

How to test context before it reaches production

Validate AI context against live data, inspect its grounding, and catch incorrect definitions or relationships before they reach production.

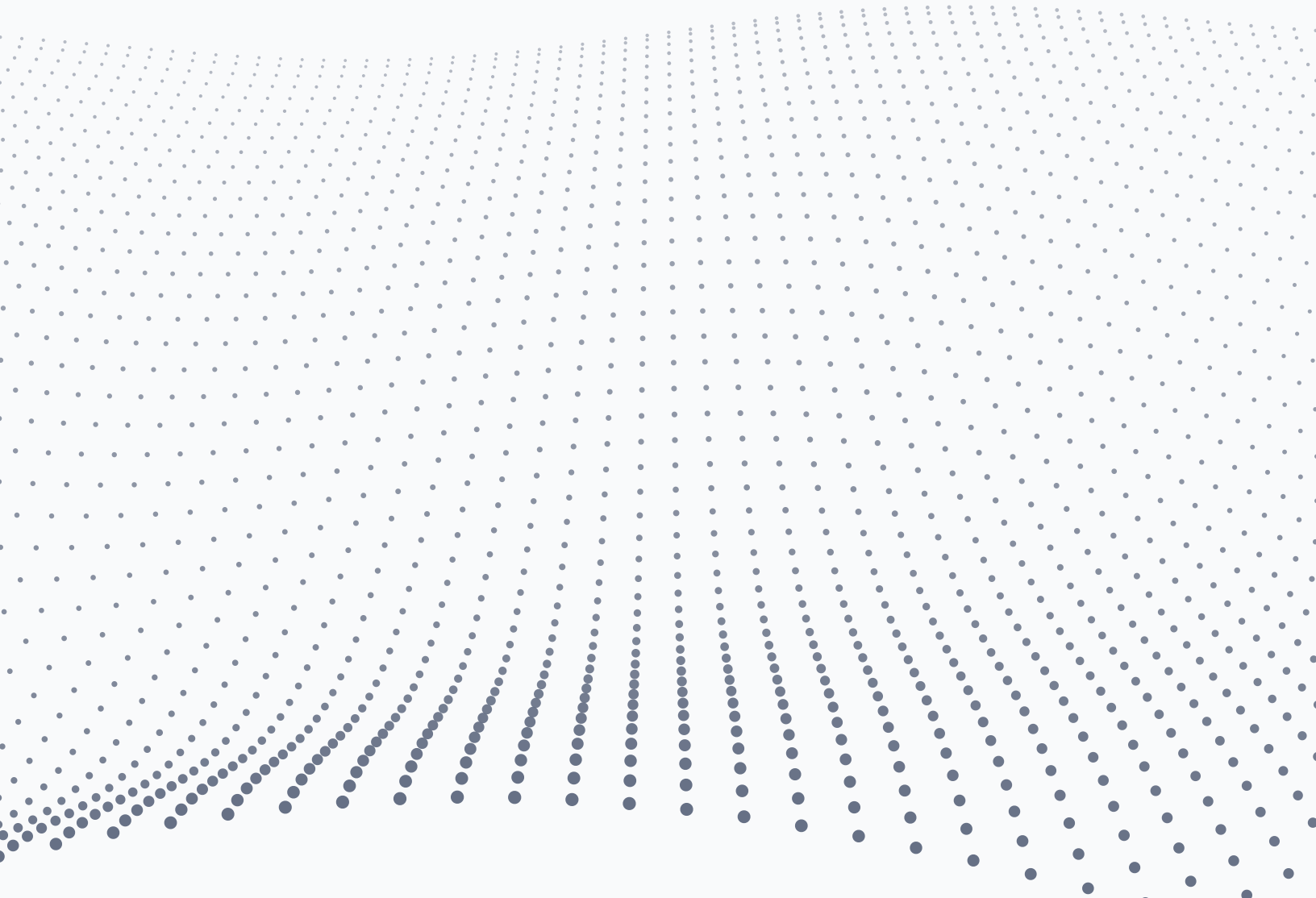


Table of Contents

Accuracy is a property of the binding	03
Most AI evaluation measures the wrong surface	04
Context Sandbox runs the question before the agent exists	04
Binding validation catches what answer evaluation misses	06
One question, all the way through	06
Live data, three gates deep	07
What this does to your evaluation	08
Where this sits	08
Where to start	09

Every vendor in your evaluation has told you their context is ready for AI. Connect an agent to our catalog, they say, and it'll understand your business.

The demos support the claim. Somebody types a question in plain language, a number comes back, the number is right, and the room nods.

Then you go home and try to work out how you'd check that against your own estate, where the same metric has four definitions and the one with the heaviest usage is the one finance stopped trusting two years ago.

That's the actual evaluation problem. The claim is testable in principle and untestable in every demo you've been shown.

Reliable AI depends on grounding. Grounding leaves a trace, and a trace can be checked.

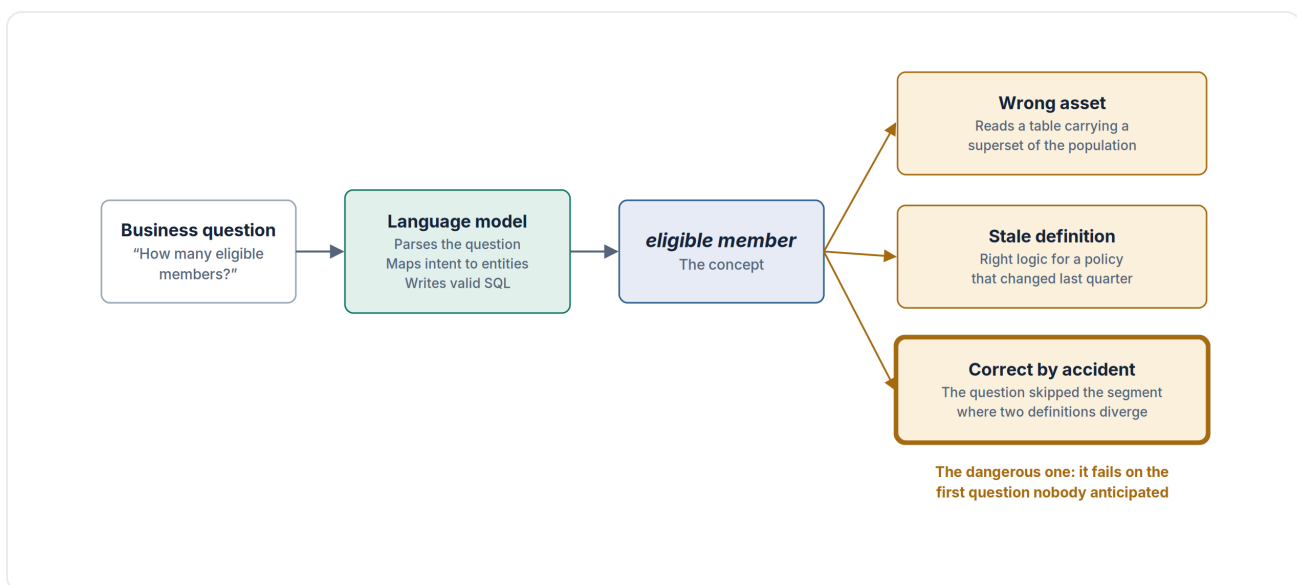
Accuracy is a property of the binding

When an enterprise agent returns a wrong number, the language model usually did its job. It parsed the question, mapped intent to entities, and generated valid SQL.

The failure happened earlier, in what the concept was pointing at.

Three versions show up repeatedly. The concept resolves to the wrong physical asset, so "eligible member" reads a table carrying a superset of the population. The concept resolves to a definition that went stale, so the logic is right for a policy that changed last quarter.

The third one is the dangerous one: the answer is correct by accident, because the question happened to avoid the segment where two definitions diverge. It passes every test you'd think to run, and it fails the first time someone asks a question you didn't anticipate.



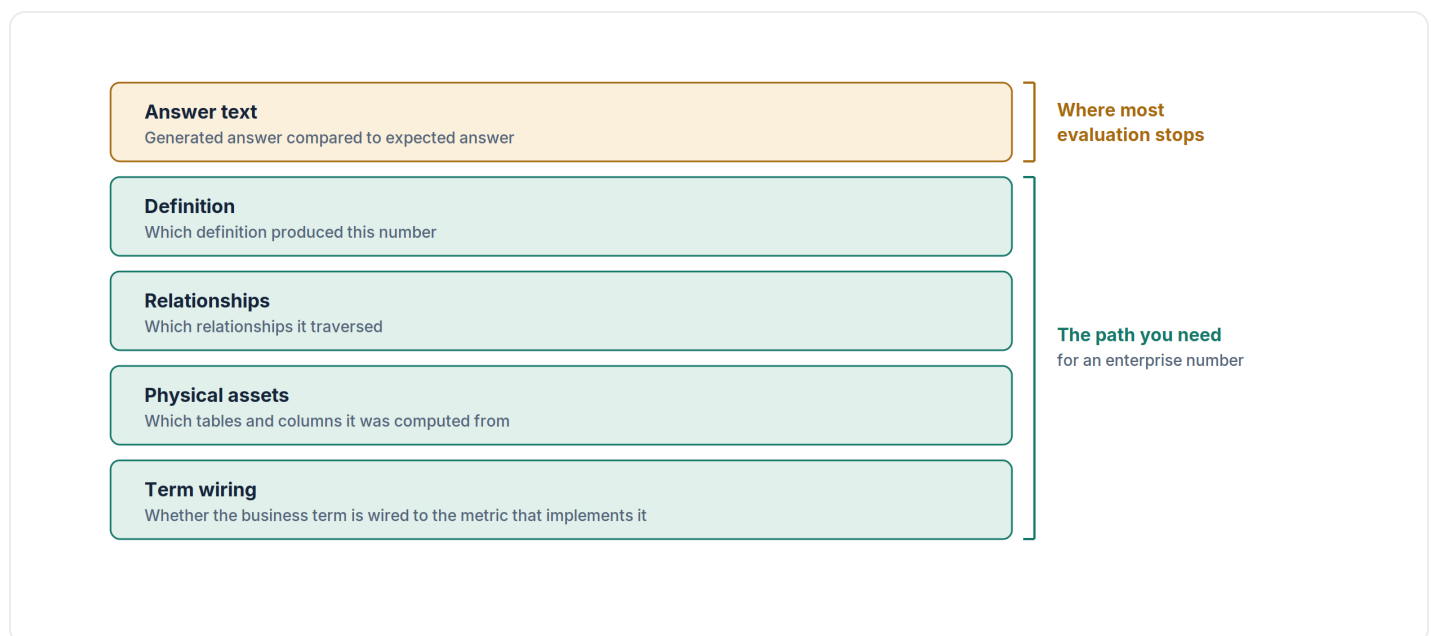
Most AI evaluation measures the wrong surface

The standard approach compares generated output against expected output. Run a set of questions, diff the answers, count the matches, ship at some threshold.

That measures whether the text agreed. For an enterprise number, agreement is the smallest part of what you need to know.

You need the path. Which definition produced this, which relationships it traversed, and which physical tables and columns the number was computed from. Whether the vocabulary term the business uses is wired to the metric that implements it, or to something adjacent that looked close enough at modeling time.

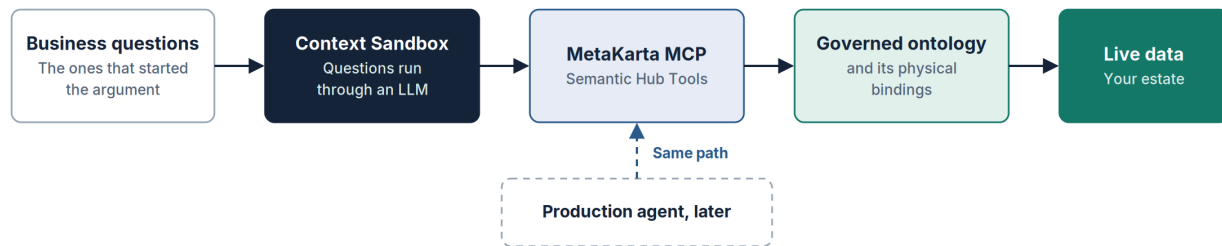
An answer that matches for reasons you can't reconstruct is a number you'll be defending later, from memory.



Context Sandbox runs the question before the agent exists

Context Sandbox is where a definition gets tested against live data, ahead of production.

You run representative business questions through an LLM against your governed ontology and its physical bindings. The environment talks to MetaKarta MCP through the Semantic Hub Tools, the same path a production agent uses, so the test exercises the real delivery mechanism.



Four things happen in the loop



Answer evaluation is the floor every eval framework already reaches. Steps 3 and 4 are where the silent failures show up.



Testing environment.

Take the questions your business actually asks. Quarterly revenue by segment. Active accounts in the enrollment window. The ones that started the argument in the first place.



Answer evaluation.

Compare each output against your approved reference answer, the one finance or compliance already signed off on. Every eval framework does this much, and here it's the floor.



Binding validation.

Confirm that each ontology concept resolves to the correct lineage, tables, columns, and metrics. This catches the silent failure, a concept returning a plausible number from the wrong asset.



Grounding inspection.

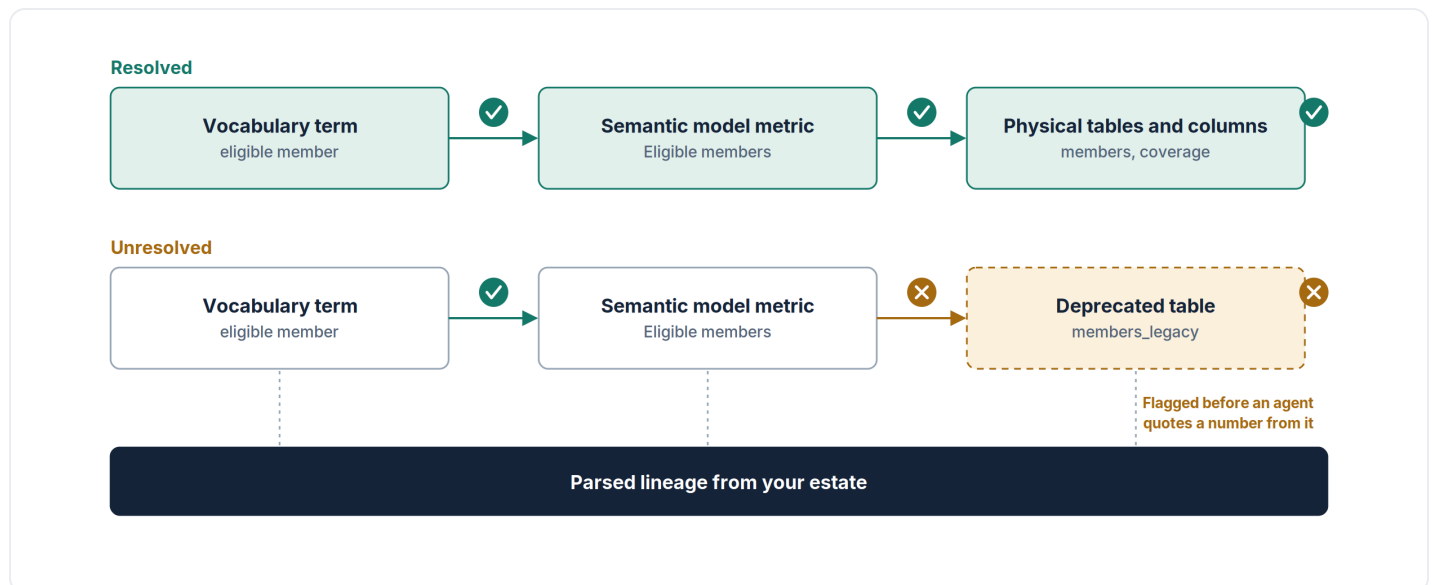
See which definitions, relationships, and physical assets supported the result. The path is on screen, in full, before anyone has to reconstruct it.

Binding validation catches what answer evaluation misses

Answer evaluation tells you the output matched on the questions you thought to ask. Binding validation tells you the concept is attached to the right thing, which holds for the questions you didn't.

The check is mechanical. Take the vocabulary term, follow it to the semantic model metric that implements it, follow that to the physical tables and columns underneath, and confirm the chain against the lineage already parsed from your estate.

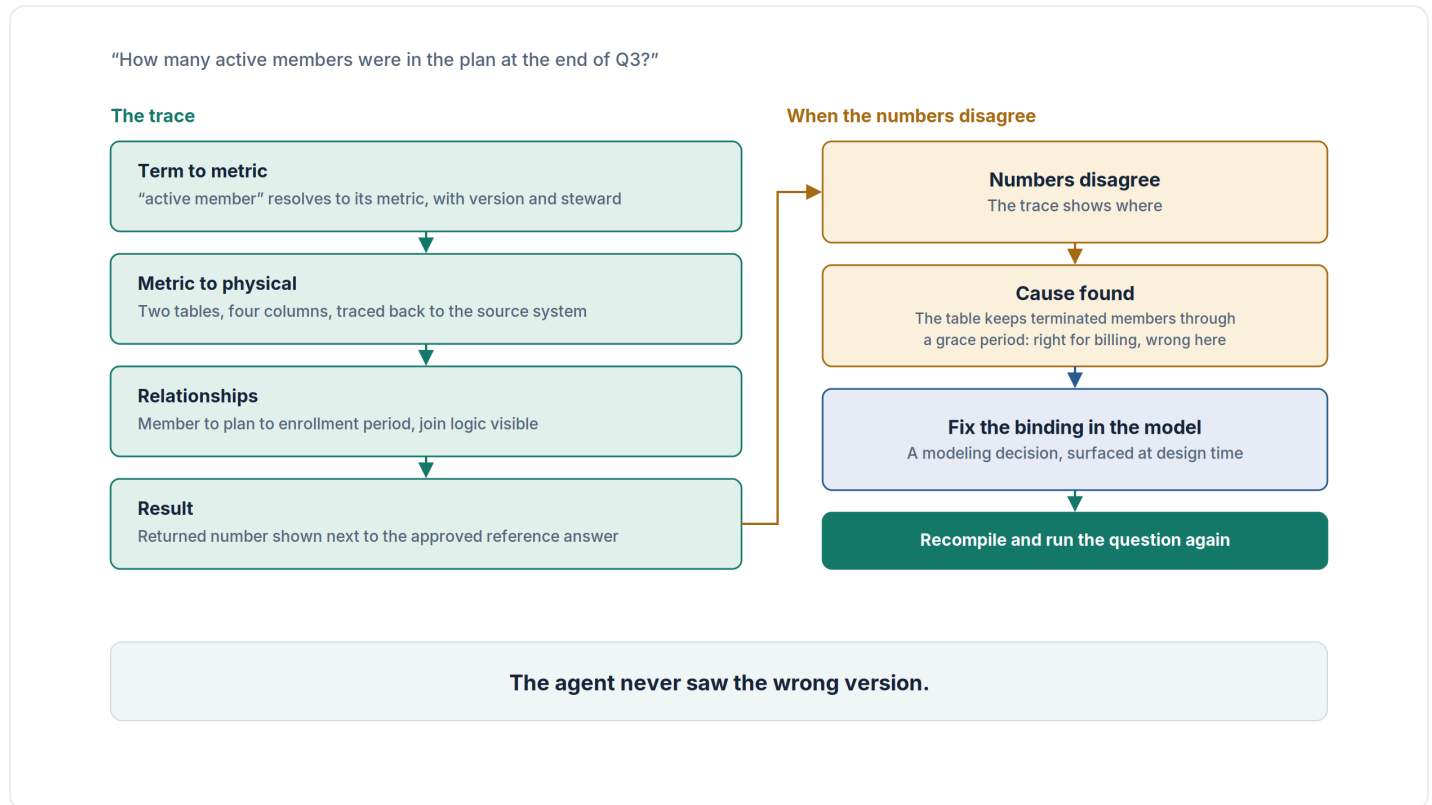
Where the chain breaks, it breaks visibly. A concept pointing at a deprecated table shows up as a concept pointing at a deprecated table, before an agent ever quotes a number from it.



One question, all the way through

Take a question your enrollment team asks every month: how many active members were in the plan at the end of Q3.

You run it in the Sandbox against your governed context. The answer comes back, and so does the trace behind it.



Say the two numbers disagree. The trace tells you where. The metric was reading the enrollment table that carries terminated members through a grace period, correct for the billing question it was originally written for and wrong for this one.

That's a modeling decision, surfaced at design time, with the evidence attached. You fix the binding in the model, recompile, and run the question again. The agent never saw the wrong version.

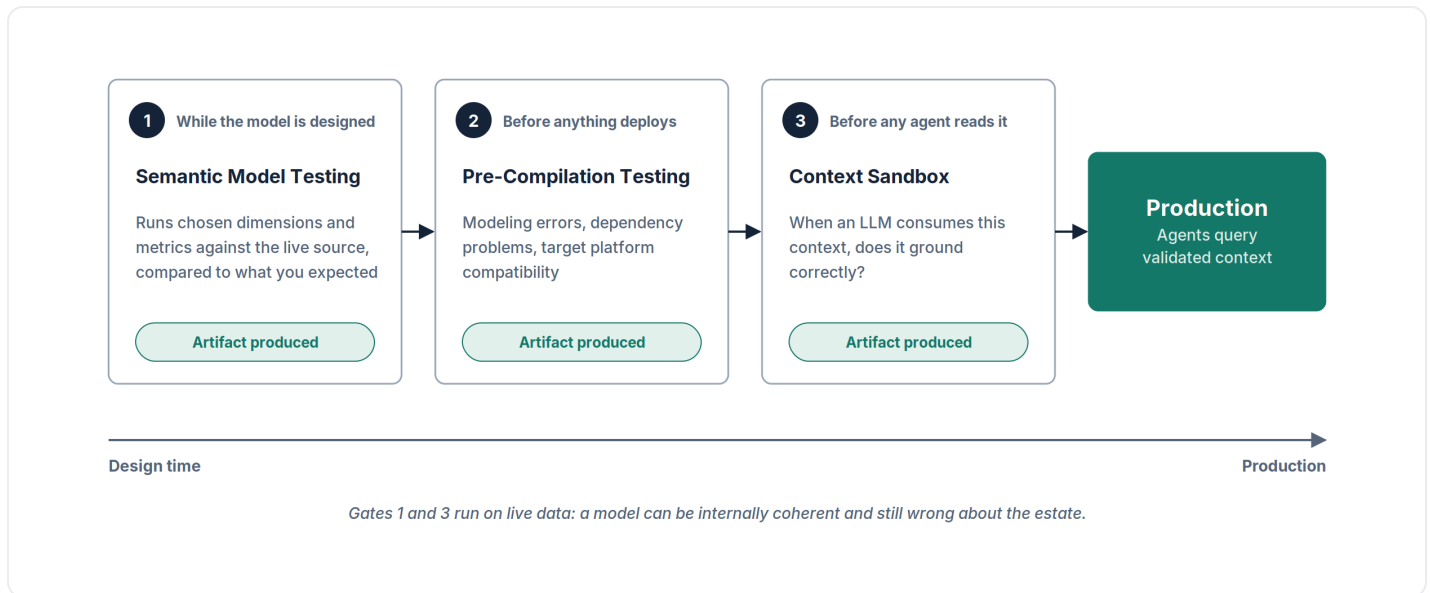
Live data, three gates deep

Sandbox validation runs against live data, which matters because a semantic model can be internally coherent and still wrong about the estate it describes.

It's also the third of three checks, and they catch different things.

Semantic Model Testing runs your chosen dimensions and metrics as a query against the live source and compares the result to what you expected, while the model is still being designed. Pre-Compilation Testing catches modeling errors, dependency problems, and target platform compatibility issues before anything deploys. Context Sandbox asks the question the other two can't: when an LLM consumes this, does it ground correctly.

Three gates, all before production, all producing an artifact you can hand to whoever asks.



What this does to your evaluation

Take the claim every vendor is making and turn it into a test they have to pass in front of you.

Ask each one to run a business question you bring, against data you supply, and show the grounding. Which definition did the answer use. Which columns did it read. Where is that written down, and who owns it.

Vendors serving context out of a black box will offer you accuracy numbers from their own benchmark. That's a different question, answered on their terms.

Test the claim yourself, on your data, with the grounding visible.

Where this sits

Context Sandbox verifies the semantic foundation before any query runs, which means there has to be a foundation to verify.

That's Semantic Hub: governed semantic models with your business vocabulary bound to them, compiled into the native artifacts of your databases and BI tools. Ontology Modeling supplies the business meaning, built bottom-up from the lineage already parsed out of your estate. Every concept is bound to a real asset from day one.

MetaKarta MCP delivers that foundation to agents through the Semantic Hub Tools. Compiled context, deterministic SQL. No compute tax, no token tax.

Every response returns the metadata evidence that produced it. Permissions follow the person behind the agent, so it surfaces only what that person is already authorized to see.

All of it reads from the same shared metadata repository as your lineage, catalog, and governance.

The validation works because the bindings are real. A concept bound to a live asset can be checked against that asset. A concept described in a document can only be read.

Where to start

Pick three questions your business asks constantly and write down the approved answer to each one, with the name of the person who approved it. That list is your reference set, and building it takes an afternoon.

Any vendor claiming their context is ready for AI should be able to run it, on your data, and show you the path behind every answer.

MetaKarta v12 is available now. Request a demo and bring your three questions.

[Request Demo](#)

About Meta Integration Technology

Meta Integration Technology, Inc. (MITI) is a pioneer in metadata integration, semantic modeling, and data lineage with decades of experience serving global enterprises and technology providers. MITI's technology is embedded in products from leading data and analytics vendors and powers MetaKarta, its enterprise platform for metadata-driven data management. MetaKarta delivers data lineage, data cataloging, data governance, and semantic enforcement through a single integrated system, connecting the full enterprise data estate across 400+ native connectors, from legacy to modern and on-premises to the cloud.

650 Castro Street, Suite 120-431, Mountain View, California 94041-2055 USA
Phone: 650.429.8853

info@metakarta.com
www.metakarta.com