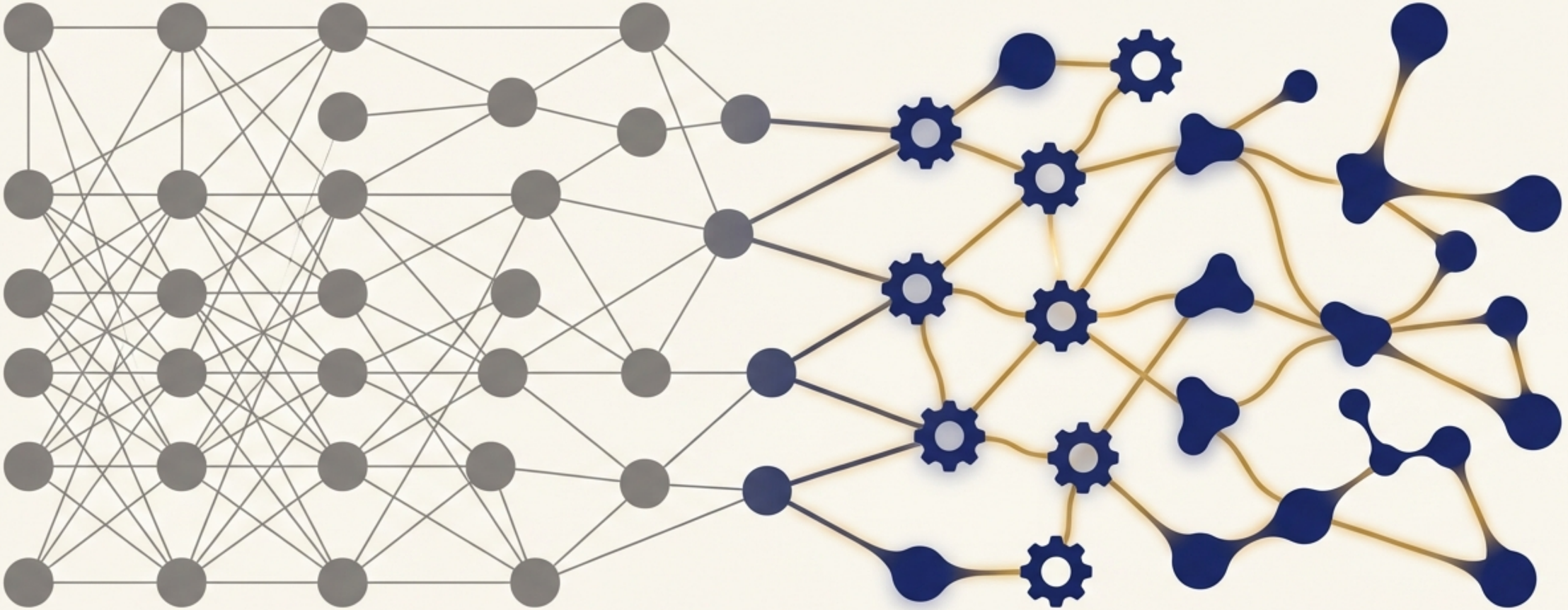


From Linked Data to Executable Knowledge

A New Paradigm for Dynamic, Self-Managing Information Ecosystems

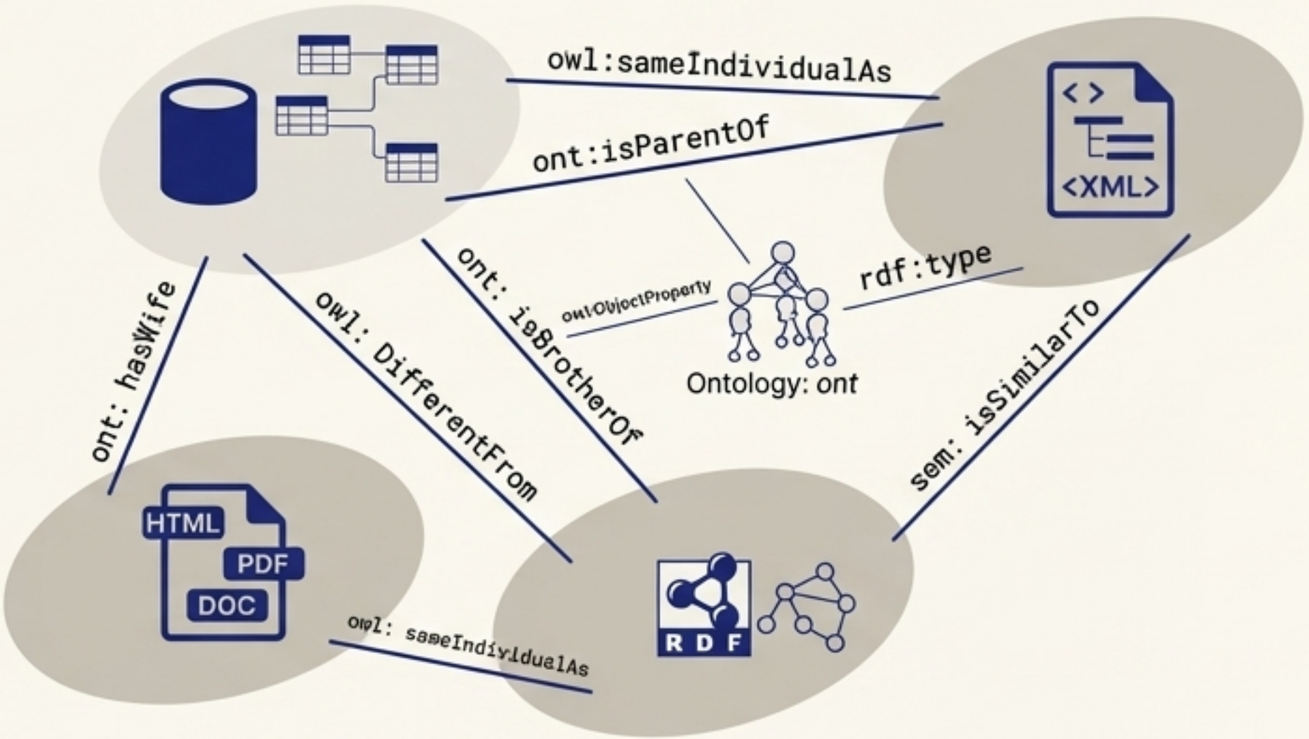
Our Web of Data is Built on Static Bridges

The Power of Linked Data

Linked Data transforms the web from a collection of documents into a unified web of data. By creating semantic connections between heterogeneous sources, it allows machines to explore and integrate information in powerful new ways.

> With linked data, when you have some of it, you can find other, related, data.
>
> — *Tim Berners-Lee*

The 'Data Islands' Metaphor

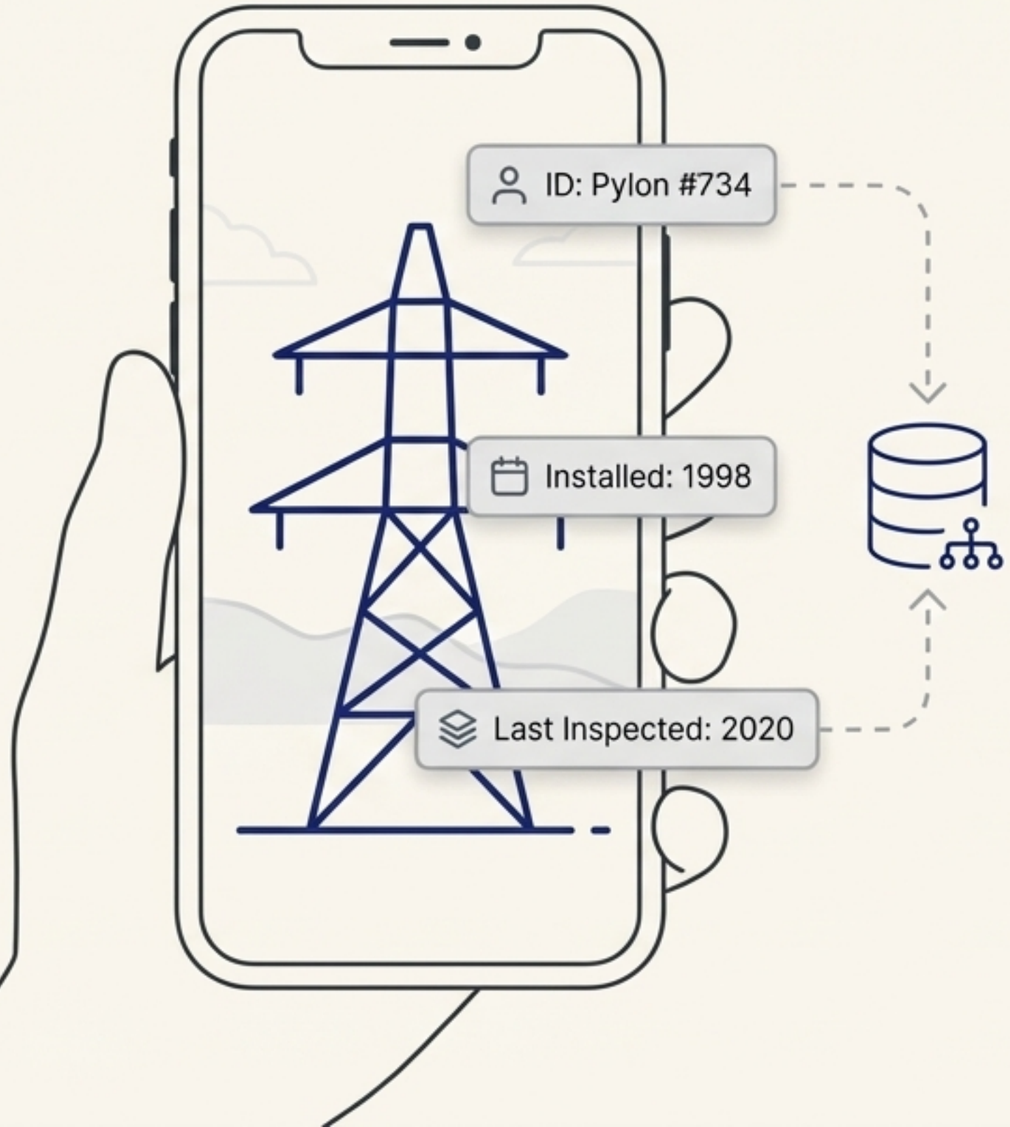


THE CORE LIMITATION

The bridges connect pre-existing facts. The data on the islands is static. We can find what *is*, but we cannot compute what *could be*.

Augmented Reality: A Magic Lens on a Pre-Written World

Core Concept: Mixed Reality merges the real and virtual, allowing digital information to overlay the physical world. The common vision is the phone as a “magic lens,” supplementing what we see with information from Points-of-Interest databases.

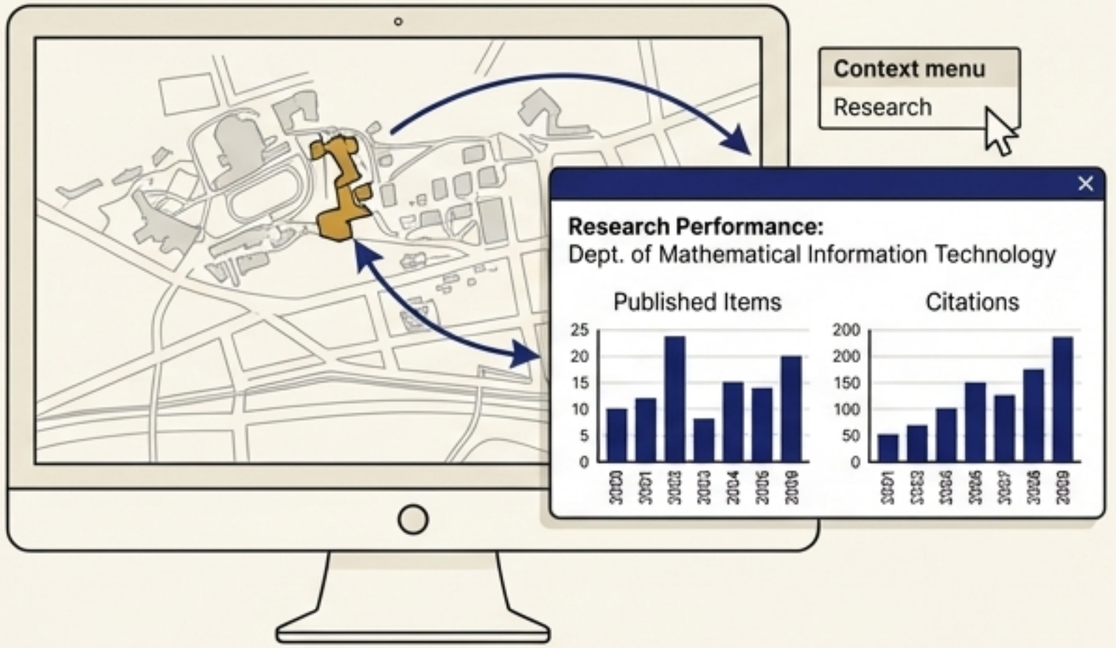


What if the lens could not only see data, but **ask questions** and get **live, computed answers**?

The Vision: Making Reality Itself Computable

We propose “Executable Reality,” an enhancement of Mixed Reality where a user’s focus can trigger on-the-fly Business Intelligence **computations** about real-world objects.

A maintenance engineer points their device at a power line and implicitly requests the last 24 hours of performance statistics, computed in real-time.

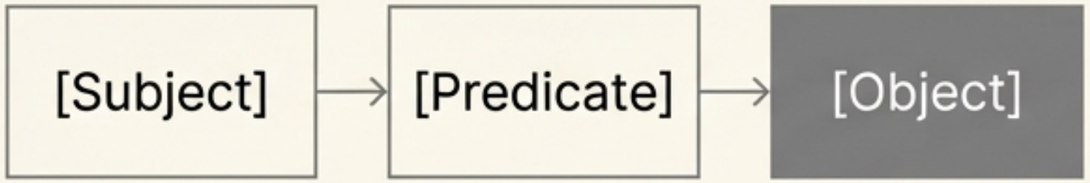


A user selects a university building and the context “research” to generate an on-the-fly report of that department’s research performance.

This user-initiated, context-aware trigger is called an “Executable Focus.”

The Breakthrough: From ‘What Is’ to ‘How to Find Out’

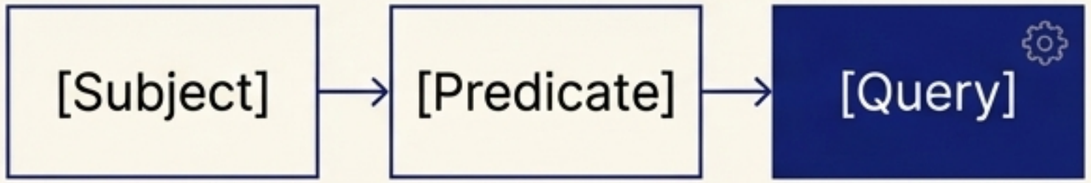
Traditional RDF Triplet




```
:John :isInLoveWith :Mary.
```

Represents a static, known fact stored in the graph. The object is a fixed value.

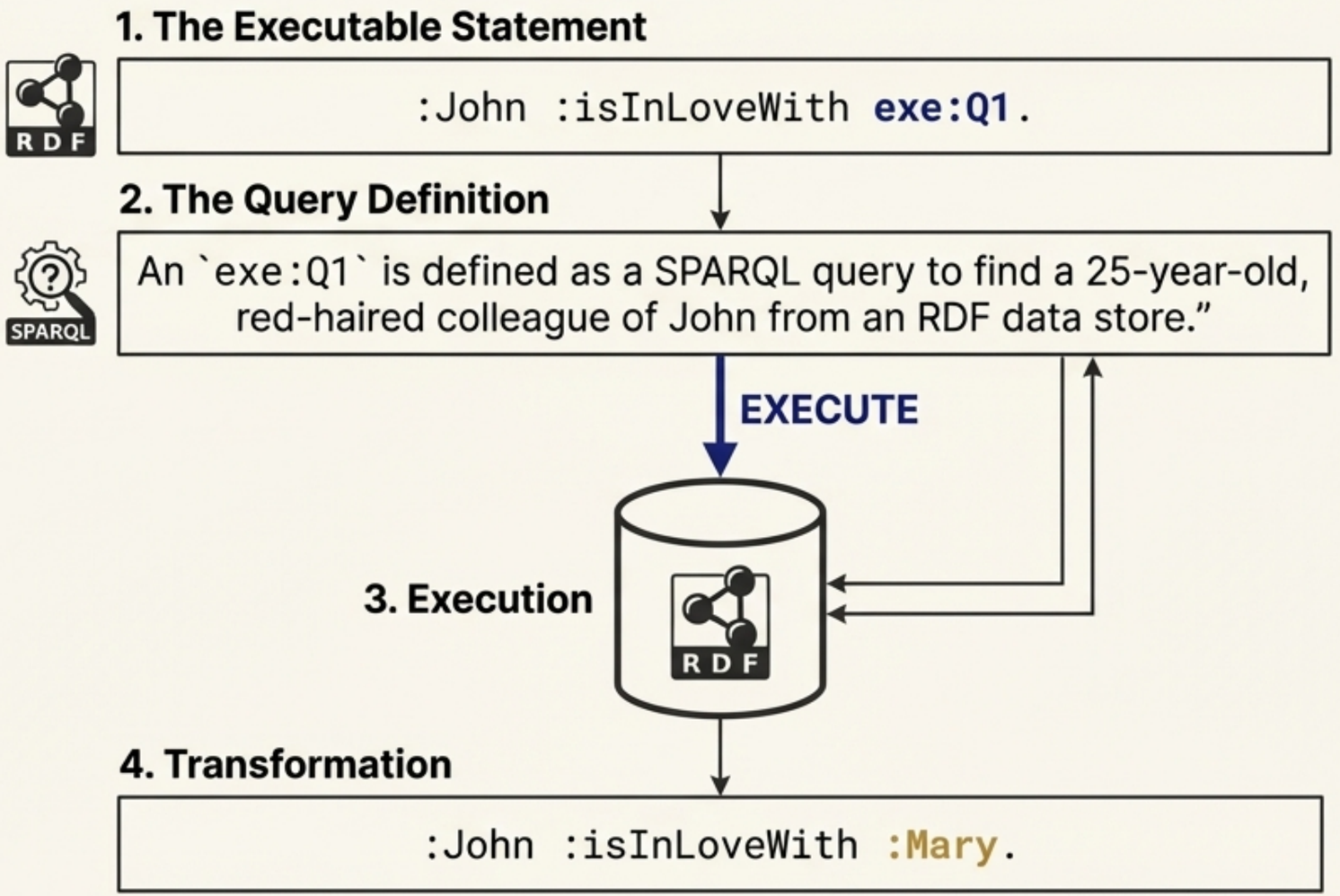
Executable Knowledge Triplet




```
:John :isInLoveWith exe:Q1.
```

Represents a *procedure* to discover a fact. The value is computed ‘on-the-fly’ by executing the query when the property is accessed.

How It Works, Example 1: Executing a SPARQL Query



Key Insight: The knowledge graph becomes dynamic. If the underlying data changes, the *same* executable statement will yield a *new* result without any change to the semantic layer itself.

How It Works, Example 2: Unifying with Relational Databases

1. The Executable Statement

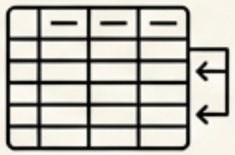

```
:AI_Department :hasAvgYoungDoctStudentsPerformance exe:Q2.
```

2. The Query Definition

```
`exe:Q2` is defined as an SQL query:
  SELECT AVG(JournalPapers) FROM AI_Department
  WHERE Title = 'PhD_Student' AND Age < '30';
```

EXECUTE

3. Execution



4. Transformation

```
  :AI_Department :hasAvgYoungDoctStudentsPerformance "7".
```

Key Insight: This is not just data linking; it is *computation linking*. We can **embed live analytics from legacy SQL systems** directly into the knowledge graph.

How It Works, Example 3: Invoking BI Software as a Service

1. The Executable Statement

```
:AI_Department :hasBusinessIntelligenceReport exe:Q3.
```

2. The Query Definition

`exe:Q3` is an S-APL query that calls a Java software module (UnitReportGenerator) with specific parameters (e.g., time window, data source).

3. Execution

4. Transformation

```
:AI_Department :hasBusinessIntelligenceReport <…/reports#rep2008-2010>.
```

Key Insight: Executable Knowledge can orchestrate complex, stateful operations, turning the knowledge graph into a dynamic application integration and automation layer.

The Broader Vision: From Managing Knowledge to **“Knowledge Computing”**

Knowledge Computing

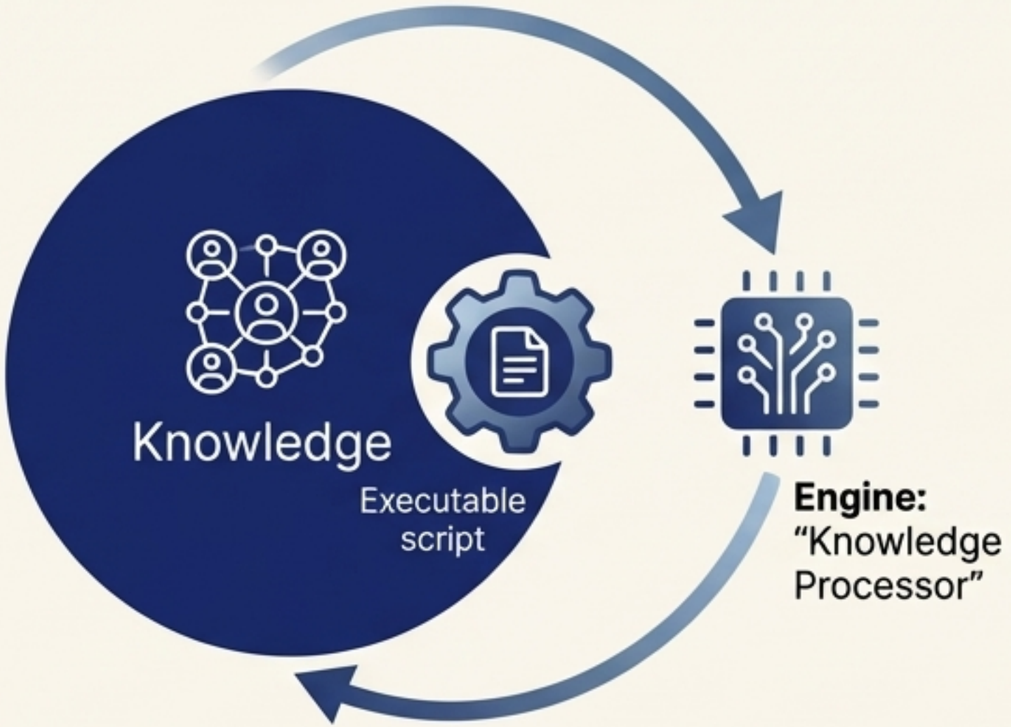
The process of executable knowledge (self-)management.

The Knowledge Store

Contains both domain knowledge and executable scripts for its own management.

A “Knowledge Processor”

An engine or agent that executes these scripts on demand.



INSTRUCTIONS FOR...

- Reasoning
- Querying
- Calling External Functions
- Mining
- Evaluation
- Diagnostics
- Prediction
- Integration

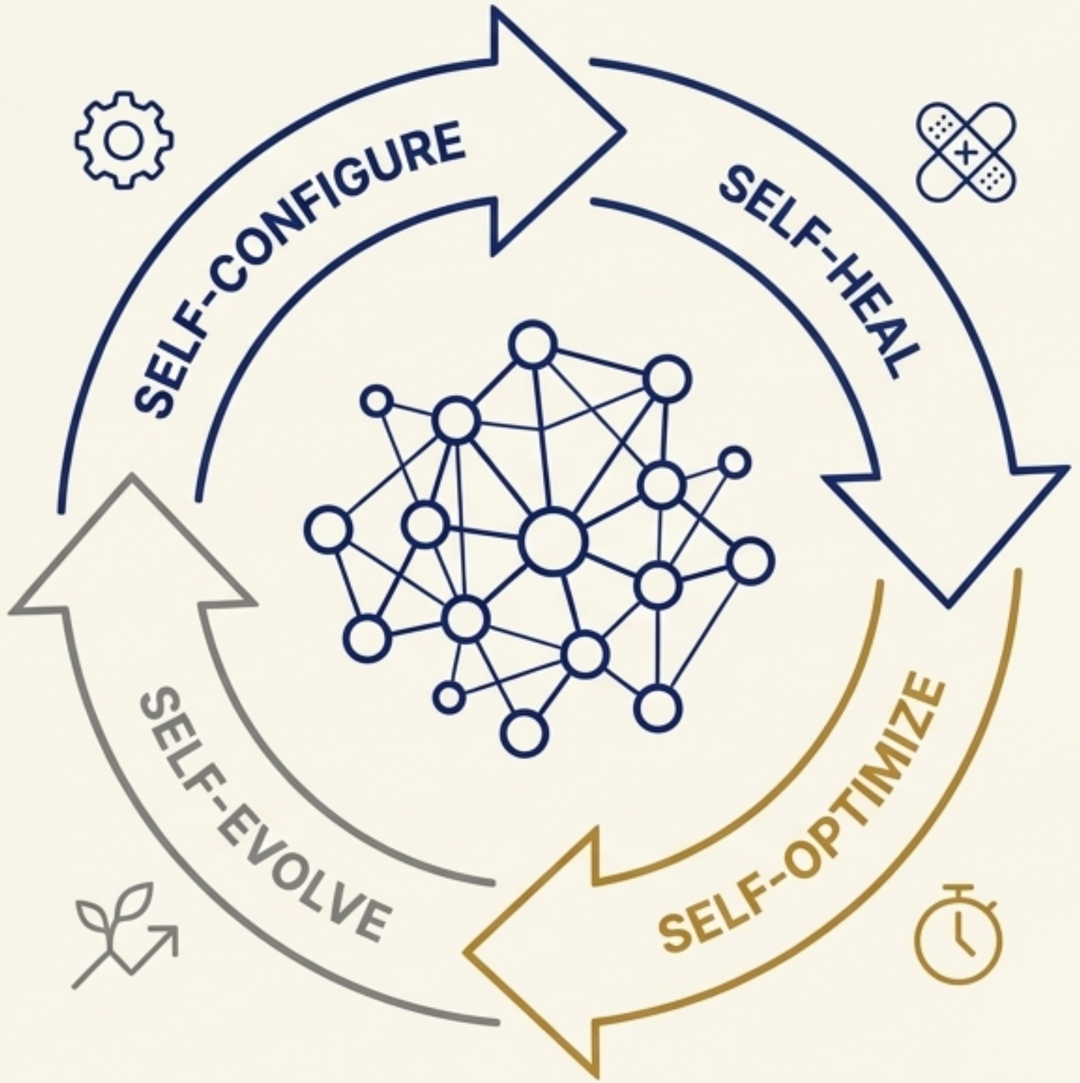
And more...

Knowledge is no longer a passive asset. It becomes a proactive, self-managing, and self-evolutionary entity.

The Foundation for Self-Managing Knowledge Ecosystems

Executable Knowledge directly enables the core properties of autonomic computing for knowledge graphs:

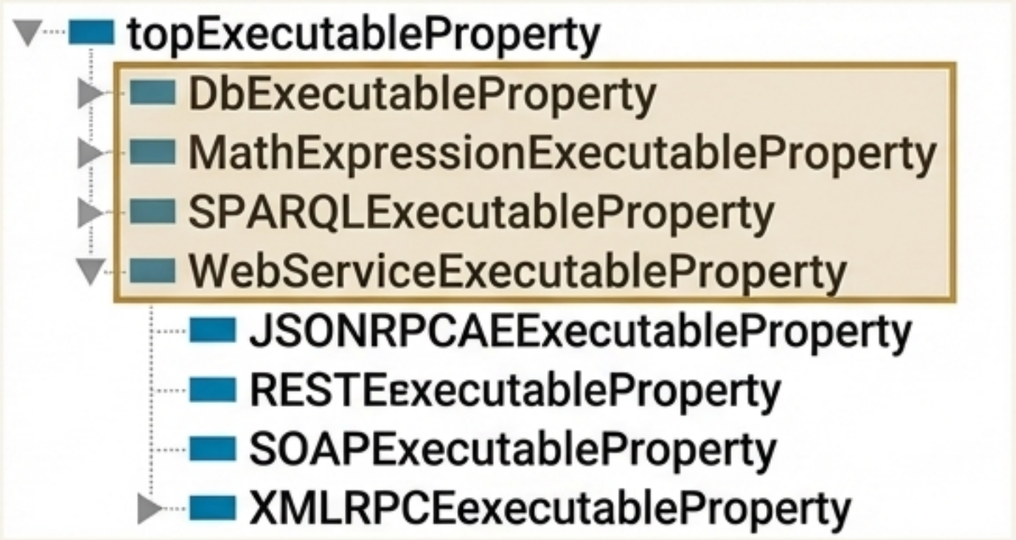
- **Self-Configuration:** Properties can automatically update their values or links based on changes in context or external data.
- **Self-Healing:** An executable link can query a backup data source if a primary one fails, preventing broken links. "A link will always have a valid target computed on the fly."
- **Self-Optimization:** Procedures can be defined to cache frequently computed values or rebalance graph structures.
- **Self-Evolution:** New analytical methods ('Quality Calculators') can be added as executable properties without altering the core ontology.



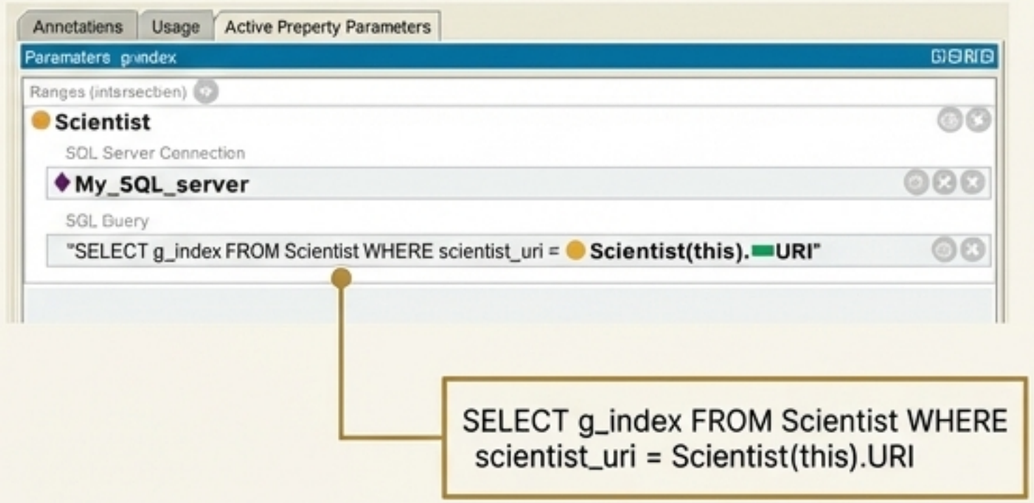
From Theory to Practice: A Proof-of-Concept Implementation

A pilot implementation was developed as a plug-in for the widely used Protégé ontology development environment.

Executable Property Types



Parameter Definition Interface

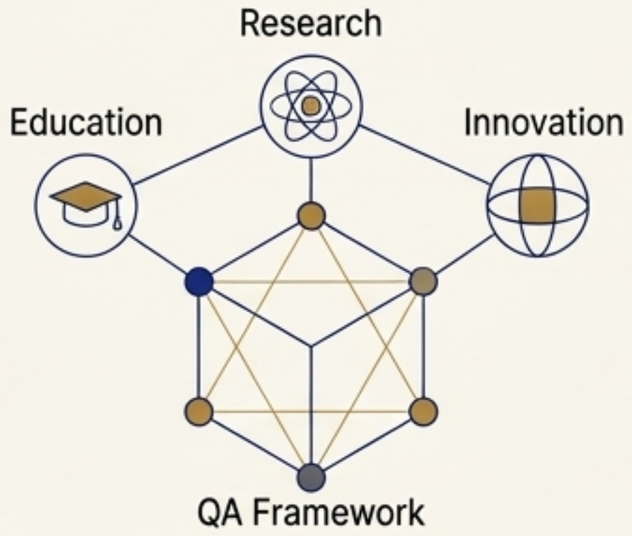


The Protégé plug-in enables the definition of executable properties directly within the ontology, specifying the procedures for SQL, SPARQL, and Web Service calls needed to compute a value.

Case Study: The EU TEMPUS-IV “TRUST” Project

CONTEXT

The project goal was to build a transparent Quality Assurance (QA) framework for Ukrainian Higher Education, integrating data from education, research, and innovation.



SOLUTION

A self-managed knowledge portal (portal.dovira.eu) built on the Executable Knowledge concept, enabling:

- **Cross-Validated Knowledge:** Executable properties query external sources like Google Scholar in real-time to update quality indicators.
- **Smart Knowledge:** End-users add their own ‘Quality Calculators’ as new executable properties for personalized performance views.
- **Self-Explanatory Knowledge:** The system can trace and explain the origin of every computed value.

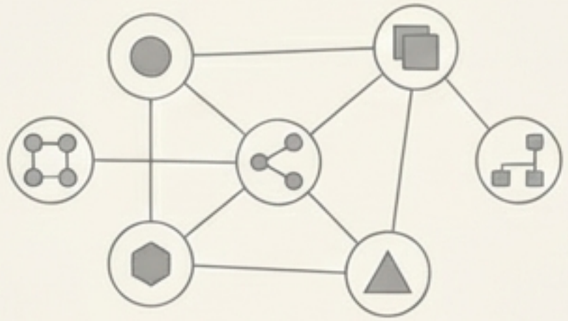
IMPACT

9.5x

increase in the speed of data logging by using executable SQL-properties, piloted on 50,000 instances.

From Static Links to a Computable Web of Knowledge

Stage 1: The Problem

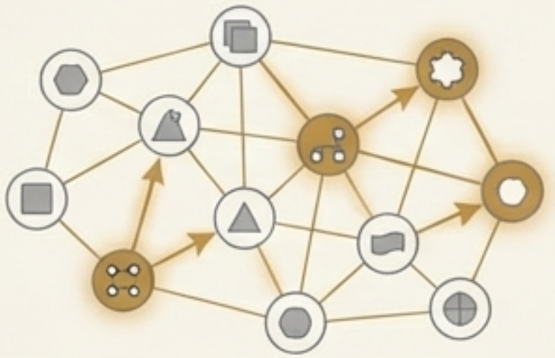


Linked Data is powerful but tied to a static web of facts.

Stage 2: The Breakthrough

Executable Knowledge embeds computation directly into the semantic graph.

Stage 3: The Horizon



This enables Knowledge Computing—the creation of proactive, self-managing, and self-aware information ecosystems.

By embedding procedural instructions within the declarative graph, we are building the foundation for a truly dynamic and intelligent web.