

How to make existing BI semantic models survive a migration

Use the semantic models your teams already built as the starting point for your migration, and compile them into every new platform.

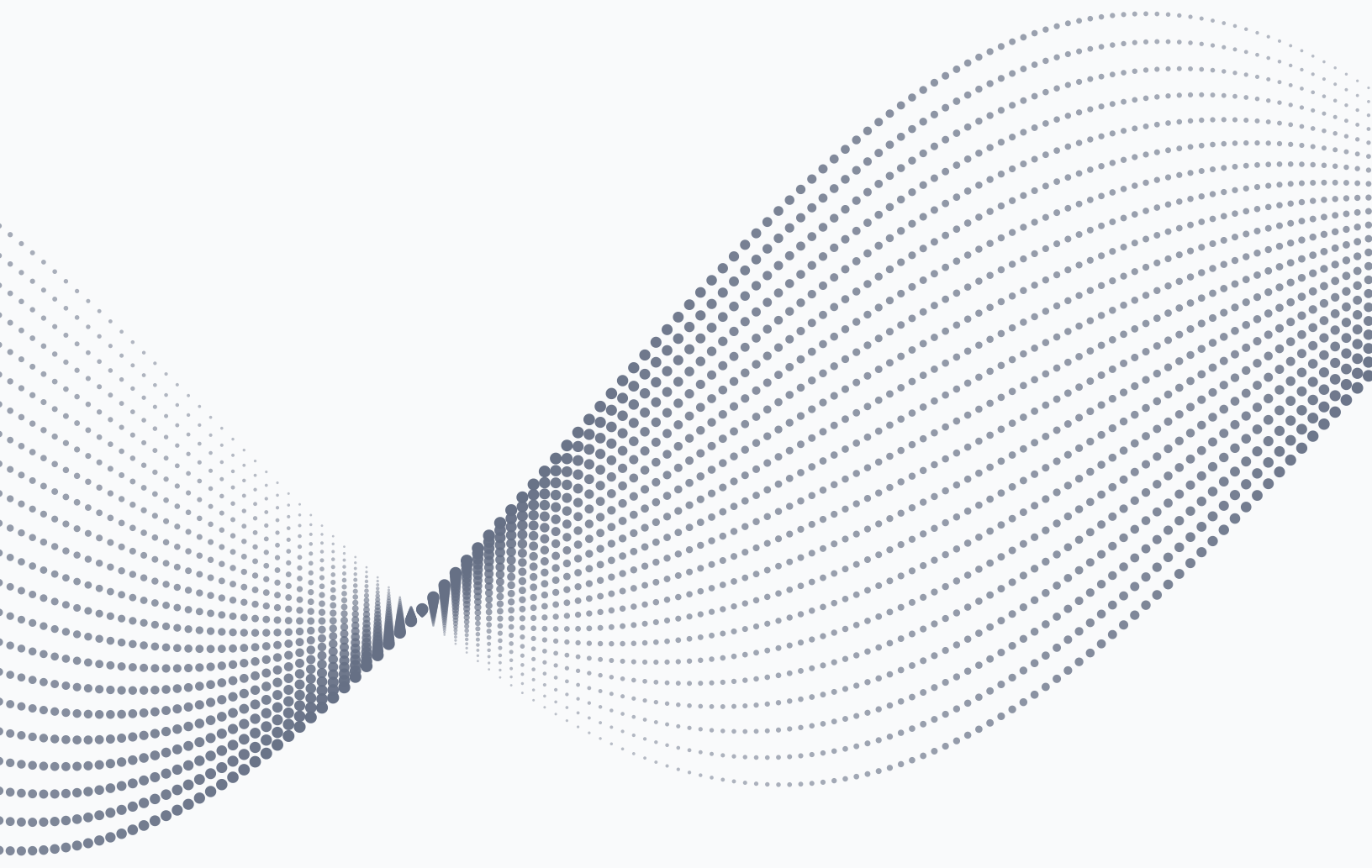


Table of Contents

Migrations fail in two directions	03
Know what breaks before you move anything	03
Your existing models are the starting point	04
What the import cleans up on the way in	05
The disagreement is the real work	06
Compile it everywhere	07
Where to start	09

The migration plan had 11 workstreams and a date on it. It stalled in week three, in a meeting about one metric.

Two teams had brought their definition of active customer. The Power BI version counted anyone with a login event in the trailing 30 days. The Tableau version required a billable transaction in the same window. Both had been correct for years, in their own tool, for their own audience.

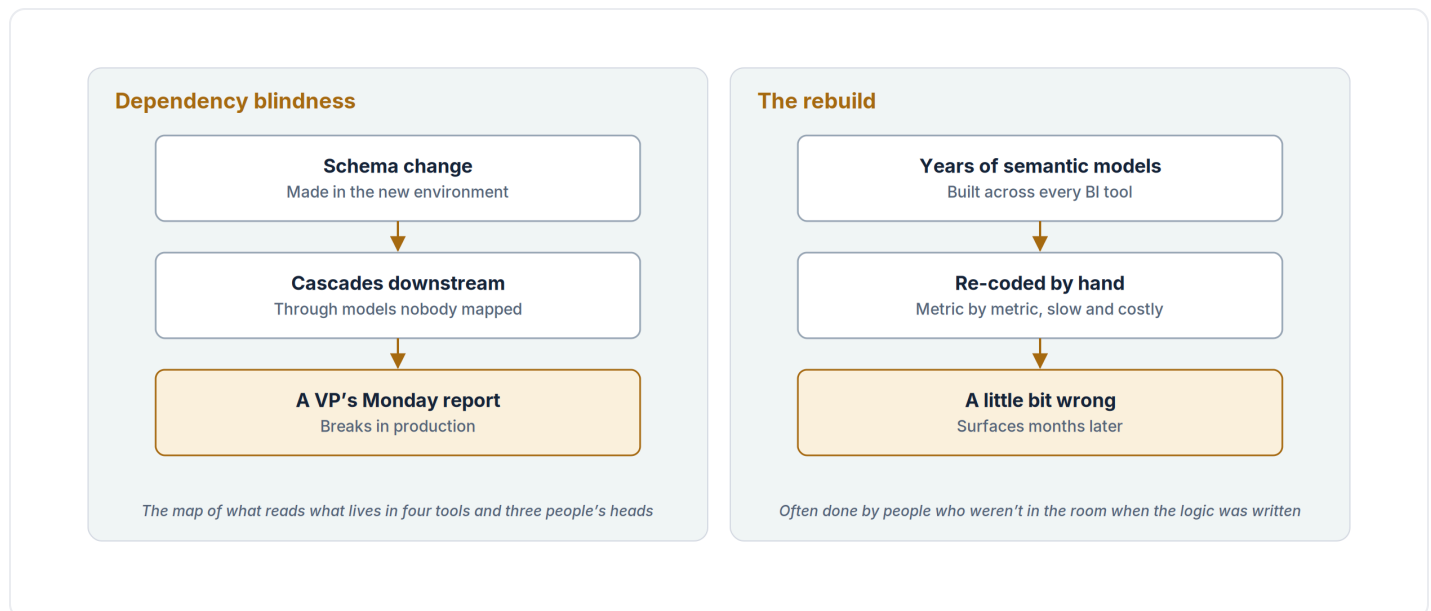
The teams couldn't settle which one the new platform should get. So active customer went into a spreadsheet with 60 other metrics, and the date started moving out.

Migrations fail in two directions

The first failure is dependency blindness. The full map of what reads what lives in four tools and three people's heads, so a schema change made in the new environment cascades into a report somebody's VP looks at on Monday. That gets discovered in production, by the VP.

The second failure is the rebuild. Years of semantic models get re-coded by hand in the new platform's native format, metric by metric, usually by people who weren't in the room when the original logic was written. It's expensive, it's slow, and the output is usually a little bit wrong in ways that surface months later.

Both are avoidable with the metadata you already generate. Most organizations have no way to reach it.

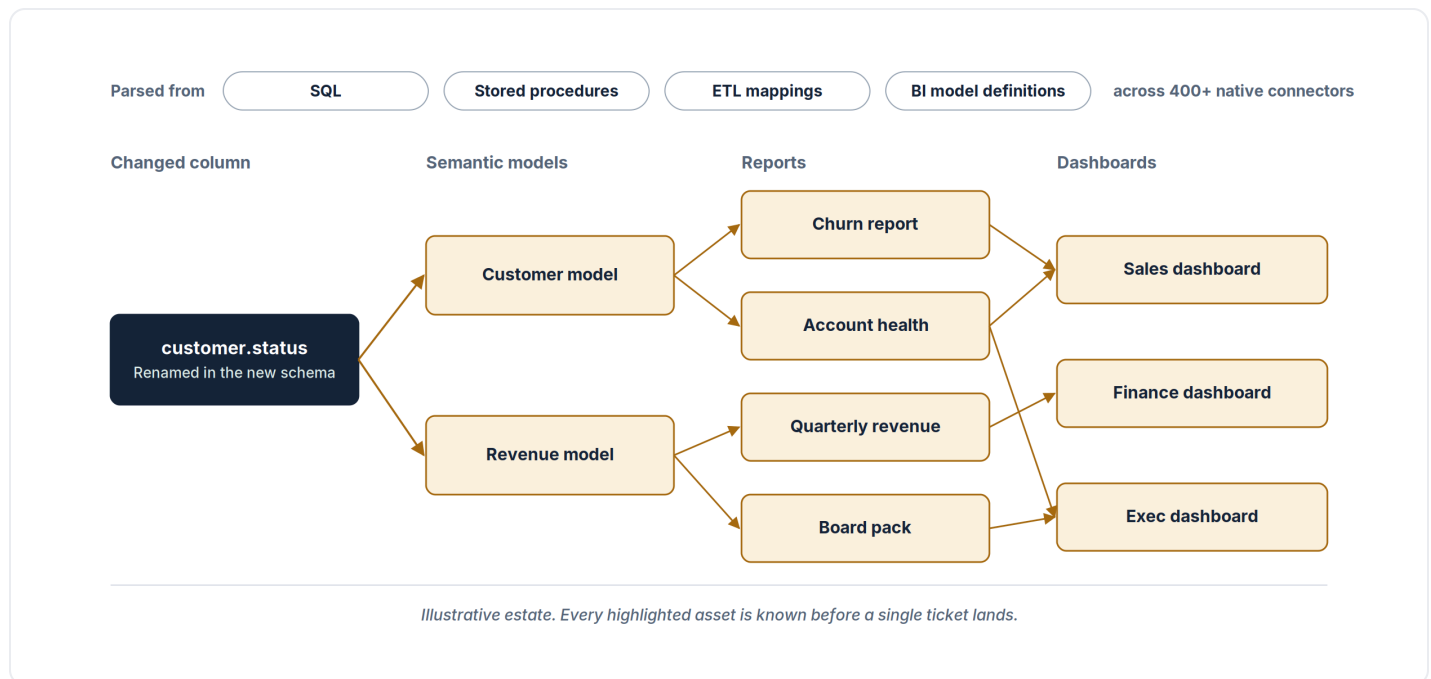


Know what breaks before you move anything

MetaKarta computes lineage by parsing the code that defines the transformation chain: SQL, stored procedures, ETL mappings, BI model definitions. Column-level, cross-system, across 400+ native connectors that span legacy on-premises platforms through modern cloud.

Lineage by design, not by observation. Nothing depends on a job having run during an observation window, which matters in a migration, where half the estate is quiet by the time anyone starts mapping it.

That gives you impact analysis before the first query touches the new environment. Change a column, see every model, report, and dashboard downstream of it. Know the blast radius before the first ticket lands.



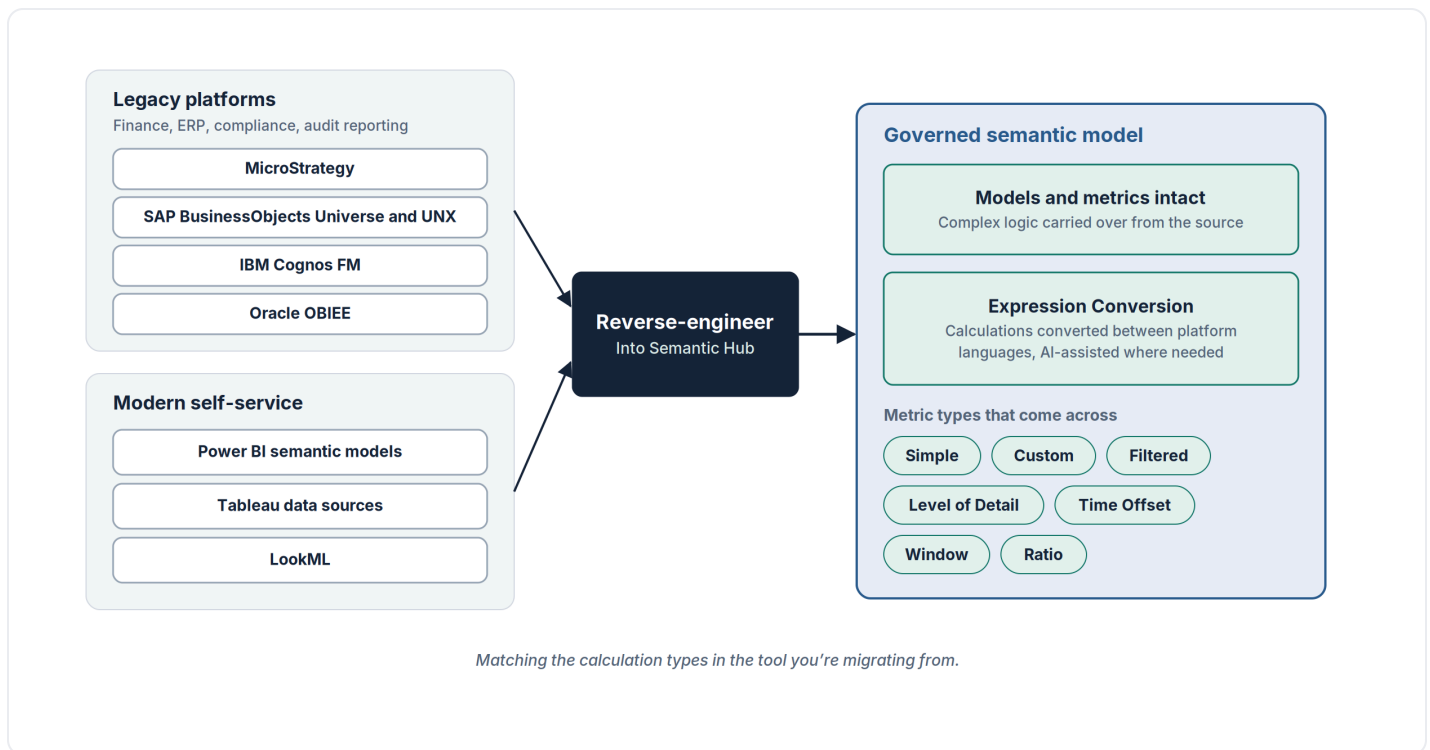
Your existing models are the starting point

Semantic Hub reverse-engineers what your teams already built.

Legacy platforms come across with their complex models and metrics intact: MicroStrategy, SAP BusinessObjects Universe and UNX, IBM Cognos FM, and Oracle OBIEE. These are the systems carrying finance, ERP, compliance, and audit reporting, and they're usually the ones with the least documentation and the fewest people left who understand them.

Modern self-service platforms come across the same way for Power BI semantic models, Tableau data sources, and LookML.

Metric expressions convert along with the models. Expression Conversion translates calculations between platform languages using MetaKarta's built-in translation technology, with AI assistance where a calculation needs it. The metric types come across too: Simple, Custom, Filtered, Level of Detail, Time Offset, Window, and Ratio, matching the calculation types in the tool you're migrating from.



What the import cleans up on the way in

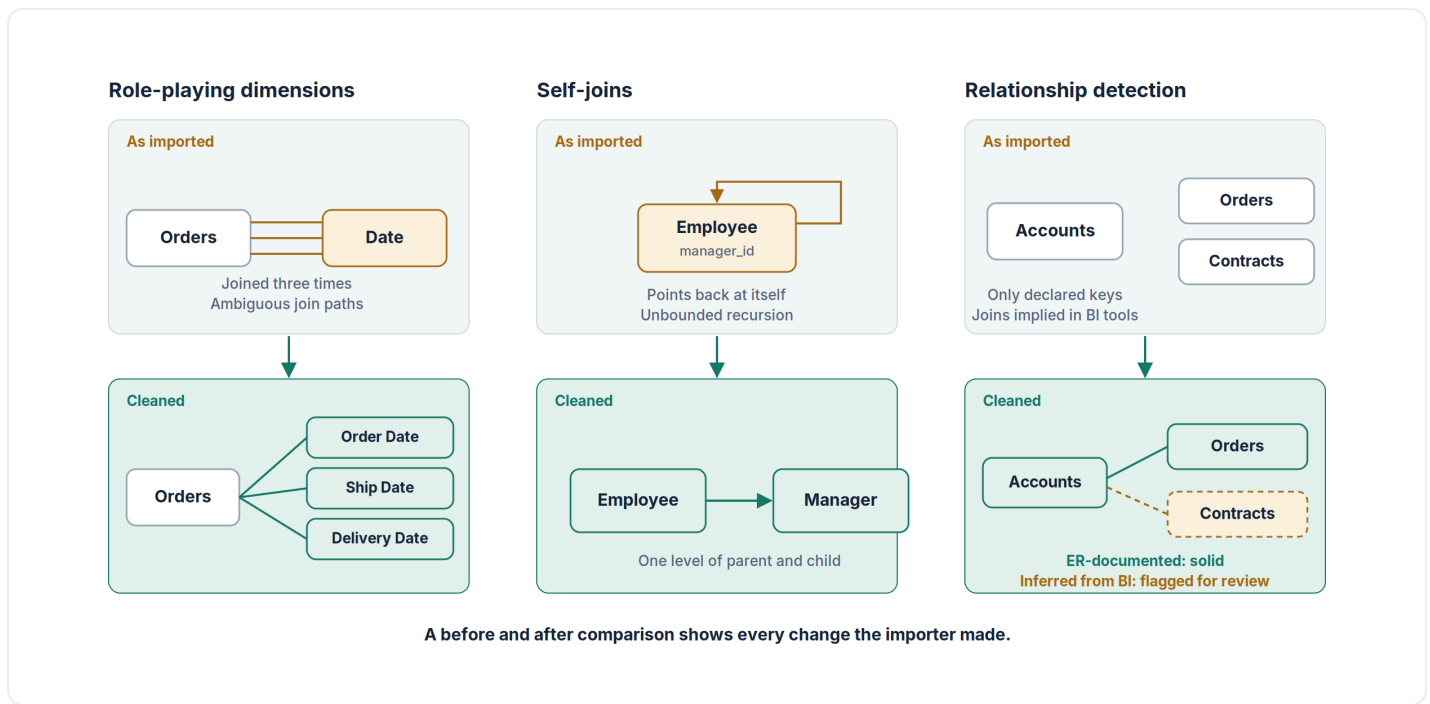
Reverse-engineering a real data estate means handling the shapes that make schemas hard to model against.

A date table joined to the same fact three times, once for order date, once for ship date, once for delivery date, resolves into uniquely named, aliased copies. No manual aliasing, no ambiguous join paths.

An employee table where manager ID points back at the same table flattens into one level of parent and child indirection, so the hierarchy is queryable without infinite recursion.

Relationships get detected past the declared keys. Semantic Hub follows documented relationships from ER diagrams and infers the joins already established inside your BI tools. Every inferred join is marked visually and flagged for review, so an assumption never lands silently in a governed model.

And you can see all of it. Before and after schema comparison toggles between the raw, as-imported schema and the cleaned, flattened, aliased version, so every change the importer made is on screen.

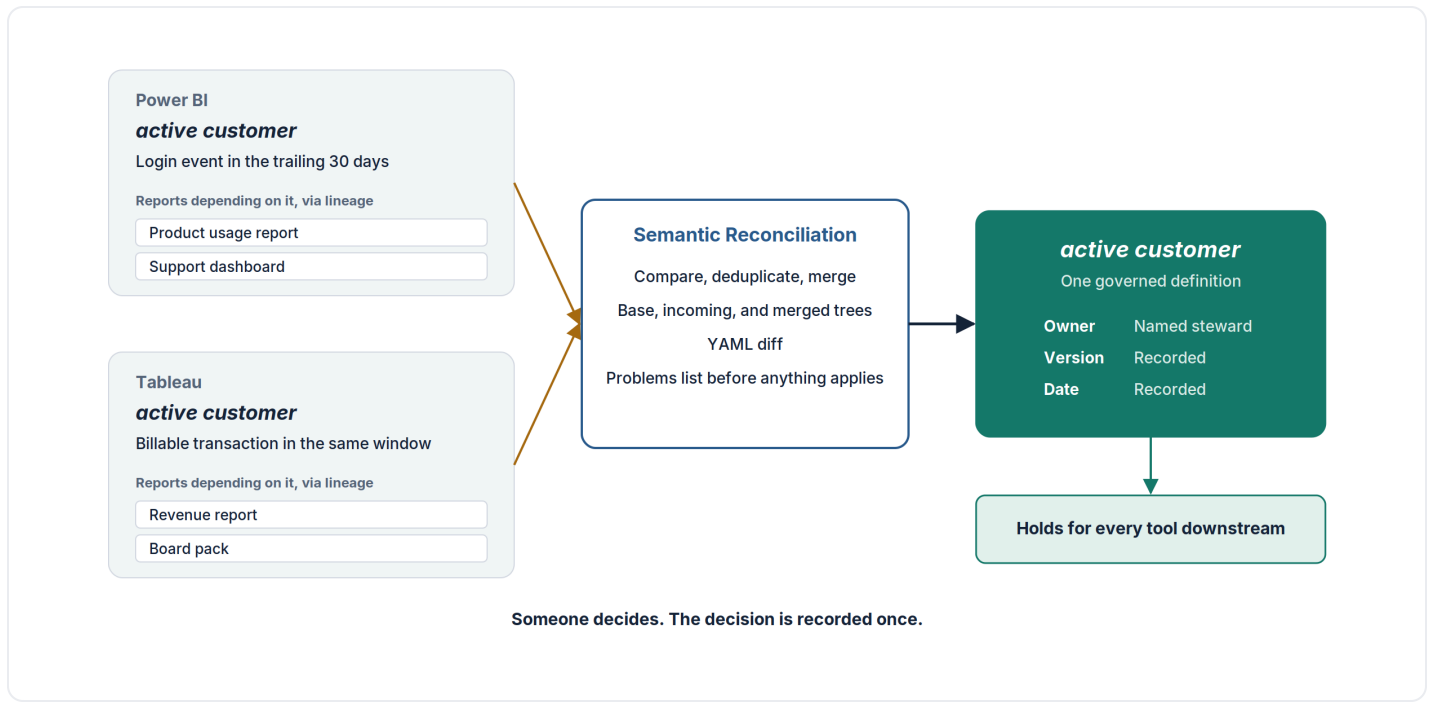


The disagreement is the real work

Importing 60 metrics from two platforms produces multiple versions built on specific business logic. The reconciliation is where a migration actually earns its go-live date.

Semantic Reconciliation compares, deduplicates, and merges semantic models so conflicting definitions get resolved once. Incremental Semantic Catalog Merge imports into an existing catalog rather than starting a new one, and the merge view puts the base catalog, the incoming import, and the merged result side by side as a tree, with a YAML diff and a problems list surfacing every conflict before anything is applied.

Which brings active customer back to the table. The two definitions are visible next to each other, with the reports depending on each one traceable through lineage. Someone decides. The decision gets an owner, a version, and a date, and it holds for every tool downstream of it.

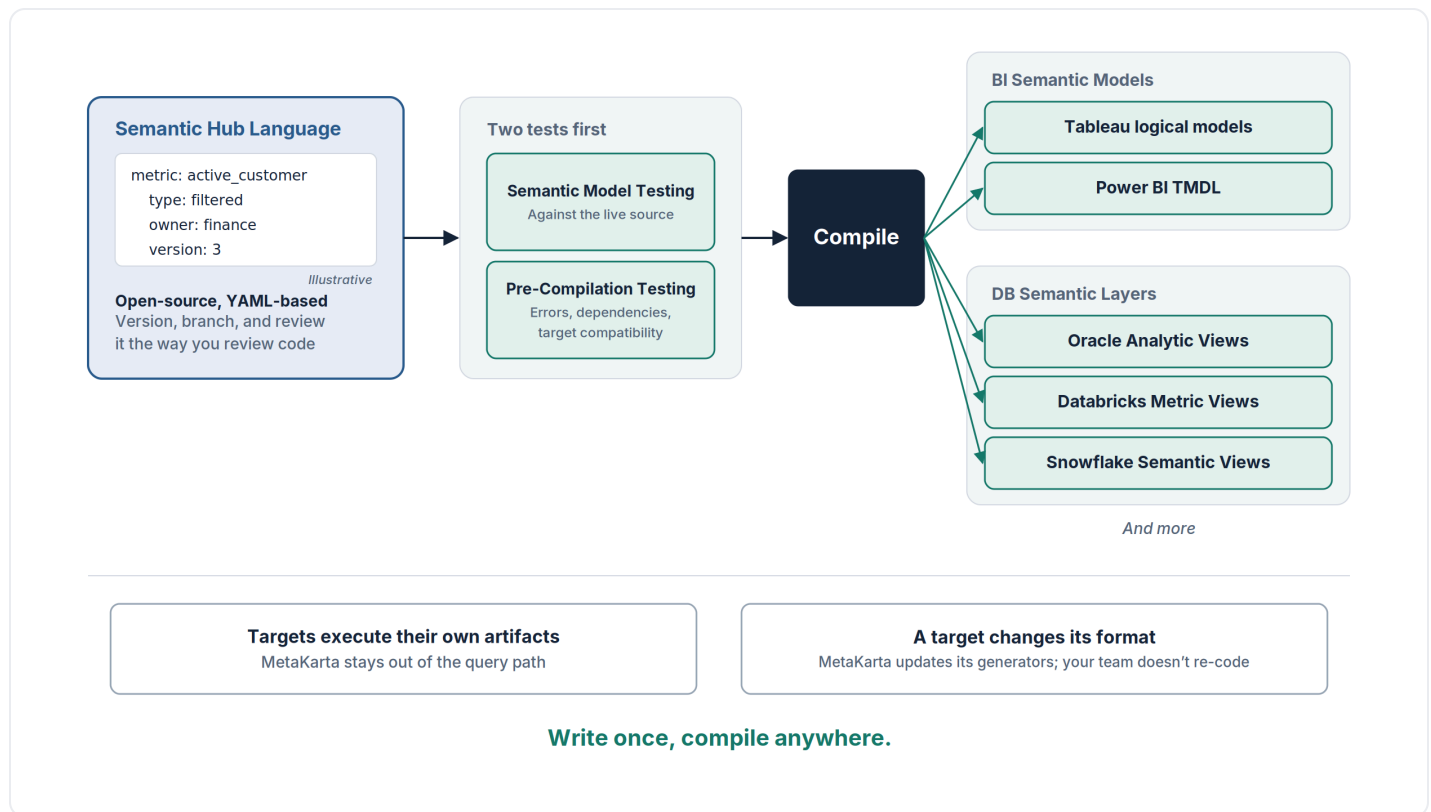


Compile it everywhere

Once the model is governed, the target platform stops being a rebuild project.

Semantic Hub compiles the definition into the native artifact each target requires: Snowflake Semantic Views, Databricks Metric Views, Oracle Analytic Views, Power BI semantic models, Tableau data sources, and more. Definitions are authored in Semantic Hub Language, a YAML-based language, so the governed model stays a text file your organization owns and your team can version, branch, and review the way it already reviews code.

Two tests run before anything deploys. Semantic Model Testing runs your dimensions and metrics against the live data source and compares the result to the expected one. Pre-Compilation Testing catches modeling errors, dependency problems, and target compatibility issues while they're still cheap to fix.



Then the artifacts land in the target systems, where those systems execute them. MetaKarta stays out of the query path, so there's no vendor runtime between a dashboard and its data. When a target vendor changes its format, MetaKarta updates its generators and your team doesn't re-code.

The next migration inherits all of it. Write once, compile anywhere.

Where to start

Pick the metric your last migration argued about longest. Count how many definitions of it exist across your BI tools, then check whether any two of them agree on the filter.

That count is the size of your migration risk, and it's measurable this week.

MetaKarta v12 is available now. Request a demo to learn how to use existing BI semantic models as the starting point for your migration.

[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