



AI-NATIVE FINANCIAL DATA FOUNDATIONS

The Dynamic Context Layer in Financial Services AI

Why Data and AI Are Not Enough Without Institutional
Context

WHITE PAPER

Release v1.0 | September 2026

A perspective on why financial systems provide partial views of the business, why AI needs a live operational representation of the institution, and how to build that layer through Model → Bind → Operationalise.

finsight-tech.com



Contents

Executive Summary

- 1. The Missing Layer Between Data and AI**
- 2. From Systems of Record to a System of Context**
- 3. What Makes Financial Context Dynamic?**
- 4. From Semantic Meaning to Operational Context**
- 5. Anatomy of a Dynamic Context Layer**
- 6. Model → Bind → Operationalise**
- 7. Three Financial Situations**
- 8. Dynamic Context for Financial AI Agents**
- 9. Evidence, Change and Trust**
- 10. Toward Context-Native Financial Services**

References

Executive Summary

Financial institutions are not short of data systems. Over decades they have built specialised platforms for trading, positions, risk, reference data, client relationships, payments, treasury, legal agreements, regulatory reporting and analytics. Modern warehouses, lakehouses, APIs and data products have made much of that information easier to access. At the same time, financial institutions increasingly have access to capable AI: foundation models, machine-learning platforms, copilots and agents that can retrieve information, interpret documents, call tools and perform sophisticated reasoning.

Yet an architectural gap remains between those two investments. The business situation that AI needs to understand rarely exists in any one system. A position store may know what a portfolio holds. A security master knows the contractual terms of an instrument. A cashflow engine projects payments. A risk engine calculates exposure. A CRM platform contains relationship information. A trading system records transactions. A mandate or policy source constrains what may be done. Each source can be authoritative for particular facts while still providing only one perspective on the wider institutional situation.

CORE THESIS Source systems provide perspectives. The context layer represents the situation. AI reasons and acts on that situation.

This paper calls the missing architectural capability the dynamic context layer. It is not another data warehouse, and it is not simply a glossary, a static semantic layer or an ontology repository. It is an operational representation of the institution's current business world: the relevant objects, identities, relationships, states, events, derived facts, rules, evidence, permissions and available actions, assembled from multiple underlying data perspectives.

The word dynamic is essential. Financial institutions operate through continuous change. Trades are executed, positions move, instruments approach lifecycle events, cashflows become due, exposures shift, markets move, new client information arrives and rules become applicable. The conceptual model of a Position or Instrument may be relatively stable; the institutional situation represented by actual positions, exposures, events and evidence is not. The model tells AI what can exist. Dynamic context tells AI what is true, relevant and actionable now.

The word operational is equally important. Classical semantic modelling is concerned with what things mean and how kinds of things relate. Those foundations remain valuable, but AI operating inside a financial institution needs more: actual business objects, current relationships, current state, recent events, derived facts, applicable rules, traceable evidence and the actions that are permitted in the current situation. In this sense, the context layer turns meaning into operational capability.

The practical approach proposed here is deliberately simple: Model the relevant institutional world; Bind that model to the multiple systems through which the world is observed; Operationalise the resulting context for AI, analytics, rules and workflows. The objective is not to replace the existing data estate or create another proprietary copy of it. It is to make the estate understandable as a coherent business world.

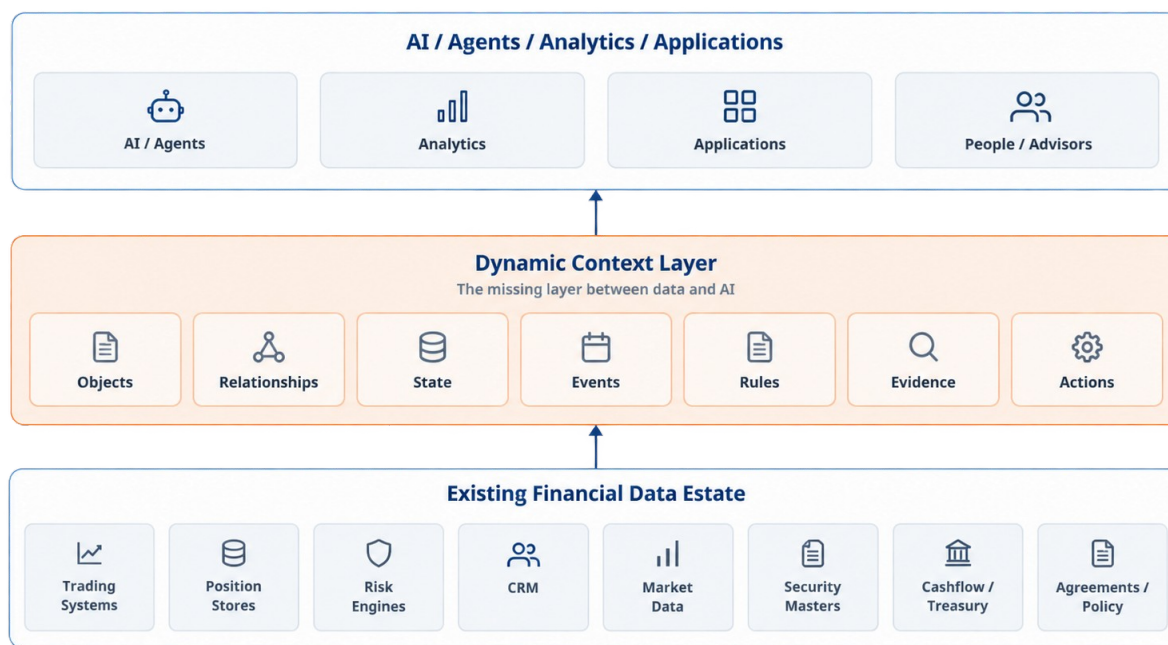


Figure 1. The missing layer in financial services AI. Existing systems of record provide the data estate; the dynamic context layer assembles business situations that AI, analytics, applications and people can use.

1. The Missing Layer Between Data and AI

For much of the last two decades, the dominant financial-services data problem was access. Data was distributed across product platforms, legal-entity stores, operational systems, risk engines and reporting environments. Enterprises invested heavily in integration, warehouses, lakes, catalogues, lineage, APIs and governance so that information could be discovered and used more consistently.

AI changes the nature of the problem. A human analyst can often compensate for fragmented institutional knowledge. They know which source is authoritative, which identifiers are equivalent, which relationships matter, which calculations are trusted and which business rules apply. An AI agent does not possess that tacit context unless the institution makes it available in a usable form.

This means that exposing more tables or tools to an AI system can increase reach without necessarily increasing understanding. The agent may retrieve the correct records and still assemble them incorrectly. It may know that a bond matures next month without understanding the portfolio that holds it. It may know that an FX exposure exists without

understanding the client's hedge policy or recent transactions. It may see a settlement status without understanding the obligation that failed or the escalation rule that became applicable.

The architectural question is therefore not simply how to connect AI to enterprise data. It is how to give AI a current operational representation of the institutional situation in which those data points matter. That representation sits between source systems and downstream intelligence. It provides the structure through which facts become a situation, a situation becomes a conclusion, and a conclusion can become an action.

PRINCIPLE The next problem is not getting AI closer to data. It is getting AI closer to the institution's operating reality.

2. From Systems of Record to a System of Context

Financial data is fragmented for a deeper reason than historical system sprawl. Different systems are designed to represent different perspectives on the same business world. A position store exists to represent holdings. A security master represents instrument terms and identifiers. A risk engine represents calculated exposures. A cashflow engine represents projected payments. A CRM represents relationship information. A trading platform represents transactions. These are not simply duplicate datasets waiting to be consolidated; they are distinct views created for distinct operational purposes.

Consider a portfolio that holds a £20 million bond position. The position system can establish the holding. The security master can establish that the bond matures in eighteen days. A cashflow service can derive the expected maturity proceeds. A portfolio-management system can establish that duration is below target. A liquidity forecast can show that £2 million of the expected cash is required for near-term outflows. The trading system can show that no replacement transaction has been executed. A mandate source can constrain what instruments may be considered next.

No single source contains the business situation. Yet the situation is exactly what matters operationally: a particular portfolio is approaching a cash event; part of the resulting cash may be available; the portfolio is below a target; no replacement has yet been observed; and the client's mandate constrains the possible response. The institution may therefore have an emerging reinvestment situation. That conclusion is not stored as a source fact. It is assembled from several authoritative perspectives.

KEY DISTINCTION A system may be authoritative for a fact without being authoritative for the situation.

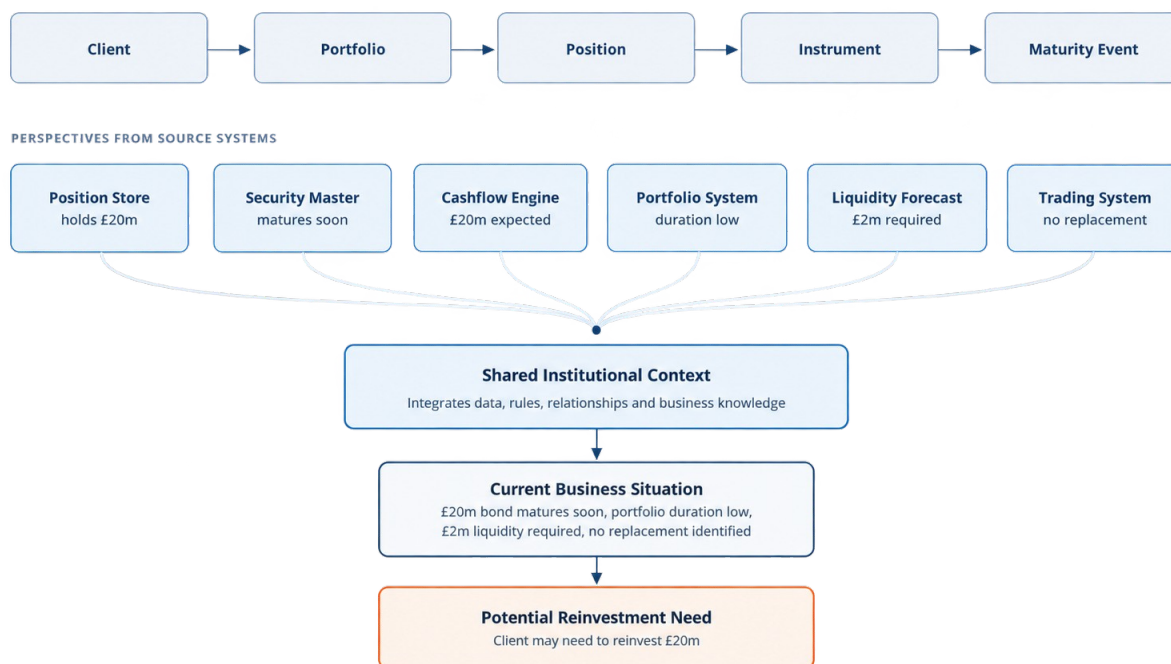


Figure 2. Multiple perspectives, one business situation. The maturing-bond example illustrates how authoritative facts from several systems can converge into a shared operational interpretation and a potential client need.

This is why the context layer should not be thought of as a conventional data-integration layer. Data integration normally asks how to move, reconcile or unify records. Context integration asks how distinct observations relate to the same institutional objects and the same point in time, which relationships make the observations relevant to one another, and what business conclusion follows when they are interpreted together.

The desired outcome is not necessarily a new canonical physical record. It is a coherent operational interpretation. A financial institution can continue to rely on specialised source systems while creating a shared layer through which those systems contribute to a common view of the business situation.

3. What Makes Financial Context Dynamic?

Financial context has two defining characteristics: it is multi-perspective and it changes. The first explains why context must assemble information across systems. The second explains why a static business vocabulary, however precise, cannot by itself represent the institutional situation that AI must operate within.

Some parts of the institutional model are relatively stable. A portfolio can contain positions. A position concerns an instrument. An instrument may have lifecycle events. A trade can alter a position. An agreement can create rights and obligations. These structures establish the kinds of things that exist and the relationships through which financial meaning is organised.

The instantiated world changes continuously. A trade is executed. A position increases. A market move changes valuation and exposure. A lifecycle date approaches. A cashflow becomes expected or due. A settlement fails. A client's liquidity requirement changes. A new mandate arrives. New evidence invalidates an earlier assumption. A previously irrelevant policy threshold becomes binding. The context layer must therefore maintain not only what things are, but what is currently true about them.

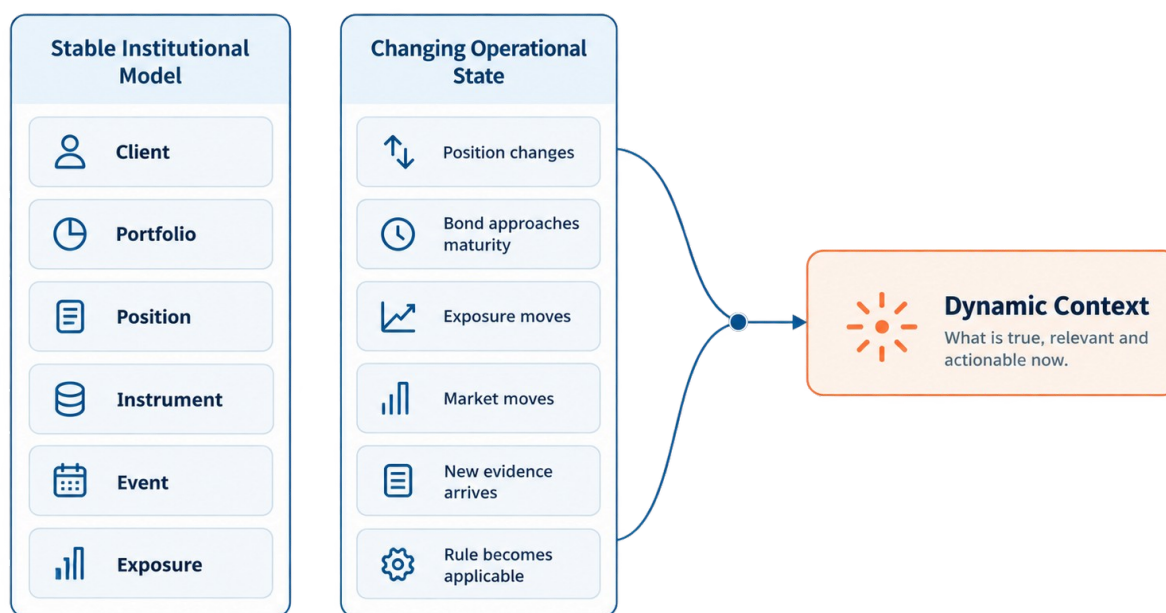


Figure 3. Stable institutional model plus changing operational state. Dynamic context emerges when durable business structure is combined with current state, new events and fresh evidence.

The important distinction is that dynamic does not mean the ontology or institutional model must constantly mutate. The conceptual distinction between Position and Instrument should remain stable even while actual positions, market values and exposures change. What changes rapidly is the context instantiated against that model.

DYNAMIC CONTEXT The model tells AI what can exist. Dynamic context tells AI what is true, relevant and actionable now.

Time is therefore not a peripheral attribute. It is part of the semantics of the operational situation. Two systems can each be correct and still describe different moments. An exposure calculated at 09:00, a hedge executed at 09:05 and a market move at 09:10 need to be aligned before an AI system decides whether the client is under-hedged at 09:15. A useful context layer must make state, event time, effective time and evidence freshness explicit enough for the intended decision.

4. From Semantic Meaning to Operational Context

Formal ontology and semantic modelling provide an important foundation for this architecture. They help distinguish concepts that are often collapsed in data models: a Legal Entity from the roles it plays; an Instrument from a Position; a Trade from the Position it changes; an Event from the State that follows; an Agreement from the obligations it creates. W3C OWL provides a formal language for defining classes, properties and individuals, while FIBO applies ontology to financial concepts and relationships.[2][3]

The operational problem begins where semantic clarification stops. AI must deal not only with kinds of things, but with actual business objects now: this client, this portfolio, this position, this agreement, this exposure, this lifecycle event. It needs to know the current relationships among them, which state currently holds, what changed, what can be derived, which rule applies, what evidence supports the conclusion and what action is available.

This operational interpretation of ontology has precedent in modern enterprise platforms. Palantir, for example, describes not only object and link types but also action types and functions as part of its Ontology, explicitly connecting semantic objects with business logic and organisational action.[1] FINOS CDM likewise demonstrates the importance of representing products, events, legal agreements and processes as related dimensions of a financial domain model.[4] The proposal in this paper is technology-neutral: the important architectural move is to make institutional meaning operational and current.

Semantic modelling foundation	Operational context extension
Defines kinds of things	Represents actual business objects
Clarifies meaning and relationships	Maintains current relationships and state
Describes possible events and states	Tracks events and state transitions
Supports logical inference	Adds calculations, rules and operational derivation
Represents knowledge	Connects knowledge to evidence, permissions and actions

The distinction is not a rejection of classical ontology. It is a layering principle. Semantic modelling gives the business world a stable conceptual structure. The dynamic context layer instantiates that structure with current state, rules, evidence and actions. In short: semantic ontology gives the business world meaning; operational context makes that meaning usable in the flow of business.

5. Anatomy of a Dynamic Context Layer

A dynamic context layer needs to represent more than entities and attributes. Its purpose is to maintain a machine-usable view of the current business situation. That view combines several categories of context that are often distributed across different technologies and organisational functions.

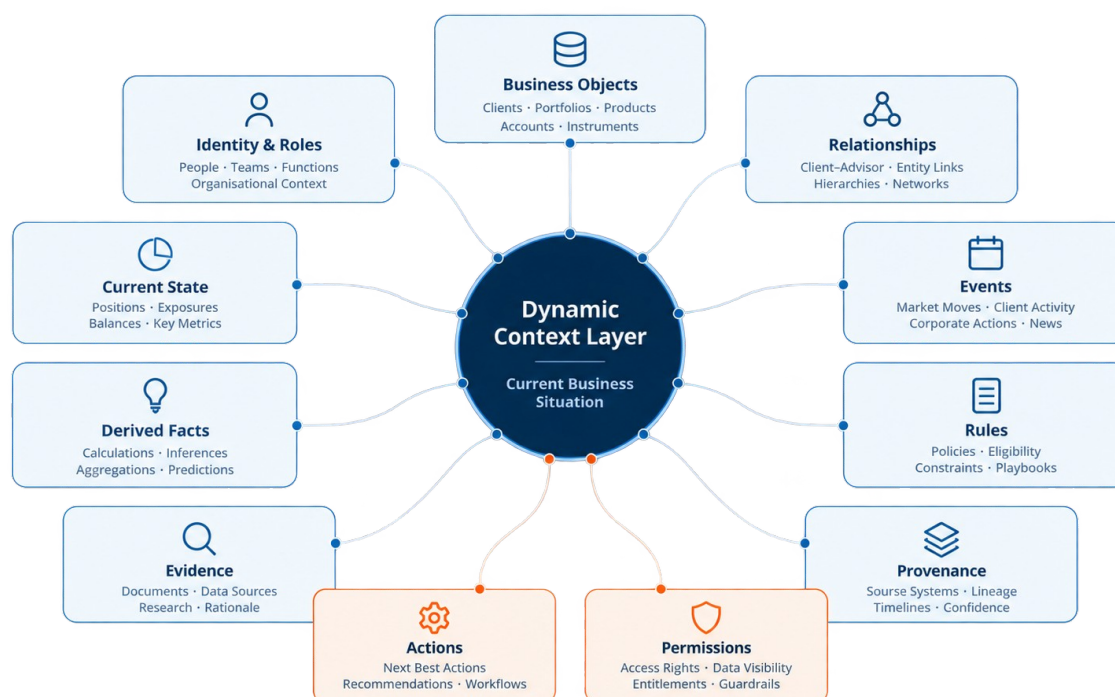


Figure 4. Anatomy of the dynamic context layer. The operational situation combines business objects with roles, relationships, state, events, rules, evidence, provenance, permissions and actions.

Business objects provide the institutional nouns: Client, Legal Entity, Portfolio, Position, Instrument, Trade, Agreement, Cashflow, Exposure and many others. Identity and role distinguish the thing from the capacity in which it participates. Relationships connect objects into a business structure. State captures the current condition of those objects, while events explain what changed and when.

Derived facts are equally important. Much of financial reality is calculated rather than directly recorded: valuation, P&L, exposure, margin, liquidity, concentration and eligibility are examples. Rules add the institution's repeatable interpretation of facts. They may encode policy thresholds, eligibility criteria, escalation logic, mandate constraints or commercial patterns. Evidence and provenance make the context traceable: which system, document, calculation or person supports a conclusion, and how fresh is that evidence?

Actions and permissions turn the layer from a passive knowledge representation into an operational one. Knowing that a settlement has failed is different from knowing which investigation workflow can be started, who is authorised to start it and which controls must be satisfied. Knowing that a reinvestment situation may exist is different from knowing which

client action is appropriate, what product universe is permitted and whether a human must confirm the conclusion before anything is sent to the client.

Not every use case needs every component on day one. The layer should be incremental. A client-opportunity capability may begin with Client, Portfolio, Position, Instrument, Event and a small set of rules. A settlement capability may begin with Trade, Obligation, Settlement Event, Failure State and Investigation. The value comes from allowing those use cases to reuse a shared institutional world rather than creating isolated context logic repeatedly.

6. Model → Bind → Operationalise

The context-layer idea is only useful if it can be delivered without turning into another multi-year enterprise transformation. A practical approach is to keep the delivery model deliberately simple: Model, Bind, Operationalise.

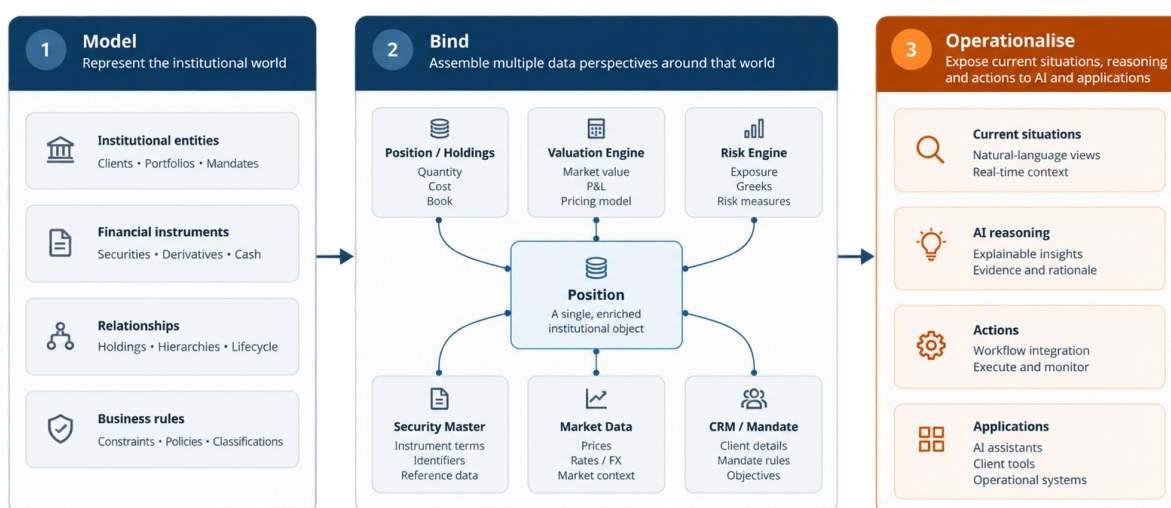


Figure 5. Model → Bind → Operationalise. The delivery pattern starts with the institutional world, binds it to multiple operational perspectives in the existing estate, and then exposes current situations, reasoning and actions.

Model

Model begins with the business world rather than the physical data estate. The question is not which tables exist, but which business objects, roles, relationships, events and states are necessary to understand the problem. The scope should be driven by a business capability rather than a desire to model the entire institution. A client-opportunity use case, for example, may initially require Client, Portfolio, Position, Instrument, Lifecycle Event, Client Need and Opportunity. The model creates a durable layer of institutional meaning above changing source structures.

Bind

Bind is the distinctive middle step. It should not be reduced to mapping an ontology class to a database column. An operational object may receive different aspects of its context from different systems. A Position may take identity and quantity from a position store, instrument terms from a security master, valuation from a valuation engine, exposure from a risk engine, market state from a market-data service and client constraints from CRM or mandate data.

Binding therefore establishes how multiple perspectives attach to a shared institutional object. It may include identity resolution, relationship construction, temporal alignment, calculations, source authority, evidence links and refresh behaviour. A source can remain authoritative for a fact while the context layer determines how that fact contributes to the wider business situation.

Operationalise

Operationalise makes the resulting context available to the capabilities that need it. That may include AI agents, analytical applications, graph queries, rule services, controls, workflows or human-facing applications. A good test is whether the layer can answer operational questions: Which clients are affected by this event? What changed since yesterday? Which rule generated this conclusion? Which evidence supports it? Which action is permitted now?

DELIVERY PRINCIPLE Keep the data estate. Add the missing context.

This approach deliberately avoids making the context layer another data platform. Trading systems should continue to trade; risk engines should continue to calculate risk; warehouses and lakehouses should continue to provide scalable storage and analytics. The context layer adds the shared operational interpretation that those systems do not individually provide.

7. Three Financial Situations

The architecture becomes clearer when the same pattern is applied across revenue, risk and operations. The details differ, but the underlying logic is consistent: several source perspectives are assembled into a current situation, institutional logic interprets that situation, and an action becomes possible.

Client opportunity: a maturing bond

The position store establishes that a client portfolio holds a bond. The security master supplies the maturity terms. A cashflow engine derives expected proceeds. The portfolio system supplies allocation and duration context. A liquidity forecast identifies near-term cash requirements. The trading system indicates whether a replacement trade has already been executed. A mandate limits the eligible response. Together, these perspectives may support a conclusion that some of the maturity proceeds are potentially available for reinvestment. The maturity event itself is not the opportunity; the opportunity is an interpretation of the wider client situation.

Risk management: an evolving FX exposure

A risk engine calculates a USD exposure. Treasury data identifies existing hedges. Transactions show recent hedge activity. A client policy or mandate describes target coverage. Market data supplies current rates and volatility. Relationship information may record the client's stated intention. No individual source can establish whether the client is currently under-hedged, whether the residual exposure is intentional or whether a new action is relevant. The conclusion belongs to the context assembled across them.

Operations: a settlement failure

A trading platform identifies the trade. Settlement infrastructure records the expected obligation and settlement date. Custody and payment systems provide actual operational state. Reference data identifies securities, counterparties and accounts. Business rules determine when a delay becomes a failure and when escalation is required. The useful context is not simply that a status field contains FAILED. It is which obligation failed, what state transition did not occur, what downstream consequences are possible, which rule now applies and which investigation action is available.

These examples show why dynamic context is institutional infrastructure rather than a feature tied to one AI use case. The same objects and relationships can be reused across commercial, risk, treasury, operational and control applications. A Position understood in client opportunity analysis should be the same institutional Position used by risk and operations, even though each capability contributes a different perspective.

8. Dynamic Context for Financial AI Agents

Agentic AI raises the importance of dynamic context because an agent is expected not only to retrieve information but increasingly to interpret situations and take or propose actions. Tool access by itself does not make an agent institution-aware. An agent can have SQL access, APIs, document retrieval and market-data tools while still misunderstanding the object it is dealing with, the relationship that makes a fact relevant or the rule that constrains an action.

Before acting, a financial-services agent may need to establish several things at once: the identity of the relevant business objects, the roles parties are playing, the current state of those objects, the event that caused a change, which conclusions are observed versus derived, which institutional rules are applicable, what evidence supports the conclusion, and what actions the agent is authorised to perform.

This leads to a useful distinction between data-aware AI and situation-aware AI. Data-aware AI asks what information it can access. Situation-aware AI asks what is happening in the institution now, why it matters and what can be done. The dynamic context layer is the architectural mechanism that makes the second question tractable without forcing every agent to reconstruct the institution independently from raw records and documents.

This also creates a better division of responsibility between AI and deterministic institutional logic. The context layer can make business objects, rules, calculations and evidence explicit. AI can then be used where interpretation, synthesis, exploration or natural-language interaction adds value, rather than asking the model to rediscover stable institutional structure or invent rules that should have been governed outside the model.

As agents become more autonomous, this separation becomes a control mechanism. The agent can reason over a context that already carries provenance, rule conclusions and permissions. It can explain why an action is proposed, and the institution can distinguish the AI interpretation from the deterministic facts and rules beneath it.

9. Evidence, Change and Trust

A dynamic context layer should not present every statement as the same kind of truth. Financial institutions work with multiple epistemic categories: an observed contractual term, a derived calculation, a business-rule conclusion, a quantitative model estimate, an AI interpretation and a human judgement are all different. Preserving those distinctions is critical to explainability and control.

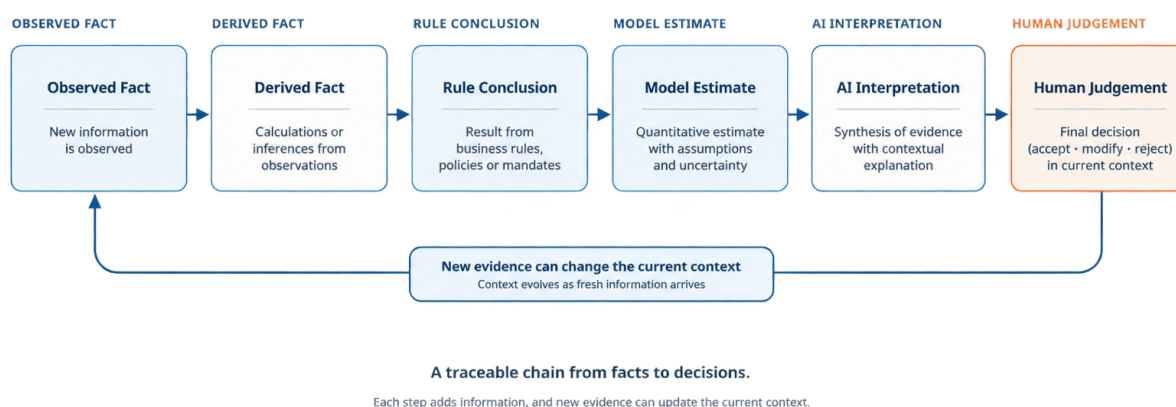


Figure 6. A traceable chain from facts to decisions. Observed facts can support derived facts, rule conclusions, model estimates and AI interpretations, while new evidence can update the context before or after human judgement.

Consider again the reinvestment example. The holding and maturity date may be observed facts. Expected proceeds may be derived from contractual terms. A rule may conclude that a portion of those proceeds is potentially investable after known liquidity needs. An AI system may interpret recent client commentary as consistent with continued fixed-income allocation. A relationship manager may then learn that the client needs the proceeds for an unplanned

redemption. The new evidence changes the situation and invalidates the earlier commercial conclusion.

This is another sense in which context is dynamic. The world does not only change because transactions and markets move. The institution's knowledge of the world also changes. A context layer should therefore carry not only conclusions but the evidence path behind them: where an observation came from, when it was valid, how a derived fact was calculated, which rule version was used, which assumptions were made and which human judgement modified the result.

TRUST PRINCIPLE Context should not merely record what the institution currently believes. It should retain why it believes it and allow new evidence to change that conclusion.

For regulated and high-stakes financial activity, this traceability is more than an explainability feature. It supports investigation, audit, model governance, control testing and responsible human oversight. It also makes AI outputs easier to challenge: a user can inspect the chain, dispute an assumption or add evidence rather than treating the model's answer as an opaque final truth.

10. Toward Context-Native Financial Services

Financial institutions have already made enormous investments in systems of record. They are now making a new wave of investment in AI. The missing architectural question is how those systems and models can share an operational understanding of the institution rather than connecting each new AI application directly to a collection of fragmented sources.

A dynamic context layer offers one answer. It treats the existing data estate as a set of complementary perspectives on a shared institutional world. It assembles those perspectives around explicit business objects, relationships, state, events, rules and evidence. It then exposes the resulting situations to AI, analytics, applications and human decision-makers.

The approach does not require a complete enterprise ontology before value can be delivered. It can start with a narrow business capability and expand. A client-opportunity use case establishes a set of core client, portfolio, position, instrument and event concepts. A risk capability can extend that context with exposure, hedge and policy. A settlement capability can add obligations, settlement events and failure states. Each new capability reuses and enriches the same institutional world instead of creating another isolated semantic island.

Over time, this changes the AI architecture of the institution. Instead of individual agents independently reconstructing context from tables and documents, applications can share a progressively richer operational model of the business. The institution moves from data-aware

AI toward situation-aware AI: systems that understand not only what information exists, but what is happening now, why it matters and what actions are appropriate.

The practical discipline remains deliberately simple. Model the institutional world. Bind it to the multiple perspectives already present in the data estate. Operationalise the resulting context. This is not a call to rebuild the financial institution's systems. It is a call to connect them through the missing layer that allows data and AI to understand one another.

CLOSING PROPOSITION Model the institutional world. Bind it to existing data perspectives. Operationalise the resulting dynamic context.

References

- [1] **Palantir Technologies**. Ontology overview. <https://www.palantir.com/docs/foundry/ontology/overview> Describes object/link types together with action types and functions in the operational Ontology.
- [2] **World Wide Web Consortium (W3C)**. OWL 2 Web Ontology Language Document Overview, Second Edition. <https://www.w3.org/TR/owl-overview/> W3C Recommendation, 11 December 2012.
- [3] **EDM Council**. Financial Industry Business Ontology (FIBO). <https://edmcouncil.org/financial-industry-business-ontology/> Financial-services ontology covering financial concepts and relationships.
- [4] **FINOS**. The Common Domain Model. <https://cdm.finos.org/docs/common-domain-model/> Current documentation describes product, event, legal-agreement, process, reference-data and namespace modelling dimensions.

About Finsight Technologies

Finsight Technologies focuses on financial-services context engineering: modelling institutional business meaning, binding it to existing data estates, and operationalising that context for AI, analytics and business workflows. The approach is designed to complement existing financial data platforms rather than replace them.

finsight-tech.com