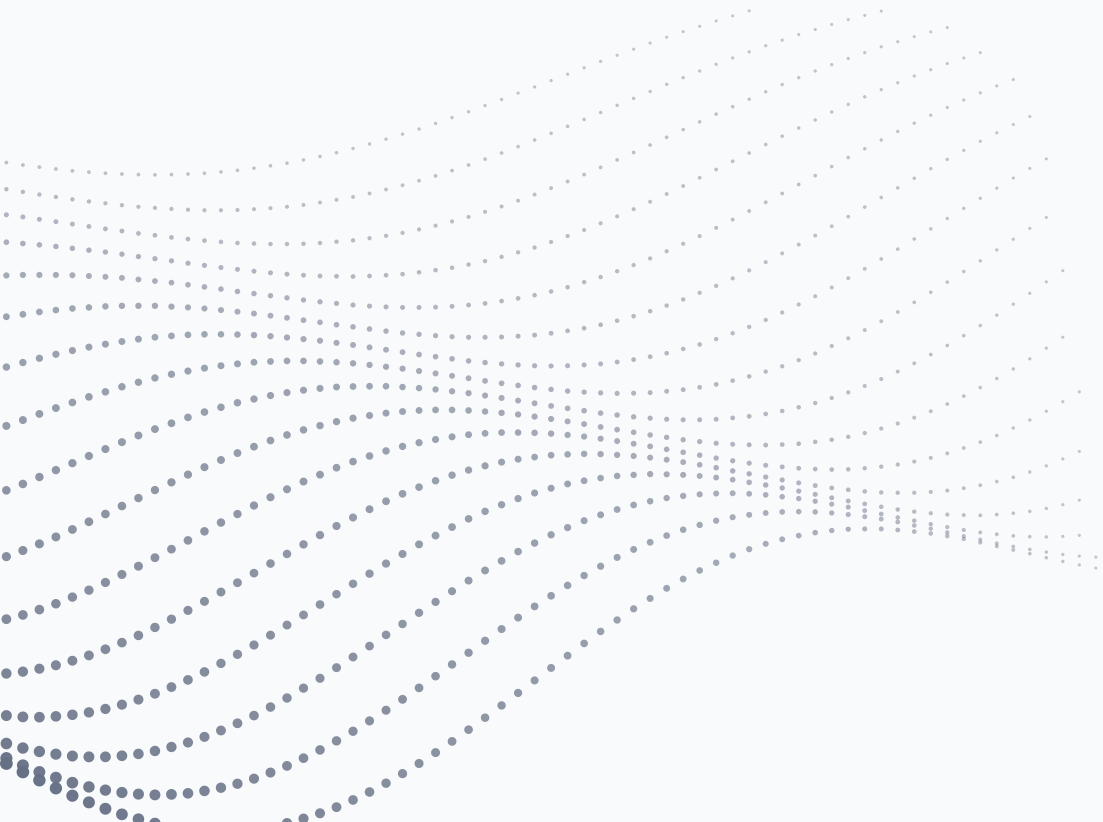
METAKARTA

Whitepaper

# Which governance platform can show your policies operating?

Evaluate data governance platforms against eight practical tests covering policy enforcement, definition accuracy, change control, audit history, and cross-platform consistency. Compare four architectural approaches and learn how to test vendors against your own glossary and data estate.



# Table of Contents


| | |
|---|---|
| Executive summary | 03 |
| What governance has to get right | 03 |
| The approaches | 04 |
| Comparing the approaches | 06 |
| The tests | 06 |
| Test 1: Coverage computed from the estate | 06 |
| Test 2: Bind a term to every field that implements it | 07 |
| Test 3: Show drift between the definition and the measure | 08 |
| Test 4: Run a definition change through review | 09 |
| Test 5: Recall a definition as it stood on a past date | 09 |
| Test 6: Make sensitivity follow the data | 10 |
| Test 7: Bring the program you already built | 11 |
| Test 8: Confirm the approved definition is the one running | 11 |
| Scoring your evaluation | 12 |
| Conclusion | 14 |

# Executive summary

Every governance vendor can show you a glossary, a stewardship workflow, and a coverage dashboard. Most of those demos look complete in 30 minutes, because the demo glossary was written for the demo.

The real test is whether a definition in your glossary can be traced to the fields that compute it, changed through a reviewed workflow, recalled as it stood on a date your auditor picks, and confirmed running in the BI tool your CFO reads.

The tests below separate platforms that can show a policy operating from platforms that can show a policy exists. Each targets a place where governance quietly falls short in production: coverage, binding, drift, change control, point-in-time history, sensitivity, program migration, and reach into the consuming systems.

Four approaches compete for this work. They're described below and compared in one table, because they differ on where governance sits relative to the estate, which decides most of what follows.

Score every vendor on the same tests, against your own glossary, before the renewal date sets the timeline for you.

# What governance has to get right

A governance program holds the organization's agreed meaning: the terms, the policies, the owners, and the rules for who may change what. The hard part is making that meaning hold in four places at once.



**In the estate.**

Every warehouse column, transformation, and report field the policy applies to.



**Over time.**

The definition today, and the definition on any date someone asks about.

**Through people.**

The steward who owns a term today, and whoever owns it after the next reorganization.

**In the consumers.**

The BI tools, the warehouse, and the agents that compute and read the numbers.

Miss any one of the four and the program has a well-maintained record of what should be true, with no way to show it is.

# The approaches

Four approaches are on the market. Three describe where governance sits. The fourth describes how the tooling gets assembled, cuts across the others, and is the one buyers evaluate least.

## Documentation-first governance

A catalog and glossary that describe the estate, with stewardship workflows, ownership, and policy records. Population comes from connectors that register assets and from stewards who describe them.

This approach is mature and well understood, and its workflow and ownership features are usually strong. The limit is position: it sits beside the systems that compute the numbers, so a definition reaches production only when someone reads it and acts on it.

## Governance native to one data platform

Data platforms now ship governance features of their own: tags, access policies, lineage, and semantic objects inside the warehouse or lakehouse. Policies there are enforced by the platform's own engine.

For anything that lives entirely inside that platform, that's a strong answer. Most estates run more than one data platform and several BI tools, and the platform's governance stops at its own boundary.

## A shared repository that harvests, binds, and compiles

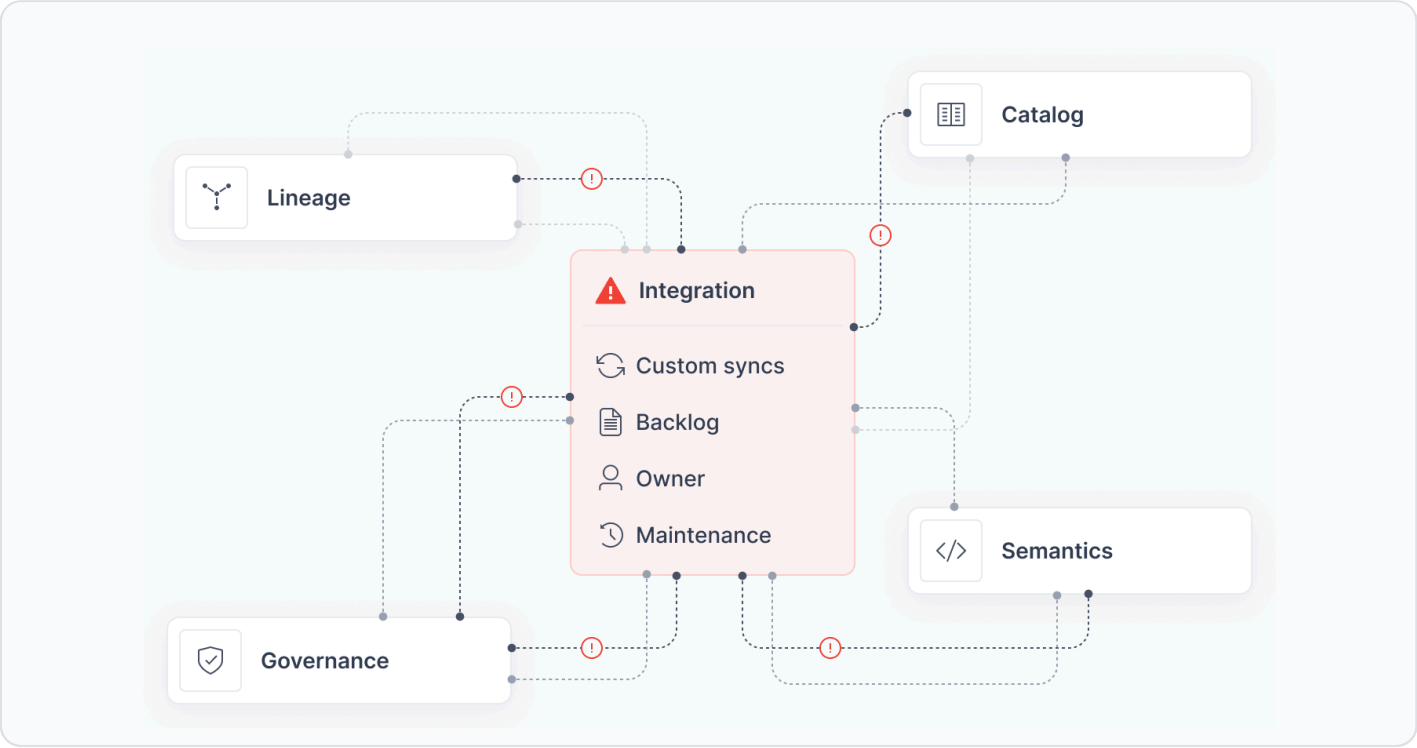
This approach harvests metadata from every system into one repository, binds glossary terms to the fields that implement them, versions every change, and compiles approved definitions into the native artifacts of the consuming systems.

Governance reaches production through the systems' own artifacts, so nothing sits in the query path. The coverage question moves to the connector library and the compile targets: a system is supported or it isn't, and the gaps are visible at evaluation time.

## The assembled solution

This one is a buying behavior, so no vendor presents it and it has no line on anyone's price list. It's also common in production.

You run a catalog for discovery, a governance workflow tool, a lineage tool, and a semantic product for BI. Each keeps its own metadata store and its own model of what a term is, and the program reconciles them by hand or by integration for as long as it owns all four.



# Comparing the approaches

| Approach | Where governance sits | Coverage denominator | Definition reaches consumers | Point-in-time history | Cross-platform |
|---|---|---|---|---|---|
| **Documentation-first** | Beside the estate | Registered assets | When someone acts on it | Varies by product | Yes, as description |
| **Native to one platform** | Inside one platform | That platform's assets | Inside that platform | That platform only | No |
| **Shared repository, compiled** | On a repository harvested from the estate | Harvested assets | Compiled into each supported target | Every governed object | Yes, per supported system |
| **Assembled solution** | Spread across tools | Varies by tool | Varies | Varies | Reconciled by hand |

Check the "Definition reaches consumers" column first. It's the property hardest to add after purchase, and the one a coverage dashboard won't tell you about.

# The tests

Run each of these against your own glossary and your own estate. A vendor's sample glossary tells you little about the terms your stewards maintain.

Each test carries a worked answer from MetaKarta, so every criterion has a concrete implementation attached. Hold every other vendor to the same standard.

# Test 1: Coverage computed from the estate

**Ask for:** the coverage of one executive report you name, measured against the fields that report contains. Then ask what the coverage percentage on the dashboard is a percentage of.

**Why it matters:** coverage measured against registered assets describes the catalog. Coverage measured against harvested assets describes the estate, and only that version tells you where the policy has gaps.

### How MetaKarta answers.

Data Catalog harvests metadata directly from the source systems across 400+ native connectors, including BI tools like Power BI and Tableau. On any harvested object, More Actions and Open Business Glossary returns a report of the glossary terms that define each field, with every undefined field listed. On a Power BI report, that's the coverage of the fields the reader sees.

A well-documented model can also bootstrap a glossary: Bootstrap from model creates terms from an existing model, with term classification or semantic mapping links back to the data elements, so new coverage starts bound.

# Test 2: Bind a term to every field that implements it

**Ask for:** one governed term from your glossary, and every warehouse column, transformation, and BI measure that implements it. Then reverse it: start from a BI measure and ask which term defines it.

**Why it matters:** a glossary term without physical bindings can't be checked against the estate. Divergence between the definition and the measure stays invisible until someone escalates a number.

### How MetaKarta answers.

Term classification and semantic mapping bind glossary terms to the physical fields that implement them. Semantic lineage runs in both directions: definition lookup traces from a field to the term that defines it, and semantic usage traces from a term to every asset that depends on it. Show Term Definitions renders the glossary definitions inside the lineage diagram, beside the data flow.

**A definition you can't trace is a definition you can't trust.**

# Test 3: Show drift between the definition and the measure

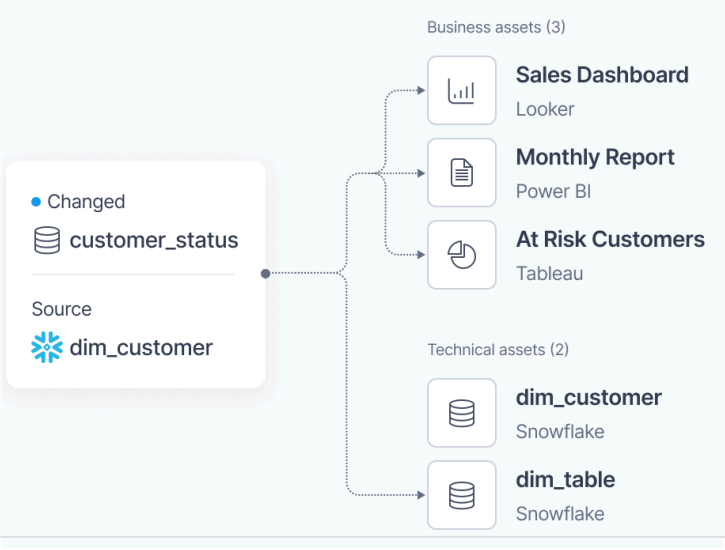
**Ask for:** a term your glossary defines one way and a BI measure that computes it another. Then ask the vendor to show where they diverge, at the expression level, before anyone escalates it.

**Why it matters:** drift is the failure that turns a governance program into a reconciliation service. A platform that can describe both the term and the measure, with nothing connecting them, leaves the comparison to people.

### How MetaKarta answers.

Parser-based lineage traces the BI measure back through the semantic model, the views, and the SQL to its source columns, and the Processes panel shows the expression at every step. With the term bound to the measure, the glossary definition and the computed logic sit on the same canvas, and the filter or join where they part ways is visible.

Impact analysis runs the same trace forward before a schema change deploys, so the drift a change would introduce shows up as a warning first.



# Test 4: Run a definition change through review

**Ask for:** a change to a governed term, proposed by one user, reviewed by a second, approved by a third, and published. Then ask to see what the approver saw before approving.

**Why it matters:** a definition change is a production change. One made by editing a page passes the day it's made and fails the first time an auditor asks who approved it.

### How MetaKarta answers.

Glossary workflow runs Candidate and Draft through Under Review, Approved, and Published, with Workflow Editor, Reviewer, Approver, and Publisher roles assigned as responsibilities. Any user can propose a term when the option is enabled. Every transition carries a logged comment.

Review Changes shows a table of differences between the published definition and the one under workflow. The Published version is what most users see, and it can't be edited directly. Deprecation runs through the same workflow, so terms are retired with a record.

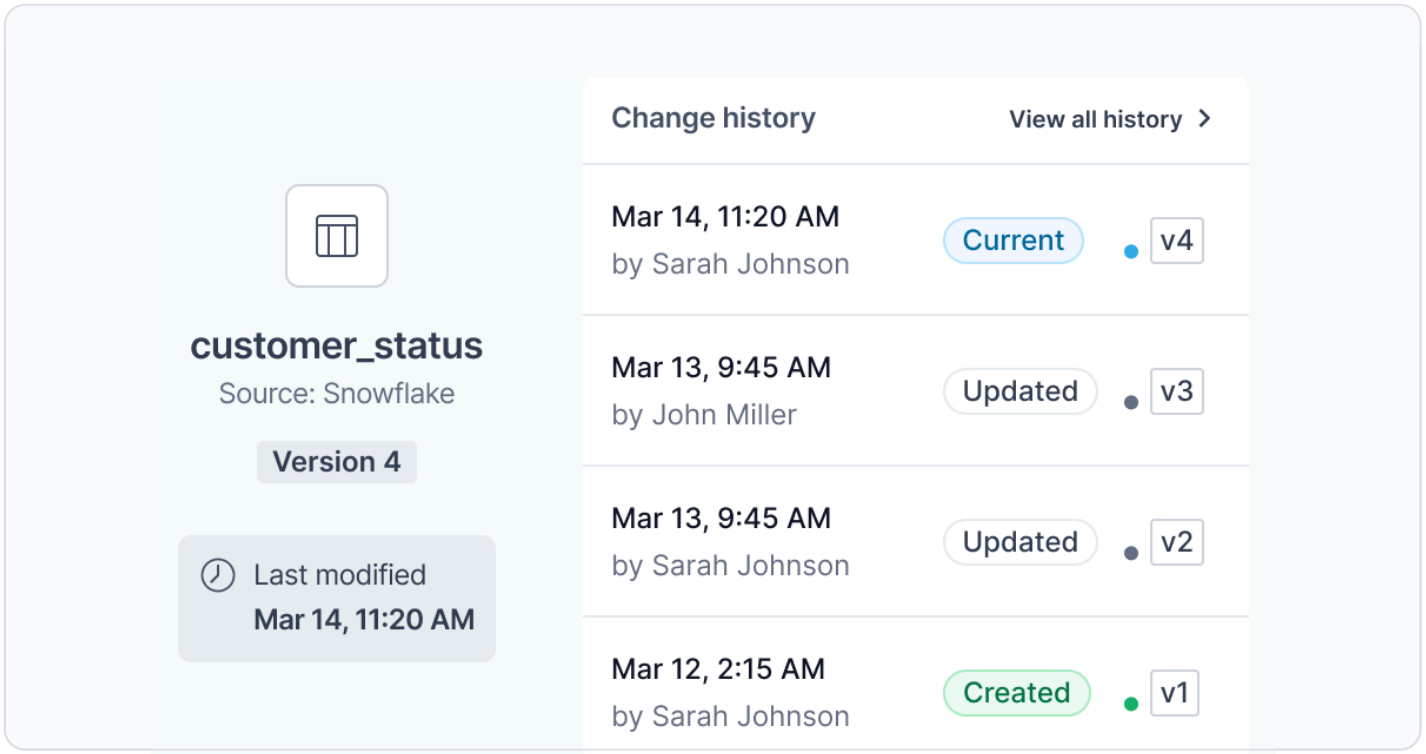
# Test 5: Recall a definition as it stood on a past date

**Ask for:** the definition of one term as it stood at your last quarter close, who changed it since, and what changed. Then ask for the same answer about the lineage behind it.

**Why it matters:** auditors pick the date. A platform that only shows the current state turns every question about a closed period into a reconstruction project.

### How MetaKarta answers.

The History tab on every object records its changes with author and timestamp, and serves as the audit log. Version and configuration management keeps configuration versions, such as a Published and a Development configuration, side by side, and Compare Metadata returns additions, changes, and removals between two versions, color-coded.




**The audit answer is a query.**

# Test 6: Make sensitivity follow the data

**Ask for:** a column classified as personal data in a source system, and the views, extracts, and reports downstream that read it. Then ask which of those carry the label, and who approved each label.

**Why it matters:** sensitive data spreads through derived assets faster than stewards can tag them. Access reviews sample the gaps.

### How MetaKarta answers.

Data classification proposes sensitivity labels from sampled and profiled data, as Sensitivity Label Data Proposed. Labels are also inferred along data flow lineage, as Sensitivity Label Lineage Proposed, through any transformation. A steward reviews and approves or rejects each proposal; a rejection stops the inferred label from propagating further in that direction, and so does data masking detected in the ETL that moves the data.

# Test 7: Bring the program you already built

**Ask for:** an import of your current glossary, with its terms, hierarchy, relationships, stewards, and workflow states intact. Then ask what happens to your existing stewardship assignments.

**Why it matters:** a governance platform that requires rebuilding the glossary by hand discards years of program work. The steward network has to survive the change for the program to survive it.

### How MetaKarta answers.

Glossaries import and export through CSV or Excel workbooks in a format shared by every custom model. Legacy attributes convert on import: a Stewards attribute becomes the Steward object role, Status and State become workflow state, and abbreviations become acronym terms. Steward assignments inherit down the glossary hierarchy, and watchers receive change notifications, so ownership carries forward with the terms.

# Test 8: Confirm the approved definition is the one running

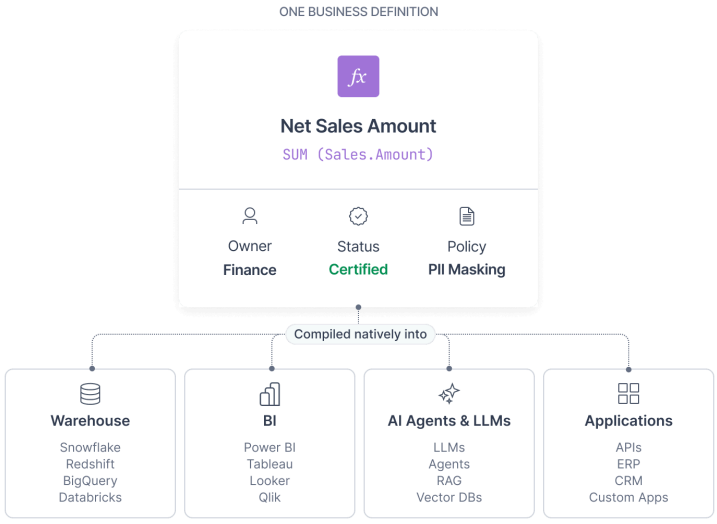
**Ask for:** a governed definition approved in the glossary, then shown running in a BI tool and a warehouse, and read by an agent. Then ask how you'd know if a consuming system ran a different version.

**Why it matters:** this is where governance either reaches production or stops at documentation. A platform that can't place the approved definition in the consuming systems leaves enforcement to each team that builds a report.

### How MetaKarta answers.

Semantic Hub compiles governed definitions into the native artifacts of each target: Snowflake Semantic Views, Databricks Metric Views, Oracle Analytic Views, Power BI semantic models, Tableau logical models, and LookML. Ontology Modeling binds glossary terms to the semantic model's metrics, so the business vocabulary and the compiled calculation stay attached.

Agents read the same definitions. Snowflake Cortex Analyst reads a deployed semantic view natively, and external agents connect to MetaKarta MCP, whose Semantic Hub Tools serve compiled context with per-user access tokens. Every response returns the metadata evidence that produced it.




**Documentation describes what should be true. Compilation makes it true.**

# Scoring your evaluation

| Test | Pass condition |
|---|---|
| 1. Coverage | Coverage of a named report, measured against its actual fields |
| 2. Binding | Every field implementing a term, and the term behind any field |

| Test | Pass condition |
| --- | --- |
| 3. Drift | The divergence between a term and a measure, at the expression level |
| 4. Review | A proposed change reviewed, approved, and published, with a diff and comments |
| 5. Point-in-time | A definition recalled as of a past date, with attribution |
| 6. Sensitivity | Labels proposed downstream from a classified source, each reviewed |
| 7. Program migration | Your glossary imported with terms, stewards, and workflow states intact |
| 8. Reach | The approved definition running in a BI tool and a warehouse, and read by an agent |

Run all eight on your own glossary. A vendor who passes seven and talks around Test 8 has told you where the program stops.

# Conclusion

A governance program is now judged on whether its policies operate in the systems that compute the numbers, on any date someone asks about. Evaluating a platform on its sample glossary tests the easy part. The tests above check whether your own definitions can be bound to the estate, changed under review, recalled as of any date, followed by their sensitivity labels, carried over from your current tooling, and compiled into the systems that consume them.

MetaKarta answers all eight from one architecture: Data Governance, Data Catalog, Data Lineage, and Semantic Hub on one shared metadata repository, harvested from the estate across 400+ connectors. Every governed definition carries its bindings, its owner, its history, and its policy into the systems that use it.

Get in touch to learn more about running these eight tests against your own glossary, starting with the policy your auditors ask about most.

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