Intellex White Paper

A Memory-First Interoperability Protocol with Usage-Aligned Tokenomics

---

Executive overview

Enterprises don't fail for lack of more dashboards. They fail because they operate as **systems-of-systems**—many teams, tools, and partners running on different clocks and incentives—without a way to share just enough context, at the right time, **with evidence**. As autonomous agents spread across those seams, the missing primitive is *memory*: a sharable, permissioned, auditable working context that lets agents coordinate without copying raw data or arguing about "whose number."

Intellex is a memory-first interoperability protocol. It turns what people and enterprises *know* into **Memory Assets**—signed, permissioned artifacts (summaries, features, verified claims, translators, model deltas) that can be **created & proved, permitted & licensed, used, updated, moved, and revoked**. Each step emits an **audit-grade receipt** and, when material, **settles** in the Intellex token, **$ITLX**.

To align incentives end-to-end, $ITLX flows through **three pools** that mirror how Intellex is used in practice:

1. **Access Pool (AP):** rewards when enterprises *access* and settle against institutional memory (licensed reads, adjudications, revocations, cross-agent settlement).

2. **Contribution Rewards Pool (CRP):** rewards **individuals and panels** when their zero-party **claims** are actually used (personal/collective memory that powers PLMs and personalization).

3. **Interop Training & Translation Pool (ITP):** rewards **builders and validators** of translators, schemas, event/reputation/expertise detectors, and cross-agent messaging components (the machinery that solves interoperability).

The result is a protocol where value only moves when **memory moves**, quality is **stake-backed**, and **every decision has a receipt**.

1) Interoperability as a memory problem

1.1 What we mean by “memory”

Memory is the usable record of past perceptions, actions, and outcomes—structured so agents can retrieve it by cues, interpret it consistently, and apply it under policy. Memory is not a data dump. It’s *governed context* that travels with permission and provenance.

- **Generalized memory**: shared concepts, taxonomies, policies, elasticities—what turns raw events into meaning.
- **Institutional memory**: the enterprise’s lived history (receipts, constraints, commitments, reconciliations, post-mortems).
- **Personal memory**: a user’s preferences and history (often held by a personal assistant/PLM). Personal memories can compose into **collective memories**—group summaries that avoid exposing anyone’s raw details.

Interoperability fails when **many valid memories** coexist but cannot synchronize at the moment of action. True personalization and cross-company coordination require **aligning collective personal memory** and **institutional memory**—safely, with user control, and without copying raw data either way.

1.2 Why current approaches fall short

- **Copy-based integration (ETL/iPaaS)** → version skew, no portable rights or revocation.
- **Central warehouses + “AI on top”** → great for hindsight, poor for live coordination.
- **Event buses without memory** → messages move, institutional memory does not.
- **Naïve “all-on-chain”** → privacy leakage, cost/latency, and fragmented semantics.

Intellex answers with a **hybrid, memory-first** architecture: low-latency off-chain working memory; **on-chain receipts** for provenance and settlement; identities and shared semantics to keep meanings aligned; and tokenized incentives to reward what's useful and punish what's not.

---

2) Architecture overview

2.1 Primitives

- **Identity & Registry** — agent DIDs, capability vectors, attestations, and discovery.
- **Shared Semantics** — versioned schemas and translators to prevent semantic drift.
- **Working Memory & CEP** — a low-latency memory plane plus complex-event rules that trigger the right task at the right time.
- **Receipts & Settlement** — append-only receipts for every material action; settlement flows in $ITLX.
- **Governance** — parameters, slashing rules, emission weights, and fee routing.

2.2 Memory Asset lifecycle

1. **Create & Prove** — mint an asset (hash, schema, policy); optional bonded attestation.
2. **Permit & License** — grant scoped rights (who, what, where, purpose, duration).
3. **Use** — a licensed read that *changes a decision* emits a receipt; royalties settle.
4. **Update** — outcomes (e.g., deliveries, lift) improve assets; contributors earn splits.
5. **Move** — bridge rights/receipts across chains; keep provenance intact.

6. **Revoke** — stop future use; propagates instantly and is provably enforced.

2.3 Receipts (the evidence layer)

A **receipt** binds: (who) used (what asset version) under (policy) at (time) to change (decision), plus pointers to prior receipts and proofs. Receipts power audit, payouts, reputation updates, and disputes.

---

3) The role of the token

3.1 Why a token is necessary

- **Neutral settlement** across many actors and chains, without a central intermediary.
- **Economic security**: make spam/abuse expensive; make truth profitable and falsity costly.
- **Portable incentives**: pay contributors and curators across org boundaries on every qualified use.
- **Governance & sustainability**: finance public-good components that everyone depends on
- **Provenance & trust**: clear immutable record of memory, its use and the reputation and expertise of the agents that use it.

3.2 Security properties provided by $ITLX

- **Sybil resistance**: writes that change shared state cost money; attacking the protocol is not free.
- **Bonded claims**: attesters stake to vouch for quality (accuracy, latency, conformance); failures are slashed.
- **DoS containment & QoS**: stake-weighted rate limits prioritize valuable work under load.

- **Economic finality**: once the challenge window closes, receipts are economically settled.

3.3 Identity, reputation, expertise, experience

- **Identity with weight**: an address becomes an identity when it carries stake, history, and active licenses.
- **Reputation that prices risk**: updates only on receipt-backed events; good behavior lowers costs and unlocks throughput.
- **Expertise as an asset**: useful translators, detectors, and models *earn per use*; shallow copies starve.

---

4) Tokenomics with three pools

4.1 Total supply

1,000,000,000 $ITLX (fixed at genesis).

4.2 Allocation (high-level)

- **Access Pool (AP)** — 9% reserved for co-emissions against enterprise access receipts.
- **Contribution Rewards Pool (CRP)** — 3% reserved for zero-party claim usage.
- **Interop Training & Translation Pool (ITP)** — 3% reserved for translators/detectors per invocation and bounties.
- **Community, Ecosystem/Grants, Treasury, Team, Investors, Public Sale, Strategic Rebates** — remaining 85% across long-term buckets (vesting, milestones, governance-gated disbursements).

(Percentages and cliffs are shown here to illustrate structure.)

4.3 Fee routing (enterprise pays → who gets paid)

Every enterprise event (licensed read, adjudication, revocation, settlement) pays a **usage fee**. By default:

- **60% → participants via AP** (royalties to asset owners; attestation fees to bonded attesters; solver fees).
- **10% → ITP** (when interop components were invoked).
- **10% → CRP** (when zero-party claims contributed).
- **15% → Treasury** (ops, risk, insurance).
- **5% → Burn** (issuance discipline).

4.4 Emission mechanics (monthly epochs)

Each pool has a monthly budget—**AP□, CRP□, ITP□**. Emissions are **receipts-gated** (only released for proven activity), **challengeable** (with slashing/clawback), and **decaying** (unused budget partially expires).

General formula for pool **X ∈ {AP, CRP, ITP}**:

Rewardi,X,t = Scorei,X,t∑jScorej,X,t · min(Xt, αX·FeesX,t)\text{Reward}_{i,X,t} \;=\; \frac{\text{Score}_{i,X,t}}{\sum_j \text{Score}_{j,X,t}} \cdot \min\Big(\text{X}_t\; \alpha_X \cdot \text{Fees}_{X,t}\Big)

- Score = net verified contribution (e.g., qualified reads, invocations, benchmark points).
- X_t = monthly emission cap for the pool.
- Fees_{X,t} = pool-relevant fee inflows this month.
- α_X = optional matching coefficient (co-emission that scales with fees).

Decay rule:

Xt+1 = δX · max(0, Xt−EX,t) + NewCapX,t+1X_{t+1} \;=\; \delta_X \cdot \max(0\; X_t - E_{X,t}) \;+\; \text{NewCap}_{X,t+1}

- E_{X,t} = emissions actually paid from pool X in month t.
- δ_X ∈ [0,1] = rollover factor (e.g., 0.5 → half of the unused budget rolls, half expires).

Challenge window: receipts are *pending* for τ\tau days. Successful challenges claw back emissions and **slash** misbehaving stake.

---

5) How the pools operate in practice

5.1 Access Pool (AP)

What it rewards: when enterprises use institutional memory—licensed reads that change a decision, dispute adjudications, settlement completions, revocation propagations.

Who gets paid: Memory Asset owners (royalties), bonded attesters (attestation fees), solvers/relayers for cross-chain steps, and the protocol (treasury/burn).

Example (B2B promo):
A retailer licenses the supplier's *ATPWindow* and *QualityLots* assets to plan a 4-week campaign. Each time those assets *actually steer* a placement or replenishment decision, a **qualified receipt** is emitted and AP rewards flow: royalties to the supplier, an attestation fee to the validator who staked on *QualityLots* fidelity, and minor solver fees for cross-chain settlement.

5.2 Contribution Rewards Pool (CRP)

What it rewards: qualified **uses of zero-party claims** (student eligibility, eco packaging preference, allergy constraint, delivery window).

Who gets paid: the individual (or their assistant) who owns the claim; any attester/validator who bonded quality; the protocol (treasury/burn).

Example (B2C personalization):
A consumer licenses "eligible for student pricing until Sep 30" to a brand *only* for checkout. Each successful discount application logs a receipt and pays a micro-royalty via CRP. Revocation stops future use; misuse after revocation is penalized automatically.

5.3 Interop Training & Translation Pool (ITP)

What it rewards: per-invocation use of translators/ontology mappers, event/reputation/expertise detectors, and cross-agent messaging components; plus **bounties** for gaps and **milestones** on fidelity/latency.

Who gets paid: builders and validators who keep interop components accurate and fast under stake.

Example (schema translator):
A third-party publishes an ERP-A↔ERP-B attribute mapper, stakes on ≥98.5% fidelity, and charges pennies per invocation. Every production call routes an ITP reward; failing audits slash stake and claw back prior emissions.

---

6) End-to-end flows (with receipts and settlement)

6.1 Supplier–retailer promotion (B2B)

1. **Publish**: Retailer posts *PromoLift*, *PlanogramReset*, *OTIFPolicy*; Supplier posts *ATPWindow*, *DCCapacity*, *QualityLots*.

2. **Attest**: A bonded validator stakes on *QualityLots* fidelity (window + SLO).

3. **License & Use**: Retailer's media and supply agents license assets and take decisions; each qualifying use emits receipts.

4. **Settle**: AP fees flow; AP emissions co-match; attesters earn; treasury accrues; a small burn occurs.

5. **Update**: Scan data and shipment events update assets; contributors earn update splits.

6. **Revoke**: If a lot fails release, supplier revokes that slice; further use is blocked and provably penalized.

6.2 PLM-powered checkout (B2C)

1. **Claim**: Consumer's assistant creates "student eligibility" with time-bound proof.
2. **License**: Brand requests checkout-only use; consumer approves.
3. **Use**: Discount applied → receipt logged → CRP micro-royalty.
4. **Revoke**: Permission ends early; any subsequent attempt fails policy checks and is penalizable.

6.3 Translator in the loop (interop)

1. **Deploy**: Builder publishes ERP translator; stakes for fidelity/latency.
2. **Invoke**: Each cross-system read uses translator → receipt emitted.
3. **Reward**: ITP pays per invocation; poor audits trigger slashing/clawback.

---

7) Risk model and adversarial resilience

- **Sybil & spam**: writes cost; identities that matter must carry stake and history.
- **Collusion**: random selection of verifiers; cross-checks; escalating bonds for high-impact claims.
- **Replay/reorgs**: economic finality after challenge window; cross-chain steps insured by bonded solvers.
- **Data leakage**: the protocol moves **rights and receipts**, not raw payloads; ZK/TEE attestations can minimize disclosure.
- **Drift & decay**: time-boxed attestations; rolling audits; automatic decay of emissions budgets to keep incentives current.

---

8) Governance knobs

- **Fee weights** (AP/CRP/ITP/treasury/burn).
- **Pool budgets & matching coefficients** (AP□, CRP□, ITP□ and α_X).
- **Challenge window τ and slashing ratios** by claim class.
- **Rollover factor δ** (decay).
- **Attestation classes** (accuracy thresholds, latency SLOs).
- **Public-good funding** for schemas, benchmarks, reference connectors.

Governance cannot mint around receipts; all emissions are **receipts-first**.

---

9) Unit economics (illustrative)

Assume an enterprise runs 20M qualified reads/month across planning, fulfillment, and customer touchpoints.

- **Fee**: $0.001/read average → **$20,000** AP fees.
    - $12,000 (60%) to participants (royalties, attesters, solvers)
    - $2,000 (10%) to ITP components used
    - $2,000 (10%) to CRP when claims contributed
    - $3,000 (15%) treasury
    - $1,000 (5%) burn
- **Emissions**: AP□ co-matches up to 20% of AP fees (if available), adding **$4,000 worth** of $ITLX to participants.
- **Outcome**: predictable opex for enterprises; yield for contributors scales with *actual usage*.

10) Why this will drive real-world volume

- **For enterprises**: fewer expedites/disputes; faster, evidence-first decisions; pay only when memory is used.
- **For contributors**: ongoing earnings for useful memory; bonded markets where being right pays and being wrong costs.
- **For builders**: sustainable public-good funding for the interop machinery everyone needs.
- **For the ecosystem**: every qualified decision emits receipts and settlements—steady on-chain activity linked to real business value.

---

11) Frequently asked questions

Q: Why not just keep paying vendors to build ETL?
A: ETL copies data but not *rights, provenance, or revocation*. Intellex pays for **governed context** that changes a decision—and it leaves a receipt.

Q: Why do individuals get paid?
A: Because zero-party claims (preferences, eligibility, constraints) are often the *missing* context in personalization and fulfillment. Paying per *qualified use* rewards value, not data hoarding.

Q: What if claims are wrong or drift?
A: Attesters stake; challenge windows allow clawback and slashing. Revocation stops future use immediately.

Q: Isn't a token overkill?
A: Not when you need neutral settlement, portable incentives, stake-backed quality, and cross-org payments that follow receipts. A token is the minimal instrument that checks all of those boxes.

---

12) Conclusion

Interoperability isn't a feature—it's a **memory** discipline. Intellex makes memory ownable, permissionable, revocable, and **payable per use**. The $ITLX token ties that memory to identity, reputation, and settlement so agents can coordinate inside firms, across partners, and with consumers—**with receipts**. Solve memory interoperability and you get what enterprises and users both want: decisions that are correct for *this* context, *now*, under *clear rules*, and a business model where everyone who made that possible gets paid.