crypto-native customers who prefer to pay in USDC; we currently don't accept it, so they pay via wire (slow, higher friction).
- Impact: Competitive disadvantage; some customers choose vendors who accept stablecoin.

WRITE A ONE-PAGE PROBLEM STATEMENT

Your problem statement should include:

- Problem: [Specific issue]
- Current state: [What happens today, with data]
- Desired future state: [What you want to achieve]
- Impact: [Cost, time, friction-quantified]
- How stablecoins might help: [Hypothesized benefit]

If you can't quantify the problem, you don't have a business case. Keep this document; you'll use it in Thing 4.

WHAT "DONE" LOOKS LIKE

You've identified one specific, quantifiable business problem. You've written a one-page problem statement with data. You've validated the problem with stakeholders (AP, intercompany, or sales ops - depending on the problem).

Thing 4: Build Your Pilot Plan and Secure Executive Sponsorship

WHY IT MATTERS

This is the “pilot gate.” Before you ask your CFO or Treasurer for permission to pilot, you need to show that you’ve thought through vendors, costs, benefits, risks, governance, and timing. A structured pilot plan - paired with executive approval - converts “interesting idea” into “resourced initiative.” This action consolidates vendor research, business case development, pilot scoping, and executive approval into one streamlined sequence.

WHAT TO DO (IN FOUR STEPS)

Step 1: Quick Vendor Research

Based on the payment model you’re leaning toward (from Chapter 4), identify the vendor categories you’ll need:

MODEL	VENDOR CATEGORIES NEEDED
Model 1 (Fiat → Stablecoin → Fiat)	On-ramp/off-ramp providers, payment API platforms
Model 2 (Fiat → Stablecoin, payee holds)	On-ramp providers, wallet providers for recipients
Model 3 (Direct Stablecoin)	Custody providers, TMS with stablecoin integration
Model 4 (Intercompany)	Custody providers with multi-entity support

Research 3-5 vendors per category. Start with providers mentioned in Chapter 4 and expand via AFP Digital Assets Community recommendations, peer networks, and your TMS vendor.

For each vendor, capture:

- Name and what they do (custody, on-ramp, integrated platform)
- Treasury-relevant features (multi-sig, approval workflows, API, reconciliation support
- Pricing model (per-transaction, monthly fee, AUM-based)
- Regulatory status (licensed, regulated in which jurisdictions)

Reach out to 2-3 vendors for exploratory calls (15-30 minutes each). Ask:

“We’re exploring [Model X] for [problem Y]. Can you walk me through how your platform would support that?”

- “What’s your typical treasury customer deployment look like?”
- “What governance controls and audit trails does your platform provide?”
- “What’s the pricing for a pilot (50 transactions over 90 days)?”

You’re not committing. You’re learning.

Step 2: Draft a One-Page Business Case

Using the problem statement from Thing 3 and the vendor research from Step 1, draft a one-page business case with these sections:

1. **Problem & Opportunity** (2-3 sentences)

"We spend $120K/year on cross-border payments to Southeast Asia, with 3-5 day settlement times. Stablecoins could reduce cost by 40-60% and settlement to <4 hours."

2. **Proposed Approach** (3-4 sentences)

"Pilot Model 1 (fiat → stablecoin → fiat) for 90 days with 5-10 suppliers in [corridor]. Use [Vendor X] for managed on/off-ramp. Process 20-50 payments, measure cost, speed, and governance compliance.

3. **Expected Benefits**
(bullets, quantified where possible)

- Cost savings: $50-$80 per payment (vs. current $100-$150) → ~$2K-$4K savings in pilot, ~$50K/year at scale
- Speed: <4 hours settlement (vs. 3-5 days) → improved supplier satisfaction
- Learning: Operational experience with stablecoin rails; data to inform scale decision

4. **Costs & Resources (bullets)**

- Vendor fees: ~$1K-$2K for pilot (based on vendor quotes)
- Internal time: Treasury ops (20 hours), Treasury manager (10 hours), IT support (~5 hours if API integration)
- Total pilot cost: ~$5K-$7K (time + fees)

5. **Risks & Mitigations (bullets)**

- Risk: Payment gets stuck or lost. Mitigation: Low-value pilot, controlled fallback to SWIFT, vendor insurance
- Risk: Governance gaps. Mitigation: Documented approval workflows, exception procedures, audit trail from day one
- Risk: Supplier friction. Mitigation: Pre-screened willing participants; they receive fiat (no crypto knowledge required)

6. **Decision Point (1 sentence)**

"After 90 days, we'll have data to decide: scale, pivot to a different model, or stop."

7. **Ask** (1 sentence)

"Requesting approval to proceed with pilot design and $5K-$7K budget for 90-day pilot execution."

Step 3: Scope the Pilot

Open Chapter 5 and use Phase 0 (Strategy & Model Selection) as your template. Answer every question for your specific pilot:

1. Problem statement: [Use Thing 3 output]
2. Payment model: [Choose one model from Chapter 4]
3. Success criteria: [Define 3-5 measurable outcomes]

Example: "Average settlement time <4 hours; cost <$15/payment; 100% governance compliance; team confidence ≥4/5"

4. Stop conditions: [Define 2-3 triggers that would pause or kill the pilot]

Example: "Loss event; governance failure; vendor insolvency"

5. Corridor and counterparties: [Choose one corridor; identify 5-10 willing participants]
6. Transaction limits: [Define per-transaction max, daily max, total pilot max]

Example: "$1K-$50K per payment; max $500K total over 90 days"

7. Timeline: [Map to weeks 1-12]
    a. Weeks 1-2: Final vendor selection and contracting
    b. Weeks 3-4: Design and dependency mapping (Phase 1)
    c. Weeks 5-8: Build and integrate (Phase 2)
    d. Weeks 9-11: Controlled execution (Phase 3)
    e. Week 12: Evaluate and decide (Phase 4)
8. Current-state governance map: Document your existing payment governance for this use case:
    a. Approval workflow (who initiates, who approves, where documented)
    b. Execution (who sends instruction, what controls prevent unauthorized execution)
    c. Settlement confirmation (how you know it settled, how long it takes)
    d. Reconciliation (who reconciles, how often, what's matched)
    e. Exception handling (what exceptions occur, who owns resolution)

Write this up as a "Pilot Scope Document" (3-5 pages). This becomes your execution plan if you get executive approval.

Step 4: Pitch Your Executive Sponsor

Schedule a 30-minute meeting with your CFO, Treasurer, or VP of Treasury (whoever owns payment strategy and innovation decisions).

Prepare a concise pitch (10-15 minutes; leave time for Q&A):

Slide 1: The Problem (from Thing 3)

"We spend $120K/year on cross-border payments with 3-5 day settlement. Suppliers complain. There's a better way."

Slide 2: The Proposed Solution (from Step 2)

"Pilot stablecoin payments (Model 1: managed service, fiat-in/fiat-out). 90 days, 5-10 suppliers, governance-first."

Slide 3: Expected Benefits (from Step 2)

Cost savings, speed improvement, strategic learning. Quantified.

Slide 4: Risks & Mitigations (from Step 2)

Loss risk (mitigated by limits, vendor insurance, fallback). Governance risk (mitigated by documented controls from day one).

Slide 5: What We're Asking For

"Approval to proceed with pilot. Budget: $5K-$7K. Resources: 20-30 hours treasury time over 90 days. Decision point: Week 12 (scale, pivot, or stop based on data)."

Bring the Pilot Scope Document (from Step 3) as a leave-behind.

Listen to their concerns. Common objections and how to respond:

- **"Is this legal/compliant?"** → "We've reviewed with legal [or will as part of Phase 1]. Both jurisdictions allow corporate use of stablecoins. We'll have full documentation."
- **"What if something goes wrong?"** → "Pilot is low-value ($50K max exposure), with SWIFT fallback ready. Vendor has insurance. Our stop conditions (documented) would pause the pilot immediately if a critical issue occurs."
- **"Do we have time for this?"** → "90 days, well-defined scope. If it doesn't work, we stop cleanly and document why. If it does, we have a scalable solution to a $120K/year problem."
- **"Why now?"** → "Stablecoin infrastructure has matured (regulatory clarity improving, institutional vendors available). AFP now offers a certificate program. If we wait another year, we're behind peers who are piloting now."

If they approve, get it in writing (email confirmation is fine). If they defer, ask what additional information they need - and schedule a follow-up in 2-4 weeks.

WHAT "DONE" LOOKS LIKE

- Vendor shortlist (3-5 vendors) with pricing and capability summaries
- One-page business case
- 3-5 page Pilot Scope Document (including current-state governance map)
- Executive approval to proceed (or clear path to approval with defined next steps)

If approved: Proceed to Thing 5 (execute the pilot).

If deferred: Focus on Path A (AFP course, peer learning) and revisit in 6 months with more market data or peer case studies.

Thing 5: Execute with Governance First or Keep Learning

WHY IT MATTERS

If you've secured approval (Thing 4), it's time to execute. But don't skip to "move money." The first four weeks are about governance design, vendor onboarding, and dependency mapping.

This is where you build the foundation that prevents governance debt.

If you're not ready to pilot yet, focus on building strategic knowledge and peer connections - and revisit Things 3-4 when you have a clearer problem or stronger governance foundation.

WHAT TO DO THIS WEEK

If You Have Approval (Path B: Piloting)

Follow Chapter 5, Phases 0-1 exactly:
Weeks 1-2 (Phase 0 recap):

- Finalize vendor selection (choose custody provider, on-ramp/off-ramp, or integrated platform)
- Execute vendor contracts (with legal review; confirm SLAs, insurance, liability terms)
- Notify pilot counterparties (confirm they're willing and ready)
- Brief internal stakeholders (treasury ops, AP, IT, compliance)

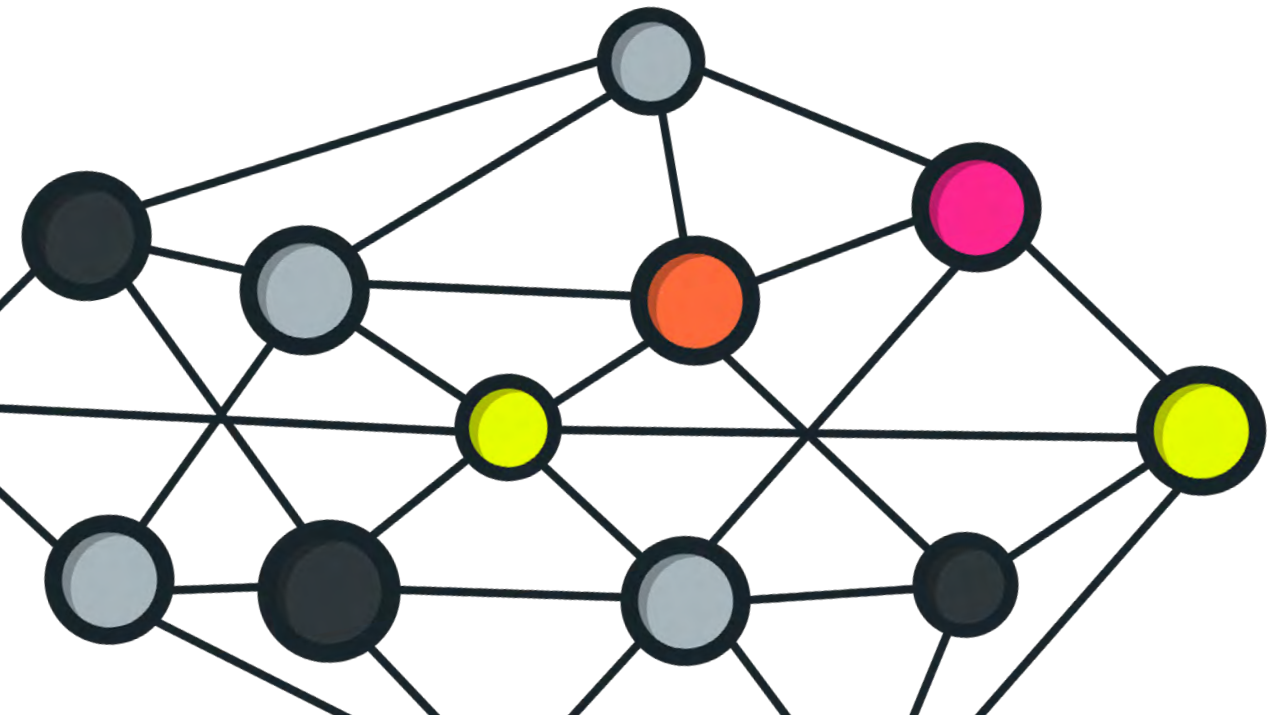
Weeks 3-4 (Phase 1):

- Map dependencies (every party, every handoff, every failure mode)
- Design approval workflow (adapt your current governance; configure multi-sig or TMS approval)
- Establish custody arrangements (if applicable; set up wallets, assign access roles, test)
- Document exception procedures (who owns what, escalation paths, fallback triggers)
- Design reconciliation process (build transaction reference table; define data flow from ERP → custody/on-ramp → on-chain → bank)
- Complete Governance Playbook Document (15-25 pages; see Chapter 5 Phase 1 outputs)

By the end of Week 4, you should have:

- Vendors onboarded and ready
- Governance fully documented
- Team trained on procedures
- No money moved yet (that's Phase 2-3)

Next steps: Proceed to Phase 2 (Build & Integrate, Weeks 5-8), then Phase 3 (Pilot Execution, Weeks 9-11), then Phase 4 (Evaluation & Decision, Week 12).



IF YOU'RE NOT READY TO PILOT YET (PATH A: LEARNING)

- Complete the AFP Digital Assets Certificate Program (4-6 weeks)
- Stay connected to treasury digital assets communities (AFP, LinkedIn, Slack)
- Attend webinars and events; take notes; ask questions
- Document what you're learning in a "Strategic Knowledge Log" (1-2 pages/month)
- Revisit Things 3-4 in 6-12 months when you have:
  - A clearer business problem worth piloting
  - Stronger foundational governance
  - Executive sponsorship or budget allocation

Either Way: Share What You Learn

Whether you're piloting or learning, share your insights with the treasury community:

- Write a short LinkedIn post about what you've learned
- Present at a treasury peer group or AFP chapter meeting
- Contribute to an AFP discussion thread or Slack community

Share:

- What worked and what didn't
- What you'd do differently
- What advice you'd give others

The treasury profession advances when practitioners share evidence, not when vendors make claims.

WHAT "DONE" LOOKS LIKE

Path B (Piloting):

- Phase 0 and Phase 1 deliverables (from Chapter 5) are complete
- Governance Playbook Document is written and reviewed
- Vendors are contracted and ready to support Phase 2
- Next step: Build & Integrate, then Controlled Execution

Path A (Learning):

- AFP course is in progress (or completed)
- You're actively engaged in treasury digital assets communities
- You've attended at least one webinar or event in the past 30 days
- You have a plan to revisit Things 3-4 when you're ready to pilot

Both: You've committed to sharing your learnings (set a calendar reminder for 90 days from now).

What Happens After These Five

If you've completed (or are progressing through) these five actions, you're no longer in the "I don't know where to start" phase. You're in one of three places:

1. **Enrolled in the AFP course and building your strategic knowledge (Path A).**

→ Next: Complete the course, earn the certificate, deepen your network, and revisit Things 3-5 in 6-12 months when you're ready to pilot.

2. **Executing a 90-day pilot with governance built in from day one (Path B).**

→ Next: Follow Chapter 5 through Phase 4. At Week 12, make the scale/pivot/stop decision based on data. Share your results with your executive sponsor and the treasury community.

3. **Decided stablecoins aren't right for you - yet.**

→ Next: Document why (not the right use case, jurisdictions are unclear, operational readiness gaps). Revisit in 12-18 months. This is a valid outcome. You made an informed decision, not a reactive one.

All three outcomes are wins. You approached stablecoins as a treasury professional: with governance first, evidence-based decision-making, and no hype.

That's the standard. That's how treasury leads.

A Final Word: Governance-First Wins

This guide opened with a principle: Outcomes first. Rails second. Governance can't be added later.

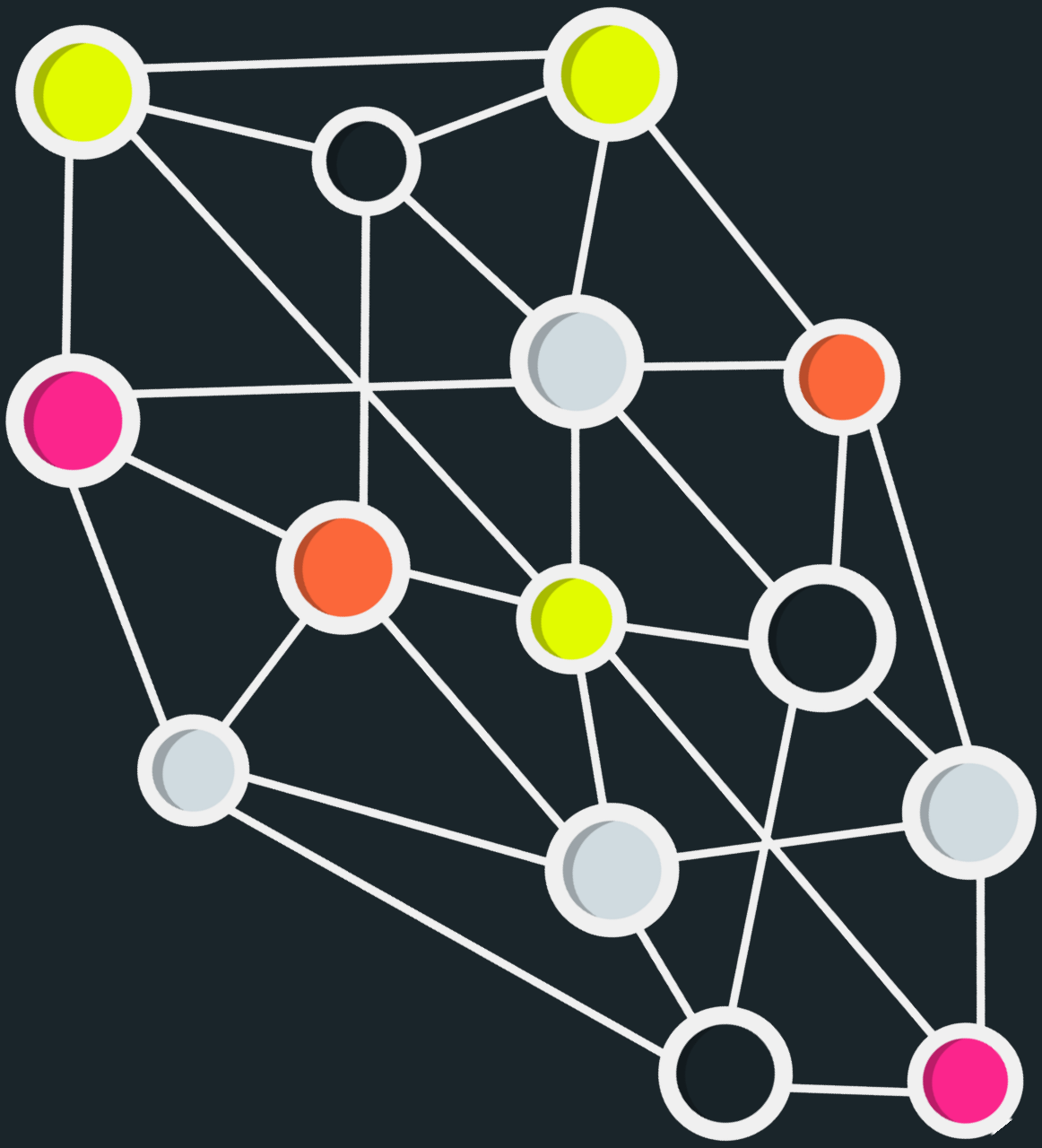
If you take one thing from these five actions, take this:

Speed, cost, and innovation are real benefits - but they're not the foundation. The foundation is governance: trusted connectivity, trusted data, and strong controls.

Build on that foundation, and stablecoins can be a powerful tool for treasury. Skip it, and you're building on sand.

Whether you're taking the AFP course, scoping a pilot, or simply staying informed, you're doing it the right way.

Welcome to the future of treasury payments. You're ready.



Contact Kyriba

1081 Camino del Rio South
San Diego, CA 92108

Phone: +1 858 210 3560

Email: treasury@kyriba.com

Website: www.kyriba.com