DP5 – Decentralized Namespace

Claim your space in the Meta-Layer — meta-domains and personal identifiers give you sovereign, portable identity, owned by you, not a platform.

Purpose of This Draft

This ML-Draft articulates Desirable Property 5 (DP5) as the condition under which people, communities, agents, artifacts, and spaces can be named, addressed, discovered, traded, and governed across the Meta-Layer without dependency on a single platform or registry.

DP5 introduces meta-domains and personal identifiers as sovereign, portable identity and addressability primitives. These identifiers allow participants to claim space in the Meta-Layer, link that space to existing web domains or decentralized identifiers, and use names as anchors for identity, ownership, trust, commerce, and governance.

The core claim is simple:

> Meta-domains and personal identifiers give participants sovereign, portable identity – owned by them, not rented from a platform.

The Meta-Layer introduces a decentralized namespace system where identity is not merely a login and addressability is not merely a URL. Names become portable anchors for people, ideas, artifacts, communities, overlays, smart tags, and virtual spaces across the open web.

DP5 guides implementation, governance design, and future ML-RFC development for decentralized naming, meta-domain registration, namespace rights, conflict resolution, and interoperable naming semantics.

Problem Statement

Why namespaces matter.

The contemporary web depends on naming systems, but most participant-facing names are not truly participant-owned. Handles, usernames, pages, tags, groups, channels, and platform identities are typically rented from centralized services. They can be revoked, shadowed, duplicated, impersonated, renamed, captured, or made non-portable by the platforms that host them.

This creates recurring failures:

- participants build identity and reputation around names they do not control

- communities lose continuity when platforms change rules or shut down access
- artifacts cannot be reliably addressed across tools
- names become vulnerable to spoofing, squatting, seizure, and censorship
- cross-system identity and ownership degrade because identifiers do not preserve meaning across contexts

DP5 reframes naming as civic infrastructure. A decentralized namespace is not only a convenience layer. It is the addressability substrate for identity, agency, data, commerce, interoperability, and governance.

Without DP5, the Meta-Layer cannot reliably answer basic questions:

- What is this person, persona, community, artifact, tag, overlay, or space called?
- Who controls that name?
- What does the name resolve to?
- What rights, policies, and histories are bound to it?
- How does the name remain interpretable across systems?

Threats and Failure Modes

Platform-rented identity

Participants build identity around handles or pages that can be revoked, hidden, renamed, or monetized by platform operators.

Failure mode: identity continuity depends on platform permission.

Namespace capture

Dominant registries or intermediaries control which names are valid, visible, or resolvable.

Failure mode: decentralized naming becomes centralized gatekeeping.

Spoofing and impersonation

Attackers create visually, semantically, or structurally similar names to mislead participants.

Failure mode: names become attack surfaces for scams and trust abuse.

Squatting and speculative enclosure

Valuable names are claimed not for use, but to extract rents from future participants or communities.

Failure mode: addressability becomes enclosure before public value can form.

Semantic drift

A name carries one meaning in one system and a different meaning elsewhere without signaling.

Failure mode: identity, trust, or ownership claims are misinterpreted across contexts.

Registry fragmentation

Multiple naming systems emerge without interoperability or conflict-resolution pathways.

Failure mode: participants cannot know which namespace claims are authoritative, compatible, or contested.

Artifact ambiguity

Objects, tags, posts, paths, and digital artifacts cannot be reliably referenced across systems.

Failure mode: knowledge, provenance, and ownership degrade because identifiers are not stable.

Non-human namespace ambiguity

AI agents, organizations, bots, and autonomous systems operate without clear namespace rights or management structures.

Failure mode: non-human actors become hard to distinguish, govern, or hold accountable.

These eight patterns are elaborated in operational detail under *Namespace Attack Taxonomy*, which adds resolver capture, ownership laundering, registry amnesia, and namespace flooding as concrete attack surfaces with required responses. The distinction is one of altitude rather than kind: this section names the structural failures a namespace must be designed against; the taxonomy names how those failures are actually attempted.

Core Principle

Names in the Meta-Layer must be portable, resolvable, governable, and resistant to capture. A namespace that cannot preserve identity, meaning, and control across systems becomes another platform dependency.

DP5 treats names as more than labels. Names are anchors for participation, reference, ownership, navigation, reputation, and coordination.

A DP5-aligned namespace must therefore support:

- participant-owned identifiers
- community-owned identifiers
- artifact and object identifiers
- namespace portability across tools
- conflict resolution and reservation logic
- verifiable ownership and control

- interpretable resolution across systems
- resistance to squatting, spoofing, and seizure

Primary Mechanisms and Structural Conditions

DP5 is enacted through mechanism families that together turn a name from a label into an accountable, resolvable, governed object. Each family addresses a distinct question that a namespace must be able to answer, and a namespace that answers some but not others will exhibit the failure modes named above.

- **Namespace objects** define *what can be named*: meta-domains for spaces, personal identifiers for participants and personas, artifact identifiers for objects, name chains for structured composition, and well-known URI anchors for publisher-controlled trust data.
- **The namespace system layer** defines *what a name is*, across five coupled layers: syntax, resolution, control, meaning, and history. Most existing naming systems implement one or two of these. DP5 requires all five, because omitting any one produces a specific, predictable failure — ambiguity, invisible override, ghost control, semantic drift, or history erasure.
- **The resolution protocol** defines *how a name is interpreted*: a governed, inspectable process that normalizes, validates, queries multiple resolvers, returns explicit state, attaches provenance, applies local policy, and renders an interface signal.
- **The state machine** defines *what a name currently is*: available, claimed, provisional, verified, active, disputed, quarantined, transferred, retired, or forked, with receipts on transitions that affect ownership, resolution, or trust.
- **The attack taxonomy** defines *what a name must withstand*, pairing each known attack with a required structural response rather than a moderation promise.
- **Registry architecture and registration rules** define *how names come into being*: ingest, candidate classification, canonicality, snapshots with checksums, pluggable indexers, and deterministic validation of labels.
- **Tradeability with bounded trust** defines *how a name may change hands* without laundering its history, through transfer receipts, provenance preservation, and visible dispute state.

The structural condition uniting these is that **a name must never assert more trust than it can demonstrate**. Every mechanism in this draft exists to close a gap between what a name appears to mean and what can actually be verified about it. Where verification is unavailable, the mechanism's obligation is to signal uncertainty rather than to default to trust — which is why explicit state, resolver provenance, and visible degradation recur throughout.

Primary Namespace Objects

Meta-domains

Meta-domains address virtual spaces within the Metaweb, similar to how traditional domains address web spaces.

A meta-domain may refer to:

- a participant-controlled overlay space
- a community zone
- a smart-tag namespace
- an application surface
- a virtual or conceptual space
- a bridge between a traditional domain and Meta-Layer objects

Example forms include:

- `boeing.com.web4`
- `apple.com.web4`
- `example.com.meta`
- `<label>.example.com.meta`

Meta-domains can link seamlessly to the broader web while functioning within the Metaweb overlay framework.

Personal identifiers

Personal identifiers address participants and personas, similar to email addresses, handles, or DIDs, but portable across Meta-Layer contexts.

Example:

- `shiftshapr.web4`
- `@jaime`
- `@jaime/artifact99`

Personal identifiers may be connected to decentralized identifiers (DIDs), credentials, proof-of-humanity mechanisms, or zone-specific identity contexts.

Digital artifact identifiers

Digital artifacts may serve as identifiers, assets, or NFTs that can be bought, sold, transferred, authenticated, or referenced.

Artifacts may include:

- posts

- paths
- smart tags
- annotations
- media objects
- overlays
- lists
- navigation components
- assertions
- credentials

Name chains

Name chains provide structured, semantic identifiers such as:

- `@user/object`
- `@community/tag`
- `@publisher/claim`

Name chains function as URI-like trust anchors, combining identity verification, object authentication, and conflict resolution pathways.

Decentralized URIs and well-known paths

DP5 also recognizes publisher-controlled decentralized URI patterns, such as:

- `/.well-known/trust.txt`

These allow trusted data to be anchored under existing publisher-controlled domains without requiring centralized registries.

Namespace System Layer: Resolution, Ownership, and Trust Semantics

This section defines DP5 as a **runtime-resolvable, multi-layer namespace system** rather than a static registry.

A DP5-compliant namespace operates across coupled layers:

Naming Layer (Syntax)

Defines canonical forms, label rules, and human-readable structure.

Examples:

- `<label>.example.com.meta`
- `@user`

- `@user/object`

Requirement:

- deterministic parsing
- canonical normalization

Failure mode: syntactic ambiguity.

Resolution Layer (Mapping)

Maps names → entities (people, agents, artifacts, spaces).

Resolution MUST include:

- multi-source resolvers (not single authority)
- explicit resolver provenance
- deterministic fallback rules
- verifiable resolution outputs

Resolution states MUST be visible:

- verified
- provisional
- disputed
- forked
- quarantined

Failure mode: invisible resolution override.

Control Layer (Authority)

Defines who can act on a name.

Control may derive from:

- cryptographic keys (wallets)
- DIDs
- registry attestations
- community governance

Control MUST be:

- transferable
- revocable
- auditable

Failure mode: ghost control (no accountable owner).

Meaning Layer (Semantics)

Defines what the name *means* in context.

Includes:

- identity type (human, org, agent, artifact)
- trust signals
- governance policies
- reputation bindings

Meaning MUST:

- travel with the name
- degrade visibly across contexts

Failure mode: semantic drift.

Temporal Layer (History)

Names are not static. They evolve.

Systems MUST preserve:

- ownership history
- resolution changes
- dispute timelines
- transfers

Failure mode: history erasure.

These five layers together define a **full-stack namespace**. Most systems today only implement 1–2 layers. DP5 requires all five.

Name Resolution Protocol (Reference Model)

DP5 requires a shared resolution model so names can be interpreted consistently across tools, overlays, registries, and communities.

A reference resolution flow SHOULD include:

1. **Normalize** the name into canonical form.
2. **Validate** syntax, reserved status, and structural constraints.
3. **Query** one or more resolvers or registries.
4. **Return state**: verified, provisional, disputed, forked, quarantined, retired, or unresolved.
5. **Attach provenance**: resolver source, timestamp, signatures, registry snapshot, and confidence level.

6. **Apply local policy**: zone rules, trust profiles, community reservations, and agent constraints.
7. **Render interface signal**: show the participant what the name means, what is uncertain, and what actions are safe.

Resolution must therefore be understood as a governed process, not a hidden lookup.

A minimal response object SHOULD include:

```
{
  "name": "@jaime/artifact99",
  "canonical": "@jaime/artifact99",
  "entity_type": "artifact",
  "controller": "did:example:123",
  "state": "verified",
  "resolver": "registry.example.meta",
  "resolved_at": "2026-04-26T00:00:00Z",
  "provenance": {
    "snapshot": "graph-snapshot-v12.json",
    "checksum": "sha256:...",
    "signatures": ["..."]
  },
  "policy": {
    "transferable": true,
    "dispute_status": "none",
    "zone_constraints": []
  }
}
```

Failure mode: **opaque resolution**, where a name appears trustworthy but participants cannot inspect how that trust was produced.

Namespace State Machine

DP5-aligned systems SHOULD expose name lifecycle states explicitly.

A name may move through the following states:

```
available → claimed → provisional → verified → active
                     ↘ disputed → resolved
                     ↘ quarantined → restored / retired
active → transferred → active
```

```
active → expired / abandoned → available or archived
active → forked → parallel-governed state
```

Core states:

- **available**: no active claim exists
- **claimed**: a participant or community has initiated registration
- **provisional**: claim exists but is pending verification or policy review
- **verified**: claim has met required proof conditions
- **active**: name is resolvable and usable
- **disputed**: competing claims or safety concerns exist
- **quarantined**: name is constrained due to suspected abuse, spoofing, or policy conflict
- **transferred**: control has changed with receipt
- **retired**: name is no longer active but history is preserved
- **forked**: multiple governance contexts recognize different meanings or controllers

State transitions MUST produce receipts where they affect ownership, resolution, trust, or participant expectations.

Failure mode: **state invisibility**, where participants cannot tell whether a name is safe, contested, provisional, or compromised.

Namespace Attack Taxonomy

DP5 treats names as attack surfaces. Naming systems concentrate trust, discovery, ownership, and memory, making them attractive targets for adversarial actors.

Common attacks include:

Spoofing

Attackers register visually or semantically similar names to impersonate trusted entities.

Examples:

- `appIe.meta` using confusing characters
- `paypaI.web4`
- near-match community names

Required response: similarity detection, warnings, and dispute pathways.

Squatting

Actors claim names for speculative rent extraction rather than use.

Required response: reservation rules, renewal logic, staking, decay, or community challenge mechanisms.

Resolver capture

A resolver becomes a de facto authority by controlling defaults.

Required response: multi-source resolution, resolver provenance, and fallback transparency.

Ownership laundering

A name is transferred repeatedly to obscure harmful history, evade sanctions, or reset trust.

Required response: transfer receipts and visible lineage.

Semantic hijacking

A name is used in a new context to imply trust or meaning it did not originally carry.

Required response: semantic profiles and degradation signaling.

Agent impersonation

Automated or AI actors adopt names that imply human identity, authority, or community status.

Required response: mandatory entity classification and controller binding.

Registry amnesia

A registry loses or suppresses prior state, disputes, or ownership transitions.

Required response: append-only logs, snapshots, checksums, and independent archival.

Namespace flooding

Attackers create many names to overwhelm discovery, governance, or trust review.

Required response: rate limits, economic friction, proof thresholds, and anti-spam containment.

Reference Patterns and Compatibility

DP5 does not replace existing naming systems. It defines how meta-layer naming can interoperate with them.

DNS-style names

DNS provides global resolution and familiar domain semantics, but is vulnerable to centralized registrar control and does not natively express trust state, provenance, or community governance.

DP5 can use DNS-linked anchors while adding overlay-level trust semantics.

DID-style identifiers

DIDs support decentralized identity and controller binding, but may be difficult for ordinary participants to read or remember.

DP5 can bind human-readable names to DID-backed controllers.

ENS / SNS-style naming

Blockchain naming systems support ownership and transfer, but often emphasize asset ownership more than contextual governance, dispute visibility, or semantic profiles.

DP5 can learn from these systems while requiring visible state, provenance, and governance.

Ordinals / BRC333-style artifacts

Ordinal inscriptions and BRC333-shaped artifacts can anchor durable metadata and namespace records.

DP5 can use inscription ordering, registry snapshots, and graph facts to support canonicality and provenance.

Well-known URI anchors

Publisher-controlled paths such as `/.well-known/trust.txt` allow existing domain holders to publish trust-relevant metadata without centralized registry dependency.

DP5 can compose these with meta-domain records and overlay resolution.

The goal is not one namespace to rule them all. The goal is interoperable naming with explicit trust semantics.

Meta-Domain Registry Architecture

A Meta-Domain Registry may operate as a public ingest and registry service for meta-domains, SNS-shaped JSON, BRC333-related ordinals, and related naming artifacts.

A registry can include:

- block-tip polling
- indexer search through services such as UniSat or pluggable alternatives
- candidate classification
- structural graph facts
- quarantined trust edges
- versioned graph snapshot releases with checksums

Such a registry SHOULD distinguish between:

- structural facts that can be merged freely

- trust assertions that require quarantine or policy review
- local drafts that remain outside public canonical state

Candidate classes

Candidate logs may include:

- `.meta` tokens
- BRC333-shaped bodies
- SNS-alias JSON candidates
- tag registry records
- anchor records
- name-chain references

Registry outputs

A registry SHOULD provide:

- canonical records
- candidate logs
- graph snapshots
- checksums
- dispute or quarantine markers
- provenance for resolver decisions

Canonicality and ordering

Where ordinal inscriptions are used, canonicality SHOULD be determined by documented ordering rules, such as lowest qualifying inscription number where adopted, rather than arbitrary string ordering.

Pluggable indexers

Registries SHOULD support pluggable indexers and resolution sources so that namespace infrastructure does not depend on a single data provider.

Failure mode: **indexer dependency**, where resolver integrity depends on one commercial or centralized API.

Registration Rules and Validation

DP5 supports concrete registration rules for product-level namespaces.

Canonical form example

A v1 product canonical form may be:

`<label>.example.com.meta`

Where:

- `<label>` is the registrant-chosen segment
- `example.com.meta` is a configurable suffix or canonical parent

Label rules

A valid label SHOULD satisfy:

- lowercase ASCII letters, digits, and hyphen only (`a-z`, `0-9`, `-`)
- no underscores or Unicode in v1 unless explicitly expanded later
- must not start or end with `-`
- length between 1 and 63 characters
- must not be an integer-only label
- must not be reserved

Structural validation regex

Label-only validation:

```
^(?!-)([a-z0-9](?:[a-z0-9-]{0,61}[a-z0-9])?)$
```

Full-domain validation example:

```
^(?!-)([a-z0-9](?:[a-z0-9-]{0,61}[a-z0-9])?)\.example\.com\.meta$
```

Regex alone is insufficient. Reserved-name and integer checks must be separate.

Recommended validation order

1. Parse `label` as the substring before the configured suffix.
2. Reject if label fails length or charset validation.
3. Reject if label matches `/^\d+$/`.
4. Reject if `label.toLowerCase()` is in the reserved set.
5. Reject if the normalized domain is already in use.
6. Accept and register.

In-use rule

"In use" means that a record already exists with the same normalized domain string under the applicable active status rule.

Storage and comparison SHOULD normalize to lowercase.

Interoperability and Tradeability

Tradeable meta-assets allow participants to exchange digital spaces, objects, and names, fostering virtual commerce and community formation.

However, tradeability must remain bounded by trust, provenance, and governance.

DP5 requires:

- transfer receipts
- provenance preservation
- dispute visibility
- constraints on reserved or protected names
- compatibility with commerce and incentive systems (DP6, DP9)
- alignment with identity and accountability requirements (DP1)

A name may be tradable, but the trust attached to the name cannot be treated as a blank commodity. Reputation, history, and community meaning must remain visible across transfer.

Governance, Accountability, and Agency Surfaces

This section defines **interface-level namespace governance**, aligning with the Meta-Layer's overlay architecture.

Namespaces are not governed only in registries. They are governed at the **point of interaction** via overlays, filters, and community rules.

Participant-facing surfaces

Participants MUST be able to:

- claim names under transparent rules
- see resolution paths (who resolved this name?)
- inspect ownership and controller state
- view trust signals and classification (human, agent, org)
- transfer names with receipts
- initiate disputes

- view dispute status in real time

Failure mode: invisible governance.

Overlay-mediated governance

Meta-layer overlays SHOULD expose:

- name provenance tooltips
- impersonation warnings
- namespace conflicts
- transfer history
- community annotations

This turns naming into a **live civic surface**, not a backend database.

Failure mode: governance hidden behind APIs.

Community governance powers

Communities MUST be able to:

- reserve names
- define namespace policies
- classify entities
- enforce local naming norms
- quarantine suspicious identities
- fork namespace rules when needed

Failure mode: centralized naming authority.

Dispute visibility

All conflicts MUST be visible as states, not silent overrides:

- competing claims
- impersonation reports
- trademark conflicts
- governance disagreements

Failure mode: silent winner-take-all resolution.

Interface as enforcement boundary

DP5 aligns with the principle that governance must live at the interface layer, where users experience identity and trust.

Browser overlays act as **civic membranes** where naming rules become visible, contestable, and enforceable in real time.

Failure mode: governance only enforced off-screen.

Incentives and Power Analysis

Namespaces concentrate value at three points, and each attracts a distinct form of capture. Understanding where the money and leverage sit explains why DP5's structural requirements — multi-source resolution, visible state, transfer receipts, registry snapshots — are economic interventions rather than technical preferences.

Registration is where scarcity is manufactured. Names are a positional good: `@jaime` has value precisely because no one else can hold it. Any registry therefore issues a monopoly with every name, and holds the power to define what may be issued, reserved, priced, or refused. Squatting is the market's rational response to unpriced scarcity, and anti-squatting mechanisms — staking, renewal, decay, community challenge — are attempts to price holding without pricing out legitimate participants. The tension is unresolved and named as such under *Open Questions and Future Work*: mechanisms that deter hoarding also burden the communities and individuals least able to pay.

Resolution is where the deepest power sits, and it is the least visible. A resolver that becomes the default for most clients does not need formal authority over the namespace; it exercises effective authority by determining what most participants see. This is the same dynamic through which DNS registrars and app-store search became control points despite holding no formal ownership. DP5's requirement for multi-source resolution and exposed resolver provenance exists because *default position is the asset*, and because capture at this layer is invisible to the participant by construction. A single commercial indexer dependency reproduces platform risk under decentralized branding.

Transfer is where accountability leaks. A tradeable name accumulates trust and can then be sold, which means reputation built by one party can be exercised by another. Ownership laundering is not an exotic attack but the predictable arbitrage of this gap: buy the history, discard the obligation. Transfer receipts and visible lineage exist to make trust non-fungible even where the name is fungible.

Several further pressures recur:

- **Speculative capital outruns civic use.** Names with obvious community value are claimed before the community exists to claim them. Reservation rules are a redistribution of the option value from speculators to prospective communities.
- **Registry economics favor amnesia.** Storing disputes, quarantines, and full ownership lineage costs money and creates liability; discarding it is cheaper and reduces exposure. Append-only logs with independent archival exist because the incentive runs toward forgetting.

- **Verification becomes a rent.** Where one attestation path is the only credible route to a verified state, verification becomes a tollgate on legitimacy. Plural control derivation — keys, DIDs, registry attestations, community governance — is the counterweight.
- **Agent identity is economically attractive to obscure.** An automated actor that reads as human or as an established organization captures trust it did not earn. Mandatory entity classification is an incentive constraint on whoever deploys agents at scale.
- **Fragmentation serves incumbents.** Where namespaces do not interoperate, each becomes a captive market. Interoperability reduces the value of holding a namespace and is therefore under-supplied by the actors best positioned to provide it.

Example: A registry publishes open registration rules and charges nominal fees, while its own resolver ranks names it has commercial relationships with above equally valid alternatives. Nothing in the naming rules changed; the namespace was captured at the resolution layer.

Why this matters: Naming power is exercised through defaults, ordering, and memory far more often than through explicit denial. DP5 therefore requires that resolution be inspectable, state be explicit, and history be preserved — not because operators are presumed hostile, but because these are the surfaces where capture is otherwise undetectable.

Community Signals Informing DP5

Community submissions and aligned work point to several recurring signals:

- desire for publisher-controlled trust anchors, such as decentralized URIs under existing domains
- need for namespace rights and management for non-human entities and AI systems
- interest in hierarchical, semantically meaningful names independent of physical nodes
- support for trust-schema-driven semantics and context-aware permissions
- demand for name chains as resolvable trust anchors for people, objects, and assertions
- need for decentralized identifiers and tags for posts, paths, and navigation objects
- concern that community identifiers may be seized, censored, or captured by platforms

These signals indicate that DP5 is not only about naming people. It is about naming the structure of the Metaweb itself.

Foresight and Failure Design

Namespaces fail slowly and then all at once. For long periods a naming system appears healthy — names resolve, transfers settle, disputes are rare — while the conditions for its capture accumulate beneath the surface. By the time failure is visible, the namespace is usually load-

bearing for identity, commerce, and provenance across many systems, which makes correction expensive and migration nearly impossible.

Predictable degradation paths include:

- **Resolver consolidation.** Multi-source resolution is implemented, but clients converge on one resolver for latency, cost, or convenience. The plural architecture remains intact while the effective architecture becomes singular.
- **State flattening.** Provisional, disputed, and quarantined states are supported by the protocol but collapsed to a single indicator in interfaces, because nuance is hard to render. Participants lose the ability to distinguish a verified name from a contested one.
- **Snapshot gaps.** Registry snapshots are produced regularly until an outage, migration, or cost review interrupts the cadence. Nothing appears broken, but a window of history becomes unreconstructable — and windows are exactly where laundering hides.
- **Reserved-set erosion.** Reservations protecting communities, institutions, or safety-critical names are relaxed case by case under commercial pressure, until the reserved set no longer reflects the risks it was built for.
- **Similarity-detection decay.** Spoofing defenses are calibrated against the confusables and scripts known at build time, while attackers move to newly supported characters, scripts, or rendering contexts.
- **Semantic profile abandonment.** Meaning-layer data is populated at registration and then left to rot, so trust signals attached to a name increasingly describe a state that no longer exists.
- **Quarantine backlog.** Suspicious names are quarantined faster than review capacity can adjudicate them, and the backlog is eventually cleared in bulk to restore throughput.

These paths compound in a characteristic sequence. Resolver consolidation makes state flattening consequential, because a single resolver's rendering becomes the participant's only view. Snapshot gaps make ownership laundering undetectable. Undetectable laundering makes semantic profiles unreliable, which makes state signals meaningless, which returns participants to judging names by appearance — the precise condition DP5 exists to escape.

Two properties make namespace failures unusually severe:

Irreversibility of trust transfer. Once a name with accumulated trust has been used to deceive, the harm is distributed across everyone who relied on it, and cannot be recalled by revoking the name. Prevention dominates remedy.

Lock-in by dependency. Names become embedded in links, credentials, contracts, archives, and other systems' data. A namespace that must be abandoned takes that dependent structure with it. This is why forkability, snapshot integrity, and cross-namespace interoperability are safety features rather than conveniences.

DP5 therefore requires safeguards designed in advance:

- **Resolver diversity monitoring**, treating concentration of resolution traffic as a measurable risk indicator rather than a market outcome
- **State-rendering conformance**, so that a compliant interface cannot silently collapse distinct name states into undifferentiated trust
- **Snapshot continuity auditing**, with checksums verified against independent archives rather than self-reported
- **Adversarial similarity testing** on a recurring schedule as scripts, fonts, and rendering surfaces change
- **Quarantine capacity provisioning** proportional to registration volume, since review capacity is the binding constraint on abuse response (DP3, DP17)
- **Fork drills**, in which a community exercises the ability to fork namespace rules and carry its records, so that the exit option is known to work before it is needed
- **Public postmortems** for namespace incidents — a successful spoof, a laundered transfer, a resolution override — linked to the specific layer that failed

Failure is expected. Failure that becomes visible only after the namespace is load-bearing is not. A DP5-aligned namespace is one where the concentration of resolution power, the erosion of state visibility, and the loss of registry memory are all observable while they are still reversible.

Relationship to Other Desirable Properties

DP5 is foundational and interdependent.

- **DP1** depends on identifiers that bind identity and accountability without forcing platform control
- **DP2** depends on participant control over names, handles, personas, and namespace portability
- **DP4** depends on identifiers that preserve data provenance without forcing cross-context correlation
- **DP6** depends on tradeable meta-assets, domains, and artifacts that preserve ownership and settlement integrity
- **DP7** depends on canonical forms and resolution semantics that work across systems
- **DP9** depends on attribution and contribution identifiers that cannot be trivially spoofed or replayed
- **DP12–DP13** depend on agent and tool identifiers that can be constrained, audited, and governed
- **DP14–DP15** depend on provenance and transparency for names, artifacts, and registry changes

- **DP18–DP20** depend on community and ownership identifiers that persist across governance and migration

Two couplings deserve emphasis, because DP5 is where they become concrete rather than aspirational.

DP3 governs the namespace itself. Reserved sets, dispute procedures, quarantine thresholds, fork legitimacy, and canonicality rules are all governance artifacts, subject to DP3's requirements for tiered decisions, receipts, visible diffs, bounded emergency authority, and memory. A registry that changes its reserved set or ordering rules without a traceable, contestable process has made a governance decision outside governance.

DP23 determines who the namespace is for. Unicode support, multilingual naming, and script integrity are the difference between a namespace that serves a global population and one that serves the subset writing in Latin script. This sits in direct tension with spoofing resistance, since script diversity expands the confusable space — a tension DP5 names openly rather than resolving by exclusion.

A failure in DP5 propagates upward. If names cannot be trusted, ownership cannot be trusted, identity fragments, and interoperability becomes deception.

Non-Goals and Explicit Boundaries

DP5 does not:

- require one global namespace for all contexts
- replace DNS, DIDs, ENS, SNS, BRC333, ordinals, or publisher-controlled URIs
- guarantee that all valuable names will be available
- eliminate disputes, squatting, or fraud completely
- treat tradeability as superior to stewardship
- require real-name identity
- collapse human, AI, organizational, and artifact namespaces into one undifferentiated model

DP5 further does not:

- prescribe a pricing, staking, or renewal model; it requires that whichever model is adopted be visible and governed
- guarantee that a name means the same thing in every context; it requires that divergence be signaled rather than hidden
- treat blockchain anchoring as a requirement, only as one available mechanism for durable provenance

- adjudicate trademark or legal ownership claims; it makes such conflicts visible as states and leaves adjudication to competent authorities and community process
- promise that a name, once held, is held permanently; expiry, abandonment, retirement, and forking are legitimate states

Failure mode: **namespace maximalism**, where DP5 is invoked to justify a single canonical registry, on the grounds that coherence requires centralization. DP5 requires interoperable naming with explicit trust semantics, which is the opposite claim.

DP5 defines conditions under which naming remains interoperable, governable, and accountable across systems.

Minimum DP5 Alignment (Non-Normative)

This section states **testable compliance criteria** rather than descriptive aspirations.

A system claiming DP5 alignment MUST pass the following checks:

Deterministic naming

- Canonical forms are defined and machine-verifiable
- Validation produces identical results across implementations

Test: same input → same validity result everywhere

Multi-source resolution

- Names resolve via more than one possible source
- Resolver provenance is exposed

Test: user can inspect where resolution came from

Explicit state signaling

Every name MUST expose state:

- verified
- provisional
- disputed
- quarantined
- forked

Test: UI or API returns state metadata

Ownership auditability

- Current controller is identifiable

- Transfer history is preserved

Test: ownership lineage query returns full chain

Portability

- Names function across at least 2 independent systems

Test: same name resolves meaningfully in multiple contexts

Anti-spoofing safeguards

- System detects or flags similarity-based impersonation

Test: registering visually similar name triggers warning or constraint

Registry memory

- Historical snapshots exist with integrity proofs

Test: past state can be reconstructed with checksum validation

Dispute mechanism

- Users can initiate and observe disputes

Test: dispute creates visible state transition

Non-human classification

- Agents and automated systems are distinguishable from humans

Test: entity type is explicitly encoded and exposed

These criteria transform DP5 from a principle into a **verifiable standard**.

These conditions define the minimum viable namespace layer of the Meta-Layer.

Open Questions and Future Work

DP5 identifies key frontier problems:

Cross-namespace interoperability

How do independent naming systems interoperate without collapsing into monopoly or fragmentation?

Semantic portability

How can meaning travel with names across cultural, linguistic, and technical contexts?

Anti-squatting mechanisms

What mechanisms balance open access with protection against speculative enclosure?

AI-native identity

How should agent identities evolve as they gain autonomy, persistence, and economic activity?

Namespace governance forks

What legitimacy models determine when a community can fork naming rules?

Human-readable vs machine-secure naming

How do we balance usability with cryptographic robustness?

Unicode and global inclusion

How do we support multilingual naming without increasing spoofing risk?

Economic design

What pricing, staking, or decay mechanisms prevent hoarding while preserving ownership?

Resolution concentration

What measures of resolver diversity are meaningful, and what interventions are available when resolution traffic concentrates without any formal change in authority?

Namespace succession

What happens to names, artifacts, and communities when a registry, resolver, or community steward ceases to operate, and how is continuity established without a central custodian of last resort?

These questions define the path toward ML-RFC maturation.

Path Toward ML-RFC

Progression toward ML-RFC maturity should include:

- standardized canonical forms for meta-domains, personal identifiers, and artifact identifiers
- shared validation rules and reserved-name policies
- resolver and registry schemas
- dispute-state semantics
- transfer receipt formats

- registry snapshot standards with checksums
- conformance tests for resolution integrity, ownership binding, and namespace portability
- implementation evidence from registry prototypes and overlay applications

Stable portions may be promoted first, especially canonical form rules, validation logic, and registry state semantics.

Closing Orientation

DP5 is the claim that participants, communities, agents, and artifacts deserve names they can carry across the Meta-Layer.

Without sovereign names, identity is rented, ownership is fragile, and discovery remains platform-shaped.

With DP5, people can claim space, communities can preserve continuity, artifacts can be addressed, and the Metaweb can become navigable without surrendering naming power to a single platform.

DP5 turns names into civic infrastructure.

To claim your space in the Meta-Layer is not merely to register a label. It is to anchor presence, meaning, and accountability in a shared world.
