



From Pipelines to Meaning:

Building the Semantic Backbone for AI-Ready Data Platforms

By Bill Inmon and Jamie Knowles

From Pipelines to Meaning: Building the Semantic Backbone for AI-Ready Data Platforms

By Bill Inmon and Jamie Knowles

Overview

For as long as most of us have been building data platforms, there's been this idea of a "single version of the truth." It shows up in strategy decks, architecture diagrams, and just about every data transformation programme. And yet, if we're honest, most modern platforms have drifted further away from it, not closer.

A modern data platform is the environment organisations use to collect, integrate, govern, analyse, and increasingly operationalise data across the business. It supports reporting, analytics, operational decision-making, and now AI-driven interactions. These platforms sit at the centre of how organisations understand customers, operations, finance, products, and risk. When meaning becomes inconsistent across that environment, the impact spreads quickly across both human and AI-driven processes.

The problem isn't that we don't have enough data. Quite the opposite. We have more than we know what to do with. The real issue is that we've lost a shared understanding of what that data actually means.

Over time, as architectures have become more distributed and pipeline-driven, integration has become something we do locally, for a specific use case, and then move on. Meaning gets buried in transformation logic, SQL, and code. It works ... for a while. Until it doesn't. Definitions start to drift, the same term means slightly different things in different places, and nobody is quite sure where the "real" version lives anymore. It's subtle at first, then it compounds. We refer to this as semantic entropy.

For years, this was manageable. Humans could compensate. Analysts would spot inconsistencies, apply context, and make judgement calls. But Artificial Intelligence (AI) doesn't work like that. It takes what it's given at face value. If meaning is inconsistent, AI doesn't fix it, it scales it. Fast.

So this stops being a theoretical concern and becomes a practical one.

This paper sets out a way to get back on track. The core idea is simple: you need a shared, explicit, and governed understanding of business meaning that sits above pipelines and systems. We describe this as a semantic backbone, built on an Enterprise Logical Data Model (ELDM). From there, everything else becomes more consistent, from governance to analytics to AI. This semantic backbone is realized through enterprise data modeling practices, which allow organizations to define, govern, and share business meaning consistently across systems.

The Mission: Defining Truth in the Modern Enterprise

For a long time, data platforms have been built on a fairly simple premise: find data, move it, clean it up, present it, and the business will figure out the rest. That worked when humans were doing the interpretation. People would take the requirements, locate the right source data, design the pipelines and structures, and ultimately deliver reports that reflected the context they had in mind. These pipelines, the automated flows that move, transform, and integrate data between systems, became the primary mechanism through which modern platforms were built. Outside of that context, though, the data moving through those pipelines often made far less sense on its own.

AI doesn't do that.

AI takes data at face value. It assumes the definitions are consistent, the relationships are correct, and the meaning is clear. When that isn't true, the results don't just degrade; they go wrong in ways that look believable. And they scale.

That's where most modern data platforms start to struggle.

We've spent the last decade optimising for speed. Pipelines are faster, tooling is more advanced, teams can deliver in days what used to take months. But in the process, we've quietly let meaning drift. Definitions live in code, integration happens locally, and the same business concept gets reinterpreted over and over again depending on the use case.

It works, up to a point. Then you try to layer AI on top of it.

The idea of a "single version of the truth" is still the right goal, but it's often misunderstood. Truth isn't the data itself. It's the meaning behind it. Without a shared understanding of what things like "customer" or "revenue" actually represent, you don't have one version of the truth; you have several, all slightly different.

The mission of the modern data platform needs to shift. It is to provide a body of knowledge that reflects how the business actually operates, expressed in language that is clear, precise, unambiguous, and defined in terms that are unique to that business. Those definitions cannot be inferred by systems alone; they must be established, agreed upon, and maintained by the people who understand the business best.

It's no longer enough to move and transform data efficiently. You have to define and maintain meaning, explicitly and consistently, across the enterprise. That definition can't live inside pipelines or individual systems. It has to sit above them, guiding how data is structured, integrated, and used. It requires a structured modeling approach where business concepts, relationships and rules are defined once and reused across systems. This is where enterprise data modeling platforms play a critical role.

Foundations: The Original Thinking of Kimball and Inmon

To understand how we arrived here, it is worth revisiting the original thinking behind enterprise data architecture.

Two of the most influential approaches to modern data warehousing were introduced by Bill Inmon and Ralph Kimball. While their methodologies differed in implementation style, both recognised the importance of aligning data to the business in a structured and consistent way.

Kimball focused on delivering business-facing analytical models quickly through dimensional structures such as star schemas, often built incrementally around reporting use cases. Inmon emphasised enterprise-wide integration through an enterprise data warehouse and enterprise data model, recasting operational data into a coherent corporate form before downstream consumption.

The following section outlines the foundational principles that shaped how organisations approached data for decades, and why those principles remain directly relevant today.

In the early days of technology, finding data and using data was simple. Technology was simple, and data was found in an obvious and predictable manner. There were magnetic tape-based master files. There were simple databases. There were reports and punched card input. The world was small enough and organized so that finding and using data was an easy thing to do.

But these simple systems soon turned into a polyglot of business applications, each designed to support a different operational area of the corporation and each generating its own data. There grew to be applications everywhere: transaction processing applications, human resource applications, finance applications, engineering applications, and many more.

Soon the computer began to pop up everywhere in the corporation.

And with this explosion of applications came challenges in finding and understanding data. It wasn't that data couldn't be found. It was that data existed in so many places, often with different definitions and values, that once you found it, you could no longer be certain which version was correct.

And there were many other problems associated with the usage of data in the corporation.

The most significant challenge was the arrival of different kinds of data. In the beginning there was structured data – data stored in clearly defined formats such as rows and columns within business systems and databases, where the meaning and structure were largely understood by the systems operating on it. But over time there started to appear other kinds of data – data from the Internet, documents, spreadsheets, emails, conversations, and so forth. And this new form of textual and unstructured data brought with it many challenges because it is very unlike structured data.

When blended together, this collection of data turned into the Tower of Babel. There was no shortage of data. But there was a real shortage of the understandability and usability of the data.

Trying to make sense of the Tower of Babel of data was a real challenge.

Trying to make sense of the data was

- Complicated
- Expensive
- Time consuming
- Confusing



**Trying to make sense
of anything was a
real challenge**

The growing challenge of managing data could not be solved effectively by simply adding more consultants or more technology. In fact, more applications, more consultants, and more technology only exaggerated the already difficult challenge of using data. Instead, what was required was a change in the architectural approach to the management of data.

What was needed was a data architecture that accounted for all of the data and all of the types of data.



**An architecture is what is
needed in order to unravel
the mess**

So what is data architecture? At its core, data architecture is the discipline of organising, integrating, and structuring corporate data so that it can be understood, trusted, and used consistently across the enterprise. What does a modern rendition of data architecture do for the corporation?

The objectives of a data architecture include:

- The accessibility of data. In order to be useful the data needs to be accessible to whoever needs to use it
- The data architecture needs to be comprehensive. All the data that needs to be used in the organization needs to be accounted for
- Data in the architecture needs to be relatable. The person using the data needs to be able to examine one element of data and determine how that data relates to another element of data. And the relationship of data must be made with confidence.
- Data in the architecture must be accurate. Incorrect data in the architecture is like sand in the oil in your car. It is corrosive and destructive.
- Affordability is always an issue. It does no good for the corporation to build an architecture that does not bring more value than it costs.

These are some of the more salient aspects of a data architecture.

What then are some of the typical constructs found in a data architecture? They include:

- A data model. The data model is a blueprint for how the data is organized.
- Names and definitions of data need to be clearly articulated and documented in a public manner
- Relationships between elements of data need to be clearly documented
- The lineage of data must be identified, showing the source of data and its transformations along the way

Not only do these aspects of data need to be collected and documented, they need to be maintained over time as well. The organization and its business are constantly changing. And the data of the corporation changes as the business of the organization changes.

At its core, these constructs are about establishing a shared and consistent understanding of data across the enterprise.

The data architecture itself is a complex structure. There are challenges to the building and maintenance of the data architecture. Some of these challenges include maintaining the architecture and achieving consensus on definitions.

The blending of textual and structured data is its own large complex subject. Structured data has keys, attributes, and records. Textual data does not. Blending text and structured data is a difficult subject. Modern AI can help with this, but only when data is reasoned against the same set of definitions.

So you wake up one day and have a data architecture. What exactly does a data architecture do for you and your organization?

With a modern and properly constructed data architecture, you can

- Find data easily and with confidence
- Understand what the data means once you have found it

- Connect the many elements of data encompassed by the architecture
- Examine data over time
- Add to the architecture in a seamless manner
- Resolve conflicts

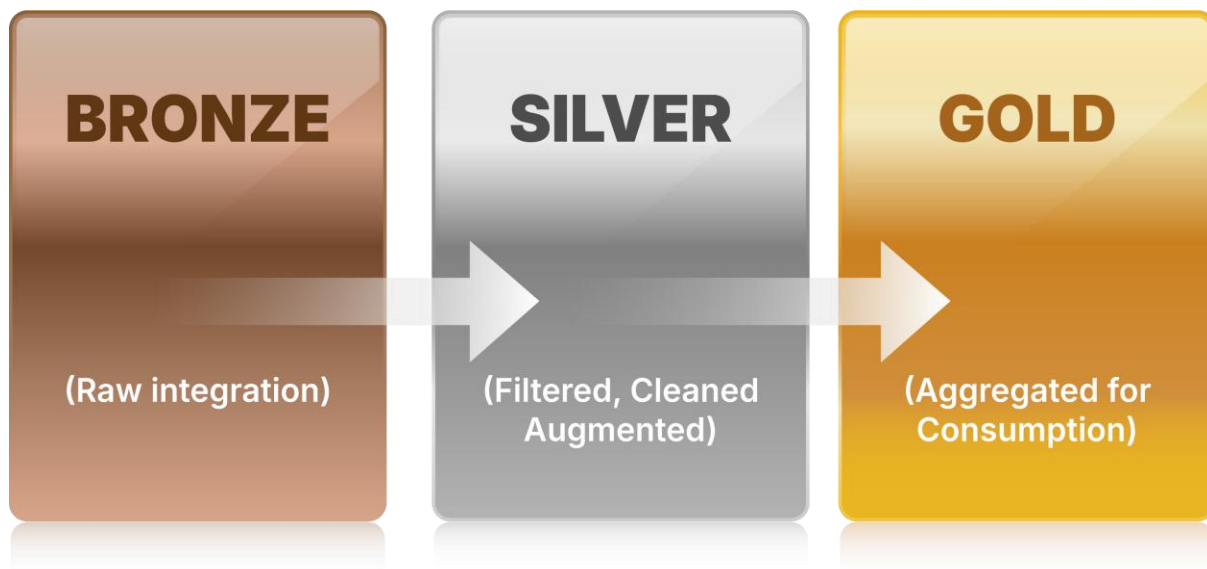
These principles remain unchanged. What has changed is the environment in which they must operate.

Modern platforms have increased in scale and speed, but the need to define, integrate, and govern meaning has not diminished. If anything, it has become more critical.

This is where the gap begins to emerge.

The Medallion Pattern

Most modern data platforms are built around what is commonly referred to as the Medallion pattern. It's a simple and practical way of organising data into layers, typically described as Bronze, Silver, and Gold. It is most commonly associated with modern lakehouse platforms, which are increasingly being used as the primary analytical environment, although many organisations still combine them with traditional warehouses.



At its core, the goal of the Medallion approach is to bring structure and control to increasingly complex data environments. It gives teams a way to land data quickly, improve it in stages, and serve different use cases without constantly reworking the same pipelines. In a world of growing data volumes, cloud platforms, and distributed teams, it provides a simple, repeatable pattern that balances speed with a degree of order.

The idea is straightforward. The Bronze layer captures raw data from source systems, usually with minimal transformation, often stored as immutable or append-only data to preserve history. The Silver layer refines that data by applying validation, cleansing, standardisation, deduplication, and some level

of integration between related source systems. The Gold layer then shapes it into structures designed for consumption, often aligned to specific reporting, analytical, or business use cases.

It is worth noting that the Medallion pattern is often treated as a practical structuring approach rather than a complete architecture in its own right. The underlying architectural considerations that have long guided data warehousing, particularly around enterprise integration, shared business definitions, historical consistency, governance, and the recasting of source data into a coherent corporate form, still apply regardless of platform.

The implementer must still make architectural decisions within the pattern itself. In many implementations, the Silver layer remains relatively close to source-oriented structures, while the Gold layer adopts dimensional models such as well-formed star schemas for analytical consumption. Other approaches may use the Silver layer as an integrated enterprise foundation aligned to a broader semantic or enterprise data model.

More formal architectural approaches, including enterprise data warehouse methodologies and Data Vault, can be applied within the Medallion pattern to introduce stronger integration, governance, and semantic consistency across the platform.

The appeal of this pattern is clear. It fits well with cloud platforms, scales effectively, and supports distributed engineering teams working in parallel. It also enables incremental delivery. Teams can land data quickly, improve it over time, and deliver value without waiting for a fully integrated design upfront.

For data engineering teams, this has been a significant step forward. It increases velocity, reduces bottlenecks, and allows organisations to respond more quickly to changing requirements.

In practice, many modern lakehouse implementations are assuming the role that enterprise data warehouses were always intended to serve: the integrated analytical foundation of the enterprise. The technologies may have changed, but the fundamental architectural challenges around integration, consistency, governance, and meaning have not.

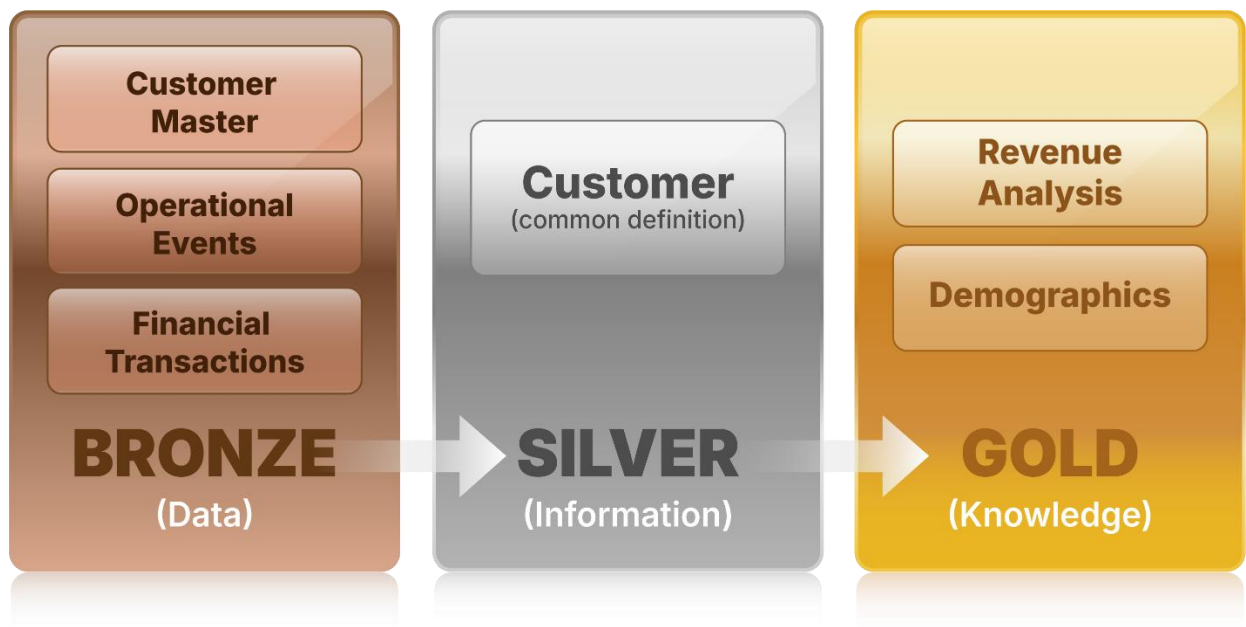
The Modern Reality: Speed Over Meaning and the Rise of Semantic Entropy

In reality, most data platforms are more complex and messy than the clean, layered diagrams make them appear.

Pipelines are often built directly from source systems to meet immediate needs. The Silver layer, rather than becoming an integrated view of the business, stays largely shaped by those sources, just cleaner and easier to work with. The Gold layer then evolves into a collection of use-case specific models, each designed to answer a particular question at a particular point in time.

Individually, these pieces work. Collectively, they give the impression of an integrated platform. But that integration is often superficial. It's a network of pipelines that looks connected, but underneath, each part is still carrying its own version of meaning.

Integration, in this model, is local and temporary. It happens within a pipeline, for a specific purpose, and rarely extends beyond it. Definitions are embedded in transformation logic, SQL, and code, rather than defined once and reused. Over time, the same business concepts are implemented slightly differently across different pipelines, depending on who built them and why.



There is no persistent, shared semantic structure that defines and maintains meaning consistently across the platform over time.

This is how semantic entropy takes hold, not as a single failure, but as a gradual drift. A field means one thing here, something slightly different there. A calculation gets copied and adjusted. A definition evolves in one place but not in another. Nothing breaks immediately, but consistency erodes over time.

In practice, this creates two very different architectures. One where data remains aligned to source systems, and meaning is fragmented across pipelines. And one where data is aligned to a shared semantic model, and meaning is consistent across the platform.

Most organisations believe they have the second. In reality, they are operating the first.

This isn't a failure of intent. The industry didn't set out to ignore enterprise integration. In many cases, it was simply deferred, fragmented, or handled implicitly within pipelines rather than defined explicitly at the enterprise level.

Semantic Entropy

Semantic entropy is the gradual loss of shared meaning across a data platform and the organisation around it.

Definitions start to drift, the same term means slightly different things in different places, and nobody is quite sure where the "real" version lives anymore. Over time, this semantic drift accumulates into a broader breakdown of shared meaning across the platform.

It doesn't happen all at once. It builds over time. A definition is interpreted slightly differently in a new pipeline. A field is reused but repurposed. A calculation is copied, then adjusted to meet a specific need. Each change makes sense in isolation. Together, they start to pull the platform apart.

At first, it's barely noticeable. Reports still run. Dashboards still look plausible. But the same terms, "customer", "revenue", "order", no longer mean exactly the same thing everywhere they appear. The differences are subtle, but they matter.

This drift isn't just technical, it's organisational. Different teams naturally develop their own interpretations. Data governance teams define business terms. Data architects and engineers shape those concepts into models and pipelines. BI teams and analysts adapt them again to answer specific questions. AI layers then consume whatever definitions they are given. Each step introduces small variations.

This is the nature of semantic entropy. Meaning drifts.

Modern, decentralised data architectures tend to accelerate this. Multiple teams working independently, building for different use cases, each making reasonable decisions in their own context. Without a shared, persistent definition of core business concepts, divergence is almost inevitable.

For years, this was manageable. People could spot inconsistencies and work around them. But as platforms grow, and as AI becomes a consumer of that data, the effects compound quickly.

Semantic entropy is not just an inconvenience. It is a loss of trust in the data itself.

This challenge is often amplified across business units. Different parts of the organisation use different language to describe similar, or sometimes identical, concepts. What one team calls a “customer” may not align with how another defines it. In many cases, this leads to the conclusion that a single shared model is unrealistic. In practice, it means each domain is operating with its own version of meaning.

The Breaking Point: AI Exposes the Cracks

For years, these issues were manageable. Data platforms weren’t perfect, but they were usable. Analysts and engineers understood the quirks, knew where the edge cases were, and could apply judgment when something didn’t quite line up.

In many cases, a single person or small team would effectively own the entire flow. They would take the requirements from the stakeholder, identify and source the relevant data, understand the context behind it, and then design the pipeline end-to-end. Along the way, they would make a series of decisions about definitions, relationships, and transformations, all based on a clear understanding in their own head of what the data meant and how it should answer the question. The final report would be well defined for its purpose, but much of that meaning lived in the pipeline and in the minds of the people who built it. Outside of that context, the data was often difficult to reuse or even fully understand.

AI changes that completely.

AI does not interpret data the way humans do. It does not question definitions or reconcile inconsistencies. It takes what it is given and attempts to reason over it. If the underlying meaning is unclear or inconsistent, the results are not just inaccurate; they are confidently wrong. Rather than recognizing uncertainty or admitting the answer may be flawed, AI often presents incorrect conclusions with complete confidence, creating what we call AI hallucinations.

That might be workable for human-led analytics, where context can be inferred, and inconsistencies can be worked around. For AI, it is far more problematic.

An AI system cannot guess which definition applies. It has to be told. If different parts of the organisation genuinely operate with different definitions, then each must be explicitly defined and structured. Whether that results in a single unifying ontology or multiple clearly defined ones, the requirement is the same: meaning must be made explicit. Without that, AI has no reliable foundation to reason from.

This is where the cracks in modern data platforms become visible.

Generative AI is only as effective as the data it operates on. Poor quality and fragmented data inevitably produce unreliable results. But even technically correct data can fail when the meaning behind it is inconsistent. AI exposes both problems immediately.

A question as simple as “How many customers do we have?” can produce multiple valid answers, depending on which pipeline or definition is used. “Revenue” may be calculated differently across reports. For a human, these differences can be spotted and adjusted for. For AI, they are treated as fact.

And AI scales those inconsistencies.

What was once a contained issue, limited to a report or a team, becomes systemic. Errors propagate across use cases. Insights diverge. Trust erodes more quickly, and in ways that are harder to detect.

What worked for dashboards, where humans could compensate for gaps in meaning, breaks down when applied to automated reasoning.

AI doesn’t create the problem. It exposes it.

Precision Matters: Customer and Revenue Example

It often starts with a simple request:

“Get me the revenue figures for all our customers.”

On the surface, this feels straightforward. The terms are familiar. Everyone in the organisation uses them. There is an assumption that “customer” and “revenue” are well understood.

But as soon as you pause and ask what those terms actually mean, the ambiguity becomes clear. An AI or human answering this question is likely to give completely different answers. Let’s look at why.

What do we mean by customer?

1. Is it B2B or B2C?
2. Is it the bill-to, sold-to, or end user?
3. Does it include only active customers, or also those who have churned?
4. Is it defined by the legal entity or operating unit?

And what do we mean by revenue?

1. Is it booked, billed, or recognised?
2. Is it calculated at a point in time or over a period?
3. Is it based on invoices, contracts, or cash received?
4. Is it reported in transaction currency or consolidated?

Then things get more complex. The calculation for Booked Revenue for B2B customers is fundamentally different from that for Booked Revenue for B2C customers. The term is the same. The meaning is not.

“Get me the revenue figures for all our customers”

Customer	Revenue
What do you mean by customer ? • Type: B2C, B2B • Role: Bill-to, Sold-to, End user • Status: Contracted, Invoiced, Active • Location: Legal entity vs operating country	What do you mean by revenue ? • Type: Booked, Billed, Recognized, Cash • Timing: Point-in-time vs over period • Allocation: Full amount vs prorated / shared • Currency: Transaction vs reporting
Revenue = ? B2C: <u>Sum(transaction_amount at sale date)</u> B2B: <u>Recognised revenue over contract term</u>	

Without explicit structure, each answer reflects a different set of assumptions. In a human-driven process, those assumptions are often made implicitly. The analyst chooses what they believe is the correct interpretation based on their understanding of the business and the context of the request. The report is correct for its purpose.

But the meaning is not explicit, and it is not shared.

To handle this properly, you need to move beyond general terms and make that variation visible and structured. Humans and AI components must be aligned on definitions if they are to interpret the question in the same way.

The Role of the Logical Data Model

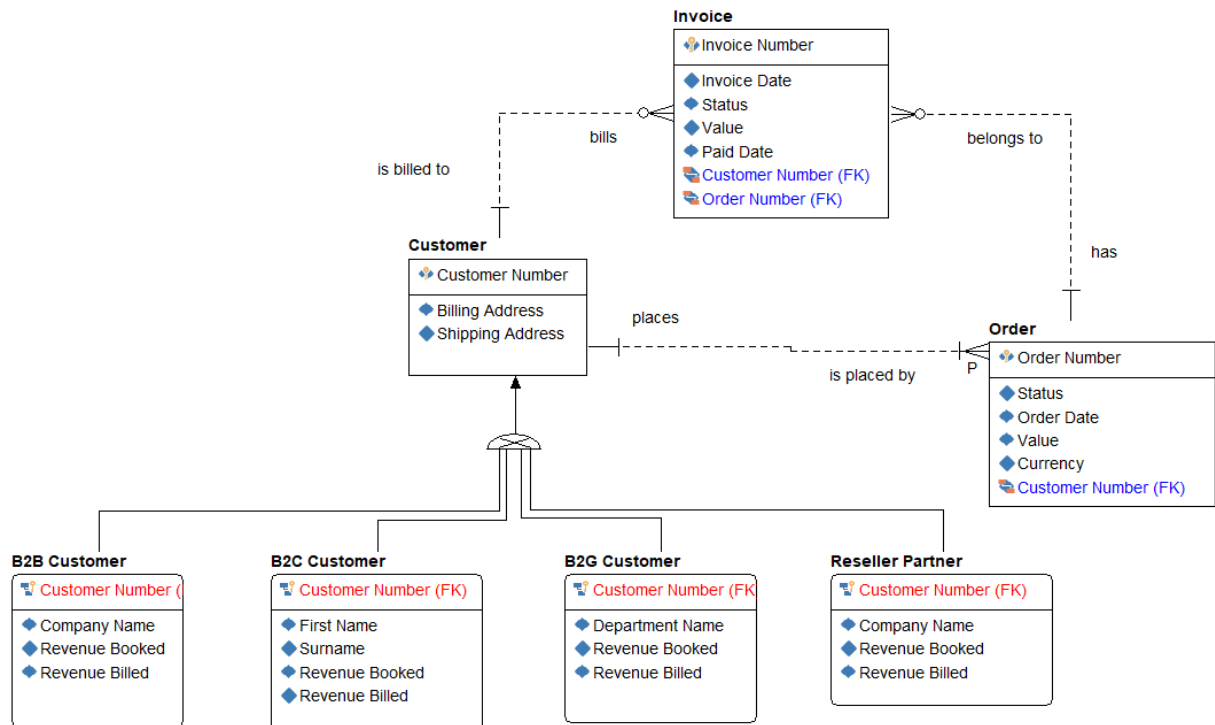
This is the role the Enterprise Logical Data Model (ELDM) was always intended to play.

An ELDM describes the business itself: the core concepts, relationships, rules, and meanings that exist independently of any specific technology or implementation. It captures how the organisation understands customers, products, revenue, orders, policies, and other business concepts in a consistent and structured way.

This differs from a physical model, which describes how data is technically implemented within a particular platform, database, warehouse, or application. Physical models are optimised for storage, performance, scalability, and operational requirements. Logical models are optimised for meaning. Historically, logical models were primarily used by architects and designers during system and warehouse development. With AI, their role becomes much broader. They provide the semantic foundation that allows both humans and machines to interpret data consistently across systems, platforms, and analytical use cases.

An ELDM provides that structure. It distinguishes between different types of customer, links them to the relevant transactions, and defines how different forms of revenue are derived. It makes explicit what was previously held in the minds of individuals or embedded within pipelines.

In the example below, we can clearly see that there are different types of customer and different forms of revenue associated with each. If we were to drill further into the model, we would see how they differ through their attributes, relationships, and business rules. The ELDM provides a precise framework for defining these concepts and their variations explicitly and consistently.



This problem is not theoretical. In many organisations, different departments legitimately produce different answers to the same question because they operate from different definitions, rules, and business contexts. Each answer may be internally correct within its own context, while still being inconsistent at the enterprise level.

The Cost of Semantic Entropy

Semantic entropy is not just a technical problem. It creates operational and financial friction across the enterprise.

In many organisations, the same business concepts are repeatedly redefined across pipelines, reports, data products, and AI initiatives. Teams build local interpretations of customers, products, revenue, risk, or contracts to meet immediate needs. Over time, this creates duplicated transformation logic, inconsistent metrics, and growing reconciliation effort between teams.

The cost is often hidden because the platform still appears to function. Reports are delivered. Dashboards are populated. AI prototypes produce answers. But underneath, organisations are paying a continuous tax on inconsistency.

Analysts spend time reconciling numbers rather than analysing them. Data engineering teams repeatedly implement the same business logic in different pipelines. Governance teams struggle to maintain alignment between glossaries, catalogs, and operational systems. Business leaders make decisions based on reports that may appear authoritative but are built on inconsistent definitions. AI initiatives slow down because teams cannot establish a trusted semantic foundation for reasoning.

As platforms scale, these issues compound. What begins as local inconsistency becomes enterprise-wide semantic fragmentation.

For AI, the impact is even more significant. AI systems reason over the meaning they are given. If definitions are inconsistent, incomplete, or conflicting, AI scales those inconsistencies into unreliable outputs, hallucinated relationships, and contradictory answers. The problem is not the model itself. The problem is the semantic foundation beneath it.

Ontology: The Missing Engineering Layer of Data

Modern data platforms are built from well-understood architectural layers. Whether implemented through Medallion patterns, enterprise data warehouses, Data Vault, or other approaches, organisations now have mature technologies for storage, processing, integration, and analytical access. These layers are engineered, automated, and scalable.

What is still largely missing is a dedicated layer for meaning.

Today, meaning is handled indirectly. It lives in business glossaries, documentation, and, most often, in code. It is interpreted by people and embedded in pipelines, rather than defined and managed as a first-class part of the architecture.

This is why consistency is difficult to achieve.

Ontology addresses this gap.

An ontology provides a structured way to represent meaning, the concepts of the business, how they relate, and the rules that govern them. It is not an abstract idea. It is an engineering layer, just like the others, but focused on meaning rather than data movement or storage.

In practice, this is what data architects have been doing for years through logical data modelling. The difference now is that the model can no longer remain a design-time artefact used only by humans. It must become an explicit and operational semantic asset, connected to governance processes, analytical structures, and AI systems through machine-readable representations that allow meaning to be shared and reused consistently across the platform.

Once it is, everything else becomes more coherent. Integration is based on defined concepts, not assumptions. Governance is grounded in structure, not just terminology. AI has a model it can reason over, rather than patterns it must infer.

The platform moves from processing data to understanding it.

The concept of defining business meaning through structured models has been at the core of enterprise data architecture for decades. What is now being described as ontology is closely aligned with the role that enterprise logical data models have always played, providing a structured, integrated view of the business and the relationships within it.

The difference today is not in the principle, but in how critical it has become. What was once a discipline used to support integration and consistency is now a foundational requirement for analytics, governance, and AI.

The challenge is not understanding this concept. It is operationalising it across the enterprise.

Why Business Glossaries Fail

Given the level of ambiguity in even simple business questions, the natural response is to define terms more clearly. This is where business glossaries come in.

Most organisations have invested in them. They define terms like “customer” and “revenue,” assign owners, and establish policies around their use. This is an important step forward. It creates a shared vocabulary and brings a level of governance that is often missing in purely technical implementations.

It seems that the business glossary is the obvious place to centre the semantic backbone of the organization. But it is not enough. Most business glossaries are not true ontologies and lack the semantic structure needed to represent meaning, relationships, and context consistently across the enterprise.

A glossary can define a term such as “Booked Revenue.” It can describe it, assign ownership, and attach policies or rules. On the surface, this looks like a complete solution.

The problem is that the definition exists in isolation.

Glossaries also have to deal with the reality of language itself. The same concept may appear under different names, “revenue”, “sales”, “turnover”. The same term may be used to mean different things in different parts of the business. Even simple issues such as spelling variations or inconsistent naming can introduce confusion.

Most glossary tools attempt to manage this through synonyms, aliases, and controlled vocabularies. This is useful. It helps standardise language and reduce obvious duplication.

But it does not solve the deeper problem. Standardising the label does not standardise the meaning.

As we saw earlier, “Booked Revenue” is not a single, uniform concept. The way it is calculated for a B2B customer may be fundamentally different from how it is calculated for a B2C customer. The timing, the source data, and the business rules can all vary.

A glossary entry cannot capture that variation in any meaningful way.

It does not describe:

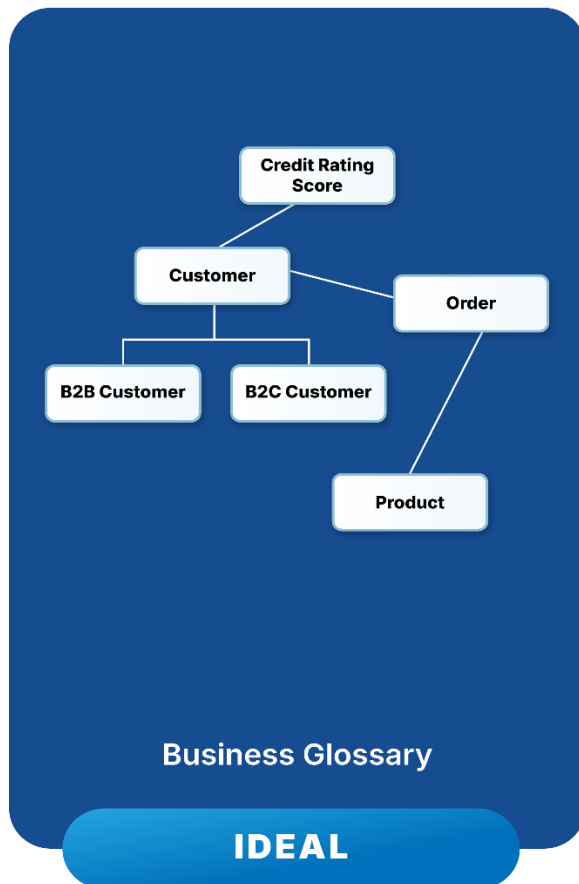
- how the concept relates to different types of customer
- how it is derived from underlying transactions
- how those relationships change across contexts
- or how the rules are applied in practice

At best, these details are captured informally in documentation. More often, they are embedded in code, pipelines, and reports.

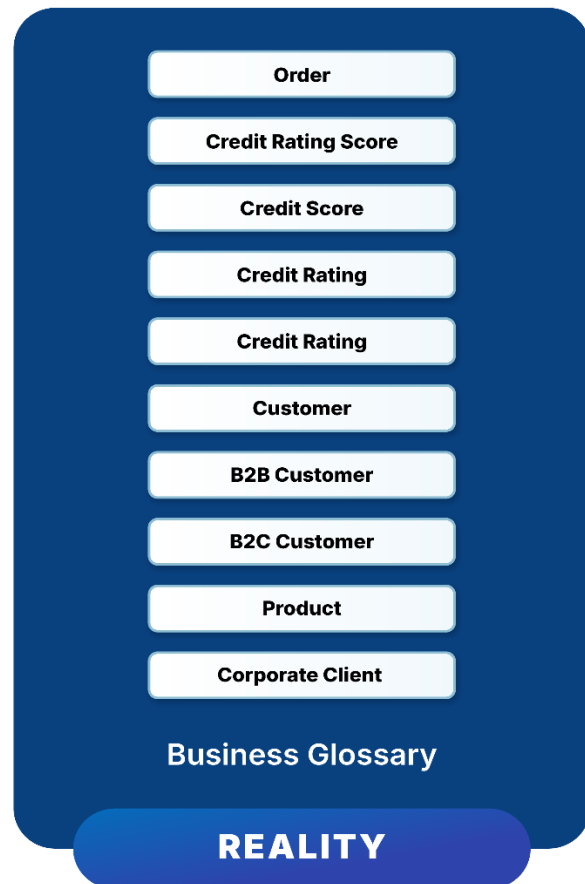
This creates a disconnect.

The glossary defines what a term is supposed to mean. The data platform implements how it is actually used. Without a structure to connect the two, they drift apart over time.

Business terms arranged into an ontology with taxonomies



A sea of disconnected terms



This is why glossaries alone struggle to support analytics, and break down entirely when used as the foundation for AI.

AI does not just need a definition. It needs to understand how concepts relate, how they vary, and how they are applied in context. A flat list of terms, however well governed, cannot provide that.

To do that, you need structure.

Why an AI Needs Ontology

AI changes how we interact with data.

Instead of writing queries or building reports, users ask questions in natural language and expect a correct answer. For that to work, the system has to understand both the question and the data it is using to answer it.

That is harder than it sounds.

As we discussed previously, when someone asks for “revenue” or “customer,” those terms are not inherently defined. The AI has to interpret them in the context of the organisation. At the same time, the data it is working with, tables, columns, fields, does not carry meaning on its own. It is just structure.

If either of those interpretations is wrong, the answer will be wrong.

Most data platforms deliver data and, at best, information. AI only works when that becomes knowledge. Without a shared model of meaning, that leap never happens.

This is where ontology becomes essential.

An ontology provides a structured and machine-readable representation of business meaning. It defines the core concepts of the enterprise, the relationships between them, and the rules that govern how those concepts are interpreted and used.

In practice, this is closely aligned to the role the Enterprise Logical Data Model has long played within data architecture. The difference is that the ontology operationalises that meaning in a form that can be interpreted consistently by machines as well as humans.

It makes explicit what would otherwise remain implied, inconsistent, or buried in code and pipelines.

In practical terms, it acts as a map. When tables, columns, and fields are linked to that structure, the AI can understand what the data represents, not just where it lives. Without that mapping, it is left to infer meaning from names and patterns. Sometimes that works. Often it doesn't.

With an ontology in place, two things change.

First, language can be interpreted properly. The AI can map terms in a question to defined business concepts rather than guessing.

Second, the data can be reasoned over consistently. Because everything is aligned to the same model of meaning, the AI is working within a shared framework, not a collection of disconnected structures.

A well-defined ontology also allows the AI to recognise ambiguity. If a term like “revenue” could mean different things, it can ask for clarification rather than returning a confident but incorrect answer.

The result is not just better answers, but more reliable ones.

The difference is subtle in design, but significant in outcome. The AI moves from guessing to understanding.

In many AI systems, this structured meaning is represented as a knowledge graph.

At a high level, there is a distinction between the definition of concepts and relationships, and the actual data that instantiates them. These are sometimes respectively referred to as the “Terminology Box” (TBox) and the “Assertion Box” (ABox).

For AI, this distinction matters. It allows the system to reason over data using an explicit model of meaning, rather than inferring relationships from raw structures.

This challenge becomes even more pronounced when incorporating textual data. Unlike structured data, text does not come pre-aligned to defined concepts. Systems such as Textual ETL can identify and classify terms within text, but without alignment to a shared semantic model, those classifications risk introducing yet another interpretation of meaning. The same principle applies: textual data must be grounded in the same ontology as structured data if it is to be used reliably in analytics and AI.

The Semantic Backbone as the Enterprise Control Plane

If the data platform is the execution engine, the semantic backbone is the control plane.

It defines how meaning is established, governed, and applied across the enterprise. Without it, every pipeline, model, and report defines its own version of meaning. With it, everything aligns to a shared understanding of the business.

This is the layer that has not been explicitly engineered.

Modern data platforms are highly capable when it comes to moving and transforming data. They provide storage, compute, and orchestration at scale. But they do not, by default, provide a consistent way to define and enforce meaning.

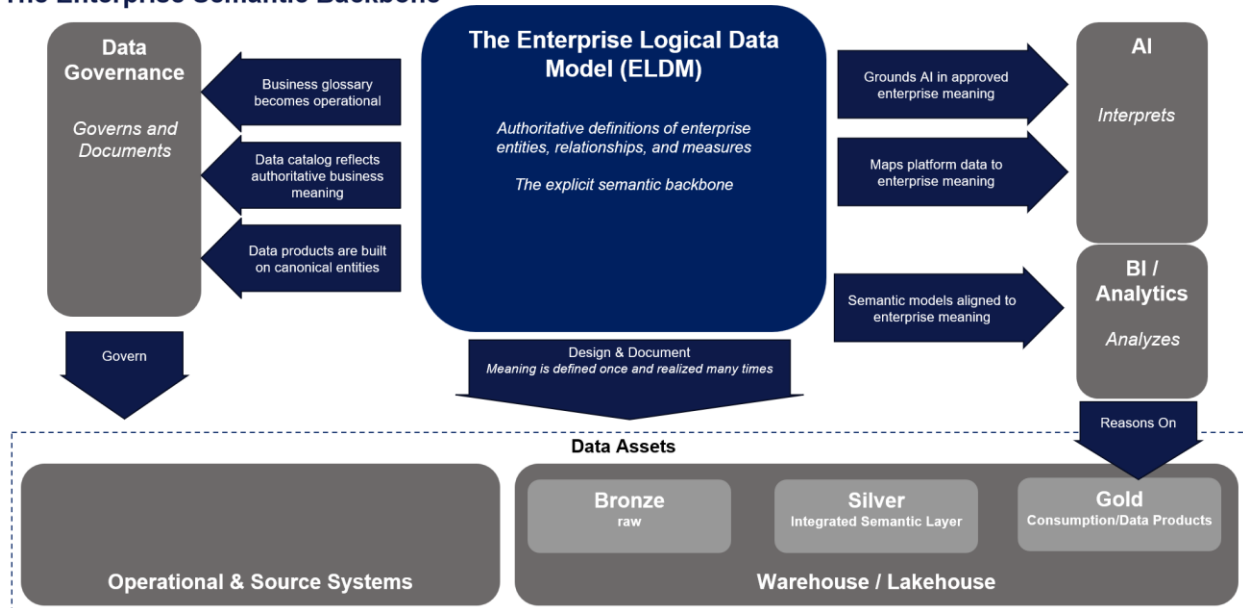
That responsibility is often distributed across teams and tools. Business teams define and maintain meaning. Governance functions establish policies, classifications, stewardship, and oversight. Architects structure those concepts into enterprise models. Engineers operationalise them in platforms and pipelines. Analysts shape them into reporting and analytical outputs. Each step introduces interpretation unless aligned to a shared semantic framework.

The semantic backbone brings that back together.

At its core is the Enterprise Logical Data Model (ELDM). This is not a documentation exercise or a static artefact. It is a working model of the business. It defines the core entities, the relationships between

them, and the rules that govern how they are used. It provides a place where meaning is agreed, structured, and maintained.

The Enterprise Semantic Backbone



The semantic backbone does not imply that IT owns business meaning. The authoritative definition of business concepts belongs to the business process and capability owners responsible for the activities that create and use that data. Their role is to define, ratify, and maintain those meanings as the business evolves. Data architects then structure and integrate that knowledge into the Enterprise Logical Data Model, while engineering teams operationalise it across platforms, pipelines, analytics, and AI systems.

The Enterprise Logical Data Model is the authoritative semantic design model of the enterprise. It provides the human-governed representation of business meaning, capturing concepts, relationships, and rules in a form that can be agreed, managed, and evolved over time.

The ontology is the machine-readable operational semantic layer derived from that model. It translates the semantic structure of the ELDM into a form that can be interpreted and reasoned over by systems, AI models, and knowledge graphs.

The Enterprise Logical Data Model is not a new concept. Data architects have used logical models for decades to capture business meaning and then design and align data assets across systems. They provide a structured way to define core business concepts, their relationships, and the rules that govern them, using a set of well-understood and reusable patterns. It is used to ensure consistency across business systems to improve quality and interoperability.

What they have always offered is a common language for the business. By defining meaning once and reusing it consistently, logical models help ensure that systems remain aligned and interoperable, even as they evolve over time.

Many organisations already have all or elements of this in place, maintained by data architecture teams. This model is rich with existing approved definitions and vital metadata, including data types and how concepts are uniquely identified by the business. It is the ideal candidate to be that shared semantic backbone.

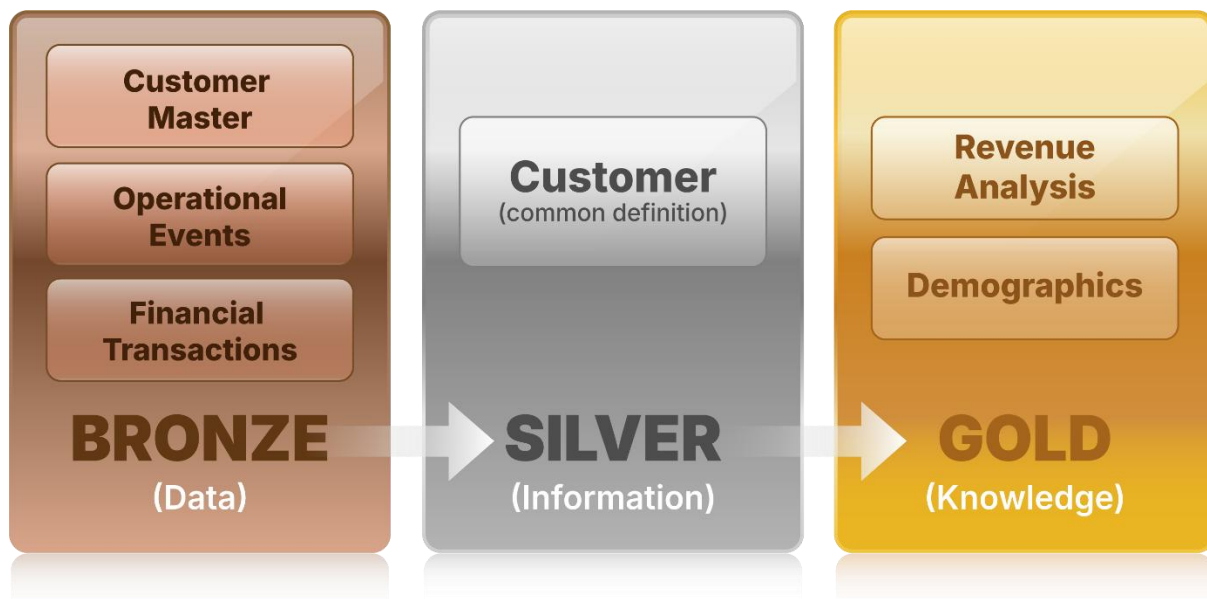
From this foundation, the rest of the platform can align.

Data governance teams have the option to use this existing enterprise logical model as the framework for governance. Some governance tools place the Enterprise Logical Data Model (ELDM) at the centre of governance activities, using it to organise business concepts, relationships, policies, and stewardship responsibilities across the organisation.

This differs from smaller logical models that may exist for individual projects or applications. The ELDM operates at enterprise scope, providing an integrated and shared model of business meaning across domains.

Alternatively, business terms with fully qualified names can be harvested by tools like ER/Studio and attached to that framework. The governance structure then gains precision and context, reducing the ambiguity of a simple flat list of terms.

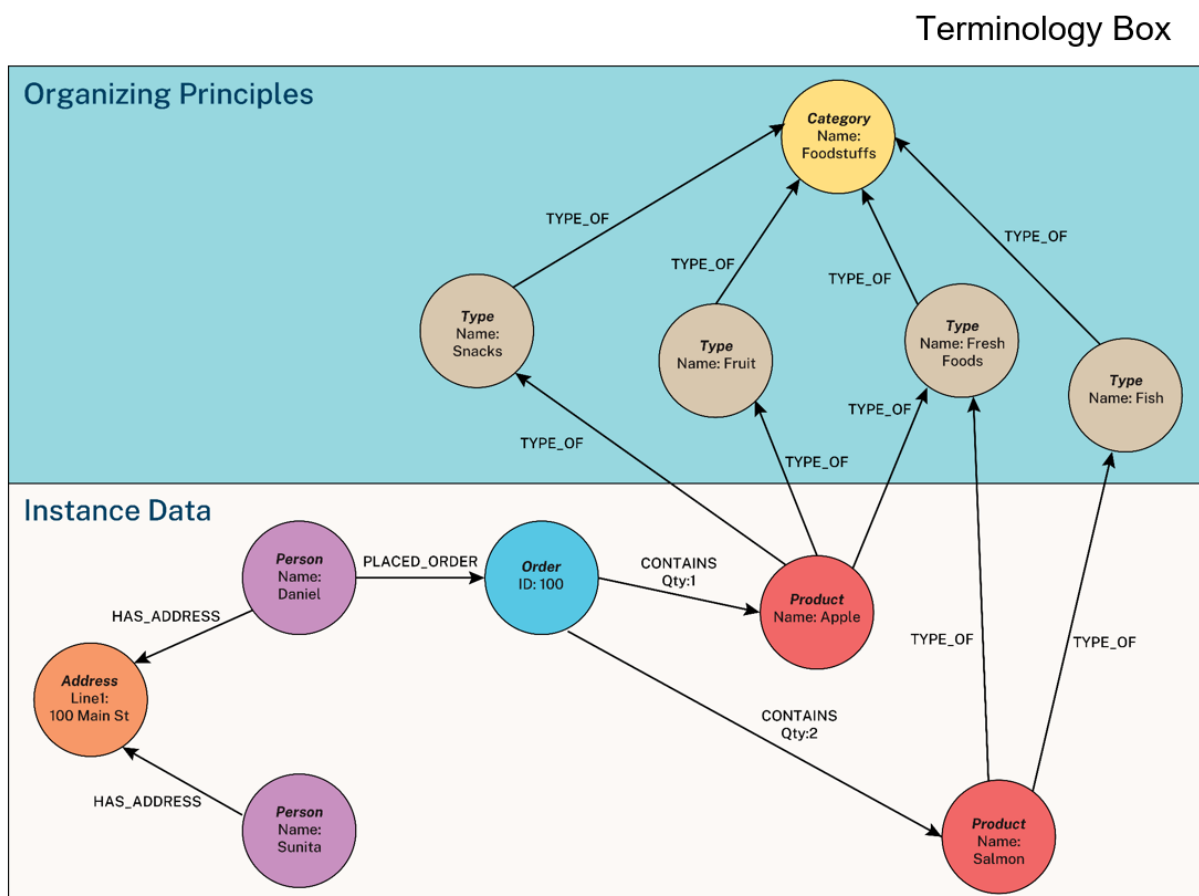
Data platform design becomes more coherent. The integrated layers of the platform, particularly the Silver layer, can be aligned to the enterprise model, ensuring that integration is based on shared meaning rather than local transformations. This does not replace existing engineering patterns. It provides the semantic foundation that makes them consistent.



Analytics becomes more consistent. Dimensions, facts, and measures are derived from the same underlying structure, reducing divergence across reports and use cases.

AI becomes more reliable. The same model of meaning that governs the data platform is used to interpret prompts and guide reasoning. The AI is no longer inferring relationships from data structures alone. It is operating against an explicit semantic framework.

In knowledge graph terms, the semantic backbone provides the definition of meaning, the TBox. The data within the platform, the records in tables, transactions, and events, forms the ABox, the instantiation of that structure.



Neo4J What is a Knowledge Graph <https://neo4j.com/blog/knowledge-graph/what-is-knowledge-graph/>

Assertion Box

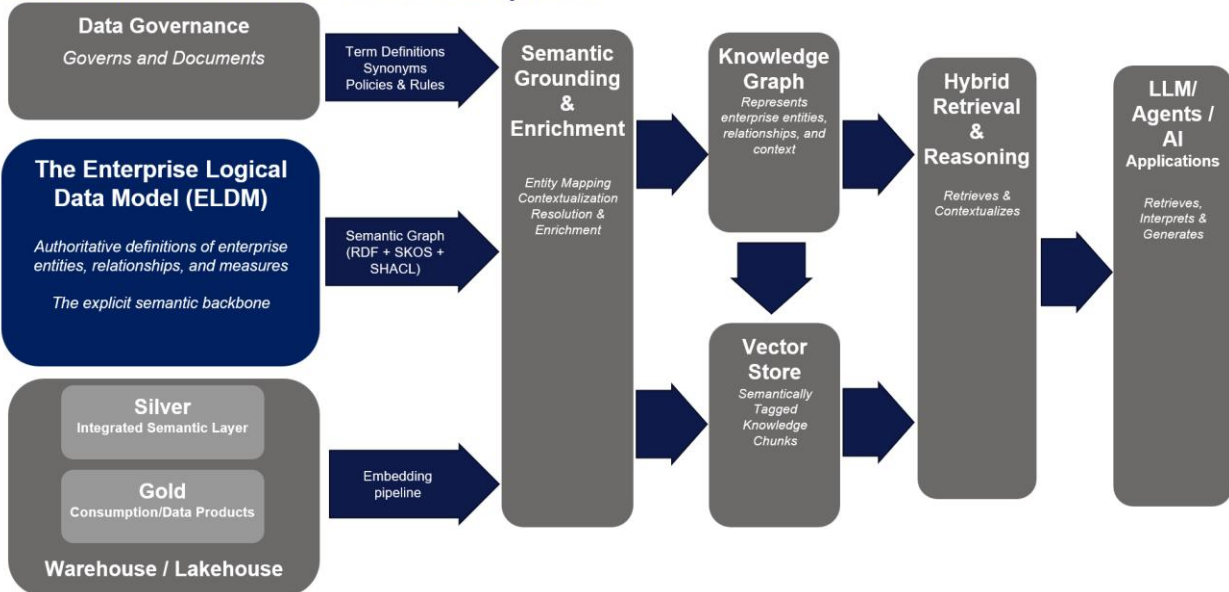
By aligning data assets to this backbone, the enterprise ensures that both humans and AI systems are working from the same model. Meaning is no longer embedded in pipelines or held in the minds of individuals. It is explicit, shared, and reusable.

The semantic backbone is not simply a documentation artefact. It becomes an operational layer that connects governance, data platforms, knowledge structures, and AI systems through a shared model of meaning.

In practical terms, the Enterprise Logical Data Model provides the semantic foundation from which machine-readable ontologies and enterprise knowledge graphs can be constructed. Through representations such as RDF, SKOS, and SHACL, the ELDM can be operationalised into semantic structures that support enrichment, AI grounding, and contextual retrieval across both structured and unstructured data.

The following conceptual architecture illustrates how these components can work together.

The Semantic Backbone for Trusted Enterprise AI



In this model, the Enterprise Logical Data Model provides the authoritative definition of enterprise concepts, relationships, and business meaning. Governance processes enrich and constrain that model through approved terminology, policies, and rules.

The warehouse and lakehouse provide the operational and analytical data foundation, while semantic enrichment processes align both structured and unstructured content to the same semantic framework. This creates a knowledge layer that AI systems can retrieve from and reason against with greater consistency and contextual awareness.

Rather than relying purely on statistical inference from raw data, the AI is grounded in an explicit model of enterprise meaning.

This does not remove the need for different representations of data. Dimensional models, data products, and analytical views still play a critical role. But they become projections of a governed model of meaning, rather than independent interpretations of it.

A common objection is that building and maintaining an enterprise model takes too long, or cannot keep up with change. In practice, the opposite is true. Without a shared model, every change must be rediscovered and reimplemented across multiple pipelines and use cases. With one, change is managed once and propagated consistently.

The semantic backbone is not about slowing down delivery. It is about ensuring that what is delivered remains consistent, reusable, and trustworthy over time.

It provides a single, accepted framework that binds governance, data platforms, analytics, and AI together. Without a semantic backbone, integration is temporary and ambiguous. With it, meaning becomes permanent and consistent.

How the Semantic Backbone Works in Practice

The semantic backbone does not replace modern data platforms. It gives them a shared understanding of meaning.

The **Enterprise Logical Data Model (ELDM) becomes the semantic backbone**. The ELDM acts as the authoritative semantic design model of the enterprise, defining the core business concepts, relationships, rules, and identifiers that describe how the business actually operates.

From this foundation, a machine-readable semantic layer can be established through ontology technologies such as RDF, OWL, SKOS, and SHACL. This operational semantic layer translates the structure and meaning of the ELDM into a form that can be interpreted and reasoned over by systems, AI models, and knowledge graphs. The resulting ontology may then be loaded into graph technologies or semantic platforms that manage the operational knowledge graph used for retrieval, linking, and AI reasoning.

Governance then aligns to that structure. Business glossaries, policies, classifications, and stewardship processes are derived from and/or attached to a shared semantic framework rather than managed as isolated terms.

As data enters the platform, source systems are mapped to those shared concepts. Integration is no longer handled independently inside individual pipelines. It is aligned to a consistent enterprise model of meaning.

Analytical models, data products, and reporting structures are then derived from that integrated foundation. Different teams can still build for different use cases, but they are working from the same semantic framework rather than creating competing interpretations.

AI systems also operate against this shared model. Instead of inferring meaning from table names or isolated metadata, AI can reason against explicitly defined business concepts and relationships. This allows prompts, structured data, and unstructured content to be grounded in the same enterprise understanding of meaning.

The result is a platform where governance, integration, analytics, and AI are all aligned to the same semantic foundation.

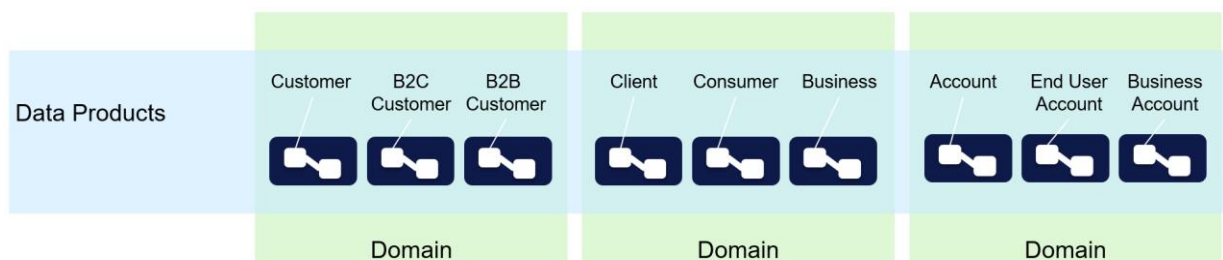
A Single Ontology, Or Many?

In an ideal world, an organisation operates from a single shared ontology.

There is one agreed understanding of core business concepts, one common language, and one semantic framework that governs how data is interpreted across the enterprise. This creates consistency not only in reporting and analytics, but also in how AI systems reason over data.

That consistency matters.

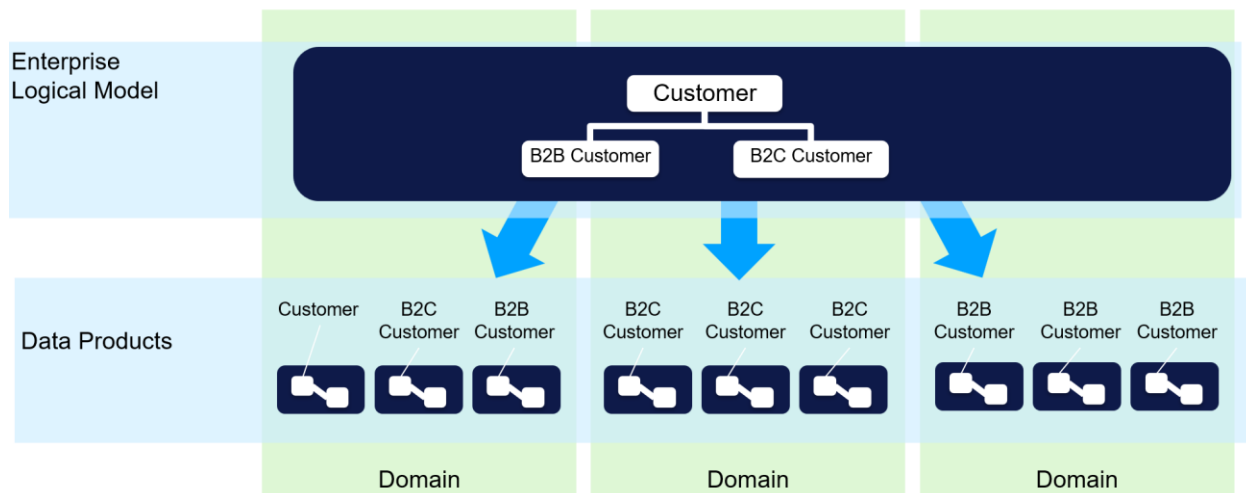
AI does not simply query data. It interprets language, maps concepts, and reasons across relationships. The more variation that exists in definitions, structures, and terminology, the harder that becomes. If every data product, warehouse model, or star schema defines concepts differently, both humans and machines are forced into a constant process of interpretation and translation.



A shared ontology reduces that ambiguity.

It provides a common semantic framework across the organisation, making data easier to integrate, govern, analyse, and reason over. It also simplifies management. Definitions are maintained once, aligned once, and reused consistently across platforms and use cases.

Ideally, this allows the organisation's data to be pooled into a coherent body of knowledge that AI can reason across as a whole, correlating information between domains rather than treating each area as an isolated island.



But reality is often more complicated.

Many organisations contain business units or domains that genuinely operate differently. They may use different terminology, apply different rules, or maintain separate operational data. In some cases, these differences are accidental. In others, they reflect legitimate differences in how the business functions.

For example, a request as simple as “How many customers do we have?” may require the organisation to determine which semantic context applies. A retail division may define a customer as an individual consumer who has completed a purchase, while a commercial division may define a customer as a contracted business account. In a shared enterprise ontology, these distinctions are explicitly modelled beneath a common customer concept. In independently managed domain ontologies, the AI or reporting system must first establish which domain context governs the request before interpreting the data correctly.

In these environments, it may be practical for domains to operate with their own independent ontologies, each aligned to its own data and definitions.

This can work, provided those ontologies are explicit and governed.

The important point is not that there must always be exactly one ontology. The important point is that meaning must be formally defined somewhere. AI cannot reliably reason over undefined or conflicting semantics. When interacting with AI, both the user and the AI system must also be clear about which ontology governs the interaction. The meaning of terms, the interpretation of relationships, and even the scope of the available data may differ between domains. Without a clearly defined semantic context, the AI may reason against the wrong definitions or combine concepts that were never intended to align.

Once multiple ontologies exist, the challenge becomes alignment.

There may be a federated approach where business units periodically synchronise common definitions. Some organisations may choose to directly reuse a shared set of canonical enterprise concepts across domains, allowing key business entities and terms to remain consistent everywhere they are used.

Others may allow domains to maintain their own independent ontologies, but map them to a higher-level enterprise ontology that acts as a semantic translation layer between domains.

The specific implementation will vary between organisations.

What does not vary is the requirement for explicit semantic structure. Whether there is one ontology or many, AI systems require a clear and governed understanding of meaning, tied directly to the data they operate on.

Without that, every interaction becomes a translation exercise, full of ambiguity and risk.

With it, meaning becomes coherent across both humans and machines, improving consistency, trust, and quality.

Conclusion: Restoring Meaning as the Foundation

Over the past decade, the data industry has optimised for speed. We built pipelines that move data faster, platforms that scale further, and tooling that allows teams to deliver in days what once took months. In doing so, we unlocked enormous value.

But we also lost something along the way.

We lost a shared understanding of what our data actually means. For years, that was manageable. Humans could bridge the gaps, interpret inconsistencies, and apply context where systems could not. AI removes that safety net.

AI does not reconcile meaning. It assumes it.

This is why semantic entropy is no longer a theoretical concern. It is a practical constraint on what organisations can achieve with their data.

The solution is not to abandon modern architectures. It is to complete them.

A data platform without a semantic backbone is an execution engine without a control plane. It can move data, but it cannot ensure that what it delivers is consistent, reusable, or trustworthy.

By establishing a shared, structured model of meaning, grounded in an Enterprise Logical Data Model, organisations create a foundation that aligns governance, data platforms, analytics, and AI.

This is not a new idea. It is a long-established discipline that has become critical.

The next generation of data platforms will not be defined by how fast they move data, but by how precisely they define meaning.

Without a semantic backbone, integration is temporary.

With it, meaning becomes permanent.

About the Authors

Bill Inmon



Bill Inmon is known as the father of the data warehouse. Bill has written 74 books published in 9 languages. Bill's books have sold over 1,500,000 copies worldwide. One of Bill's books BUILDING THE DATA WAREHOUSE has sold over 500,000 copies. Bill was named by Computerworld as one of the ten most influential thought leaders in the history of computing.

Bill developed the technology for textual disambiguation, which reads raw text and turns that text into a standard data base. Bill founded and took public Prism Solutions, the original ETL company. Bill founded LLM Management.

Bill started his career as a pro golfer but got tired of starving to death and decided to get real job. Bill lives in Denver, Colorado with his wife and his two Scotty dogs – Rollie and Lena.

Jamie Knowles



Jamie Knowles is Product Director for ER/Studio at [IDERA Software](#), where he leads the strategy and development of the ER/Studio enterprise data modelling platform.

With more than 25 years of experience in data modelling and enterprise architecture, he focuses on governance, semantic modelling, and AI-ready data platforms, working with customers and industry partners on approaches that connect business meaning, analytics, and AI through a shared semantic foundation.