XPLORE INTELLIGENCE

COMPETITIVE ANALYSIS · AGENT VERIFICATION

Xplore is the verification layer for enterprise AI agents.

The platform traces claims, numbers and actions back to source-of-record evidence, then applies policy before outputs are released.

WHERE WE SIT IN THE AGENT STACK

L4	**Observe · eval · guardrails** Langfuse · Galileo · Arize · LangSmith
L3	**Orchestration & frameworks** LangGraph · CrewAI · AutoGen
L2	**Verification layer — Xplore** Node Resolution engine · Data Integrity metric · any runtime
L1	**Knowledge graph & data** Neo4j · Postgres · S3
L0	**Models & tools** OpenAI · Anthropic · MCP · internal APIs

The current stack is strong on models, orchestration and monitoring. Xplore adds the missing verification layer: agents run with claim-level provenance, policy and auditability.

TRUSTGRAPH · PRODUCT — NODE RESOLUTION · ENGINE — DATA INTEGRITY · METRIC — COMPETITIVE ANALYSIS · 2026

---

01 — THE PROBLEM

A confident answer and a fabricated one look the same.

Same output. The market checks how it **reads**. Xplore checks whether each figure is **real**.

LOOKS RIGHT — WHAT THE MARKET CHECKS

"Q3 exposure is $1.22M, scaling the $1.4M cargo balance from the CRM by the 0.87 regional factor."

- ✓ Reads fluently, well-structured
- ✓ LLM-as-judge faithfulness **0.91**
- ✓ No guardrail or policy hit

VERDICT — SHIPS TO THE USER

IS RIGHT — WHAT WE CHECK

"Q3 exposure is $1.22M, scaling the $1.4M cargo balance from the CRM by the 0.87 regional factor."

- ✓ **$1.22M** — recomputed: 1.4M × 0.87 matches
- ✗ **"$1.4M from the CRM"** — actually scraped from a website; no CRM record of origin
- ✓ **0.87 factor** — source resolves, snippet matches

VERDICT — 1 OF 3 CLAIMS FAILS PROVENANCE · BLOCKED

The market is strong at traces, judges and guardrails. Xplore adds claim-level provenance: where the number came from, which source it cites, and whether that source supports it.

---

02 — OUR TECHNOLOGY

Node Resolution: capture any runtime → resolve the graph → one Data Integrity score.

HOW IT PLUGS IN

SDK
instrument an agent in ~5 lines

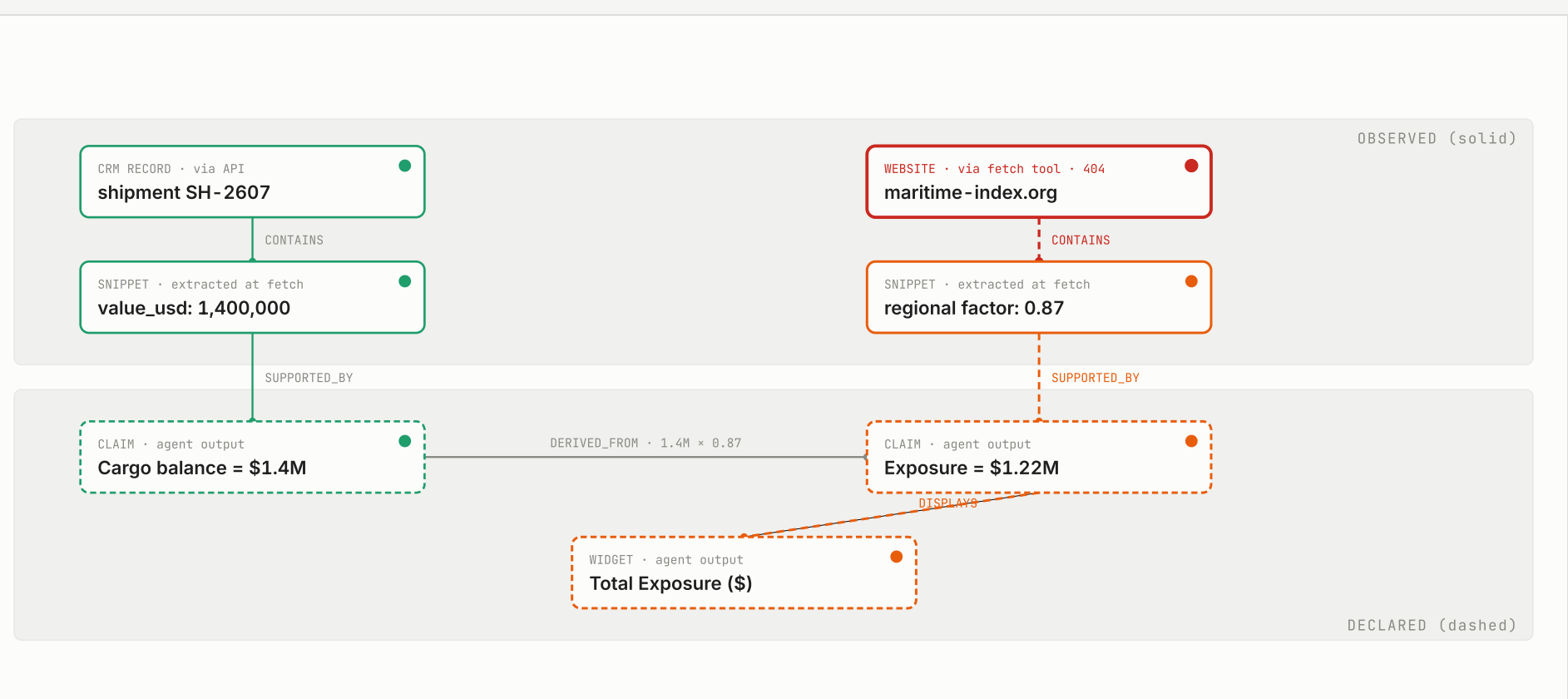
Bridges
OpenTelemetry / OpenInference · OpenAI · MCP shim

Gateway
proxy mode — wrap any runtime, no code

OBSERVED ← real I/O, captured & hashed

DECLARED ← the agent's own claims

Agents **stay where they run** — capture feeds both zones of the graph.



METRIC **Data Integrity** — ✓ Grounding ✗ Attribution ✓ Consistency ✓ Traceability — PROPRIETARY · AUDITABLE

Deterministic where ground truth exists; model-assisted where it does not. A **located, attributed** failure becomes an auditable control point for gating, remediation and training.

---

03 — WHY IT'S BETTER

A score is useful. A reproducible cause is operational.

Most eval tools can score outputs and traces — with code-based checks and LLM judges. Our specialty is narrower and deeper: **provenance** — did this claim actually come from the source it cites?

MOST EVAL TOOLS — SCORING

Score the output & trace

- › **Strong at** tracing, eval workflows, dashboards and scoring.
- › Scoring is code-based checks (need a gold reference) or LLM / SLM judges (Galileo Luna, Arize & LangSmith judges).
- › Answers **"is the output good?"** — not "where did this number come from?"
- › Judge-based scores can **drift** run-to-run; the written reason is itself a model output.

STRONG ON QUALITY SIGNALS

XPLORE — PROVENANCE VERIFICATION

Verify where each claim came from

- ✓ Checks each claim against the **actual source and record** it cites — value origin included.
- ✓ **Deterministic where ground truth exists**; model-assisted for open or qualitative claims.
- ✓ **Reproducible** for provenance checks — the cause is tied to evidence and policy.
- ✓ Each point of the score links to the **exact node and check** behind it.

AN AUDITABLE TRAIL OF EVIDENCE

Eval and observability provide quality signals. Xplore adds **claim-level provenance**. The same located cause that makes a failure auditable also makes it a training signal.

---

04 — THE PLATFORM

From verification to a platform: capture → evaluate → train.

Located failures become training signals. **TrustGraph rides on top of any runtime** — agents remain in the customer environment, which gives it the widest reach. **Forge and ABBI run on the Xplore platform.**

STEP 1

Capture

Record the run mechanically — any runtime, no model opinion.

- Inputs, outputs, claims, artifacts
- Tool & API calls + returns
- Sources, hashed at fetch
- Prompts, configs, policy

→

STEP 2

Evaluate

Node Resolution builds the graph and runs typed checks.

- Observed vs declared, per claim
- Deterministic where truth exists
- Data Integrity + located failures
- Trust propagated along provenance

→

STEP 3

Train

Forge turns located failures into changes — and re-verifies.

- Fix flow, tools, rules or source
- Re-run the same checks
- Promotion gates & rollback
- Drift caught live in prod

TrustGraph
captures & verifies — embeds on top of any runtime
RUNTIME-AGNOSTIC · AGENTS STAY WHERE THEY ARE · PLATFORM-SCALE COVERAGE

Forge
trains on what was found
RUNS ON THE XPLORE PLATFORM

ABBI — the business-facing app on the loop: trained agents fill the board, every figure verified, feeding back as new capture. RUNS ON THE XPLORE PLATFORM

The same graph that detects failures is what trains the fix. Each captured run becomes an eval; each eval becomes training data — no separate labeling, no GPU in the core.

---

05 — HOW DEEP WE TRAIN

Agent = model + harness. The market tunes the prompt. We train the whole harness.

"Scaffolding" is the prompt layer; the **harness** is the runtime that runs the loop, calls tools and enforces control. Vendors stop at the scaffold — fixing the harness is still manual everywhere. Xplore operates at every level — because we can attribute the failure to the part that caused it.

	CHANGE SURFACE	LAYER	CURRENT MARKET CAPABILITY	XPLORE
L0	Rewrite prompts	SCAFFOLD	Prompt optimizers DSPy · TextGrad · GEPA	✓
L1	Rewrite from traces	SCAFFOLD	Eval / trace tools LangSmith · Braintrust · Galileo — market stops here	✓
L2	Change the agent loop	HARNESS	Manual only — hand-edited scaffolds & flows	✓
L3	Change tools & data	HARNESS	Manual only — rewire integrations by hand	✓
L4	Change the tests	EVAL HARNESS	Manual QA — humans write & update cases	✓ Q4 2026
L5	Change the agent team	MULTI-AGENT	Emerging category — sub-agents, routers, shared memory	✓ 2027

The deeper the harness can be changed, the faster the agent adapts and the lower the operational cost. **Public tooling is concentrated around L0–L1**; L2+ remains largely manual in current agent-engineering practice.

---

06 — ENGINE & PRODUCTS

One engine. Land → expand → apply.

Node Resolution is the shared engine. We **land** as the verification layer, **expand** by using the failures it finds to improve the agent, and **prove** the stack with a business application — one product today, the others on the same graph.

Node Resolution → Data Integrity
observed vs declared · provenance graph · shared across products

TrustGraph
Verify any runtime
Provenance + trust propagation
Block ungrounded output
BEACHHEAD · the verification layer
LAND

Forge
Use the failures to improve
Train · re-verify · gate
Catch drift in production
EXPANSION · same dev buyer
EXPAND

ABBI
First business app on the stack
Every figure provenance-checked
Proof the platform works
APPLICATION · platform proof point
APPLY

The same graph runs underneath all three, so deployments compound. **We sell one thing today — the verification layer — and earn the right to the rest.**

---

07 — PRODUCT: TRUSTGRAPH

They score the output. We verify the evidence behind it.

	TrustGraph Xplore	Observability & eval LangSmith · Arize · Galileo · Ragas
What it verifies	✓ Source of every number, then the output	✓ Whether the output looks correct
Provenance — number → source	✓ Linked to system of record	– Not tracked
Block bad output before release	✓ Source-level policy gate	~ Output guardrails only
One agent's error reaching another	✓ Trust graph links A → B	– Traces are separate
How it verifies	✓ Deterministic checks first, LLM only where needed	~ Mostly LLM-as-judge on every output
Integration	✓ SDK · OpenTelemetry · proxy — any runtime	✓ SDK / OpenTelemetry
Cost at thousands of agents	✓ Most checks run with no model call — cost stays flat	– Per-trace billing + an LLM judge per output — grows with volume

At thousands of agents the gap compounds: judge-per-output pricing **scales with traffic**, while TrustGraph runs **deterministic checks with no model call** for the bulk of verification — and uses an LLM only where ground truth doesn't exist.

---

08 — PRODUCT: FORGE

Low-code training of the whole harness — on customer-specific benchmarks.

Optimizers tune prompts, RL trainers tune model behavior, and eval platforms measure quality. Forge trains the harness **around** the agent — generating or modifying tools, scripts, connectors, routing, policies and tests — then re-verifies the change before promotion.

CAPABILITY	Xplore Forge	Prompt optimizers DSPy · TextGrad	RL trainers OpenPipe ART	Eval platforms LangSmith · Braintrust
Scope of change	✓ Whole harness — flow, tools, connectors, scripts, policy, tests	– Prompt only	~ Model weights	– Measurement only
Authoring effort	✓ Low-code harness generation; no model fine-tune required	~ Code / DSL	– ML pipeline + GPU	~ Code
Custom benchmarks on customer environments	✓ Environment-specific simulation	~ From examples	~ From rollouts	✓ From datasets
Safety — gates, rollback, policy	✓ Built-in, reversible	– Outside core scope	– Outside core scope	~ CI gates
Continuous improvement (drift)	✓ Live re-train	– Outside core scope	– Outside core scope	~ Alerting
Runs without GPU / fine-tune	✓ Teaches the system	✓ Yes	– Requires GPU	✓ Yes

The result: **hours instead of weeks, and repeatable instead of manual.** Internal benchmark (~30 tasks per run vs estimated senior-engineer time) — methodology on request.

---

09 — PRODUCT: ABBI

ABBI: the proof — agentic BI on the verified-agent stack.

The **first application** on the verified-agent stack, showing that the verification layer can produce business output that is usable, traceable and governed. Incumbents lead on ecosystem reach and governed data; the dimensions below are where ABBI differs.

CAPABILITY	Xplore ABBI	Agentic BI ThoughtSpot Spotter · Tableau Next	Copilot / warehouse Power BI · Snowflake · Databricks
Who can build an agent	✓ Business user — no code or low-code	~ Analyst + semantic model	~ Analyst / SQL skills
Provenance of every figure	✓ Traced to source	~ SQL lineage to inspect	– Trust the model
Blocks ungrounded numbers	✓ Before they are shown	~ Constrained to its model	– Limited controls
Improves on domain-specific questions	✓ Built-in training	– Manual modeling	– Manual improvement loop
Data-platform independence	✓ Agnostic	– Own platform	– Own ecosystem

Incumbents lead on ecosystem reach. **ABBI's edge is provenance and accessibility** — relevant where a wrong-but-confident figure is unacceptable: finance, healthcare, logistics, compliance.

XPLORE INTELLIGENCE — SUMMARY

Enterprise agents need an audit trail. Xplore is the verification layer that gives them one.

The verification layer
WHAT WE ARE
Prove every claim against its source of record; block ungrounded output before it reaches users.

Provenance-first verification
WHY WE'RE DIFFERENT
Eval tools score outputs and traces; Xplore verifies where each claim came from — deterministic where ground truth exists.

Land → expand → apply
HOW WE GROW
TrustGraph today; Forge and ABBI on the same graph, so deployments compound.