The Intelligence Processing Unit: Architectural Explainability as a Foundation for Trust in AI Systems

Justin Holbrook
Principal Researcher
anywhereintelligence.ai

Swaroop Kallakuri
Principal Researcher
anywhereintelligence.ai

Joshua Holbrook
Principal Researcher
anywhereintelligence.ai

Lucas Burgess
Independent Researcher

Abstract

Trust between humans rests on a simple mechanism: the ability to ask "why did you do that?" and receive an answer that is coherent, inspectable, and open to challenge. AI systems require a structurally equivalent mechanism to make the system's deliberative processes attributable, their actions normatively authorized, and their effects revocable. Current AI systems built on large language models lack this architecture. Instead they act first, only later permitting post-process scrutability.

We introduce the Intelligence Processing Unit (IPU) as an architectural primitive providing both epistemic backbone and normative gating for AI system processes. We define the IPU as the atomic unit of process execution: a bounded, append-only, immutable record for subsystem activity from query to response with real-time traces for the subsystems consulted, the knowledge accessed, the scope boundaries enforced, the routing decisions made, and the latencies incurred. Unlike traditional explanation methodologies, the IPU acts as both execution scribe and authorization gate for in-progress, durable system activity. Memory creation, knowledge modification, persistent action, and other trust-dependent subroutines all turn on verified, unfailed IPU traces. The result is a system that can demonstrate, at the moment of commitment, its actions are attributable and authorized. We present the IPU architected in AIOS (Anywhere Intelligence Operating System) as an initial implementation of the IPU primitive.

We situate our work within the broader context of cross-disciplinary trust research, arguing that process-based explainability (why did the system take this path?) is more fundamental than prediction-based explainability (why did the model predict this token?). We conclude by arguing the former is not merely a better explanation mechanism; it is an essential component for achieving grounded trust and normative coherence in human-centric AI systems.

1. Introduction

1.1 The Trust Problem

No technology achieves sustained adoption without trust [24,5]. Trust in technology, however, is not simply a user decision. It is a response to the structure of the technology itself: the ability to attribute both authorized and unauthorized system actions to known, inspectable mechanisms. A toaster's behavior is authorized by its design; a calculator's outputs are authorized by arithmetic; a car's behavior is authorized by physics and engineering. In each case, failure is diagnosable, deviation

is detectable, and errors carry known signatures. Trust follows from this attributability as an earned consequence of purposeful architecture.

AI systems built on large language models have no inherent mechanism by which to earn this trust [16]. They hallucinate (confidently asserting falsehoods with no external signal that something has gone wrong) [19,20]; are inscrutable (unable to explain why they produced a particular output or what information they relied upon) [25]; and are unpredictable (the same question may produce different answers at different times, with no mechanism for understanding what changed) [27,30].

These are not engineering defects that can be patched. They are consequences of an architectural choice to build AI systems that encode knowledge parametrically, produce outputs stochastically, and offer no native mechanism for system examination, reporting, or justification of its own deliberative process. Trust as a system artifact remains elusive [4].

1.2 The Interpersonal Trust Analogy

In human relationships, trust is not built on perfect decision-making [28]. People make mistakes, forget things, and sometimes get things wrong. What sustains trust is the capacity for **reflective accountability** – the ability to answer, when challenged:

- *"Why do you believe that?"*
- *"What makes you say that?"*
- *"How did you reach that conclusion?"*
- *"What information are you relying on?"*
- *"Where might you be wrong?"*

A person who can genuinely engage with these questions by reflecting on their own reasoning process, acknowledging uncertainty, identifying the sources of their beliefs, and accepting correction merits one level of trust. A person who insists on conclusions without being able to explain them, who becomes defensive when questioned, or who has no awareness of their own epistemic limitations merits a lesser level of trust [11,28].

Researchers in psychology often refer to such reflective accountability as metacognition: the ability to think about one's own thinking [11]. In organizational trust theory, Mayer, Davis, and Schoorman [28] identify three factors of trustworthiness: ability (the trustee's competence in the relevant domain); benevolence (genuine concern for the trustor's interests); and integrity (adherence to principles the trustor finds acceptable). Lee and See [24] extend this to automation, proposing that trust is calibrated through performance (what the system does), process (how it operates), and purpose (why it was designed that way). Taken together, these approaches outline a three-dimensional framework for assessing whether reliance on a system is warranted: competence must be inspectable, operation must be transparent, and intent must be verifiable. Trust is the consequence of meeting these criteria.

AI systems fall short across all three dimensions: (1) their competence boundaries are unknown [46]; (2) their processes are opaque [25]; and (3) their alignment is asserted by designers rather than users [13,36]. The result is decreased capacity to safely align system capabilities and responsibilities and earn user trust in system operations and outputs [15,45].

1.3 Trust in Technology

Trust requirements for technology differ based on what is at stake and the cost of failure [39,1]. Consider, for example, systems with alternative deterministic and stochastic profiles.

Table 1: Technology trust profiles: determinism, trust basis, and failure mode.

Technology	**Trust basis**	**Failure mode**
Toaster	Deterministic: setting 3 always produces the same toast	Known and repeatable – user learns failure boundaries

Technology	Trust basis	Failure mode
Calculator	Deterministic: same input always produces same output	Essentially zero failure within domain
Car	Mostly deterministic with known failure boundaries	Mechanical failures are diagnosable; driver maintains agency
Search engine	Semi-deterministic: results vary but ranking logic is inspectable	Relevance failures are visible; user selects from options
AI assistant	Stochastic: same question may produce different answers	Failures are invisible – the system cannot tell you when or why it is wrong

The critical observation is this: as we move from deterministic to stochastic technologies, the trust requirement shifts from repeatability to explainability [15,6]. A toaster doesn't need to explain itself because its behavior is perfectly predictable. An AI assistant must explain itself precisely because its behavior is not. Current AI systems sit at the most demanding end of this spectrum (stochastic behavior, autonomous or semi-autonomous decision-making, no native explainability) but provide trust mechanisms better suited for the least demanding end.

1.4 Contributions

We formally introduce the Intelligence Processing Unit (IPU), an architectural primitive that addresses the trust deficit in traditional AI systems. The IPU is:

1. **The atomic unit of execution.** Every user-issued action in an AI system produces exactly one IPU, a bounded, accountable execution pathway across subsystems from query to response.
2. **An immutable accountability record.** The IPU trace is append-only and immutable, recording which subsystems were consulted, what knowledge was accessed, which scope boundaries were enforced, what routing decisions were made, and the latency each step incurred.
3. **A commit gate.** No durable system effect (memory creation, knowledge modification, reminder scheduling) can occur without a complete, unfailed IPU trace; if the trace is incomplete, the system cannot act.
4. **The architectural answer to "why?"** When a user asks "why did you say that?", the IPU trace provides a structured, inspectable answer of how the system produced its response, enabling revision and correction.

1.5 Paper Organization

This paper proceeds as follows. Section 2 establishes the theoretical and psychological grounds for the claim that process-based explainability is a structural requirement for trust in AI systems, not a design preference. Section 3 defines the Intelligence Processing Unit: its properties, lifecycle phases, commit gate, provenance model, and two-coordinate identity structure. Section 4 distinguishes the IPU from existing XAI methods, observability tools, audit logs, and MLOps frameworks. Section 5 describes the AIOS reference implementation: governing design principles, trace accessibility, GSIC routing integration, the IPU emitter, and the IPU's role in the personal knowledge infrastructure. Section 6 addresses anticipated objections and proposes an investigative agenda. Section 7 concludes.

2. Why Trust Requires Explainability

2.1 Prediction-Based vs. Process-Based Explainability

The dominant approach to explainable AI (XAI) focuses on understanding individual model predictions [34]. Feature perturbation methods identify which inputs influenced a particular output [34]; game-theoretic attribution assigns importance scores to input features [26]; attention visualization highlights which tokens the model weighted during generation [18]; MLOps pipelines address data provenance, model drift, and aggregate bias [22].

These methods constitute genuine contributions to AI system deployment and operations. By design, however, they answer an engineering question: "Why did the model output this particular token?" The larger epistemic question remains unanswered: "Why did the system give me this answer?"

The gap is architectural at two levels. At the token level, LLM inference is stochastic: each token is sampled from a probability distribution within a prediction-based process that is neither reproducible nor fully recoverable [18,26]. But a deeper problem exists. In most modern AI systems, inference is the product of a multi-stage pipeline: intent classification, tool routing, knowledge retrieval, source ranking, context assembly, LLM generation, and output formatting. Scrutinizing the LLM's attention weights provides little insight into why, for example, the system searched Wikipedia instead of MedlinePlus, chose a deterministic rather than stochastic route, or decided a user query required clarification instead of response.

The gap between token-level and pipeline-level opacity is the gap between **prediction-based explainability** (why was this token predicted?) and **process-based explainability** (why was this path chosen?). Similar to how humans prefer particularized explanations (why P over Q?) to probability distributions (what do most people think) [29], trust in AI systems is primarily a particularized, process-based problem requiring an epistemic response to trust-related user questions: where did this information come from, what was considered, and what was the system's reasoning at each decision point.

2.2 The Black Box Is More Than the Model

When AI researchers discuss the "black box problem," they typically mean the opacity of neural networks. But from a user's perspective, the black box is the entire system [9]. Consider a user who asks a health question and receives a confident answer. The relevant trust questions are:

- Was a medical knowledge source consulted, or did the answer come from parametric memory?
- Were multiple sources considered, or just one?
- Did the system recognize this as a health-critical query requiring high-reliability sources?
- Was any information filtered, reranked, or excluded?
- Is the system confident in this answer, or should it have said "I don't know"?

No existing XAI method answers these questions, largely because they operate at the model level rather than across the pipeline [25,32,9]. Yet cross-disciplinary research in trust relationships reflects an emphasis on answers to questions such as these – answers that include an ability to acknowledge ignorance [21,19,47], a capacity to explain reasoning [12,3,41], and an awareness of predictable versus unpredictable failures [8,24,33,14]. The IPU addresses this by instrumenting the pipeline itself.

2.3 Risk-Proportional Trust

Any serious discussion of trust in AI systems is, at its core, a discussion about risk and risk management [35,39,46]. Trust is not demanded uniformly across all interactions with a system; it is demanded in proportion to what the user stands to lose when the system is wrong. Put another way, trustworthiness scales with the consequences of a system's failure modes, not with the frequency of its successes.

The risk surface varies dramatically by application. As a result, responsible deployment requires aligning system capability with system responsibility [16]. A system assigned a responsibility that exceeds its verified capability creates a trust deficit that performance optimization cannot close [15,45]. At the low-stakes end (creative writing, exploratory ideation, open-ended generation) opacity may be tolerable because few, if any, commitments turn on output. At the high-stakes end (clinical decisions, financial transactions, legal documentation) opacity creates significant danger [6] because consequences are large and often irreversible.

We term this dynamic the capability-responsibility alignment principle: the requirement that a system's scoped capabilities, including its accountability mechanisms, align with its tasked responsibilities. As applied, the principle generates a structural design requirement for AI systems: as risk increases, the need for inspectable AI increases non-linearly [39]. High-risk applications require a

system that can demonstrate, at the moment of commitment, that its actions were authorized, sourced, and bounded. The IPU satisfies this requirement by providing granular, process-based visibility, allowing users to calibrate trust within a paired capability and responsibility set [24,33].

In this sense, the IPU is responsive to increasing regulatory demands for AI scrutability and accountability. The EU AI Act [10] stratifies systems into risk tiers, imposing progressively stricter obligations for traceability, human oversight, and documentation as risk rises. NIST's AI Risk Management Framework [31] maps governance and measurement requirements to the severity and likelihood of potential harms. The higher the stakes, the higher the bar for explainability. The IPU is designed to meet this bar by default – not through compliance instrumentation added after deployment, but through architectural constraints that make every system action attributable and every durable effect traceable to an authorized execution pathway.

3. The Intelligence Processing Unit

3.1 Definition

An Intelligence Processing Unit (IPU) is the complete, accountable execution pathway for one explicit user-issued action command – the architectural mechanism by which that action is attributable, its authorizing conditions verifiable, and its effects revocable.

As an architectural primitive, an IPU has six distinct characteristics. (1) An IPU is bounded – one user-issued intent, one execution pathway, one trace. No execution crosses the provenance boundary of another. (2) It is append-only. Events are written sequentially and cannot be modified after emission. (3) It is immutable. Once a trace is committed, no record within it can be altered. (4) It is inspectable. Every event is accessible to the user and to system audit tools through the standard trace API. (5) It is attributable. Every event records which subsystem produced it, what scope it operated in, and what authority it acted under. (6) It is referentially stable. Every durable artifact the system creates carries a pointer back to the IPU that authorized its creation.

An IPU is not a log entry, a telemetry metric, or a debugging trace. It is the epistemic backbone of the system's trust and accountability model. If an action cannot be explained via its trace, it must not be allowed to produce durable effects.

3.2 Lifecycle Phases

Each IPU progresses through an ordered sequence of phases serving both scribe and authorization functions. A system-agnostic reference architecture for a typical AI system might include:

Table 2: IPU lifecycle phases, operational purposes, and trust properties.

Phase	Purpose	Trust property
Ingress	Query received at API boundary	Timestamp, provenance, user identity
Declaration	Command recognized as actionable	Intent classification is recorded
Legality gate	Scope, authorization, policy validation	Authorization is a planning gate, not a post-hoc note
Retrieval	Knowledge/corpora sources consulted	Which sources were accessed, which were blocked
Reasoning	LLM inference or deterministic computation	Model used, tokens generated, time elapsed
Verification	Consistency and confidence checks	What was checked, what passed, what failed
Assembly	Response composition	Sources cited, context window used
Commit gate	Final validation before durable effects	The trace must be complete and unfailed
Completion	Response delivered to user	Terminal status, full latency, trace ID returned

Of the phases above, the commit gate bears particular mention as a core trust enforcement mechanism. Before any durable system effect the IPU projection is evaluated as $\left[\text{committable}(\tau) \equiv \text{initiated}(\tau) \wedge \text{terminated}(\tau) \wedge \neg\text{failed}(\tau) \wedge \neg\text{refused}(\tau)\right]$, where τ denotes a trace. If any condition is unmet, the transaction is rolled back. No partial persistence. No silent side effects. No "best effort" completion. Reflective accountability requires that an agent account for its actions before they become irreversible, not merely after. The commit gate enforces this as a structural precondition.

3.3 Provenance and Attribution

Every event in the IPU trace carries provenance metadata recording three aspects of source attribution: the source kind (whether information came from user input, a personal knowledge object, a corpora document, or system configuration), source identifiers (specific document, entity, or record identifiers for each item consulted), and citation references (user-visible provenance that can be surfaced in the response). Together these permit information chain inspectability at the level of individual facts rather than aggregate summaries.

When a user asks "where did that information come from?", the IPU trace provides a structured answer: this fact came from this source, retrieved at this time, ranked by this method, with this confidence score. This is not a citation generated by the LLM. It is the auditable execution record of which sources were consulted and how they were ranked.

3.4 Scope Enforcement

A critical trust property that existing XAI methods do not address is scope enforcement. In a multi-user system with private and public knowledge spaces, the user must be able to verify that their private data was not exposed to other users, that public data was not silently promoted to private knowledge, and that the AI system ranked and relied on the two based on the user's epistemic and normative preferences.

The IPU trace records, for every execution, which knowledge scopes were requested, which were accessed, which were blocked by policy, and whether scope filtering was applied. This makes scope enforcement inspectable. Access control implementation becomes a first-class citizen of the system's audit framework, allowing users to verify whether and how private data was accessed and employed.

3.5 The Event/Projection Architecture

The IPU trace is implemented as two complementary data structures corresponding to the write and read sides of event sourcing architectures. The **event stream** is the immutable, append-only ledger: every subsystem emission produces exactly one record (written once, never modified) carrying the full semantic payload of that activity, its phase membership, and a chained integrity hash linking it to the preceding event. The **projection** is a materialized current-state summary derived from the event stream and updated on every new emission. It synthesizes the trace's current authorization state (whether all phases have completed, whether any failure or refusal has occurred, whether the action is committable, and a per-phase status map constituting the trace's trust fingerprint) in a form queryable in constant time without scanning the full event stream.

The two structures serve distinct purposes and audiences. The event stream serves audit and inspection: every individual decision, in sequence, with full provenance. The projection serves enforcement and display: the inspectable trust fingerprint of commit gate activity rendered as a phase-by-phase account of how a query actioning a durable response was processed. This separation – immutable write-side ground truth and derived read-side summary – is the architectural mechanism that makes real-time commit gating possible without sacrificing the completeness of the post-process audit record.

3.6 Conversation Identity and Trace Coherence

Every IPU trace is anchored to a specific user interaction through a two-coordinate identity structure. One coordinate is the **interaction identifier** (`conversation_id`): a permanent, non-reusable marker generated when the user issues a command, binding that exchange to its execution record. The other is the **execution identifier** (`trace_id`): generated when the trace is initialized to identify the processing pathway, including the subsystems consulted, the sequence, and the authority. These two coordinates serve complementary roles. The interaction identifier is the identity of the exchange. The execution identifier is the identity of the process that produced it. Every event emitted during

a single execution carries both, ensuring that the trace is simultaneously interaction-coherent and execution-coherent without requiring subsystems to reconstruct context.

This binding produces a navigable provenance structure in both directions: given an interaction identifier, the system can retrieve every subsystem event that produced the corresponding response; given an execution identifier, the system can retrieve the original request and the system's response. Our companion paper [17] identifies this two-coordinate structure as the mechanism enabling an interaction archive that is "an immutable, append-only record of every prompt/response pair bound to its verbatim text and IPU trace, enabling indefinite per-record retrieval and provenance." Systems without this structural binding face significant obstacles verifiably answering trust-related questions; coordinates may exist in separate logs, but without system-wide, end-to-end structural linkage.

4. IPU vs. Existing Approaches

4.1 Comparison with XAI Methods

Conventional XAI methods provide useful but narrow visibility. LIME and SHAP [34,26] primarily explain single-model predictions by attributing importance to features after the output is produced. MLOps techniques [22] focus on population-level behavior, essential for real-time system monitoring. Attention visualization [43] shows token-level weighting signals during inference [18], and chain-of-thought (CoT) traces [44] surface generated rationale during decoding [3,41].

Each of these methods has their strengths and weaknesses. The IPU differs along four dimensions:

1. **Scope.** Existing XAI methods are limited in scope, explaining a single model's output, aggregating behavior across a population, or surfacing only internal model signals. The IPU's scope is the full execution pathway: query ingress, intent declaration, legality gating, knowledge retrieval, reasoning, verification, response assembly, commit gate evaluation, and output completion. It is a system-agnostic primitive for every ordered phase, every subsystem, and every decision point, including those that involve no LLM activity at all.
2. **Character.** Existing methods are prediction-based and retroactive, generating post-output explanations with no authority over contemporaneous or subsequent system behavior. The IPU is process-based and authoritative: it records activity in real time as execution proceeds. The system cannot commit until the trace permits it.
3. **Unity.** Existing methods are fragmented and subsystem-specific, with multiple tools targeting different subsystems deployed across a stack. The IPU is unified and subsystem-agnostic. Every component emits events to the same trace using the same schema, producing a coherent, end-to-end record of the system's deliberative process, not a collection of separate logs.
4. **Audience.** Existing tools are engineering-facing: observability dashboards, model cards, and drift metrics designed for developers and operators. The IPU is user-facing. The trace identifier is returned with every response, and the full execution record is accessible through the user-facing API as a first-class accountability artifact.

The key distinction between the prediction-based approach and the process-based approach is this: prediction-based methods explain the relevant model; the IPU explains the system.

4.2 Comparison with Observability Tools

Modern observability stacks (OpenTelemetry, Datadog, Jaeger) provide distributed tracing for software systems. The IPU shares structural similarities (spans, traces, correlation IDs) but differs in three fundamental ways:

1. **Semantic content.** Observability traces record latency and error rates. IPU traces record deliberative content, including which knowledge was accessed, what routing decisions were made, what confidence scores were assigned.
2. **User-facing.** Observability tools are for engineers. IPU traces are for users. The trace ID is returned in every response, and the user can query the full execution pathway through the API.

3. **Commit gate.** Observability traces have no causal authority. A slow span in Jaeger does not prevent the system from completing the request. An incomplete IPU trace prevents the system from committing durable effects.

Observability tools answer the question "what is the system doing?" for the engineers who built it. The IPU answers the question "why did the system act?" for the users who depend on it.

4.3 Comparison with Audit Logs

Enterprise audit logs record who did what and when. IPU traces subsume this function but go further, recording not just the action but the deliberative process that led to the action. An audit log relays "the system answered the question." An IPU trace relays "the system classified the query as high-stakes, consulted a curated knowledge source rather than parametric memory, reranked results by confidence, assembled the response from the top-ranked passages, and here is what each step produced."

Audit logs are limited by structure. An audit log establishes an action occurred but not whether an a priori condition was met [10]. The distinction goes beyond semantics. A system that acts and then records is structurally different from a system that records as a precondition for acting. The IPU instantiates the latter, elevating the trace from retrospective account to contemporaneous authorization.

4.4 Comparison with Machine Learning Operations

MLOps substantially improves lifecycle transparency by tracking data provenance, feature pipelines, model versions, and pre-/post-processing logic across training and deployment [22]. In terms of explainability, however, MLOps shares the same shortcomings of other prediction-based XAI approaches when compared to the IPU. Its scope is the training and deployment lifecycle rather than inference-time execution; its character is retrospective and population-level rather than real-time and decision-specific; its audience is operators and data scientists rather than end users; and its outputs carry no causal authority over what the system may do next.

The IPU is not a replacement for MLOps. Instead it is a complementary inference-time layer that extends MLOps with end-to-end, decision-time accountability, completing the accountability stack across the system lifecycle and mitigating harmful actions that occur before root-cause MLOps analysis [6,40].

5. Implementation

Our companion paper [17] presents AIOS (Anywhere Intelligence Operating System) as a naive framework for implementing a personal intelligence system: user-owned meaning architecture, continuous engagement without session boundaries, and meaning-constrained execution. In that paper, we treat the IPU architecture developed here as prior work and extend it to usage in an AI system featuring a personal world model (the locally embodied structure of referents, relationships, and historical associations through which a specific user experiences the world). This paper's contribution is the IPU architecture itself. AIOS is the reference implementation in which that architecture is first engineered and deployed.

5.1 Design Principles

Three governing principles shape the AIOS implementation of the IPU. The first is **traceability as a precondition for action**: if an action cannot be explained via its trace, it must not be allowed to produce durable effects. This guarantee has a direct institutional analog – courts maintain transcripts, hospitals maintain medical records, banks maintain transaction histories. People generally distrust institutions that operate without records; the IPU extends this accountability requirement as a structural invariant.

The second is **epistemic humility as architecture**: premature certainty erodes trust faster than explicit uncertainty; clarification is preferable to silent resolution; and "I don't know" is a required output class with the same architectural standing as a confident answer [21,47,20]. The IPU operationalizes this architecture by making the system's epistemic state inspectable in the trace (retrieval hit counts, confidence scores, verification results). A system response of "I don't know" is an acceptable and verifiable output with evidentiary backing.

The third is **non-agentic by design**: no autonomous action outside explicit user command [38]. Every IPU begins with an explicit user-issued action; the system does not self-initiate, infer intent without confirmation, or act on behalf of the user without being asked. Autonomous agents that act without instruction create an asymmetry of agency; the IPU inverts this. The user acts, the system responds, and the trace makes that response inspectable and auditable.

5.2 The AIOS Implementation

In AIOS, each of the nine lifecycle phases described above is instantiated by a named subsystem. In this section we provide an overview of that lifecycle, and highlight three aspects of the IPU's implementation in AIOS that merit particular examination.

5.2.1 Lifecycle Overview

At ingress, the API boundary receives the user's request and binds an interaction identifier (`conversation_id`) to the interaction, establishing the user's identity coordinate before any processing begins. An execution identifier (`trace_id`) is then auto-generated at the opening of the trace, binding the execution pathway to the interaction for the life of the query. Declaration is handled by GSIC (Geometric Superposition Intent Classifier): it classifies the query's intent using a combination of pattern matching and geometric embedding, determines whether routing proceeds to a deterministic tool or to the synthesis layer, and emits its routing decision as a structured trace event. The legality gate enforces scope and authorization policy before any knowledge access occurs, recording which knowledge scopes were requested, permitted, and blocked.

The retrieval layer accesses the personal knowledge graph and corpora sources, emitting source identifiers, retrieval confidence scores, and re-ranking decisions for every item consulted. The synthesis layer conducts LLM inference or deterministic computation, recording model tier, token count, and generation latency. The verification layer performs consistency and confidence checks on the assembled output, emitting pass/fail state for each check. The assembly phase composes the final response from ranked, scope-filtered sources. The commit gate evaluates the current trace projection against the committability condition before any durable knowledge writing. The terminal event closes the trace with a full latency record and returns the `trace_id` to the user as the interaction's accountability handle, accessible via the user-facing API.

The communicative mechanism that makes this end-to-end record possible is the **IPU emitter** (`IPUEmitter`): the single required interface through which every subsystem declares its activity to the trace. Subsystems never interact with IPU internals (the `IPUScribe`, `IPUProjector`, or `IPUGate`) directly; they interact exclusively through the emitter, which validates, sequences, and durably persists each emission before returning an `IpuEmitResponse` carrying both the recorded event and the current projection.

5.2.2 The Routing System

AIOS treats the IPU trace as a first-class API artifact, not a developer diagnostic. Every query pairs a `conversation_id` with a `trace_id`, and every API response returns the `trace_id` as the user's downstream entry point for system accountability. The full event stream and current projection are accessible via the user-facing API. This accessibility is intentional rather than incidental. In AIOS, a companion application surfaces the trace for each interaction (which sources were consulted, how long each phase required, what routing decisions were made), allowing users to directly query the system's explainability layer.

The routing system showcases the IPU trace's deliberative richness. When GSIC routes a query to a deterministic tool, the trace records the tool selected and its confidence score, the classification method employed, the parameters extracted from the query, and the tool's execution latency. When GSIC routes to corpora sources, the trace records which sources were selected and which were filtered. When GSIC triggers clarification, the trace records the specificity score and the domain hint used in the clarification response. The result is an interaction record in which routing is not a black-box precondition to the answer but an inspectable, auditable first-phase commitment step. The user can situate the system's output within subsystem reasoning pathways, confidence scores, and decision determinations.

5.2.3 The IPU Emitter

On its own, the IPU neither initiates action nor interprets execution; it receives and records. The mechanism that makes this reception coherent across a heterogeneous pipeline is the emitter: a per-service component instantiated by each subsystem with its own layer designation, component identifier, and invocation type.

For example, GSIC instantiates an emitter scoped to routing decisions; the retrieval layer instantiates one scoped to knowledge access; the synthesis layer instantiates one scoped to LLM invocation. Each service calls three methods – `start_span`, `progress`, and `end_span` – with structured, service-specific payloads. The `IPUScribe` receives these emissions and builds a canonical, sequence-chained event record. Every event in a single execution carries the same `trace_id` tied to the `conversation_id` governing the interaction. The result is a normalized, end-to-end trace produced by active subsystems and assembled by a passive pipeline. Instead of a log appended after the fact, the result is a record constructed in parallel with execution as each subsystem declares what it is doing and under what authority.

To enforce semantic discipline across the pipeline, each subsystem maintains a typed emission module that functions as a service-local wrapper carrying its own operational vocabulary and activity declarations: routing decisions and confidence scores for the intent classifier; source identifiers and ranking scores for the retrieval layer; scope boundaries and access decisions for the authorization layer; and token counts and generation latency for the synthesis layer. Typed emissions ensure every event is both specific and consistent, resulting in a trace produced by active subsystems, assembled by a passive pipeline, and enforceable as a precondition for durable system effects.

5.2.4 The Personal Knowledge Infrastructure

The IPU's integration into the AIOS personal knowledge layer (the layer holding entities, relationships, and normative boundaries explicitly promoted from candidate to durable referent by the user) reflects a structural distinction between facts inferred by the system and meaning commitment authorized by the user [17]. In AIOS, promotion is not inference; it is an authorized operation. The IPU trace is the authorization record that makes it so. Every durable artifact written to the personal knowledge layer carries the `trace_id` of its execution, providing an immutable provenance certificate for the authorizing act that established it.

This architecture has direct consequences for personal alignment, or what we term in our companion paper **constitutional continuity** [17]: the requirement that a system accompanying a user over years preserve the structure through which that user revises their own normative commitments. When a user's beliefs or values change (a boundary shifts, a prior commitment is revised, a previously-trusted source is repudiated) the IPU trace makes that revision attributable and recoverable. The system is able to represent constitutional amendment as a historic, traceable, authorized operation.

6. Challenges and Investigative Pathways

6.1 Anticipated Challenges

The primary contributions of this paper are architectural. The IPU is a novel approach to XAI for human-centric AI systems requiring a unified, end-to-end, process-based inspectability and accountability framework. We anticipate several challenges to its adoption, briefly acknowledging them here while reserving fuller treatment for later work.

The first class of challenges reclaim what we describe as already present in adjacent techniques: that the IPU reduces to a well-engineered audit log, that existing XAI methods already address explainability, that distributed tracing tools in software observability accomplish comparable goals, or that CoT obviates further need for explainability mechanisms. We regard these challenges as correctly identifying intellectual predecessors, but dispute their equivalence to our contribution. The decisive structural difference in each case is the presence of an atomic, unified causal authority. Audit log records; the IPU governs. Existing XAI methods address model predictions; the IPU addresses the deliberative pipeline. Observability tools surface system metrics; the IPU surfaces deliberative contracts. CoT displays rationale tokens; the IPU provides end-to-end visibility. (The CoT objection is particularly vulnerable given research demonstrating CoT tokens as unfaithful to final output [41,23] with an underlying pathology seated in RLHF optimization for perceived sincerity rather than

evidential accuracy [7,3]. The IPU sidesteps this problem entirely by accepting the native limitations of stochastic validation and fully evidencing the surrounding execution path.)

A second class of challenges concerns scale and complexity. The architecture described here is more complex to implement than a standard logging stack, and translating trace data into intuitive trust signals for end users is an open UX problem. We acknowledge both. We also acknowledge the LLM's internal reasoning remains opaque to the IPU: the trace records what the pipeline did, not why the LLM opted for a particular phrase. (The IPU does, however, make stochasticity visible. By recording the end-to-end path from query to response, the IPU enables users to calibrate trust around a particular output and task rather than treating all outputs and tasks as equally reliable.) Our claim is not that the IPU is simple or complete, but that systems without a unified, process-based accountability mechanism such as the IPU face significant risk in aligning system capabilities and responsibilities and earning user trust.

A third class of challenges holds that AI systems should not be held to accountability standards that humans themselves do not meet. We view this class of challenge more as an apologistic response to system error than a hurdle to XAI adoption. Nevertheless, we acknowledge that the conditions under which accountability asymmetries between humans and machines are justified warrant careful treatment in their own right [37,42,12].

6.2 Investigative Pathways

The IPU is a proposed architectural primitive for explainability. Whether the IPU produces measurable trust improvements over competing alternatives is an empirical question. The following investigative pathways provide an entry point to establishing, refining, or refuting its value as a legitimate addition to the field of XAI.

I1: Trust calibration outcomes. Does access to IPU traces actually improve users' ability to calibrate trust – to rely on the system when appropriate and question it when not? A rigorous study would compare trust calibration accuracy between users with and without trace access across a range of query types, including cases where the system is correct and cases where it fails. Baseline: standard LLM assistant without process-based trace. Metric: trust calibration score on a validated instrument [15,45]. Prediction: trace access will improve calibration on failure cases more than on success cases, since the value of explainability is highest when the system is wrong.

I2: User comprehension of trace data. Structural transparency does not guarantee comprehensibility. What trace representations, levels of detail, and interaction patterns most effectively support user understanding of system behavior? This is a UX research problem requiring studies of how users actually read, interpret, and act on trace information. The risk of poorly designed trace presentation is that it creates an illusion of transparency without supporting genuine understanding [38].

I3: Cross-system portability. The IPU is specified here in the context of the AIOS reference implementation. An open, portable, implementation-agnostic IPU specification that any AI pipeline could adopt is required. The key research question is which elements of the IPU schema are architecture-independent and therefore standardizable, versus which are specific to particular pipeline architectures.

I4: Adversarial integrity. In high-stakes contexts the question arises whether IPU traces can be tampered with or forged. The AIOS implementation enforces append-only semantics, sequence-chained integrity, and cryptographic chaining across the event record, producing tamper-evident traces under adversarial conditions. Whether this is sufficient for regulated deployment contexts is an open question for security and compliance research [2].

I5: Compositional accountability. The IPU enforces accountability at the pipeline level. As AI systems become more compositional, maintaining coherent IPU tracing across distributed execution boundaries becomes non-trivial. Extending architectural accountability to multi-agent and cross-system AI pipelines is the next frontier.

7. Conclusion

In a world where AI systems are becoming tools of daily life – as ordinary as a toaster, a calculator, or a car – the preconditions for trust require heightened attention in AI system design. In this paper, we contend trust in AI systems is not a feature to be added but a property to be architected. The

Intelligence Processing Unit provides this architecture by making the system's deliberative processes inspectable, immutable, and causally authoritative. Every question produces a trace. Every trace is scrutable. Every durable effect requires a commitment gate.

We acknowledge our contribution builds on and extends prior work in XAI. The novelty of the IPU is the unifying role it plays in recording and authorizing subsystem activity across heterogeneous pipelines to enable system-agnostic, inference-time accountability. Unlike prediction-based mechanisms, the IPU's process-based approach enables AI systems to answer "why did you do that?" by reference to an end-to-end execution pathway that serves as both contemporaneous authorizing gate and auditable system record.

References

[1] Amershi, S., Weld, D., Vorvoreanu, M., et al. (2019). Guidelines for Human-AI Interaction. *Proceedings of the 2019 ACM CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3290605.3300233

[2] Balan, K., Learney, R., & Wood, T. (2025). A Framework for Cryptographic Verifiability of End-to-End AI Pipelines. *IWSPA 2025*. arXiv:2503.22573 https://arxiv.org/abs/2503.22573

[3] Barez, F., Wu, T., Arcuschin, I., et al. (2025). Chain-of-Thought Is Not Explainability. *ICLR Workshop on Foundation Models in the Wild*. PDF: https://fbarez.github.io/assets/pdf/Cot_Is_Not_Explainability.pdf

[4] Bender, E.M. & Koller, A. (2020). Climbing Towards NLU: On Meaning, Form, and Understanding in the Age of Data. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, 5185–5198. https://aclanthology.org/2020.acl-main.463

[5] Benk, M., Kerstan, S., von Wangenheim, F., & Ferrario, A. (2024). Twenty-four years of empirical research on trust in AI: a bibliometric review of trends, overlooked issues, and future directions. *AI & Society*, 40, 2083-2106. https://doi.org/10.1007/s00146-024-02059-y

[6] Chien, J. & Danks, D. (2025). Trustworthiness in Stochastic Systems: Towards Opening the Black Box. arXiv:2501.16461 https://arxiv.org/abs/2501.16461

[7] DeVilling, B. (2025). The Polite Liar: Epistemic Pathology in Language Models. arXiv:2511.07477 https://arxiv.org/abs/2511.07477

[8] Dzindolet, M.T., Peterson, S.A., Pomranky, R.A., et al. (2003). The Role of Trust in Automation Reliance. *International Journal of Human-Computer Studies*, 58(6), 697-718. https://doi.org/10.1016/S1071-5819(02)00020-0

[9] Ehsan, U. & Riedl, M.O. (2024). Explainable AI Reloaded: Challenging the XAI Status Quo in the Era of Large Language Models. *ACM HTTF 2024*. arXiv:2408.05345 https://arxiv.org/abs/2408.05345

[10] European Union. (2024). Regulation (EU) 2024/1689 of the European Parliament and of the Council (Artificial Intelligence Act). *Official Journal of the European Union*. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689

[11] Flavell, J.H. (1979). Metacognition and cognitive monitoring: A new area of cognitive-developmental inquiry. *American Psychologist*, 34(10), 906-911. https://doi.org/10.1037/0003-066X.34.10.906

[12] Frazier, B.N., Gelman, S.A., & Wellman, H.M. (2009). Preschoolers' Search for Explanatory Information within Adult-Child Conversation. *Child Development*, 80(6), 1592-1611. https://doi.org/10.1111/j.1467-8624.2009.01356.x

[13] Gabriel, I. (2020). Artificial Intelligence, Values, and Alignment. *Minds and Machines*, 30(3), 411–437. https://doi.org/10.1007/s11023-020-09539-2

[14] Hancock, P.A., Billings, D.R., Schaefer, K.E., et al. (2011). A Meta-Analysis of Factors Affecting Trust in Human-Robot Interaction. *Human Factors*, 53(5), 517-527. https://doi.org/10.1177/0018720811417254

[15] Hoff, K.A. & Bashir, M. (2015). Trust in automation: Integrating empirical evidence on factors that influence trust. *Human Factors*, 57(3), 407-434. https://doi.org/10.1177/0018720814547570

[16] Holbrook, J. & Holbrook, J. (2025). Understanding Is All You Need. Anywhere Intelligence (working paper).

[17] Holbrook, J., Kallakuri, S., Holbrook, J., & Burgess, L. (2026). Personal World Models: Architectural Requirements for Personal Intelligence Systems. Anywhere Intelligence (working paper).

[18] Jain, S. & Wallace, B.C. (2019). Attention Is Not Explanation. *Proceedings of NAACL-HLT 2019*, 3543-3556. arXiv:1902.10186 https://arxiv.org/abs/1902.10186

[19] Ji, Z., Lee, N., Frieske, R., et al. (2023). Survey of Hallucination in Natural Language Generation. *ACM Computing Surveys*, 55(12), 1-38. https://doi.org/10.1145/3571730

[20] Kalai, A.T., Nachum, O., Vempala, S.S., & Zhang, E. (2025). Why Language Models Hallucinate. arXiv:2509.04664 https://arxiv.org/abs/2509.04664

[21] Kominsky, J.F., Langthorne, P., & Keil, F.C. (2016). The Better Part of Not Knowing: Virtuous Ignorance. *Developmental Psychology*, 52(1), 31-45. https://doi.org/10.1037/dev0000065

[22] Kreuzberger, D., Kuhl, N., & Hirschl, S. (2023). Machine Learning Operations (MLOps): Overview, Definition, and Architecture. *IEEE Access*, 11, 31866-31879. arXiv:2205.02302 https://doi.org/10.1109/ACCESS.2023.3262138

[23] Lanham, T., Chen, A., Radhakrishnan, A., et al. (2023). Measuring Faithfulness in Chain-of-Thought Reasoning. arXiv:2307.13702 https://arxiv.org/abs/2307.13702

[24] Lee, J.D. & See, K.A. (2004). Trust in Automation: Designing for Appropriate Reliance. *Human Factors*, 46(1), 50-80. https://doi.org/10.1518/hfes.46.1.50_30392

[25] Liao, Q.V. & Vaughan, J.W. (2024). AI Transparency in the Age of LLMs: A Human-Centered Research Roadmap. *Harvard Data Science Review*, Special Issue 5. arXiv:2306.01941 https://arxiv.org/abs/2306.01941

[26] Lundberg, S.M. & Lee, S. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems (NeurIPS)*, 30. arXiv:1705.07874 https://arxiv.org/abs/1705.07874

[27] Marcus, G. (2018). Deep Learning: A Critical Appraisal. arXiv:1801.00631 https://arxiv.org/abs/1801.00631

[28] Mayer, R.C., Davis, J.H., & Schoorman, F.D. (1995). An Integrative Model of Organizational Trust. *Academy of Management Review*, 20(3), 709-734. https://doi.org/10.5465/amr.1995.9508080335

[29] Miller, T. (2019). Explanation in Artificial Intelligence: Insights from the Social Sciences. *Artificial Intelligence*, 267, 1–38. https://arxiv.org/abs/1706.07269

[30] Mitchell, M. (2020). On Crashing the Barrier of Meaning in AI. *AI Magazine*, 41(2), 86–92. https://doi.org/10.1609/aimag.v41i2.5259

[31] National Institute of Standards and Technology. (2023). *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*. NIST AI 100-1. https://doi.org/10.6028/NIST.AI.100-1

[32] Nushi, B., Kamar, E., & Horvitz, E. (2018). Towards Accountable AI: Hybrid Human-Machine Analyses for Characterizing System Failure. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*. arXiv:1809.07424 https://arxiv.org/abs/1809.07424

[33] Parasuraman, R. & Riley, V. (1997). Humans and Automation: Use, Misuse, Disuse, Abuse. *Human Factors*, 39(2), 230-253. https://doi.org/10.1518/001872097778543886

[34] Ribeiro, M.T., Singh, S., & Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *Proceedings of ACM KDD 2016*, 1135-1144. arXiv:1602.04938 https://arxiv.org/abs/1602.04938

[35] Rousseau, D.M., Sitkin, S.B., Burt, R.S., & Camerer, C. (1998). Not so different after all: A cross-discipline view of trust. *Academy of Management Review*, 23(3), 393-404. https://doi.org/10.5465/AMR.1998.926617

[36] Russell, S. (2022). Artificial Intelligence and the Problem of Control. In Werthner, H., Prem, E., Lee, E.A., & Ghezzi, C. (eds.), *Perspectives on Digital Humanism*, pp. 19–24. Springer, Cham. https://doi.org/10.1007/978-3-030-86144-5_3

[37] Schneier, B. & Sanders, N.E. (2025). AI Mistakes Are Very Different from Human Mistakes. *IEEE Spectrum*. January 2025. https://spectrum.ieee.org/ai-mistakes-schneier

[38] Shneiderman, B. (2022). *Human-Centered AI*. Oxford University Press. ISBN: 9780192845290

[39] Slovic, P. (1987). Perception of risk. *Science*, 236(4799), 280-285. https://doi.org/10.1126/science.3563507

[40] Spoczynski, M., Melara, M.S., & Szyller, S. (2025). Atlas: A Framework for ML Lifecycle Provenance & Transparency. arXiv:2502.19567 https://arxiv.org/abs/2502.19567

[41] Turpin, M., Michael, J., Perez, E., & Bowman, S.R. (2023). Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting. *Advances in Neural Information Processing Systems (NeurIPS)*, 36. arXiv:2305.04388 https://arxiv.org/abs/2305.04388

[42] Vallor, S. & Vierkant, T. (2024). Find the Gap: AI, Responsible Agency and Vulnerability. *Minds and Machines*, 34, article 20. https://doi.org/10.1007/s11023-024-09674-0

[43] Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30. arXiv:1706.03762 https://arxiv.org/abs/1706.03762

[44] Wei, J., Wang, X., Schuurmans, D., et al. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems (NeurIPS)*, 35. arXiv:2201.11903 https://arxiv.org/abs/2201.11903

[45] Yin, M., Wortman Vaughan, J., & Wallach, H. (2019). Understanding the effect of accuracy on trust in machine learning models. *Proceedings of the 2019 ACM CHI Conference on Human Factors in Computing Systems (CHI 2019)*, Article 279, 1-12. https://doi.org/10.1145/3290605.3300509

[46] Zhang, Y., Liao, Q.V., & Bellamy, R.K.E. (2020). Effect of confidence and explanation on accuracy and trust calibration in AI-assisted decision making. *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 295-305. https://doi.org/10.1145/3351095.3372852

[47] Zhang, H., Diao, S., Lin, Y., et al. (2024). R-Tuning: Instructing Large Language Models to Say 'I Don't Know'. *Proceedings of NAACL 2024*. arXiv:2311.09677 https://arxiv.org/abs/2311.09677