TECHNICAL RESEARCH NOTE · SR-TR-001 · v1.0

Payment Finality Is Not One Event

A Systems Taxonomy for Stablecoin Payment Infrastructure

Authors: StoneReason Research (StoneReason Research) **Published:** 2026-09-20 **Revised:** 2026-09-20 **Version:** 1.0 **Type:** Technical Research Note
Keywords: payment finality, settlement finality, stablecoin payments, blockchain confirmation, payment verification, reconciliation, payment evidence, asset identity, agent payments

Abstract

Payment systems routinely collapse a chain of distinct technical and operational events — authorization, signing, submission, provider acceptance, network observation, execution, confirmation, native finality, evidence issuance, and business resolution — into a single word: "confirmed" or "final." This report argues that the collapse is a category error with operational consequences, and develops an eleven-stage event model that keeps the stages separate. Drawing on protocol specifications and payment-system standards rather than on any single vendor's framing, it defends a set of non-identities — provider success is not native finality, a signature is not execution, execution is not finality, finality is not business resolution, webhook delivery is not payment occurrence, a reconciliation action is not blockchain truth, and signed evidence is not blockchain consensus — and examines how the meaning of "final" differs materially across Bitcoin, Ethereum, Solana and Stellar. The report is conceptual: it associates no original empirical dataset, and it states where empirical work would be required to quantify the timing gaps it describes. A closing section notes, neutrally, how the StoneReason payment stack implements parts of the model, kept distinct from the report's evidence.

1. Research question

Payment systems, and the software built on top of public blockchains in particular, routinely answer one question — *"did the payment go through?"* — with one word: *confirmed*, or *final*. This report asks a narrower and more precise question:

> What distinct technical and operational events are commonly collapsed into "payment confirmed" or "payment final," and why should payment infrastructure keep them separate?

The claim of the report is that the collapse is a category error: a single label is made to stand for a chain of events that occur at different times, are established by different parties, carry different failure modes, and reverse under different conditions. Where money and irreversibility meet, conflating them is not a simplification but a source of loss.

2. Scope and non-claims

This is a **conceptual and synthesis** technical note. Its contribution is a taxonomy and a set of defended distinctions, not a measurement.

- IN SCOPE a stage model of a payment's life on a public ledger; definitions; the argument for keeping the stages apart; network-specific meanings of “final” for four protocols.
- NOT CLAIMED no original empirical dataset; no global measurement of how long each stage takes; no ranking of networks as “better” or “faster”; no legal opinion on settlement finality in any jurisdiction. Timing figures quoted are attributed to protocol documentation, not measured here.

Throughout, individual claims carry a class label so the reader can tell what kind of statement each is: OBSERVATION (directly observed), DERIVATION (mechanically derived), INTERPRETATION (our analysis), HYPOTHESIS (needs testing), EXTERNAL (sourced to a cited work), and PRODUCT FACT (a fact about current StoneReason capability). These classes are not interchangeable and are not collapsed.

3. Why “finality” becomes ambiguous in payment systems

In a traditional account-based system, settlement finality has a precise, legally grounded meaning: the point after which a transfer is irrevocable and unconditional, defined by the rules of the system and, ultimately, by settlement in central-bank money (CPMI-IOSCO, 2012). The Principles for Financial Market Infrastructures make finality a *property conferred by a system's rulebook*, not a physical fact about a message. EXTERNAL

Public blockchains break the assumption that finality is a single, rulebook-defined instant in two ways. First, several widely used networks offer only *probabilistic* finality: a transaction becomes exponentially harder to reverse as blocks accumulate, but is never, at any finite depth, unconditionally irreversible (Nakamoto, 2008)(Garay et al., 2015). Second, even networks that provide *deterministic* finality do so through a distinct consensus event that occurs some time *after* a transaction first appears in a block (Buterin et al., 2020). The BIS has made the same point from the payments side: probabilistic settlement sits awkwardly with the finality that a payment system is expected to guarantee (BIS, 2025). EXTERNAL

The ambiguity is compounded by the software layer. Between a merchant's intent to be paid and the ledger's own notion of finality sit wallets, signers, submission endpoints, RPC providers, indexers and webhook senders. Each of these introduces its own success signal — “accepted,” “200 OK,” “delivered” — that is easy to mistake for the ledger's. INTERPRETATION

4. Definitions

We fix the following terms for the remainder of the report. Where a network's own protocol uses a materially different concept, that is noted in Section 13 rather than forced into this vocabulary.

Table 1. Terms fixed for use in this report.

Term	Definition used here

Term	Definition used here
Payment obligation	A stated intention that a specific value in a specific asset be delivered to a specific destination, before anything has happened on-chain.
Authorization	A decision that a payment *may* proceed, made by whoever holds that authority (a policy engine, a human, a rulebook). Distinct from the cryptographic act that later effects it.
Signature	A cryptographic operation binding a transaction to a key. Proves consent to a specific transaction; proves nothing about its effect.
Submission / provider acceptance	A transaction handed to an endpoint (node or RPC provider) and acknowledged by it. An acknowledgement of receipt, not of inclusion.
Network observation	An observer seeing the transaction in the mempool or in a block, via one or more sources. A claim about what an observer saw, conditioned on which sources it asked.
Execution / state effect	The transaction being executed by the network so that ledger state changes (a balance moves). Distinct from any judgement that the change is permanent.
Confirmation / commitment	An intermediate strengthening of a transaction's position — blocks on top, or a supermajority vote — short of the protocol's strongest guarantee.
Native (protocol) finality	The strongest irreversibility guarantee the protocol itself defines, in the protocol's own terms — and, for some protocols, only ever probabilistic.
Provider assurance	A statement by an intermediary (RPC, gateway, processor) that a payment succeeded. A claim about a provider's belief, not a protocol fact.
Evidence issuance	The production of a portable, verifiable record asserting facts about the payment.
Business resolution	A payee's operational decision that an obligation is satisfied (ship the goods, credit the account) — a policy choice layered on top of the technical facts.
Reconciliation	The workflow of resolving discrepancies between expected and observed payments (underpaid, overpaid, late, reorged), without altering the ledger's record.

5. A payment event model

We model a payment on a public ledger as an ordered sequence of eleven stages. The order is causal, not temporal-uniform: real gaps between stages range from milliseconds to tens of minutes, and some stages can fail or reverse after later stages appear to have succeeded. INTERPRETATION

Table 2. The eleven-stage payment event model, with what establishes each stage and how it characteristically fails or reverses.

#	Stage	Established by	Characteristic failure / reversal
1	**Obligation / intent**	Payee or payer	Wrong amount, asset, or destination stated.
2	**Authorization**	Policy / human / rulebook	Authorized beyond budget or outside policy.
3	**Signature**	Key holder	Valid signature over a transaction that never executes, or executes differently than intended.
4	**Submission / acceptance**	Node / RPC provider	Accepted then dropped; accepted by one provider, unseen by others.

#	Stage	Established by	Characteristic failure / reversal
5	**Network observation**	Observer + sources	Seen via one source that later disagrees; mempool sighting that never lands.
6	**Execution / state effect**	Network	Executed in a block that is later orphaned/re-orged.
7	**Confirmation / commitment**	Protocol (partial)	Confirmed-but-not-final; reversible under the protocol's own rules.
8	**Native finality**	Protocol (strongest)	For probabilistic chains, never absolute; for BFT chains, requires the finality event to occur.
9	**Provider assurance**	Intermediary	Provider says “paid” on the strength of stage 4, 5 or 6, not 8.
10	**Evidence issuance**	Evidence producer	Evidence that outruns the fact it attests; signed truth mistaken for consensus.
11	**Business resolution**	Payee policy	Goods shipped on a stage-6 signal that later reverses.

The remaining sections defend the specific non-identities between adjacent and near-adjacent stages that are most often collapsed in practice.

6. Observation is not execution

Seeing a transaction is not the same as its effect being real. A transaction observed in the mempool (stage 5) has been broadcast, not executed; a transaction observed in a block has been executed *on that fork*, which is not yet the same as executed on the canonical chain. An observation is always relative to the sources the observer consulted: a single RPC endpoint can report a state that a second, independently operated endpoint does not yet share. DERIVATION

This yields a rule that is easy to state and easy to violate: `OBSERVED_BY_ONE_SOURCE != ESTABLISHED`. Agreement between sources known to be operated independently is evidence; agreement between two calls to the same provider is availability, not corroboration. INTERPRETATION

7. Provider assurance is not native finality

`PROVIDER_SUCCESS != NATIVE_FINALITY`. An intermediary that returns “success” is reporting a belief formed at some stage — often submission (4), observation (5) or execution (6) — not the protocol's finality event (8). The two can differ by seconds on a fast BFT chain and by an hour on Bitcoin, and they can *diverge*: a provider can report success for a transaction that is subsequently re-orged out of the canonical chain. DERIVATION

The distinction is not pedantic. A payee that resolves business on provider assurance has outsourced its irreversibility guarantee to an intermediary's SLA, which is a credit relationship with that intermediary, not a property of the money. INTERPRETATION The payments-standards literature draws the same line between an operational acknowledgement and rulebook finality (CPMI-IOSCO, 2012).

8. Signature is not execution

`SIGNATURE != EXECUTION`. A signature (stage 3) proves that the holder of a key consented to a specific transaction. It does not prove that the transaction executed, that it executed as intended, or that it executed at all. A signed transaction can be dropped before submission, replaced by a higher-fee transaction, revert on execution while still consuming its nonce, or execute with an outcome the signer did not anticipate. DERIVATION

This separates two authorities that are frequently merged. *Authorization* (stage 2) decides whether a payment *may* happen; *signing* (stage 3) is the mechanical act that commits to a particular transaction. A system that lets the signer be the authorizer has no layer at which a payment can be refused before it is irreversibly effected. This is the crux of the agent-payments threat model, taken up as future work in Section 16: `AUTHORIZATION != SIGNING`, and an agent that can sign its own authority can widen it. HYPOTHESIS

9. Execution is not finality; finality is not business resolution

`EXECUTION != FINALITY`. Execution (stage 6) changes ledger state on some fork; finality (stage 8) is the guarantee that the change will not be undone. On probabilistic chains the gap is a matter of degree that never reaches certainty; on BFT-finality chains it is a discrete later event. DERIVATION

`FINALITY != BUSINESS_RESOLUTION`. Even perfect native finality does not tell a payee whether the *obligation* is satisfied. The amount may be short, the wrong asset, past a deadline, or to a stale address. Business resolution (stage 11) is a policy decision that *consumes* technical finality as one input among several. A merchant may rationally ship on a strong-but-not-final signal for a low-value order and wait for finality on a high-value one; that is a risk policy, and keeping it separate from the technical fact is what makes it a policy rather than a hidden assumption. INTERPRETATION

10. Webhooks are transport evidence, not payment occurrence

`WEBHOOK_DELIVERY != PAYMENT_OCCURRENCE`. A webhook is a message about a payment; its delivery establishes that a sender believed something and that a receiver received the message. It does not establish that the payment occurred, at what stage, or that it remains true now. Webhooks can be delivered late, out of order, duplicated, or for a state that a later reorg invalidates; and their absence is not evidence of non-payment. DERIVATION

The correct reading of a webhook is as a *transport signal that points at a fact to be verified against the ledger*, carrying at most the stage its sender had reached when it fired. Treating receipt of a webhook as the payment itself reintroduces exactly the provider-assurance error of Section 7 at the transport layer. INTERPRETATION

11. Signed evidence and its limits

`SIGNED_EVIDENCE != BLOCKCHAIN_CONSENSUS`. A cryptographically signed record asserting facts about a payment (stage 10) is valuable: a third party can verify, with a published key, that a specific issuer asserted specific facts, without trusting the issuer's database. But a signature authenticates *the issuer and the statement*, not the truth of the world the statement describes. Evidence can be issued about a

stage-6 execution that a reorg later undoes; the signature remains valid while the asserted fact becomes false. DERIVATION

Sound evidence therefore names the stage it attests and the basis of that attestation, and remains verifiable independently of whoever issued it — so that its scope cannot quietly widen from "the issuer says this executed" to "this is final." The distinction between authenticating a statement and establishing a fact is the same one the verifiable-credentials literature draws for identity claims, and it transfers directly to payment evidence. INTERPRETATION

12. Reconciliation without rewriting chain truth

`RECONCILIATION_ACTION != BLOCKCHAIN_TRUTH`. Reconciliation is the workflow for the cases the happy path omits: underpaid, overpaid, late, pending, or reorged. The essential discipline is that a reconciliation action — assigning a case, adding a note, marking it resolved — changes *workflow state*, never the ledger's record of what happened. INTERPRETATION

Collapsing the two is how ledgers silently drift from reality: if marking a case "resolved" also rewrote the recorded payment truth, the audit trail would record decisions as facts. Keeping `BILLING_STATUS != BLOCKCHAIN_TRUTH` means an operator can decide how to treat a short payment without anyone later being unable to tell what the chain actually said. DERIVATION

13. Cross-network terminology considerations

"Final" is not one guarantee across networks; it names materially different protocol events. The model above is deliberately network-neutral, but applying it requires reading each protocol in its own terms. The following characteristics are sourced to protocol documentation, not measured here. EXTERNAL

Table 3. What "final" denotes across four networks, in each protocol's own terms (sourced to protocol documentation, not measured here).

Network	Finality model	What "final" means in its own terms
Bitcoin	Probabilistic (Nakamoto)	No absolute finality at any depth; reversal probability falls exponentially with confirmations, the "six blocks" heuristic being convention, not a protocol threshold (Nakamoto, 2008)(Garay et al., 2015).
Ethereum	BFT finality gadget (Gasper: Casper-FFG + LMD-GHOST)	A checkpoint is *finalized* across epoch boundaries; documentation describes finalization on the order of two epochs (~12.8 minutes) absent faults, with accountable safety (Buterin & Griffith, 2017)(Buterin et al., 2020)(Ethereum.org).
Solana	Two-tier (Tower BFT)	An *optimistically confirmed* block (>2/3 stake vote) is distinct from a *rooted/finalized* block; the commitment levels `processed`, `confirmed` and `finalized` are not synonyms (Anza/Agave).
Stellar	Federated Byzantine Agreement (SCP)	A value is final when nodes *externalize* it on ledger close (~5s), given correct quorum-slice overlap; finality is a property of agreement, not of accumulated depth (Mazières, 2015).

Two consequences follow. First, a single cross-network "confirmed" flag cannot preserve these meanings; translating them into one shared word is where the guarantee is lost. INTERPRETATION Second,

this report makes no ranking claim: probabilistic and deterministic finality answer different questions, and “faster to a confirmation” is not “stronger guarantee.” NON-CLAIM

14. Implications for payment infrastructure

- Expose the stage, not a boolean. A payment API that answers “confirmed: true” has thrown away the information a payee needs to set its own risk policy. INTERPRETATION
- Report finality in the network's own vocabulary and keep it apart from provider assurance and from business resolution. INTERPRETATION
- Treat webhooks as pointers to verify, not as the fact. DERIVATION
- Let evidence name the stage it attests, and keep reconciliation workflow state separate from ledger truth. INTERPRETATION
- Separate authorization from signing so that a payment can be refused before it is irreversibly effected. INTERPRETATION

15. Application in StoneReason

This section describes how one implementation applies the model. It is product description, not part of the report's evidence, and nothing here should be read as support for the distinctions above — those stand or fall on Sections 3–13.

- **Payment Truth** keeps occurrence, amount, destination, execution, native finality and timing as separate axes, each network's settlement condition in its own terms rather than a shared boolean. PRODUCT FACT
- **Payment Passport** issues signed evidence that names what it attests and is verifiable with a published key — an instance of Section 11, including its limit that signed evidence is not consensus. PRODUCT FACT
- **Reconciliation** records workflow state (underpaid, overpaid, late, reorged) without rewriting recorded payment truth, per Section 12. PRODUCT FACT
- **Guard** decides what may be paid before execution and never signs or moves value, implementing the authorization/signing separation of Section 8. PRODUCT FACT

16. Limitations

- The report is conceptual. It quantifies none of the timing gaps it describes; the intervals between stages are asserted qualitatively and attributed to protocol documentation. LIMITATION
- Four networks are examined; the taxonomy's generality to others (e.g. UTXO variants, rollups with their own finality inheritance, DAG ledgers) is argued but not demonstrated. LIMITATION
- Protocol characteristics evolve. Network-specific claims are true as of the cited documentation and should be re-verified against current specifications. LIMITATION
- The model is descriptive, not normative: it does not prescribe which stage a given business should resolve on. LIMITATION

17. Future empirical work

The distinctions defended here are testable. A controlled measurement program — not a claim over private customer traffic — could quantify, for a defined test population on named networks: observation latency; the interval from execution to native finality; the gap between provider assurance and finality; and webhook timing relative to on-chain occurrence. Such work would have to preserve `PROVIDER_TIMING != NATIVE_FINALITY_TIMING` and `CONTROLLED_TEST_POPULATION != GLOBAL_PAYMENT_POPULATION`, and disclose sample size, observation window, inclusion and exclusion criteria, source dependencies and missing-data treatment. It is described here as a direction, and is deliberately not reported as a result. HYPOTHESIS

Methodology

Definitional and synthesis analysis. The event model is constructed from first principles of distributed-ledger settlement and payment-system reasoning, then checked against primary protocol documentation for four networks and against the CPMI-IOSCO settlement-finality standard. No original measurement was performed; every network-specific characteristic is sourced to protocol or standards documentation, not inferred from any implementation.

Data availability

No original empirical dataset is associated with this report. It is a conceptual/synthesis technical note. Quantitative timing claims are attributed to the cited protocol documentation; StoneReason has not measured them here. Future empirical work is described in Section 16.

Conflict and affiliation disclosure

StoneReason develops payment infrastructure (Payment Truth, Payment Passport, Reconciliation, Guard) related to the topics discussed in this research. The distinctions defended here are argued from protocol and payment-system evidence and are intended to stand independently of that product; Section 15 states, separately and neutrally, how the product applies them. This report was authored by StoneReason Research; it is not the work of an independent university or institute, and no individual researcher is named.

References

1. [A] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf
2. [A] Garay, J., Kiayias, A., & Leonardos, N. (2015). The Bitcoin Backbone Protocol: Analysis and Applications. EUROCRYPT 2015. https://eprint.iacr.org/2014/765
3. [A] Buterin, V., & Griffith, V. (2017). Casper the Friendly Finality Gadget. arXiv:1710.09437. https://arxiv.org/abs/1710.09437
4. [A] Buterin, V., Hernandez, D., Kamphefner, T., et al. (2020). Combining GHOST and Casper. arXiv:2003.03052. https://arxiv.org/abs/2003.03052
5. [A] Ethereum.org. Gasper — Proof-of-stake consensus documentation. https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/gasper/
6. [A] Anza / Agave. Optimistic Confirmation (Solana proposals documentation). https://docs.anza.xyz/proposals/optimistic_confirmation
7. [A] Mazières, D. (2015). The Stellar Consensus Protocol: A Federated Model for Internet-level Consensus. Stellar Development Foundation. https://stellar.org/papers/stellar-consensus-protocol.pdf

8. [A] CPMI-IOSCO (2012). Principles for Financial Market Infrastructures (PFMI). Bank for International Settlements. https://www.bis.org/cpmi/publ/d101.htm
9. [A] Aldasoro, I., Aquilina, M., Lewrick, U., & Lim, S.H. (2025). Stablecoin growth: policy challenges and approaches. BIS Bulletin No. 108, Bank for International Settlements. https://www.bis.org/publ/bisbull108.pdf
10. [A] Vogelsteller, F., & Buterin, V. (2015). EIP-20: Token Standard. https://eips.ethereum.org/EIPS/eip-20
11. [A] Chain Agnostic Standards Alliance. CAIP-19: Asset Type and Asset ID Specification. https://chainagnostic.org/CAIPs/caip-19
12. [A] International Organization for Standardization. ISO 4217 — Currency codes. https://www.iso.org/iso-4217-currency-codes.html

How to cite

```
StoneReason Research. "Payment Finality Is Not One Event: A Systems Taxonomy for Stablecoin Payment Infrastructure." StoneReason Technical Research Note SR-TR-001, version 1.0, 2026. https://stonereason.com/research/reports/payment-finality-is-not-one-event/
```

Revision history

Revision history of SR-TR-001.

Version	Date	Change
1.0	2026-09-20	Initial publication.