Network Working Group
Internet-Draft
Intended status: Informational
Expires: 17 July 2026

B.A. Fisher
DPA R&D

13 January 2026

Invariant-Closed System Design (ICSD)
draft-dpa-icsd-00

Abstract

Invariant-Closed System Design (ICSD) defines an architectural principle for building systems in which all representable states are valid and compliant by construction. Rather than relying on runtime detection, validation, or remediation of invalid or abusive behaviour, ICSD ensures that invalid states are structurally unrepresentable through constrained state models, invariant-bound transitions, and authority-anchored truth sources.

This document introduces the ICSD concept, its core properties, design trade-offs, and applicability to compliance-critical domains such as payroll, taxation, financial reporting, and regulated infrastructure. It is intended for discussion and experimentation in environments requiring high assurance of correctness, auditability, and resistance to abuse.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 17 July 2026.

Copyright Notice

Table of Contents

1. Scope and Status

This document is an Internet-Draft and represents work in progress. It is published to enable structured technical review, experimentation, and architectural discussion around Invariant-Closed System Design (ICSD).

The material is a research artefact. It does not claim technical completeness, production readiness, or endorsement by the IETF or any other standards body, and it is not presented as a standards-track specification.

ICSD is presented as an architectural principle rather than a protocol. It is applicable across many system types and does not mandate a particular programming language, database technology, or deployment model.

Ongoing research, implementation experience, and domain-specific profiles may be pursued separately.

2. Executive Summary

Many systems attempt to achieve correctness and compliance by accepting broad inputs, then using validation, monitoring, moderation, and repair mechanisms to detect and handle invalid behaviour after it occurs. This reactive approach increases operational complexity, creates ambiguous intermediate states, and expands the attack and abuse surface.

Invariant-Closed System Design (ICSD) proposes a different approach: design the system such that only valid states can exist. In an ICSD system, invalid and non-compliant states are not "handled" at runtime because they are not representable in the system’s state space.

ICSD shifts complexity from runtime policing towards upfront modelling. It improves auditability by ensuring every persisted state is valid, and it reduces abuse opportunities that rely on malformed, partial, or inconsistent internal state.

3. Terminology

Requirements Language: The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174] when, and only when, they appear in all capitals.

System State: The complete set of values that determines system behaviour at a given instant, including persisted data and any in-scope in-memory state.

Representable State: A state that the system can encode in its data model and reach through its defined transitions.

Invariant: A condition that MUST hold for all representable states of the system (e.g., legal constraints, accounting identities, authorisation constraints, or schema requirements).

Invariant-Closed: A property of a system where the set of representable states is closed under the invariants; i.e., every representable state satisfies every invariant.

Authority-Anchored Truth Source: An external or internal source of truth that is treated as authoritative for specific facts (e.g., statutory registries, regulated rulesets, signed identity assertions, or versioned policy artefacts).

Reactive Enforcement: Any mechanism that detects, mitigates, or remediates invalid behaviour after it has been encoded as a system state (e.g., validation errors on persistence, moderation queues, anomaly scoring, or repair jobs).

4. Introduction

Contemporary systems often permit a broad range of inputs and states, depending on reactive mechanisms - validation checks, monitoring, moderation, or repair - to handle invalid, non-compliant, or abusive behaviour after it occurs. This reactive approach increases operational complexity, maintenance burden, and exposure to errors or attacks that exploit inconsistent or intermediate states.

Invariant-Closed System Design (ICSD) offers an alternative paradigm: systems are structured so that only valid states can exist. Invalid or non-compliant states are not detected and corrected - they are prevented by design, as they have no representation in the system’s state space.

The core axiom of ICSD is: If a system state can exist, it is already valid. Correctness, regulatory compliance, and legitimacy are embedded directly into the type system, input constraints, transition rules, and data sources.

This document outlines the ICSD principle, contrasts it with traditional reactive designs, and explores its benefits in domains where trust assumptions are minimal and external authorities define ground truth. Examples draw from production experience in payroll and taxation systems, where invariants are non-negotiable.

ICSD is particularly relevant to high-assurance environments, including financial systems, identity management, regulatory reporting, and secure infrastructure protocols.

5. Core Principle

The foundational axiom of ICSD is: *If a system state can exist, it is already valid.*

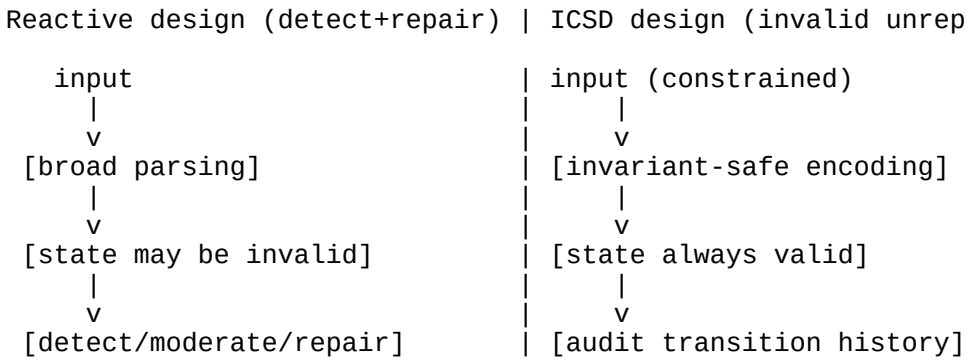

Figure 1: Reactive enforcement versus invariant-closed design

In a reactive design, the system must cope with invalid-but-encoded states and expend effort detecting, triaging, and correcting them. In an invariant-closed design, invalid states are unreachable: the system's representational model does not permit them, and transitions preserve invariants by construction.

ICSD does not imply that a system is small or trivial. Rather, it implies that as complexity increases, the modelling discipline increases as well: each additional state and transition is admitted only if it can be expressed in a way that preserves invariants.

6. Invariant-Bound State Space

An ICSD system defines a closed set of invariants: conditions that must always hold true. These invariants define the system's state space. The system MUST ensure that:

* all representable states satisfy all invariants;

* all state transitions preserve invariant validity; and

* no mechanism exists to bypass or temporarily suspend invariants.

In practice, this typically requires modelling state as a finite set of well-defined types (or records) whose construction is only possible through invariant-preserving transitions.

Examples of non-negotiable invariants in compliance-critical domains include:

* a payroll run cannot exist without a verified legal entity and an applicable tax context;

* a regulatory submission artefact cannot exist unless schema and policy validation have passed; and

* an authorisation grant cannot exist without a verified identity and an explicit scope of authority.

In ICSD, invariants are not optional checks; they define what the system can represent.

7. Authority-Anchored Truth

Where possible, ICSD systems anchor truth to authoritative sources rather than user assertion. This reduces trust assumptions and limits the risk of fabricated, inconsistent, or self-contradictory state.

Authority anchoring commonly appears as:

* company identity derived from statutory registries;

* tax calculations constrained by authoritative rulesets and versioned guidance;

* roles and authority derived from official records or cryptographically signed policy artefacts; and

* identity claims derived from verified credentials rather than mutable profile fields.

User input is treated primarily as selection, initiation, or proposal - not as the authoritative ground truth. Where user input is necessary (e.g., choosing a tax code), the representational model SHOULD restrict it to a set of invariant-safe options.

8. Non-Representable Invalid States

A defining property of ICSD is the elimination of representable invalid states. In an ICSD system, there are no:

* invalid-but-saved records;

* broken intermediate states;

* speculative or partially complete entities; or

* privileged objects created before authority has been verified.

If a configuration or object cannot be valid, it has no representation in storage or memory. This removes entire classes of bugs and abuse vectors associated with partial, inconsistent, or order-dependent state.

9. Elimination of Reactive Enforcement

Because invalid behaviour cannot be encoded, ICSD systems minimise or eliminate the need for reactive enforcement mechanisms such as:

* heuristic abuse detection and anomaly scoring;
* moderation queues and manual review pipelines;
* background repair jobs for data quality;
* error-driven state reconciliation; and
* ad hoc "fix-forward" scripts to correct invalid persistence.

Reactive enforcement is replaced by structural impossibility. The system does not monitor for invalid behaviour; it cannot represent it.

This does not remove all monitoring needs. Systems may still monitor for availability, performance, and suspicious but valid activities. ICSD specifically targets the class of failures where invalidity arises from representable internal state.

10. Auditability and Compliance

In ICSD systems, auditability emerges as a side effect of correct design:

* every persisted state is valid;
* audit logs describe state transitions, not error recovery;
* compliance evidence is implicit in system history; and
* reconciliation becomes verification rather than investigation.

A practical consequence is that auditors and operators can reason over the system using a smaller set of cases: "what valid state did we enter, when, and under what authority?" rather than "what went wrong and how did we patch it?"

ICSD is especially valuable where invariants are externally imposed (e.g., law and regulation) and where failure modes have significant cost (e.g., financial penalties, reputational harm, or operational shutdown).

11. Design Trade-offs

ICSD deliberately shifts complexity away from runtime policing and remediation towards upfront modelling and invariant definition. This increases initial design discipline but can reduce long-term maintenance cost, operational overhead, abuse surface area, and regulatory exposure.

Common trade-offs include:

* *Reduced flexibility:* inputs and workflows may be more constrained than in permissive systems.

* *Upfront modelling cost:* invariants must be identified, formalised, and tested early.

* *Migration complexity:* legacy datasets containing invalid state may require transformation.

* *Profile management:* different jurisdictions or policy regimes may require explicit profiles.

ICSD is not a statement that "validation is bad". It is a statement about where validation belongs: as part of constructing representable state, rather than as a recurring operational burden after invalidity has already been encoded.

12. Comparison with Traditional Designs

Traditional design	ICSD design
Accepts broad input	Accepts only invariant-safe input
Detects invalid behaviour	Cannot represent invalid behaviour
Repairs bad states	Prevents bad states
Relies on monitoring	Relies on structure and modelling
Compliance as process	Compliance as property
Security controls often	Many controls become inherent to

| layered on top | the state model |
+---+---+

Table 1: Traditional design compared with ICSD

ICSD does not remove the need for good engineering practices such as testing, change control, or operational security. It changes the baseline: correctness and compliance become properties of what the system can represent.

13. Applicability

ICSD is particularly well suited to domains where:

* correctness is mandatory and errors have high cost;
* external authorities define ground truth;
* trust cannot be assumed; and
* regulatory exposure is high.

Example domains include payroll and taxation, financial systems, identity and access control, regulatory reporting, and secure infrastructure and protocol design.

13.1. Applicability to Identity-Bound Security Systems

In identity-first architectures, a common failure mode is the existence of "privileged but invalid" objects, such as a grant that exists before identity is verified, or a session that exists before policy checks are applied. ICSD mitigates this by making such objects unrepresentable: the only representable grants and sessions are those that are already bound to verified identity and explicit authority.

This principle complements identity-bound frameworks and transports such as [UZPIF], [UZP], and [TLS-DPA], where correctness and authorisation are expected to be intrinsic properties of the connection and its negotiated state.

14. Security Considerations

ICSD can reduce attack surface by eliminating representable invalid states. Many classes of injection, abuse, and privilege escalation rely on malformed, partial, or inconsistent internal state; ICSD systems make such states unreachable by design.

However, ICSD is not a complete security model. A representable state being "valid" with respect to invariants does not guarantee that it is desirable. Attackers can still:

* perform harmful actions that are valid within policy (e.g., authorised but abusive use);

* target the integrity of truth sources (e.g., poisoning authority data);

* exploit incorrect invariant definitions (i.e., specification bugs); and

* attack availability (e.g., denial-of-service) regardless of state validity.

Designers should treat invariants and authority sources as security-critical artefacts: they SHOULD be versioned, reviewed, tested, and auditable. Where possible, authority inputs SHOULD be authenticated (e.g., signatures, provenance records) and changes SHOULD be subject to explicit change control.

15. Conclusion

Invariant-Closed System Design reframes correctness from a runtime concern into a structural property. By designing systems where invalid behaviour is not prevented but structurally impossible, ICSD can reduce operational complexity, improve auditability, and strengthen trustworthiness in compliance-critical environments.

ICSD is best viewed as a discipline: it rewards rigorous modelling and clear definitions of invariants and authority. Where those conditions hold, it can remove entire classes of failures that otherwise demand continuous monitoring and remediation.

16. One-Sentence Summary

Invariant-Closed System Design is the practice of building systems where invalid behaviour is not prevented - it is structurally impossible.

17. IANA Considerations

This document has no IANA actions.

18. References

18.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

18.2. Informative References

[TLS-DPA] Fisher, B. A., "TLS-DPA: An Identity-Bound Security Protocol for Traditional, Overlay, and Zero-Port Transports", Work in Progress, Internet-Draft, draft-dpa-tls-dpa, <https://datatracker.ietf.org/doc/html/draft-dpa-tls-dpa>.

[UZP] Fisher, B. A., "UZP: Universal Zero-Port Transport Protocol", Work in Progress, Internet-Draft, draft-dpa-uzp-transport, <https://datatracker.ietf.org/doc/html/draft-dpa-uzp-transport>.

[UZPIF] Fisher, B. A., "The Universal Zero-Port Interconnect Framework (UZPIF): An Identity-Centric Architecture for Post-Port Networking", Work in Progress, Internet-Draft, draft-dpa-uzpif-framework, <https://datatracker.ietf.org/doc/html/draft-dpa-uzpif-framework>.

Author’s Address

Benjamin Anthony Fisher
DPA R&D Ltd (https://www.dpa-cloud.co.uk)
Email: b.fisher@dpa-cloud.co.uk
URI: https://orcid.org/0009-0004-4412-2269