DIAGNOSTIC AND REPLACEMENT SERIES · PAPER 8

HOST-CONSTRAINED PRIMITIVE MISMATCH

The Structural Decay of Relationship-Coded Systems Inside Transaction-Coded Hosts

Jamie Forrester

Independent Systems Architect · Edinburgh, UK

hello@jamieforrester.com · jamieforrester.com

April 2026 · v4.0 · Working Paper

If you do not own your payment grammar, you do not own your business model.
If you cannot name the primitive, you cannot name the exit.

Abstract

A platform enters host-constrained primitive mismatch when its core economic primitive cannot be expressed in the governing interface of its host, and it cannot replace the host without losing critical access to the population, surface, or commercial pathway the host controls. At that boundary, the host does not merely tax the platform. It rewrites the platform's operating object — and every corrective action that does not name that rewriting will fail.

This paper names the failure class, traces its full mechanism through the Patreon/Apple case (the cleanest available instance, where every mechanism is publicly evidenced), and extracts four portable instruments: the Primitive Switch, Representational Loss, the Trust Tax Corollary, and the Platform Decay Spectrum. It provides a nine-question scored Host Constraint Test, a decision flowchart, and a universal four-exit redesign model with admissibility conditions integrated into each exit.

The verdict: if a platform needs an external grammar to speak to its own users, it does not fully control the business it believes it operates.

For Platform Leaders — Read This First

You are in host-constrained primitive mismatch if all four conditions hold: your platform has a native economic primitive; a host controls the interface through which it becomes commercially real; the host's grammar cannot represent that primitive without loss; and you cannot replace the host without losing critical access to the population, surface, or

commercial pathway the host controls.

The fee is not the problem. It is one downstream consequence among many. The problem is that your founding economic logic must pass through a grammar that cannot represent it — and every fix you have deployed has operated inside that grammar rather than naming it.

Four structural exits exist: Decouple (hold native primitive internally, expose host-compatible form at boundary), Allocate (change the economic object so the host sees a different compatible entity), Exit (move the commercial event outside the host interface), or Abandon (accept the host's grammar and become a different kind of platform). None is easy. None is optional. Each exit is mapped in Section IX with its admissibility condition and primary failure mode.

Concept Hierarchy — How the Instruments Relate

Failure class: Host-Constrained Primitive Mismatch

Governing law: Hostage Platform Law

Core mechanism: Representational Loss

Primary event: Primitive Switch

Downstream burdens: Trust Tax, Value Leak, Refund Paradox

Progression model: Platform Decay Spectrum

Strategic response: Four Exits (Decouple, Allocate, Exit, Abandon)

Representational Loss and the Primitive Switch are the load-bearing mechanisms. The Trust Tax, Value Leak, and Refund Paradox are their downstream consequences. All are expressions of the Hostage Platform Law. The Platform Decay Spectrum maps which stage of the failure class a platform has reached. The Four Exits are the only structural responses to a confirmed case.

Series Position

This is Paper 8 in the Diagnostic and Replacement Series. Papers 1–3 extracted the diagnostic laws: interface-legitimacy mismatch, function collapse under scale, and why correction loops cannot generate redesign from within the same frame. Papers 4–6 named the replacement mechanics: burden transfer, the five-stage replacement pipeline, and admissibility. Paper 7 named the Governed Correction Sequence. Paper 8 applies the full toolkit to a new failure class — externally imposed primitive compression — and extends the toolkit with four new instruments. Every prior paper named failures internal to the system or to its correction loops. Paper 8 names a failure that arrives from outside.

In Paper 9's terms, host-constrained primitive mismatch is kernel compression: the platform's native kernel configuration (Value/Allocation, Meaning/Interpretation, Sovereignty/Consent, Continuity/Memory) is forced through a host grammar that can only represent a subset of it. The instruments named here — Primitive Switch, Representational Loss, Trust Tax — are the observable consequences of that compression at the operational layer.

Reader Map

Reading Path	Sections
Full read — platform architects, researchers, institutional designers	All sections and appendices
Executive read — CPO, CEO, platform lead	For Platform Leaders box, then Sections I, II, III, VIII, IX, XII
Series reader	Sections II and XII only
Practitioner / diagnostic read	Sections VIII, IX, X

Section I The Hidden Failure Is Not the Fee

What kind of dispute is this?

"App Store fee dispute" is not a category. It covers three structurally different conflicts. Fee disputes are disagreements about commission rates resolvable through negotiation without changing what either party is — Netflix and Apple have one, and it produces no primitive mismatch because Netflix's standardised monthly subscription is exactly what Apple's IAP grammar represents. Grammar disputes are conflicts where the platform's native economic object cannot be expressed inside the host's interface without structural loss — cutting Patreon's commission to zero would not restore per-creation billing to the iOS app. Sovereignty disputes are the terminal stage: the platform has lost authority over the commercial event itself regardless of fee rate. This paper addresses grammar disputes.

The structural question that could not be answered

In January 2026, one creator — Josh 'Cheeseness' Bush — suspended billing and cancelled memberships in public protest, after eighteen months of documented correspondence with Patreon in which the structural rationale for a billing migration was never explained in a way that preserved his economic relationship with his community. He was asking why billing his fans on a monthly calendar, regardless of his publishing output, was economically equivalent to the per-creation model he had built his community around. Patreon could not answer that question. Not because it was unwilling. Because the answer would have required naming something it had not yet named. This paper explains what that thing was.

The refusal

Every public account of the Patreon/Apple conflict treats the 30% commission as the problem. The fee is real. The checkout friction is real. The legal uncertainty is real. But these are downstream of the break. The break is upstream: at the precise moment when a platform's native economic object must pass through a host interface that cannot represent it. Everything visible is a consequence of that passage. Nothing that operates at the visible layer can fix what operates at the invisible one.

This paper makes no claim about bad faith. Apple's IAP grammar is coherent on its own terms. The failure class arises from structural incompatibility under conditions of platform dependency, not from malice. The secondary cases — Spotify, Kindle, Epic Games — are lower-resolution class checks. Their inclusion establishes transfer plausibility, not full evidential equivalence with

Patreon. This paper does not assess commercial fairness, assign moral failure, or propose regulatory remedies.

Section II The Failure Class — Defined Before the Evidence

The case instantiates the class. It does not derive it.

Host-Constrained Primitive Mismatch — Canonical Definition

A system enters this failure class when its core economic or coordination primitive cannot be expressed in the governing interface of its host platform, and the platform cannot replace the host without losing critical access to the population, surface, or commercial pathway the host controls.

The four necessary conditions

- The platform has a native economic primitive — a specific form of value exchange it was built to hold.
- A host platform controls the interface through which that value exchange must be commercially initiated.
- The host's interface grammar is structurally incompatible with the native primitive: it cannot represent it without structural loss. Loss of logic, not just loss of formatting.
- The platform cannot replace the host without losing critical access to the population, surface, or commercial pathway the host controls.

The Hostage Platform Law

A relationship-coded economic system cannot be faithfully executed inside a transaction-coded payment interface without structural loss. When the payment interface is controlled by an external authority whose grammar is incompatible with the platform's native economic logic, every corrective action that does not rename the primitive will fail at the interface and produce compensatory burden at every adjacent layer.

Representational Loss

Representational loss occurs when the host interface can process a numeric or transactional substitute for the platform's native object, but cannot preserve the governing logic that made that object meaningful.

Translation preserves logic in a new form. Compression discards logic and keeps only the number.

The Hostage Platform Law names the structural condition. Representational Loss names the specific mechanism by which primitive compression occurs. It is a subordinate mechanism of the governing law, not a separate governing principle.

The interface as a filter

Interfaces are not neutral pipes. An interface encodes a grammar — a set of objects it can represent, relationships it can hold, and authorities it assigns. Apple's IAP interface behaves like a filter: it allows the payment event through and strips the relationship logic away — the output cadence, the fan-set cap, the creative act as the billing trigger. The relationship is not translated by passing through this filter. It is truncated. What passes through is not what entered.

The predictable structural consequences — the class signature

- Authority over the commercial event shifts to the host — payment initiation, refund authority, and remittance timeline governed by the host regardless of the platform's internal definitions.
- The platform's internal logic becomes compensatory relative to the host's grammar.
- Ghost structures form to carry the functions the product can no longer perform at the boundary.
- Failure repeats regardless of local optimisation — the primitive is not named; the fix is compensatory; the failure returns.
- The legal surface opens when the product surface is exhausted. When a system reaches the legal surface, it is a diagnostic signal that the product surface has been completely exhausted. Lawyers enter the room when the frame-level gate was never reached.

Section III The Primary Instance — Patreon under Apple IAP

Patreon satisfies all four conditions. The case is used here as proof, not as topic.

The canonical primitive

Patreon's native primitive is creator-defined patronage: a fan supports a specific creator on creator-defined terms, with the economic relationship governed by creative output, cadence, fan-set cap, and community meaning — not by a standardised subscription interval. That primitive is what the platform was built to hold, what creators and fans understood they were entering, and what the host interface cannot represent.

Dimension	Patreon Native (Relationship-Coded)	Apple IAP Grammar (Transaction-Coded)
Billing trigger	Creative act	Calendar interval
Duration logic	Output-linked	Time-linked

Dimension	Patreon Native (Relationship-Coded)	Apple IAP Grammar (Transaction-Coded)
Fan authority	Fan-set monthly maximum	None — no fan-set maximum
Billing variance	Irregular, creator-defined	Standardised
Economic object	Ongoing patronage relationship	Recurring purchase event
Memory type	Operational — preserves output history	Audit — records subscription event only
Authority at refund	Creator’s tier rules	Apple only

The four conditions demonstrated

- Condition 1 — Native primitive confirmed: "Creators define their own economic relationship with fans." Tier A: Patreon Help Centre and founding documentation.
- Condition 2 — Host controls decisive interface confirmed: iOS is "the number one platform for fan engagement and new membership sign-ups." Apple’s IAP is mandatory for payment initiation inside that app. Tier A: Patreon’s public statements.
- Condition 3 — Incompatibility confirmed: "Apple’s in-app purchase system only supports creators on subscription billing." Per-creation migration requires 1:1 support and cannot be automated. Tier A: Patreon Help Centre, March 2026.
- Condition 4 — Cannot replace host confirmed: three compliance iterations in eighteen months, formal complaint filed, sustained compliance under protest. Tier A: Patreon for Creators announcements, 2024–2026.

The Primitive Switch — the first structural break

The moment a fan taps "Join" inside the Patreon iOS app. Before it: the system presents creator-defined patronage. At it: the relationship-coded support intent must become a transaction-coded payment event in Apple’s governed purchase surface. Per-creation billing logic is not representable. Fan-set monthly maximums are not representable. The creator’s economic grammar disappears at the boundary. Everything downstream traces to this moment: the 30% fee, the Refund Paradox, the Value Leak, the multiplier, the ghost structure, the legal complaint. None are independent problems. They are the downstream consequences of the Primitive Switch operating without correction at the frame level.

The Transfer Test — resolved (Paper 1 method)

What Patreon signals: creator-defined patronage economics, billing linked to creative output, fan-set monthly caps. What Patreon delivers under constraint: host-conditioned subscription-compatible payment events at the iOS boundary, with the creator’s economic grammar absent from every transaction Apple processes. Positive divergence confirmed. The Transfer Test is resolved.

Section IV The Anatomy of the Mismatch — Mechanism

Economic compression — Representational Loss in operation

The multiplier mechanism is the clearest proof of Representational Loss. It takes the maximum number of paid posts in any single month over the prior three months and converts that to a monthly subscription price. A creator who published zero posts in two months and four in one is priced at four posts per month — regardless of future output. The billing event is severed from the creative act it was designed to reflect. The number survives the filter. The output-linked, fan-capped, irregular logic does not.

Patreon's own support architecture confirms the distinction: per-creation migration cannot be automated because the conversion is not a formatting change — it is a primitive substitution requiring human judgment. When a platform cannot automate translation between two of its own billing models, the translation is not a formatting problem. It is a primitive conflict.

Audit memory vs operational memory (Paper 2 applied)

From Paper 2's Four-Function Law: audit memory preserves retrospective defensibility; operational memory preserves usable state for the next person who must act. The subscription record created by multiplier migration is audit memory — it records what billing event occurred. It does not preserve the output-linked history of the original per-creation contract. The operational form no longer exists in the system. The relationship has been rewritten in audit form.

The Value Leak — sensing gap mechanised

Apple's payment remittance cycle introduces a structural sensing gap — typically approximately 45 days after month-end, up to 75 days in documented cases. If a fan cancels and Apple processes a refund, Patreon may not know for weeks. The creator continues delivering exclusive content against a commercial relationship Apple has already terminated. Sensing held by one party; accountability held by another. That is the Value Leak.

The Refund Paradox — accountability without authority

When a fan requests a refund on an iOS purchase, Apple governs the outcome. The creator's tier rules do not apply. Apple determines whether the refund is granted and when the membership ends. The creator retains full delivery accountability but holds no commercial authority at the point of consequence.

Accountability without authority, at the exact layer where accountability and authority must be coextensive for a creator-fan economic relationship to remain coherent.

The Four-Function Law at the class level (Paper 2)

Function	Condition	Mechanism of Failure
Sensing	Split and degraded	Patreon cannot sense Apple's payment status during the remittance cycle. The Value Leak operates in this gap.
Interpretation	Fragmented	Apple interprets its rules unilaterally; Patreon interprets compliance; creators interpret economic impact. Documentation proliferation is structural evidence of this fragmentation.
Authority	Externalised and unequal	Apple governs payment initiation, refund, and remittance. App update blocked late 2025.
Memory	Degraded to audit only	The subscription record does not preserve the output-linked history of the original per-creation contract. The relationship is not preserved. Its numeric shadow is.

The Collapse Cascade (Paper 2) — traced through Patreon

Stage	Patreon Instantiation
1. Personalisation of responsibility	Creators blamed for Apple’s pricing; Patreon blamed for Apple’s rules. Apple’s structural authority at the boundary disappears from the visible account.
2. Documentation inflation	Separate help articles for fans, creators, migration, iOS pricing, fee handling, refund authority, and web checkout. Audit memory grows; operational continuity does not.
3. Process sprawl	Migration deadline, multiplier, repricing tool, 1:1 support, web checkout, external browser migration, annual membership. Each clarifies a compliance procedure without relocating the primitive conflict.
4. Exception overload	Per-creation creators cannot self-serve migration. Each requires human judgment. Support capacity is finite; Apple’s deadline is not.
5. Informal substitution	Creators perform the routing function the platform cannot. The off-platform instruction layer becomes load-bearing.
6. Legibility inversion	The iOS pricing help article makes the structural failure legible to observers while making the product harder to inhabit at the point of commercial consequence.

Section V The Ghost Structure and the Trust Tax

The ghost structure

When a product cannot perform a function the commercial relationship requires, users build infrastructure to perform it themselves. Paper 2 named this the ghost structure. In Patreon’s case, it is off-platform — displaced onto the creator layer. This makes it the most expensive form: it consumes the creator’s relationship capital rather than an internal operational resource. Discord servers, Twitter threads, creator community posts — all teaching fans how to bypass the iOS app and subscribe through the web. This is not creator creativity. It is unpaid structural load. The size and density of the ghost structure is a direct measure of the mismatch: as it grows, the platform is losing its category definition as frictionless membership infrastructure.

Three burdens creators carry

Communication burden: each Apple policy reversal reaches creators through Patreon’s communications. Creators relay each reversal to their fan communities through their own channels, using trust they built through their creative work. Three reversals in eighteen months means three draws on an account the platform did not build and cannot replenish.

Instruction burden: every post saying "sign up on the web, not in the app" is a measurable instance of Patreon losing its category definition as frictionless membership infrastructure. The creator absorbs the awkwardness of the instruction and the friction in the subscription flow that results.

Price-explanation burden: Patreon publishes a help article specifically for creators to share with fans, explaining why iOS prices are higher. The structural cost of Apple’s commission is

communicated at the creator-fan relationship layer, by creators, using trust they built for other purposes.

> **The Trust Tax Corollary**
>
> When a platform cannot absorb the cost of a host's constraint, the cost is displaced onto the most relationship-dependent actor in the system. That displacement is invisible in the platform's accounts and appears as a tax on trust.
>
> The Trust Tax does not appear on Patreon's balance sheet. It is paid in full, every time, by creators. In Paper 4's terms, the Trust Tax is a form of avoidable burden that has not yet been made avoidable. Competitors understand this. They are competing with the Trust Tax Patreon cannot stop collecting.

Josh 'Cheeseness' Bush's situation is the Trust Tax becoming visible as a relational event in a specific person's life. The platform could not answer his question because the answer would have required naming a primitive mismatch the correction loops were not built to surface. The ghost structure cannot explain itself. It can only perform the function it exists to perform, until the person it depends on stops performing it.

Section VI Why Ordinary Fixes Fail

From Paper 3's Redesign Law: execution improves performance within a frame; diagnosis improves control within a frame; redesign changes the frame itself. The first two do not reliably produce the third. Every corrective action Patreon has deployed is execution or diagnosis correction. None is redesign. None changes the frame.

Fix Deployed	Correction Layer	Why It Cannot Resolve the Mismatch
Migration deadline (Nov 2025)	Execution — compliance timeline	Reversed after Epic ruling; created further creator uncertainty
Web checkout in iOS (May 2025)	Execution — acquisition surface	Apple required move to external browser; in-app continuity broken
External browser checkout	Execution — fee avoidance	US only; breaks impulse support; contingent on Epic ruling under Supreme Court review
Multiplier mechanism	Execution — billing translation	Representational loss — compresses the primitive; does not preserve it
1:1 migration support	Diagnosis — transition management	Manages the migration; does not resolve the mismatch
Formal complaint against Apple	Legal surface — product surface exhausted	Legal surface opens when the product surface has been completely exhausted (Paper 3 recognition bias). No native redesign lane in correction loops; legal is what remains.

Fix Deployed	Correction Layer	Why It Cannot Resolve the Mismatch
Tools proposed to Apple	Diagnosis — policy advocacy	All declined — consistent rejection confirms primitive conflict, not tooling gap

No tool resolves a conflict at the level of the primitive by operating at the level of the tool.

Apple declined every tool Patreon proposed. A tooling gap would yield at least one accepted proposal. Consistent rejection means the constraint is not resolvable at the tooling layer. The problem is a conflict at the level of the primitive.

Section VII The Pattern Beyond Patreon — Lower-Resolution Class Checks

Spotify, Kindle, and Epic Games are lower-resolution class checks, not full diagnostic artifacts. Their inclusion establishes transfer plausibility, not full evidential equivalence with Patreon. Each confirms the four necessary conditions and receives a placement on the Platform Decay Spectrum.

Stage	What It Looks Like	Trust Tax Status	Case
Active Friction	Mismatch visible as product absence. Users can see what is missing.	Paid by users explaining the missing feature	Spotify
Structural Crisis	Mismatch breaking founding logic. Platform in public dispute with host.	Paid by creators in relationship capital	Patreon
Total Exit	Mismatch forced total war. Platform chose removal over compliance.	Paid by users navigating platform restrictions	Epic
Total Absorption	Mismatch invisible. Product rebuilt entirely around host grammar.	Paid by users; normalised as product friction	Kindle

Spotify — Active Friction

Native primitive: continuous listener-artist relationship. Host constraint: Apple’s IAP cannot represent a platform-level media relationship without governing the payment event. Four conditions confirmed. The off-platform instruction layer is Spotify’s official iOS onboarding experience for new subscribers — the ghost structure has been formalised as product. Institutional Exit Signal: formal complaint to the European Commission. Decay stage: Active Friction.

Kindle — Total Absorption

Native primitive: book ownership as a durable, accumulating library asset. Host constraint: Apple’s IAP restriction prevents selling books inside the iOS app without commission. Four conditions confirmed. No "Buy" button. Books purchased elsewhere, loaded to the reading

surface. The compensatory architecture has been absorbed so completely it has become the product's official user experience. New iOS Kindle users learn this as normal. This is the destination Patreon must refuse. Decay stage: Total Absorption.

Epic Games — Total Exit

Native primitive: cross-platform game economy with persistent in-game currencies and direct player-developer commerce. Host constraint: Apple's IAP mandatory for in-app commerce; Epic's direct payment mechanism is structurally incompatible. Four conditions confirmed. Epic built its own payment mechanism into Fortnite and accepted App Store removal as the consequence. Outcome as of April 2026: partial, contested, Supreme Court pending, Apple's stay granted April 6, 2026. Epic is not evidence that exit works. It is evidence of what exit costs. Decay stage: Total Exit.

Platform	Native Primitive	Mismatch Point	Decay Stage	Legal Signal
Patreon	Creator-defined patronage	Support intent → IAP event	Structural Crisis	Formal complaint
Spotify	Continuous media access	Subscription → app store event	Active Friction	EU complaint
Kindle	Library accumulation	Book purchase → app store event	Total Absorption	Absorbed
Epic	Cross-platform game economy	Direct commerce → single-host event	Total Exit	Supreme Court

Section VIII The Host Constraint Test

Use this test when a platform is experiencing persistent friction with a host — recurring policy cycles, compensatory fixes that do not resolve the underlying issue, or ghost structures carrying functions the product cannot. Score each element 0 (not identifiable or unclear), 1 (partially specified), 2 (specific, evidenced, and structurally useful). A score of 2 requires a specific answer, not a general impression. Maximum score: 18.

Q	Question	What the Answer Reveals
1	What is the platform's native primitive — not what it says it does, but what economic object it actually creates, governs, and is commercially dependent on?	The native primitive. Compare against Q3 to identify the mismatch point.
2	What host interface controls the decisive commercial event — where must the native object pass through an external grammar to become commercially real?	The location of the Primitive Switch. This is where structural loss occurs.

Q	Question	What the Answer Reveals
3	What grammar does the host recognise — what billing forms and economic objects can it represent, and what cannot pass through unchanged?	The host's representational limit. The gap between Q1 and Q3 is the Representational Loss.
4	What part of the native primitive becomes unrepresentable at the host boundary — what logic is stripped while the number passes through?	The specific mechanism of compression. The surviving number versus the lost logic is the Primitive Switch in operation.
5	Where does authority shift — who governs the payment event, the refund, and the remittance?	The structural authority map. The entity that governs these holds authority regardless of commercial intent. If Apple governs all three, the dependency condition is confirmed.
6	What compensatory burden appears downstream — what layers have been built to absorb what the product can no longer perform at the boundary?	The load-bearing evidence of structural failure. Each compensatory layer is a ghost structure in formation.
7	What ghost structure has formed — what are users doing off-platform to carry functions the product cannot?	The ghost structure. External ghost structures consume relationship capital — the most expensive form. If present, institution substitution (Paper 2) is confirmed.
8	Is each fix execution or diagnosis correction — does any corrective action reach the frame level?	The correction layer reached. If all fixes are execution or diagnosis, the frame-level gate has not been reached and correction collapse (Paper 3) is confirmed.
9	Which exit is structurally available — what does the Platform Decay Spectrum predict, which exit preserves the truth kernel, and what admissibility conditions must each exit satisfy?	The strategic position. Class membership is confirmed if a specific exit can be specified with its admissibility conditions. Inability to specify an exit means the platform is still in compliance.

Total Score	Status	Action
12–18	Class membership confirmed	The four conditions are met. Apply the four-exit model in Section IX. Specify the truth kernel before evaluating any exit. Begin with Exit A unless the category change of Exit B or the legal stability of Exit C is already in place.
6–11	Partial — further investigation required	At least one condition is unclear or partially evidenced. Identify which question scored lowest — that is where the structural picture is incomplete. Do not assume class membership until Q1–Q5 are all clearly specified.
Below 6	Insufficient evidence or class does not apply	Either the platform has not yet been examined closely enough, or the failure class does not apply. Check boundary conditions in Section X. If the host grammar is a superset of the native primitive, the failure class does not apply regardless of fee friction.

Decision Flowchart

Step	Condition	Branch
Step 1	Does the platform have a native economic primitive?	No → framework does not apply. Yes → Step 2.
Step 2	Does a host control the decisive commercial interface?	No → constraint is negotiable friction, not a structural trap. Yes → Step 3.
Step 3	Is the host grammar structurally incompatible with the native primitive?	Host grammar is a superset (e.g. Netflix/IAP) → no mismatch. Native primitive becomes stripped or compressed at boundary → Step 4.
Step 4	Can the platform replace the host without critical loss?	Yes → dependency condition fails; constraint is painful but not a structural trap. No → failure class confirmed. Apply the Host Constraint Test. Proceed to Step 5.
Step 5	Where does the case sit on the Platform Decay Spectrum?	Active Friction → Structural Crisis → Total Exit → Total Absorption. Stage determines urgency and available exits. Proceed to Section IX.

The truth kernel

Before any exit is evaluated: what is the truth kernel — the minimal durable continuity structure any redesign must protect? (Paper 5.) For Patreon: the patronage relationship between a specific fan and a specific creator, on creator-defined terms, with output-linked billing logic, preserved in operational form across commercial events. Exit A preserves it internally. Exit B transforms it. Exit C attempts to protect it by moving outside the host. Exit D destroys it. No exit is correct without first specifying what must be preserved.

Section IX The Four Exits — A Universal Redesign Model

Every platform confirmed in the failure class has four structural exits and one non-exit. These are the only four moves structurally available. Everything else is compensation. Each exit includes its admissibility condition and primary failure mode.

The non-exit — Comply

Remain inside the mismatch. Accept primitive compression. Build compensatory layers. This is the default state before the mismatch is named. Compliance compounds: its cost grows with every policy cycle and every creator who reaches the Migration Threshold.

Ungoverned translation is a patch. Governed translation is an architecture. Compliance is neither.

Exit A — Decouple (Boundary Decoupling)

Hold the native primitive as a first-class object inside the platform’s own system. Expose only host-compatible events at the interface boundary. The host sees a subscription. The platform holds the relationship economics behind it.

Success condition: a fan joins a per-creation creator’s iOS membership, pays through Apple IAP, and the creator receives a net economic outcome equivalent to the pre-Apple relationship — output-linked logic preserved inside Patreon’s system, invisible to Apple’s billing grammar.

Admissibility condition: the internal layer must satisfy refusal integrity (Paper 6) — it must be able to structurally refuse compression of the patronage logic it holds. Specifically, if Patreon’s internal system begins interpreting creator relationships as "monthly subscriptions" because Apple does, the decouple has failed. The internal system must remain sovereign to its own primitive. Creators must be able to inspect that the internal layer is holding their economic logic faithfully.

Primary failure mode — the Successor Trap (Paper 6): as the translation layer scales, maintenance pressure may cause it to drift toward the external subscription logic. Governed translation becomes ungoverned compression. The mismatch reappears inside Patreon’s own architecture, invisible to Apple and to creators.

Exit B — Allocate (Allocation Shift)

Change the economic object so the host sees one compatible transaction while native allocation logic operates behind it. The fan holds one relationship with Patreon; the platform allocates to creators according to creator-defined logic, including output-linked cadence, invisible to the host.

Admissibility condition: fans must accept a relationship with the platform rather than directly with creators. Creators must accept the platform as the economic intermediary. This is a category change, not a feature change.

Primary failure mode: creators whose identity is built on direct fan relationships may refuse this exit not because the economics are worse but because the relationship structure is different.

Exit C — Exit (Host Exit)

Move the commercial event outside the host interface. Web checkout is the partial version. The admissibility challenge here is specific: Apple currently holds the admissibility layer. For a fan, paying through Apple is "official" — it carries the trust and familiarity of a platform they already have payment details stored in. Moving to web checkout requires Patreon to build its own admissibility from scratch with each fan. That is why the external-browser friction is so structurally damaging: it is not merely inconvenient. It is a challenge to the user’s sense of officialness at the exact moment commercial intent is highest.

Admissibility condition: the web checkout route must simultaneously satisfy legal stability (currently contingent on the Epic ruling under Supreme Court review, stay granted April 6, 2026), geographic accessibility (currently US fans only), and impulse economics (external browser breaks the impulse support moment at the highest-value point in the funnel). None of these three are currently met in full. Exit C is not fully admissible in its present form.

Primary failure mode — legal reversal: if the Supreme Court reverses the Epic ruling, the web checkout option disappears. Epic is not evidence that exit works. It is evidence of what exit costs.

Exit D — Abandon (Category Abandonment)

Accept the host's grammar and become a different kind of platform. Stop holding the native primitive. Kindle is the advanced-stage case. Patreon's sustained "we strongly disagree" position is the platform refusing this exit explicitly.

Admissibility condition: always technically admissible because the host already accepts what it proposes. The question is internal: can the creator and fan base accept the change in what the platform is? Primary failure mode: every creator who joined on the promise of creator-defined patronage has, under Exit D, been told that promise was temporary. The Trust Tax becomes a single catastrophic draw.

Strategic triage

Structural Situation	Direction
Cannot exit the host; bilateral positioning non-negotiable	Exit A (Decouple). Specify the truth kernel first. Design refusal integrity into the translation layer from the start. Monitor for Successor Trap.
Cannot exit the host; category change acceptable	Exit B (Allocate). Validate creator and fan acceptance of tripartite structure before build. Strategic decision, not a product decision.
Host constraint is the only obstacle; legal foundation stable	Exit C (Exit). Do not build against an unstable legal ruling as a stable foundation. Watch Epic's Supreme Court outcome first.
Founding primitive no longer commercially defensible	Exit D (Abandon). Name it to the creator base before the Migration Threshold names it for you.
Still deploying compensatory features	You are in compliance. Stop when: same problem returns under new deadline; 1:1 support exists for automated processes; ghost structure is the only coherence layer; legal surface has opened.

Section X Boundary Conditions — What This Paper Does Not Claim

Non-Claim	Why the Boundary Matters
Host grammar is always incompatible with native primitives	The failure class applies only where the host grammar cannot represent the native primitive without structural loss. Where the host grammar is a superset of the native primitive (Netflix/IAP), no mismatch exists regardless of fee friction.
The fee is the problem	Cutting the fee to zero does not restore per-creation billing to the iOS app. The class is about grammar incompatibility, not commission rate.
Host platforms act in bad faith	Apple's IAP grammar is coherent on its own terms. The failure class arises from structural incompatibility under conditions of platform dependency, not from malice.
All platforms in the failure class should exit	Exit viability depends on admissibility conditions specific to each case. An exit that is structurally correct may still be inadmissible. Paper 6 specifies admissibility conditions.
The framework applies to non-payment constraints	The framework is most precise in payment-mediated, bilateral relationship systems. Content moderation restrictions, hardware

Non-Claim	Why the Boundary Matters
	certification requirements, and non-payment-mediated coordination constraints require different frameworks.
Completing the analysis guarantees exit success	Even a correctly specified exit does not automatically become admissible. Building toward a correct exit and ensuring that exit becomes institutionally real are different problems. Paper 6 addresses the second.
Physical infrastructure or coercive state power	The framework applies less directly where primary constraints are physical infrastructure, coercive state power, or coordination problems not mediated by payment interfaces.

Section XI The Cost of Continuation

Cost Category	Mechanism
Trust erosion	Each policy reversal spends creator trust Patreon cannot recover. Three reversals in eighteen months. Competitor platforms exploit the instability.
Category definition loss	Every creator post instructing fans to "sign up on the web" is a measurable instance of Patreon losing its definition as frictionless membership infrastructure.
Primitive compression	Every creator migrated under the multiplier has had their economic relationship with fans changed at its root. The billing event is severed from the creative act.
Value Leak	Creator delivers exclusive content against a commercial relationship the platform cannot yet confirm is active. The creator bears the delivery cost.
iOS conversion degradation	External-browser web checkout breaks the impulse support moment at Patreon’s highest-value acquisition surface. International fans have no alternative.
Legal uncertainty	Web checkout for US fans contingent on Epic ruling under Supreme Court review. Apple’s stay granted April 6, 2026. The last available structural remedy is legally contested.
Engineering capacity locked	The last eighteen months of Patreon engineering are compliance iterations, not creator innovations. The opportunity cost is visible in the product timeline.

These costs do not accumulate separately. They reinforce one another. Trust erosion increases migration pressure; migration pressure increases competitor opportunity; legal uncertainty prevents stable redesign; engineering capacity remains locked in compliance. Each policy reversal amplifies all of the others. A platform that does not exit the failure class does not reach a stable equilibrium — it reaches a point where the Migration Threshold is crossed and the compounding accelerates.

Section XII Conclusion — Sovereignty Is the Primitive

The structural condition

Host-constrained primitive mismatch is not a dispute about fees, policy, or platform fairness. It is the structural condition that appears when one system's native economic object must become another system's recognised event before it can exist commercially. At that boundary, the host does not merely process the platform. It changes what the platform is allowed to mean.

What this paper adds to the series

Contribution	Definition
Host-Constrained Primitive Mismatch	The failure class: a platform's native economic or coordination primitive cannot be expressed in the governing interface of its host, and the platform cannot replace the host without losing critical access to the population, surface, or commercial pathway the host controls.
Hostage Platform Law	A relationship-coded economic system cannot be faithfully executed inside a transaction-coded payment interface without structural loss. Every corrective action that does not rename the primitive will fail at the interface.
Representational Loss	The mechanism: the host interface processes a numeric substitute while stripping the governing logic. Translation preserves logic; compression discards it.
Primitive Switch	The primary event: the moment a native economic object must pass through a host interface that cannot represent it. Everything downstream traces to this moment.
Trust Tax Corollary	When a platform cannot absorb a host's constraint, the cost displaces onto the most relationship-dependent actor. Invisible in accounts; paid in relationship capital.
Value Leak	The sensing gap: creator delivers content against a commercial relationship Apple has already terminated but Patreon cannot yet see. Accountability held without sensing.
Refund Paradox	Accountability without authority at the exact layer where they must be coextensive. Apple governs refund outcomes; the creator bears delivery obligation.
Platform Decay Spectrum	Four observable stages: Active Friction, Structural Crisis, Total Exit, Total Absorption. A predictive model for where a given case is heading and what exit remains available.
Four Exits Model	Decouple, Allocate, Exit, Abandon — the only four structural responses to confirmed class membership, each with its admissibility condition and primary failure mode.
Host Constraint Test	A nine-question, scored diagnostic instrument for confirming class membership and locating the primitive conflict.
Decision Flowchart	A five-step gate sequence for determining whether the failure class applies and which exit is available.

Contribution	Definition
Concept Hierarchy	The ordering of all Paper 8 instruments under the governing law, preventing concept overload by making the dependency structure explicit.

Falsifiable Prediction

If the Hostage Platform Law is correct, any relationship-coded platform forced through a transaction-coded host interface will produce representational loss, compensatory ghost structures, Trust Tax displacement, and progression toward one of the four decay stages — regardless of fee rate, legal structure, or management quality.

The law would be weakened by a relationship-coded platform operating inside a transaction-coded host without representational loss, ghost structure formation, or Trust Tax displacement — or by a platform that exits the failure class through continued optimisation without redesigning the primitive. Across the cases examined, no such counterexample has been found.

A note on protocol-first architecture

The logical endpoint of the analysis is a design posture called protocol-first architecture: building platforms so that the sovereign record of the economic relationship — the patronage history, the output-linked ledger, the fan-set maximums — is held in a form the platform controls, and the interface (iOS, Android, web) is one temporary skin through which that record is expressed. Under this posture, no single host controls the grammar. The primitive is sovereign because it is not held in the host's interface at all. That is a future paper. The existing analysis is sufficient to identify the failure, name its costs, and specify the exits.

This paper's own admissibility acknowledgment

This framework holds the same structural position it describes: it names a primitive mismatch that correction loops were not built to surface, and it does not yet possess a formal recognition channel through which that diagnosis can be officially relied upon under consequence. By its own criteria, it is a redesign signal without a redesign interface — holding truth at the structural layer while lacking the admissibility conditions that would make it institutionally real. Whether it earns those conditions depends on whether the exits specified here produce lighter systems than the compliance iterations they are designed to replace.

The 30% fee is not the price of distribution. It is the price of primitive dependency — the structural cost of a relationship-coded platform whose founding economic logic must pass through a transaction-coded host before it can exist commercially.

The topology names the cause.
The interface names the constraint.

The primitive names the exit.

If you do not own your payment grammar, you do not own your business model.

Jamie Forrester · hello@jamieforrester.com · April 2026

If this maps to a platform you operate or a constraint you are designing against, you can reach me at hello@jamieforrester.com

Appendix A — Evidence Reference

Tier	Source Type	Contents and Weight
A	Primary public documentation (direct)	Patreon Help Centre: billing grammar, iOS pricing articles, per-creation migration documentation, refund authority statements, March 2026. Patreon for Creators announcements: migration timeline, compliance iterations, formal complaint filing, January 2026. Patreon public statements: iOS as "number one platform for fan engagement and new membership sign-ups." Apple IAP developer documentation: billing grammar, subscription models, refund authority.
B	Secondary case evidence	Spotify: formal complaint to European Commission; official iOS onboarding flow as documented ghost structure. Kindle: absence of "Buy" button as documented product constraint; Total Absorption as confirmed decay stage. Epic Games: Fortnite payment mechanism, App Store removal, legal proceedings through April 2026; Supreme Court stay granted April 6, 2026.
C	Structural inference	The Primitive Switch, Representational Loss mechanism, Trust Tax Corollary, Platform Decay Spectrum, and Four Exits model are structural inferences from the primary evidence and the framework developed across Papers 1–7. Clearly marked as inference throughout.
D	Cross-series application	Paper 1 (Transfer Test resolved), Paper 2 (Four-Function Law applied at class level, Collapse Cascade traced), Paper 3 (Redesign Law applied to every deployed fix), Paper 4 (Trust Tax as avoidable burden), Paper 5 (truth kernel specified), Paper 6 (Successor Trap named in Exit A, admissibility conditions per exit), Paper 7 (frame-level gate absence confirmed). All applications are explicit and section-referenced.

Appendix B — Glossary of Named Concepts

Term	Definition and Source
Host-Constrained Primitive Mismatch	The failure class: a platform’s native economic or coordination primitive cannot be expressed in the governing interface of its host, and the platform cannot replace the host without losing critical access to the population, surface, or commercial pathway the host controls. [Paper 8]
Hostage Platform Law	A relationship-coded economic system cannot be faithfully executed

Term	Definition and Source
	inside a transaction-coded payment interface without structural loss. Every corrective action that does not rename the primitive will fail at the interface and produce compensatory burden at every adjacent layer. [Paper 8]
Representational Loss	The mechanism by which primitive compression occurs: the host interface can process a numeric or transactional substitute for the native object, but cannot preserve the governing logic. Translation preserves logic; compression discards it and keeps only the number. [Paper 8]
Primitive Switch	The precise moment at which a native economic object must pass through a host interface that cannot represent it. Everything downstream traces to this moment. [Paper 8]
Trust Tax Corollary	When a platform cannot absorb the cost of a host's constraint, the cost displaces onto the most relationship-dependent actor in the system. Invisible in accounts; paid in relationship capital. [Paper 8]
Value Leak	The structural consequence of sensing being held by one party (Apple) and accountability being held by another (creator): creator delivers content against a commercial relationship the platform cannot yet confirm is active. [Paper 8]
Refund Paradox	Accountability without authority at the exact layer where they must be coextensive. Apple governs refund outcomes; the creator retains full delivery accountability. [Paper 8]
Platform Decay Spectrum	Four observable stages of mismatch progression: Active Friction, Structural Crisis, Total Exit, Total Absorption. [Paper 8]
Four Exits	Decouple (hold native primitive internally, expose host-compatible form at boundary), Allocate (change economic object so host sees compatible entity), Exit (move commercial event outside host interface), Abandon (accept host's grammar). [Paper 8]
Fee Dispute	A disagreement about commission rates resolvable through negotiation without changing what either party is. Distinct from a grammar dispute. [Paper 8]
Grammar Dispute	A conflict where the platform's native economic object cannot be expressed inside the host's interface without structural loss. The class this paper addresses. [Paper 8]
Hostage Platform Law	See above. [Paper 8]
Transfer Test	Compare what the system optimises for in execution with what it signals in recruitment or interface. Divergence is the tell. [Paper 1]
Four-Function Law	Institutions fail when sensing, interpretation, authority, and memory remain fused at the point of consequence. [Paper 2]
Redesign Law	Execution improves performance within a frame; diagnosis improves control within a frame; redesign changes the frame itself; the first two do not reliably produce the third. [Paper 3]
Migration Law	Institutions persist while the burden of their incoherence can be

Term	Definition and Source
	transferred outward; they begin to be replaced when a superior coordination architecture makes that burden avoidable. [Paper 4]
Admissibility Law	A substitute inherits institutional reality not when it surpasses the incumbent in design, but when consequence-bearing actors can safely rely on it without the incumbent's permission. [Paper 6]
Successor Trap	A substitute may clear the admissibility threshold initially and still decay into the topology it replaced. [Paper 6]

Appendix C — Series Reference

Forrester, J. (2026a). The Expertise Illusion in AI Task Marketplaces. SSRN Working Paper.

Forrester, J. (2026b). The Four-Function Law of Scalable Institutions. SSRN Working Paper.

Forrester, J. (2026c). Why Systems Can't Fix Themselves: The Missing Redesign Layer. SSRN Working Paper.

Forrester, J. (2026d). Institution Migration: How Better Coordination Makes Legacy Institutions Unnecessary. SSRN Working Paper.

Forrester, J. (2026e). The Institutional Replacement Pipeline. SSRN Working Paper.

Forrester, J. (2026f). The Admissibility Problem: Why Better Substitutes Still Fail to Replace Worse Institutions. SSRN Working Paper.

Forrester, J. (2026g). The Governed Correction Sequence. SSRN Working Paper.

Forrester, J. (2026h). Host-Constrained Primitive Mismatch: The Structural Decay of Relationship-Coded Systems Inside Transaction-Coded Hosts. SSRN Working Paper. [This paper]

Forrester, J. (2026i). The Kernel Reduction of Institutional Systems. SSRN Working Paper.

Forrester, J. (2026j). Kernel Recomposition Patterns: How New Institutional Categories Emerge from Novel Coordination Arrangements. SSRN Working Paper.

---

hello@jamieforrester.com · jamieforrester.com · Edinburgh, UK · Paper 8 v4.0 · April 2026