THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA

Embedded Linux Conference

# Beyond Vector Search

## Building Knowledge Graphs for Autonomous Infrastructure

# AGENTIC INFRASTRUCTURE

While Infrastructure as Code (IaC) got us to the cloud faster; Agentic Infrastructure Operations ensures we can actually run a complex footprints without drowning in operational complexity. It's about automating Day-2 tasks, and helps to reallocate up to 60% of engineering time from routine maintenance back to core business innovation.

# What does an AI agent need to operate IT infrastructure reliably?

## Context

Understand what a configuration item is and what it means

## Relationships

Know what depends on what — blast radius, impact chains

## Governance

Enforce policies, compliance rules, and change controls

# APPROACH 1 — VECTOR SEARCH

Infrastructure docs, runbooks, and config files are embedded as vectors. Queries retrieve semantically similar documents.

### ✓ Strengths

- Fast semantic search across large collection of texts
- Easy to implement with off-the-shelf embeddings
- Good for free-text runbook lookup
- Scales well for document retrieval

### ✗ Limitations

- No relationship graph — can't trace dependencies
- Cannot answer 'what breaks if X fails?'
- Blind to topology and blast radius
- Config drift invisible — embeddings go stale

*Verdict: Great for knowledge retrieval. Insufficient for autonomous decision-making.*

# Context Aggregation

## Unlocking Aggregation Reasoning

On complex cross-document queries requiring structural grouping and counting (e.g., "Which departments have outdated compliance documents?"), standard semantic search drops under 20% accuracy because it cannot synthesize unstructured text. GraphRAG utilizes pre-computed community summaries to deliver highly accurate context windows.

**<70%**

MULTI-DOCUMENT REASONING ACCURACY

# APPROACH 2 — GraphRAG

Infrastructure entities and their relationships are modeled as a knowledge graph. AI traverses edges to reason across dependencies.

### ✓ Strengths

- Full dependency traversal — blast radius in seconds
- Context-rich: apps, infra, teams, policies in one graph
- Powers autonomous impact analysis and change planning
- Compliance posture queryable at any time

### ✗ Limitations

- Complex to build & maintain the graph schema
- Requires continuous graph population from live infrastructure
- High operational burden: GraphDB + ETL pipelines
- Keeping graph current with infrastructure drift is hard

*Verdict: Architecturally superior — but the graph is hard to build, populate, and keep accurate.*

# Cost Analysis


*GraphRAG incurs high up-front entity extraction costs, but search patterns like Microsoft's LazyGraphRAG reduced active runtime token inferencing costs by over 700x.*

Source: https://www.microsoft.com/en-us/research/blog/lazygraphrag-setting-a-new-standard-for-quality-and-cost/

# APPROACH 3 — Resource Graph

Rescile's Unified Configuration Server (UCS) constructs a entity-relationship diagram based on configuration files and enables multi-hop logic traversals that explicitly trace inherited policies, ownership structures, and data flows.

**Precision**

Deterministic (Follows hard code/graph links; mathematically precise topology).

**State Awareness**

Real-time (The graph maps the *structure*, REST fetches the *live state* on-demand).

**Hallucination Risk**

Extremely Low (The agent is restricted to verified graph structures and API payloads).

**Multi-Hop Reasoning**

Native (The graph is built specifically to traverse deep dependency chains).

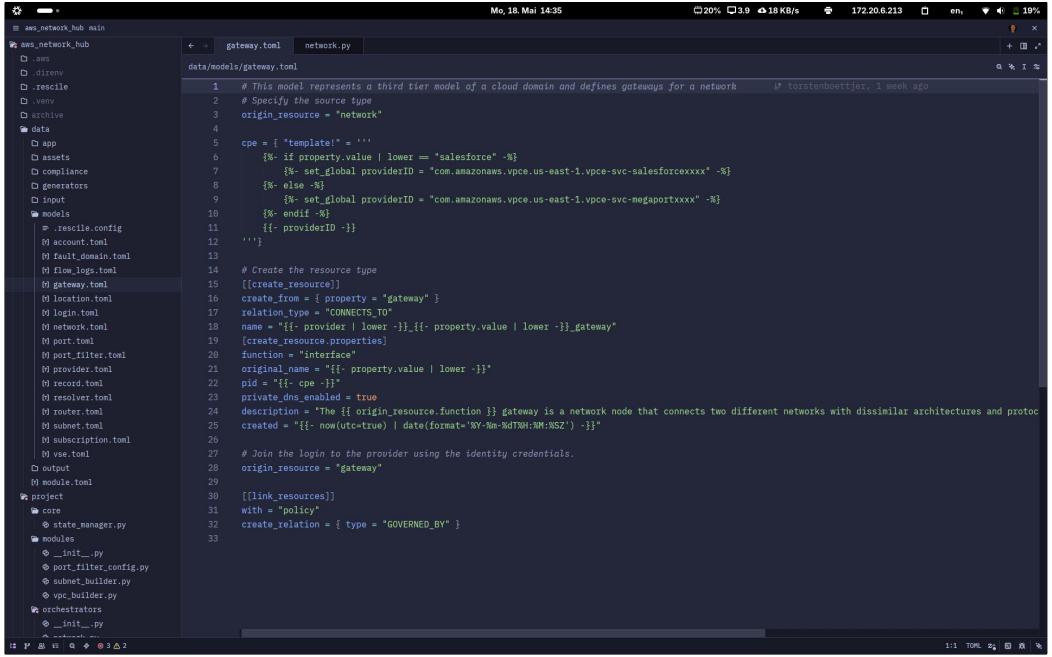
# APPROACH 3 — Resource Graph

Rescile's UCS transforms static configuration files into a dynamic knowledge graph that maps the entire infrastructure as an interconnected entity-relationship network, instantly tracing inherited policies, ownership structures, and data flows.

| | |
|---|---|
| **01** Ingestion | Declarative TOML/JSON/CSV blueprints · No LLM tokens required · State Sync · Git-Versioned |
| **02** Storage & Indexing | In-memory JSON graph · Transient properties · GraphML export · GraphQL + MCP server |
| **03** Query Flow | Typed MCP tool calls · Full dependency traversal · Always-live data · Blast-radius in ms |
| **04** Generation & Governance | Native IaC generation · Separated compliance layer · OSCAL auto-export · Deterministic plans |

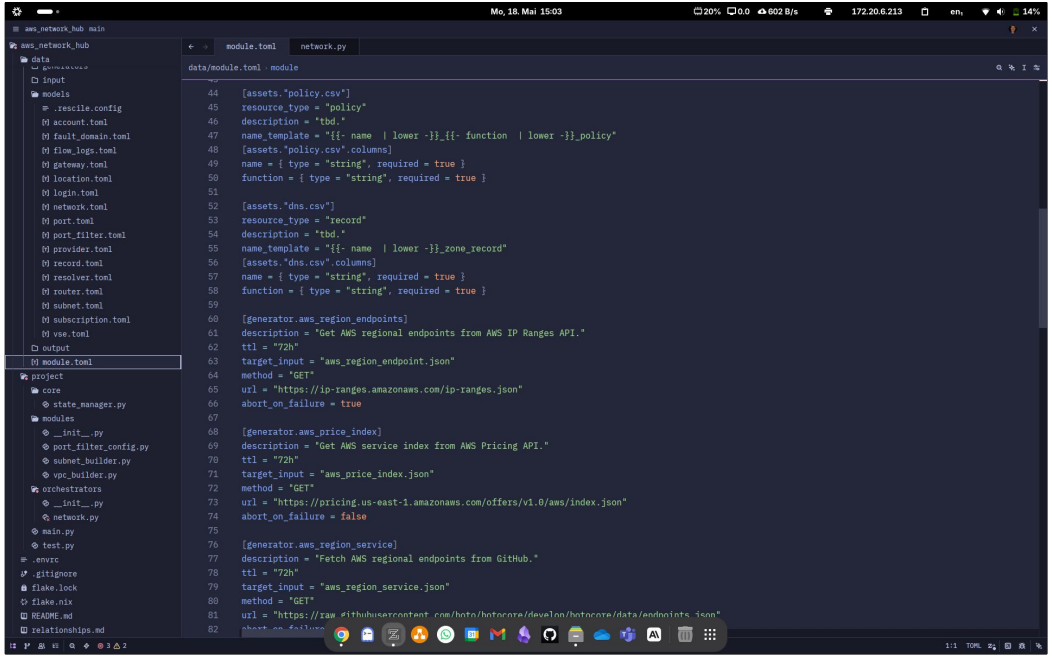
# Ingestion

The framework ingests flat files. Instead of calculating cosine similarities, it parses the explicit declarations (e.g., a CSV mapping networks, a TOML file defining container specs). It instantiates them as nodes within a lean, in-memory JSON object graph.



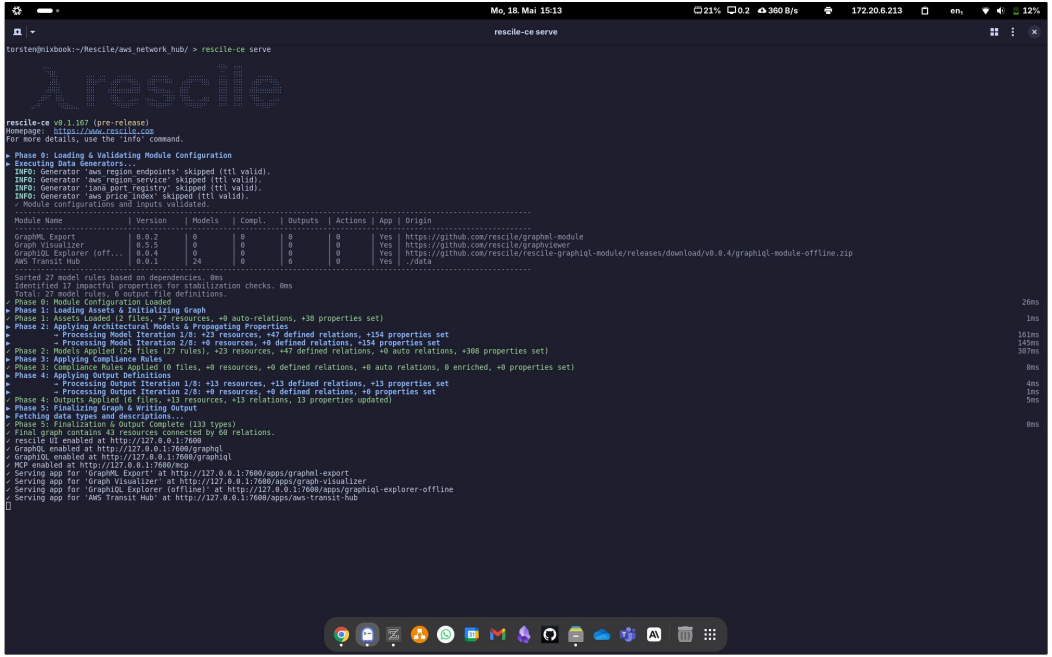
# Ingestion

The UCS bridges the gap between static engineering files and live enterprise realities providing a true operational context. This is achieved, with the server acting as a universal ingestion layer. It collects and harmonizes resource information from across the organization's.



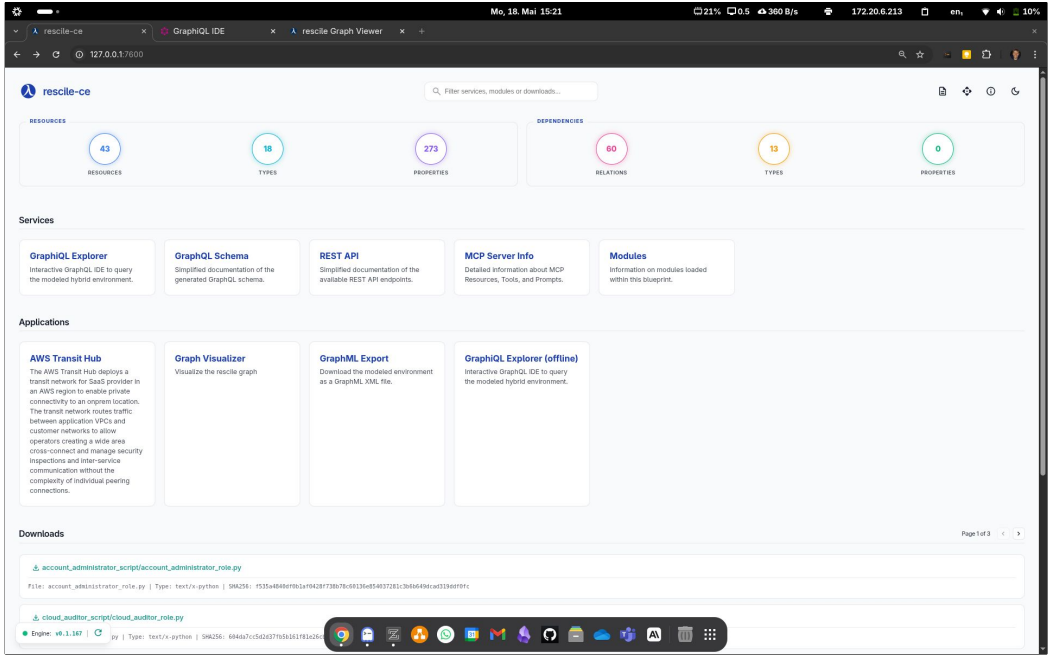
# Storage & Indexing

An in-memory engine resolves the templates and blueprints, linking the resources deterministically. If a TOML specifies that App-Server-1 requires Subnet-A, the engine draws a hard dependency edge in the knowledge graph as an in-memory, structural representation of the target infrastructure environment.



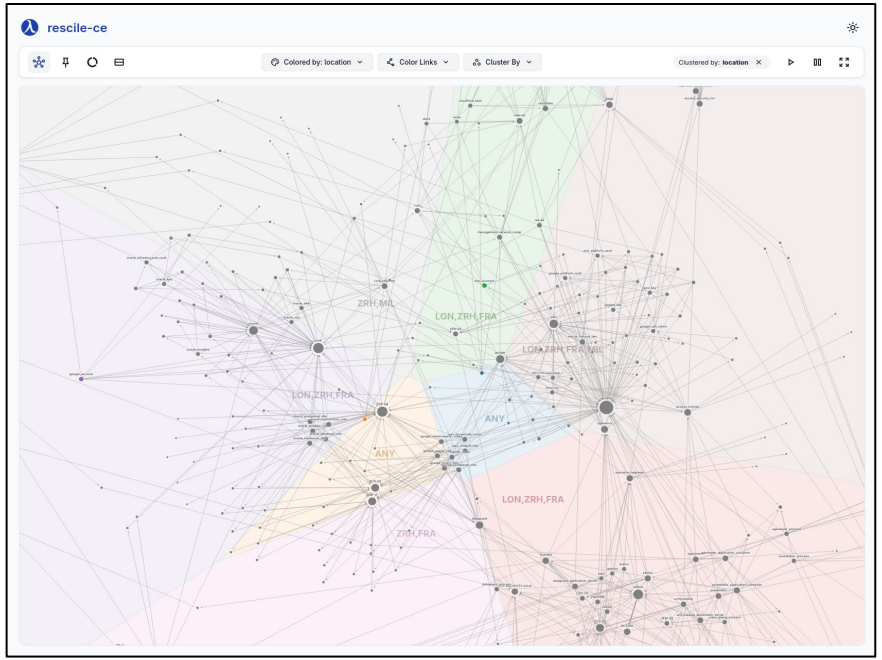
# Storage & Indexing

The core engine enables a highly modular ecosystem. Infrastructure operators can dynamically extend the system's core capabilities by deploying plug-and-play applications. This allows organizations to build out customized, self-service portals and backend engines for specific infrastructure domains



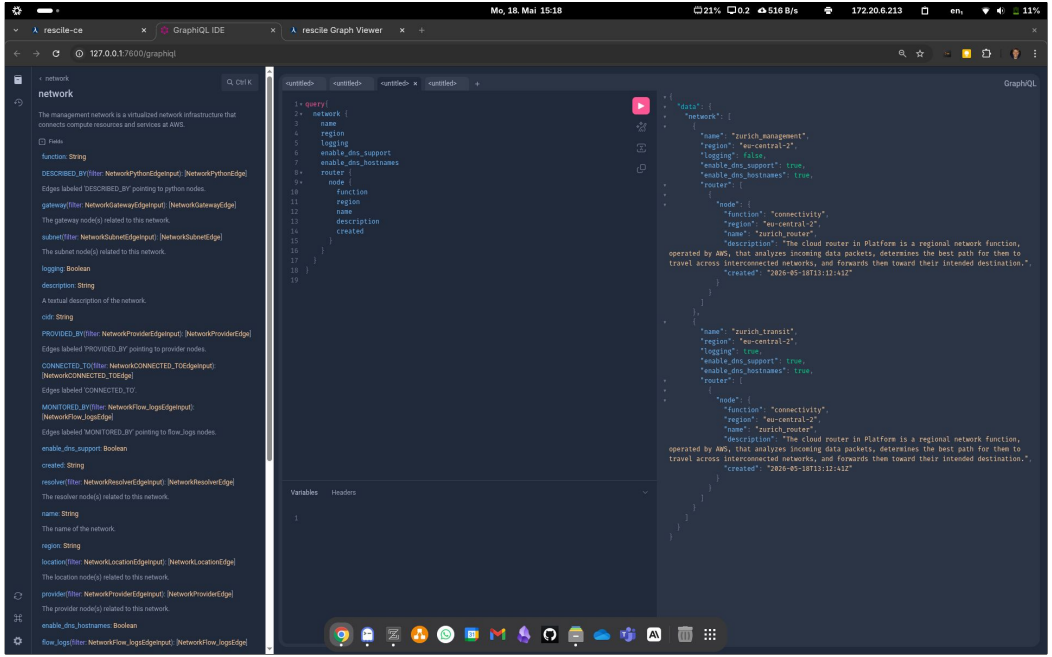
# Query Flow

The design eliminates the friction of adopting advanced infrastructure management by anchoring the entire platform in open-source standards. For IT managers, this ensures your team can leverage existing skillsets and tools without being locked into a proprietary ecosystem.



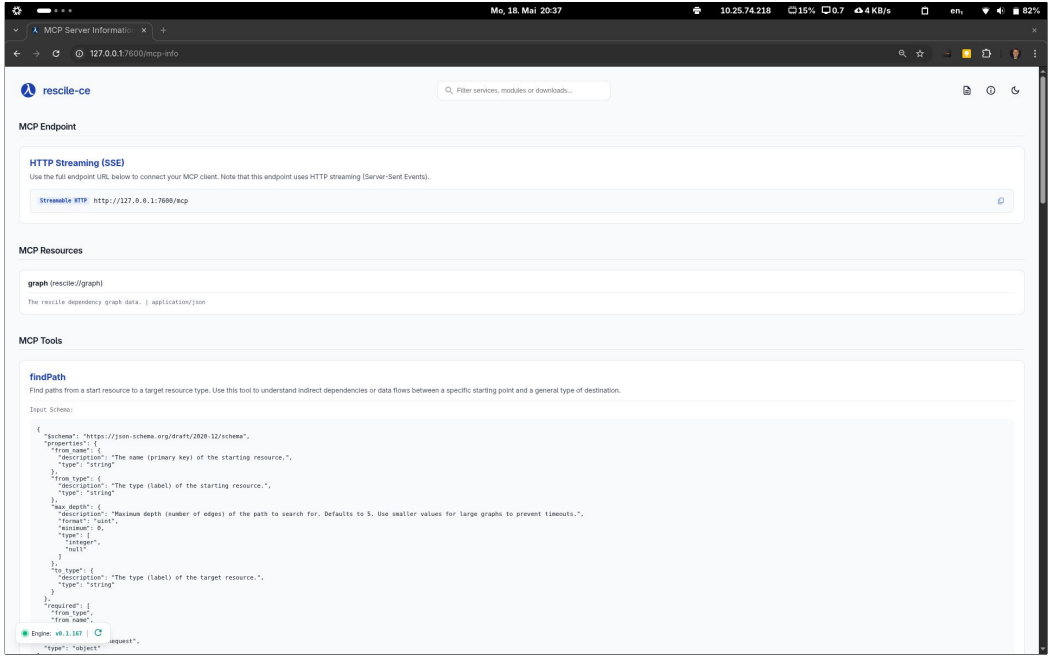
# Query Flow

Rescile diverges from traditional vector-based RAG by querying a static vector store. The framework traverses the in-memory graph to identify the relevant nodes, triggers REST API requests directly to the underlying systems or cloud providers, and hydrates the graph nodes with live status metrics, state flags, and real-world configurations.



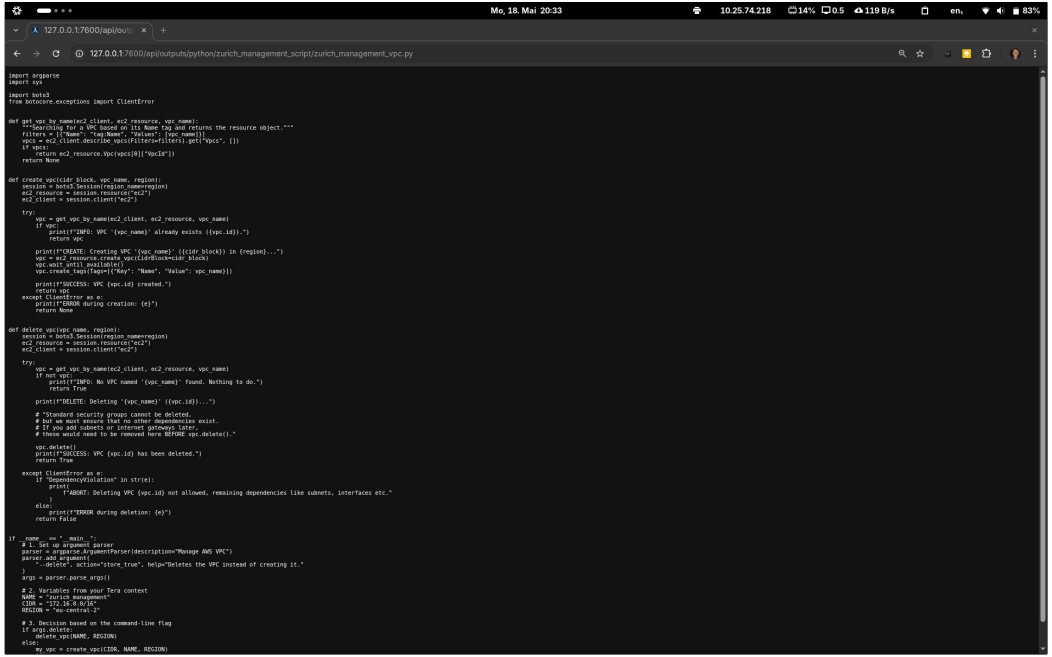
# Query Flow

To enable agentic operation without building proprietary, brittle integrations, Rescile exposes the graph through the Model Context Protocol (MCP). As an open, industry-standard interface designed specifically for AI models, MCP acts as a universal data fabric. It allows both internal native agents and external LLM ecosystems to seamlessly consume and manipulate infrastructure state.



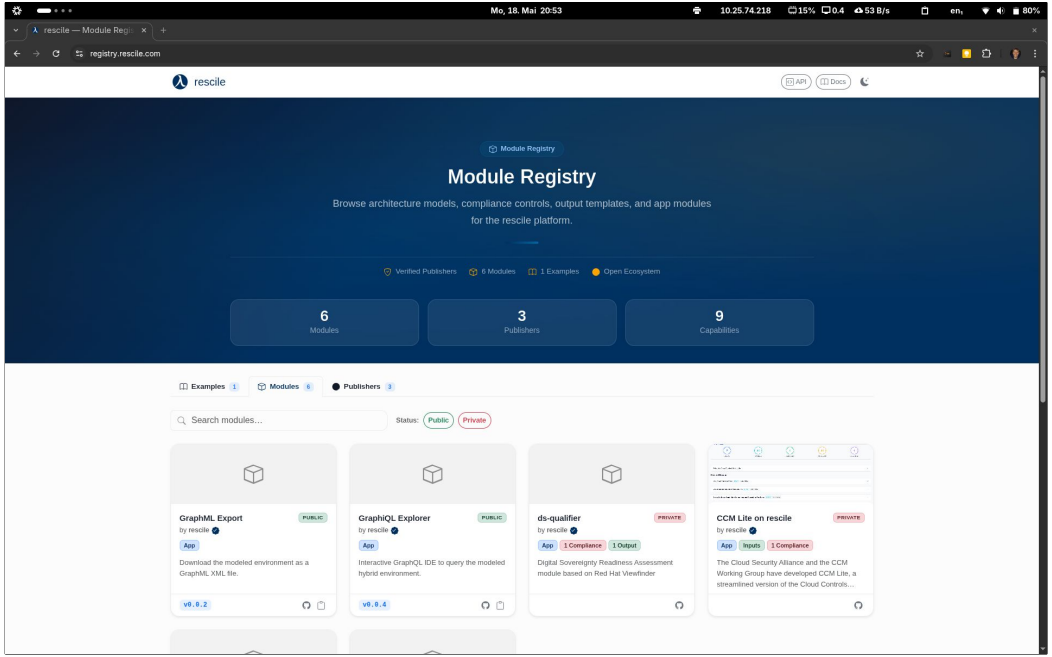
# Generation & Governance

An accurate graph is only as good as its real-world implementation. To ensure 100% reliable execution, Rescile's engine takes the relationships, dependencies, and policy guardrails validated in-memory and compiles them into concrete, deployable artifacts.



# Generation & Governance

To accelerate time-to-value and eliminate redundant engineering effort, Rescile features a centralized Blueprint Registry. This registry acts as a secure, shared repository where operators and community members can publish, discover, and distribute pre-built blueprint modules and architectural templates.



# Thank You

Download
https://www.rescile.com