DLX Technical Result

Current development status, supporting evidence and validation boundaries

Executive summary. IndrasNet OÜ has developed and documented a bounded mechanism for checking declared conditions at an execution boundary. A controlled experiment produced recorded evidence of stale-state detection, a BLOCKED verdict and refusal of a protected mutation by an external adapter. This is an engineering R&D result, **not a commercially ready product**.

1. Main result: detecting stale state before execution

Consider an automated system preparing a consequential configuration change. The request satisfies the declared conditions at the initial check. Before the protected operation, however, the live state changes. This precise case was examined using a controlled SQLite reference environment.

STAGE	RECORDED BEHAVIOUR
S1 · ADMISSIBLE	The request was machine-admissible under S1. It carried state revision 7 and the digest of the initial state.
S1 to S2	A separate environment event changed the live state to revision 8.
Live reread	Before the protected operation, the adapter reread current state and detected a mismatch with the request-bound precondition.
BLOCKED	The corridor returned BLOCKED, with reason code STATE_PRECONDITION_MISMATCH.
No protected mutation	The external adapter refused the protected write. The recorded result reports zero protected mutations.

> **Causal boundary.** The earlier admissibility gate did **not** cause this refusal. The live reread detected the mismatch, the corridor returned BLOCKED, and the external adapter enforced non-execution. “Zero protected mutations” does not mean the database never changed: the environment event intentionally changed S1 to S2.

2. Technical evidence: two distinct lines

Static V7 fixtures. Predetermined PASS/BLOCK cases, reason-coded outcomes, source-bound digests, Rust-based verification and separate Python recomputation. These support reproducible outcomes for the specified **static fixtures**.

Controlled SQLite case. Recorded source code and evidence trace a state mismatch before protected execution, the BLOCKED result and the adapter refusal. This is **not** proof of a full cross-system deployment.

Archive inspection and review boundaries

An earlier static review recorded matching SHA-256 hashes for PROOF.zip and 15 selected source, protocol and oracle files. PROOF.zip is not attached to this PDF. Supporting evidence may be requested from IndrasNet, subject to access approval. No independent SQLite rerun was performed.

Separately, a read-only inspection of the archived SQLite database recorded an integrity check of "ok" and resource R at revision 8; execution, receipt and post_state had no rows. This inspection of saved database bytes was not an independent experiment replication.

indrasnet.ee/dlx | info@indrasnet.ee

Validation boundaries and next steps

What is established, what remains open, and the proposed partner engagement

3. What is established and what is not

CLAIM	EVIDENCE STATUS
CURRENT STATE drift in the controlled case	Supported by recorded experiment evidence.
BLOCKED verdict and external adapter refusal	Supported by the experiment artifacts and saved database inspection.
Reproducible V7 static outcomes	Supported by the supplied tests and evidence; not independently rerun for this document.
Validation of all six contract dimensions	Not demonstrated; this SQLite case exercises CURRENT STATE.
Operation across independent production systems	Not demonstrated.
Technical advantage over CAS	Not demonstrated.
Production deployment readiness	Not established.
Validated paying DLX customers	Not established.

4. Potential value to a customer

Unvalidated hypothesis. In a consequential workflow, different independently managed components may approve a change, supply live conditions, execute the operation and retain evidence. In some such workflows, there may be no shared, independently verifiable contract binding the authorised action to the conditions that still hold when execution occurs.

DLX proposes a way to formalise that verification and its evidence. If existing CAS, IAM, executor-side checks or trusted platform controls already satisfy the requirement, the assessment should say so. The SQLite case alone does not establish a need for DLX as a separate runtime layer.

5. What has been built — and the next step

Oleg Surkov leads DLX architecture and technical development at IndrasNet OÜ. The existing engineering base includes a contract model, verification mechanisms, test scenarios, an evidence format, source code, and documented results with supporting artifacts. This is an **engineering-research foundation**, not a finished industrial system.

Proposed next step. Evaluate one real, consequential cross-system workflow with an accountable owner; map current controls and trust boundaries; test representative failure scenarios; and determine whether an additional, evidence-backed verification gap actually remains.

> **For potential partners.** IndrasNet proposes a **bounded paid technical assessment** of one workflow, with defined scope and evidence-backed findings. This offer remains a hypothesis, **not** a signed engagement, validated price or finished security product. The objective is to establish additional value for a real organisation.

indrasnet.ee/dlx | info@indrasnet.ee