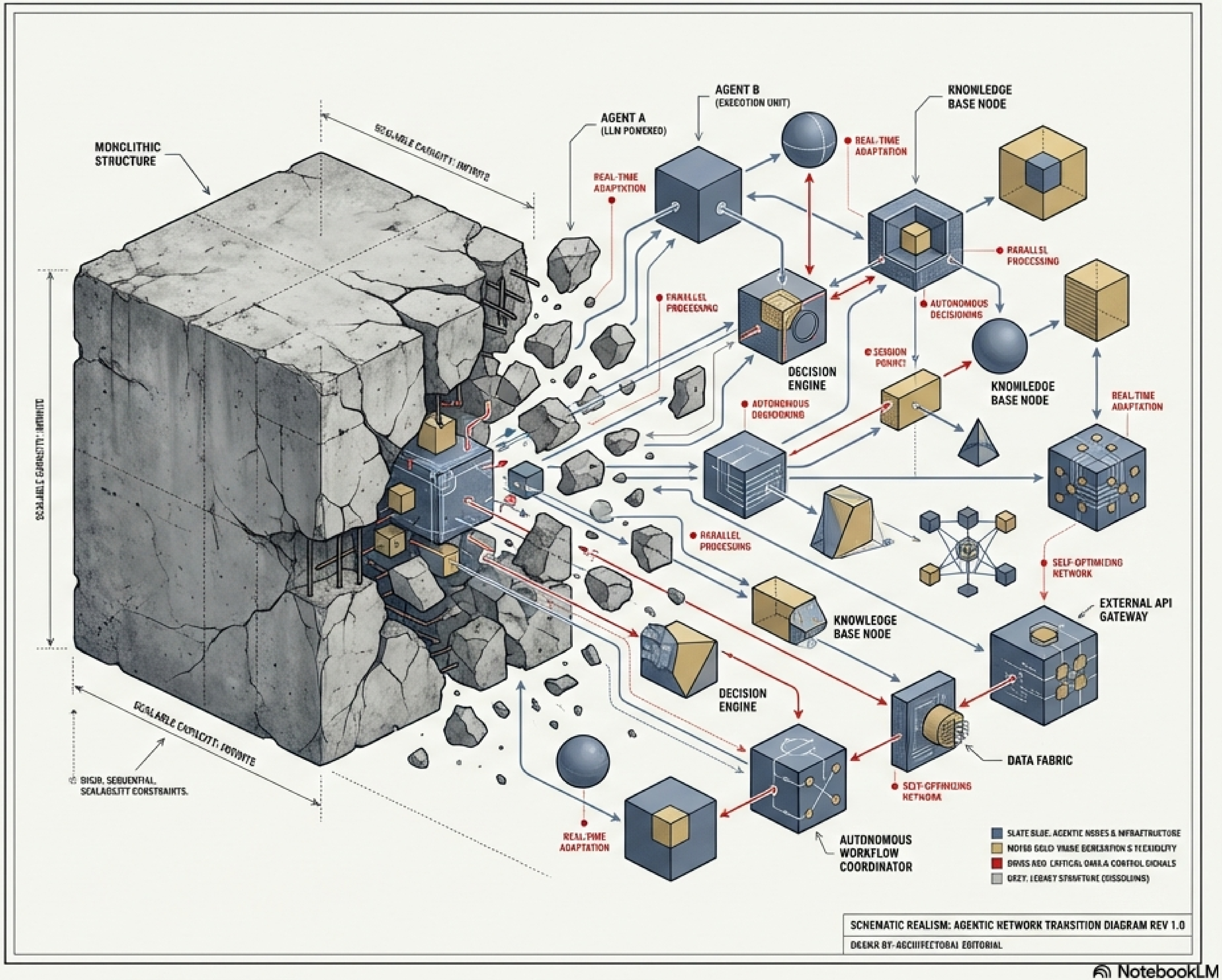
The TrillionDollar Shift to Agentic Software Architecture

Moving beyond the Monolith: How LLMs and Agents are restructuring the software economy.

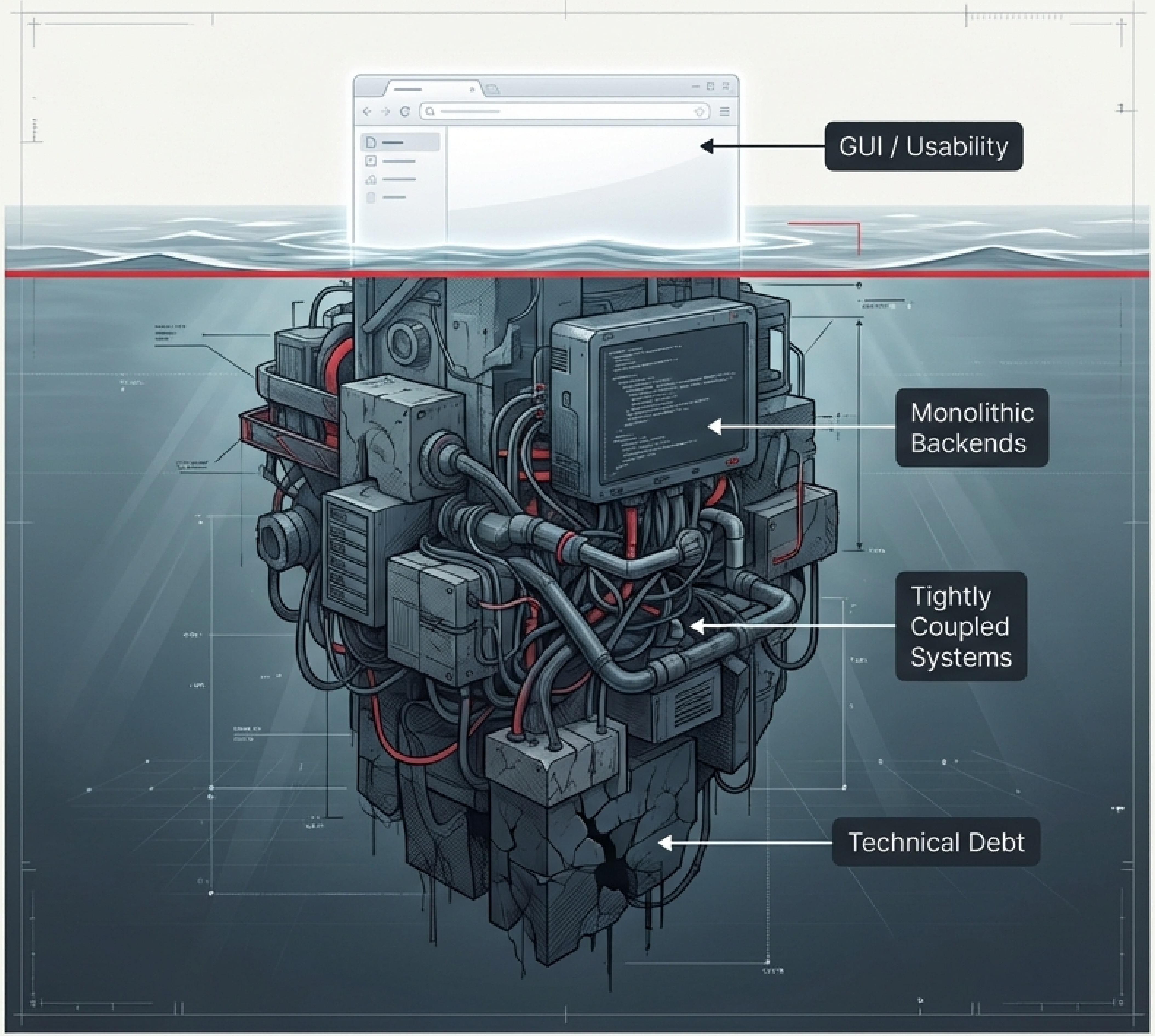


The Illusion of the Interface: Why GUIs are only the tip of the iceberg.

For decades, software has been shaped by the WIMP paradigm (Windows, Icons, Menus, Pointers). While this improved usability, it masked the architectural reality.

The Reality: GUIs function as the visible tip of an iceberg. Beneath the surface lies a mass of monolithic code—hard to maintain, difficult to evolve, and a major source of compounding technical debt.

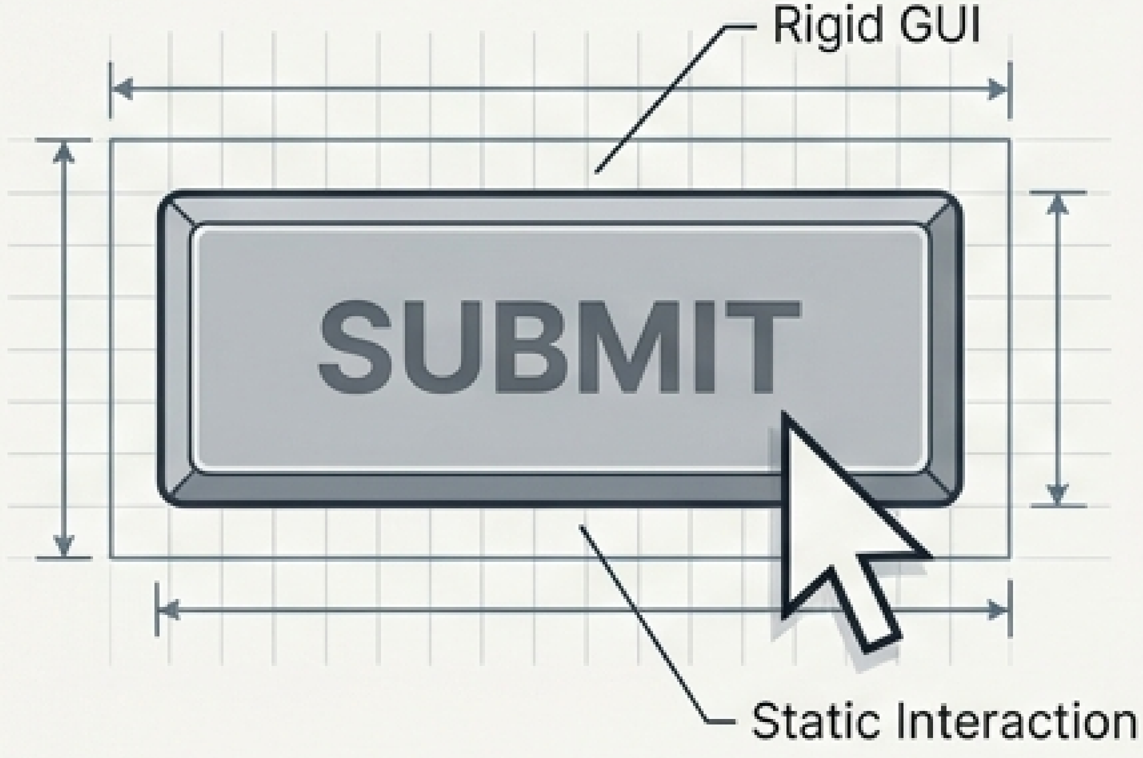
Open standards promised modularity, but inconsistent API implementations forced developers into pragmatic compromises, resulting in tightly coupled systems.



The Catalyst: LLMs and the Return of Loosely Coupled Software

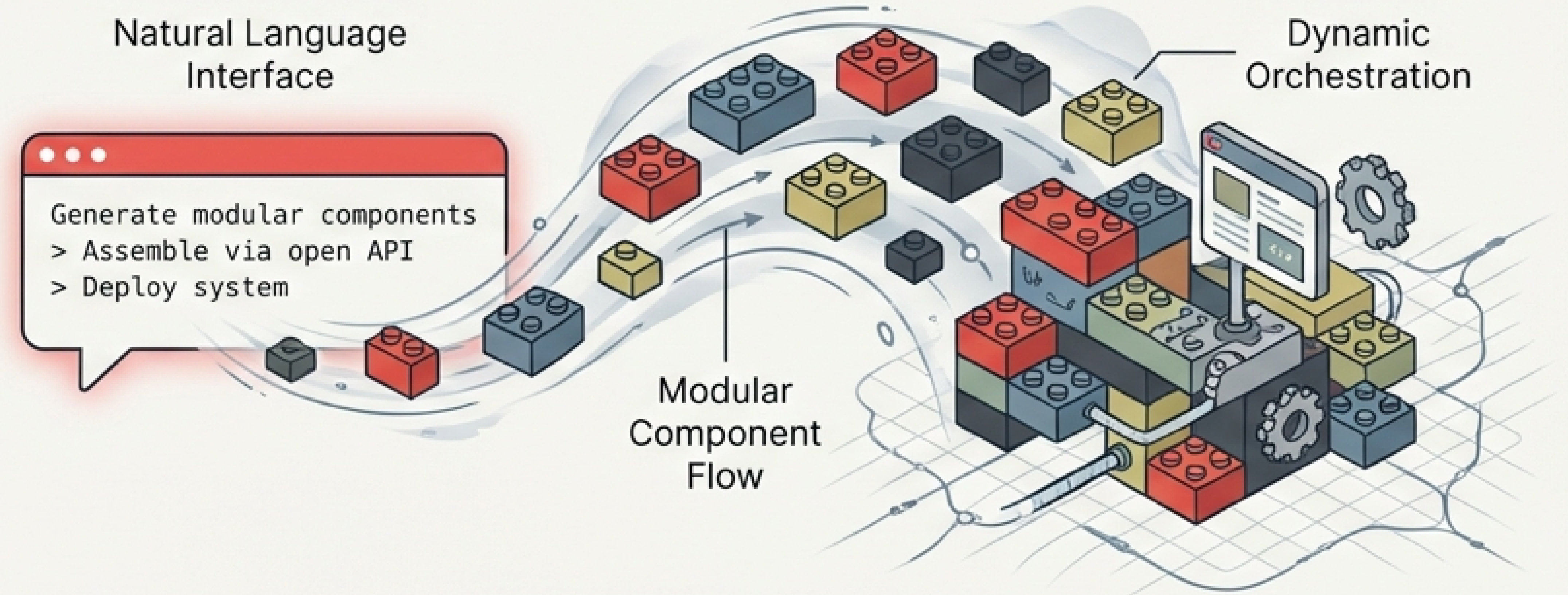
The emergence of Large Language Models (LLMs) disrupts the dominance of the GUI. Natural Language is no longer just for *using* software; it is a practical interface for *building* and *orchestrating* it. This revives the architectural ideal of "Lego block" software: systems composed of loosely coupled components connected via open standards.

The Old Way



Rigid GUI

The New Way



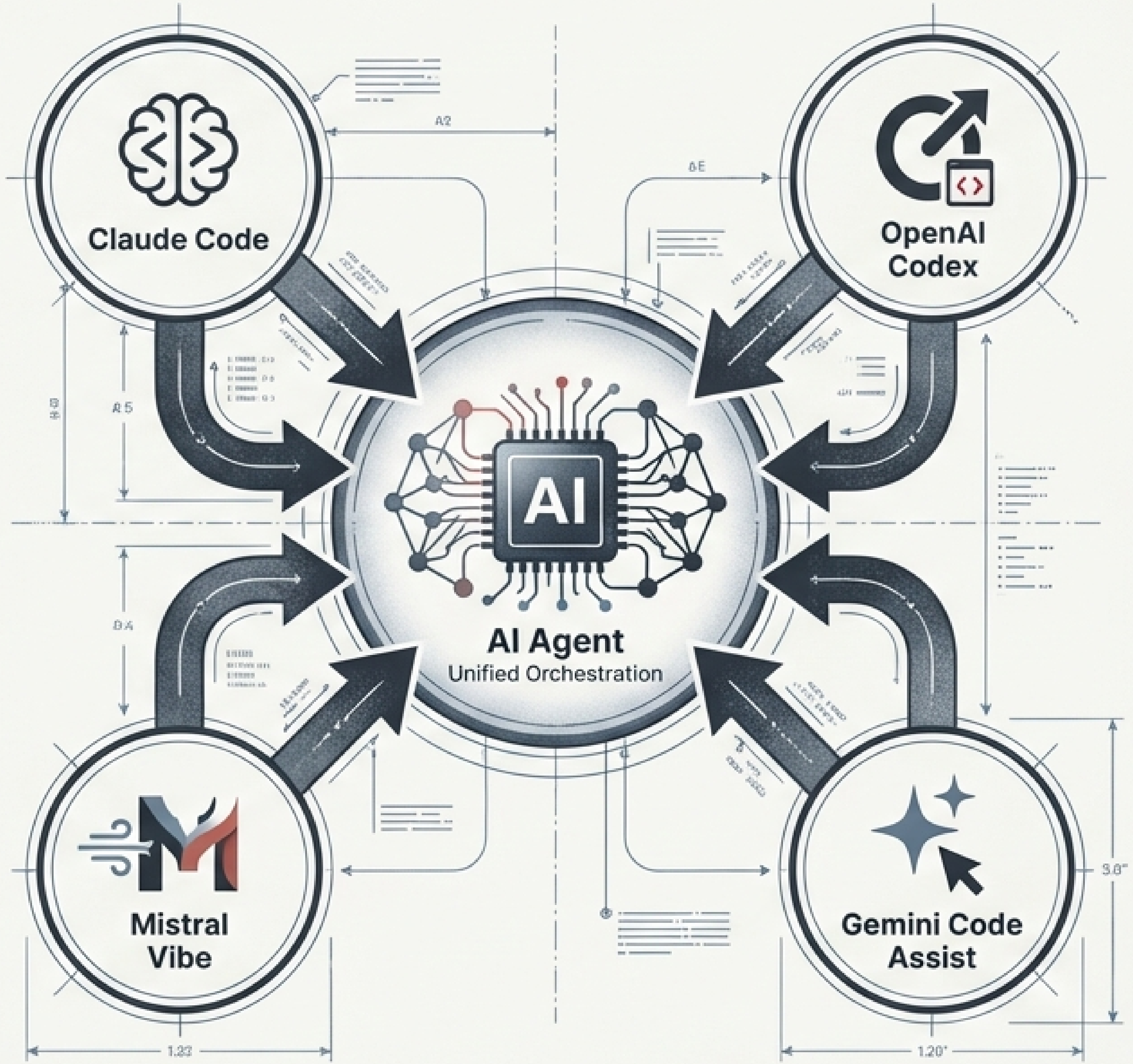
```
Generate modular components
> Assemble via open API
> Deploy system
```

Self-Assembling Systems

From Writing Code to Describing Intent

Software development is moving away from writing lines of code toward describing desired outcomes. Tools like Claude Code, OpenAI Codex, Mistral Vibe, and Gemini Code Assist began in the command line but are now integrating into broader environments.

The Result: Software becomes a composite system orchestrated by agents, not a handcrafted application.



Architectural Editorial

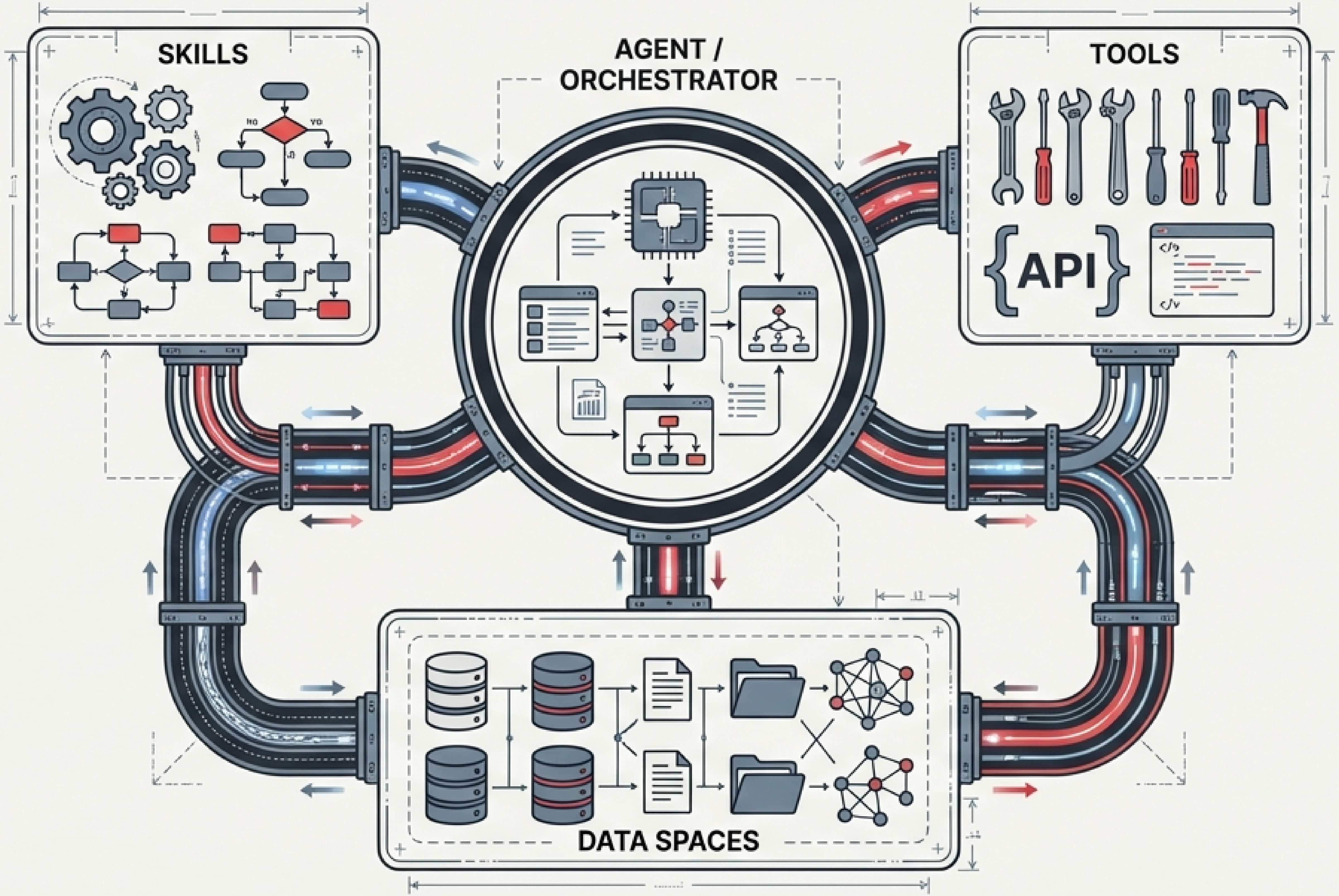
The Anatomy of Agentic Software.

In this new architecture, Agents act as Orchestrators, dynamically coordinating three distinct components to deliver user intent:

1. **Skills:** Workflow modules and domain logic.

2. **Tools:** Execution components for specific operations.

3. **Data Spaces:** Databases, knowledge graphs, and APIs.

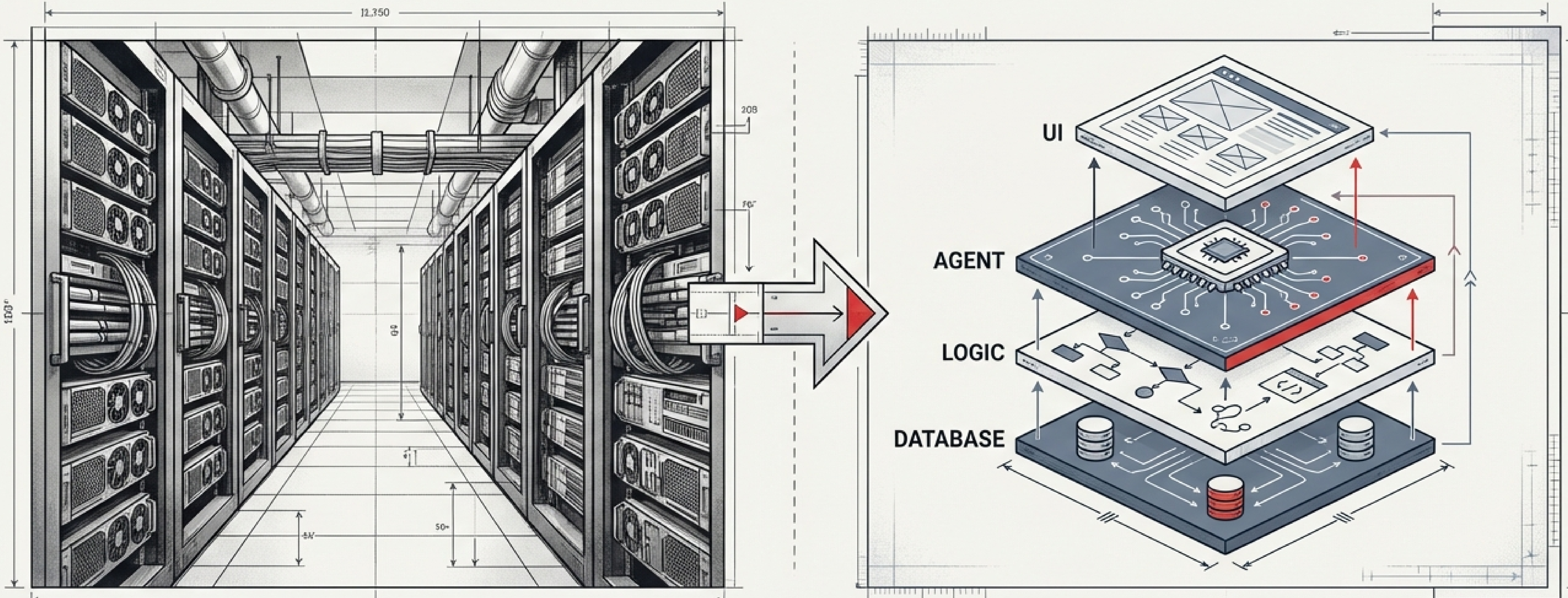


NotebookLM

The ‘GPU Moment’ for Software Infrastructure

Just as data centres redesigned for GPUs, software must redesign for Agents

The rise of LLMs forced a global redesign of data centres to accommodate GPUs. A similar structural shift is now required for software architecture itself. Agent-oriented infrastructure is becoming as foundational to software as GPUs are to AI.



Physical Infrastructure Shift (GPUs)

Logical Software Shift (Agents)

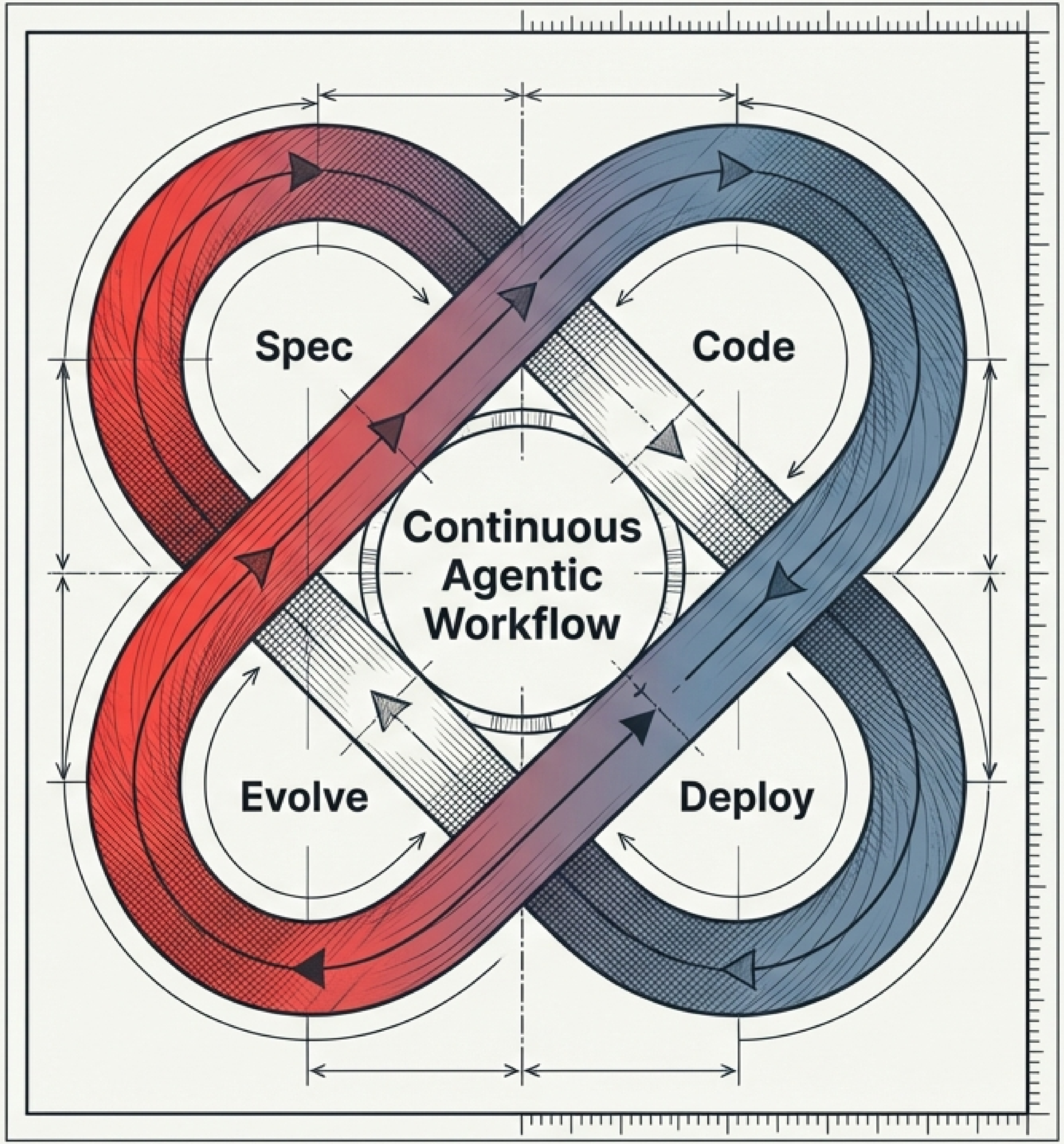
Changing the Economics of Development.

Broadened Participation: Users and domain experts collaborate in a unified process.

The New Assembly Language: Markdown and Natural Language replace proprietary logic.

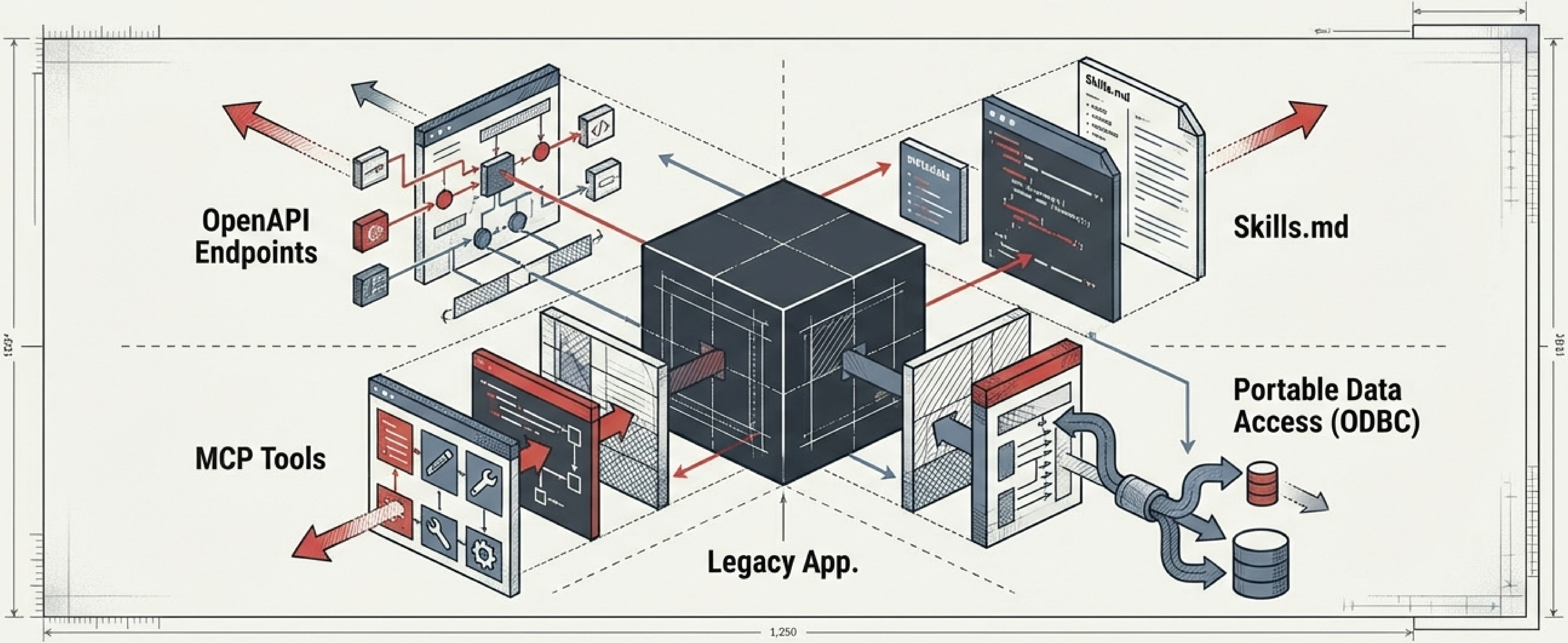
Compressed Workflow: The traditional silos of Product Spec, Programming, Deployment, and Evolution are compressed into a single, continuous loop.

ROI: Hand-offs are reduced. Reliance on monolithic systems diminishes, preventing technical debt.



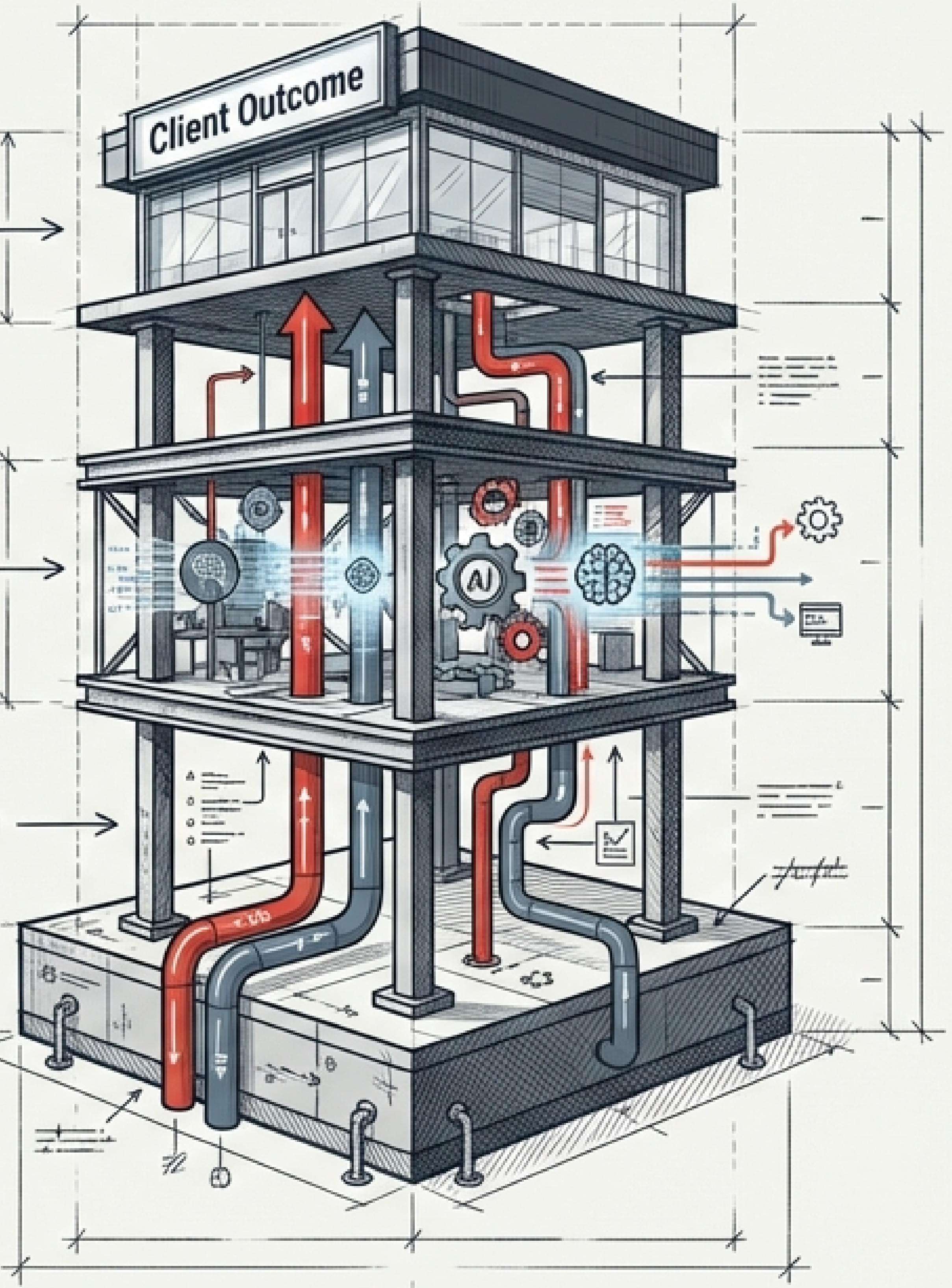
Strategic Imperative for Horizontal ISVs.

ISVs must evolve from selling standalone applications to providing **Agent-Ready Service Layers.** The goal is to turn applications into reusable toolkits for agent orchestration.



Strategic Opportunity for Vertically Focused VARs

Value-Added Resellers (VARs) gain a massive opportunity by packaging domain expertise into agent-enabled solutions. They must shift from System System Integrators to Domain Orchestrators.



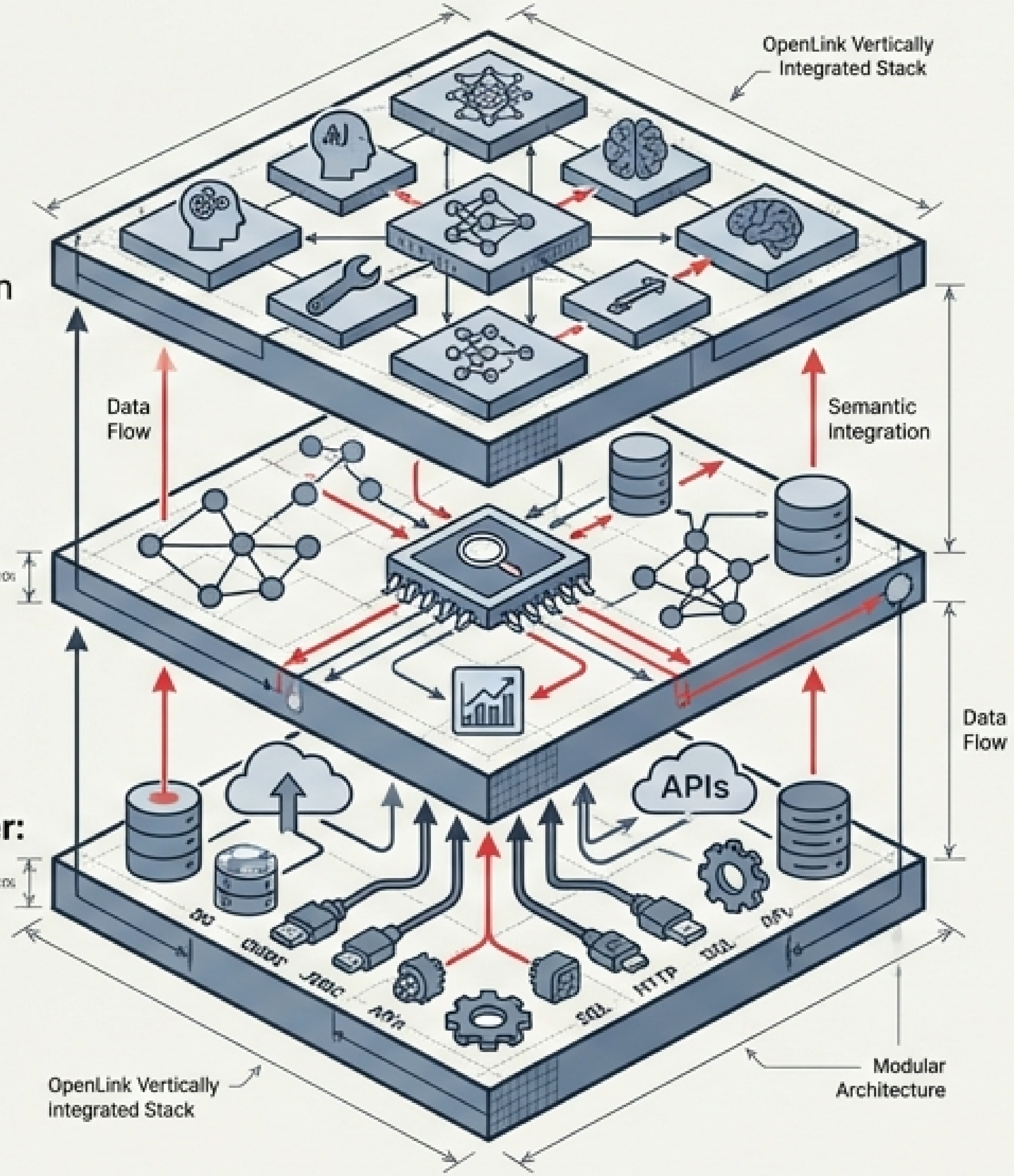
Client Outcome

Agent Layer (MCP / Skills)

Industry Record System (Finance/Healthcare)

Enabling the Shift: The OpenLink Ecosystem.

To transition from the Monolith to the Agentic era, we need a platform designed for loose coupling and semantic harmonisation. OpenLink provides a vertically integrated, standards-based portfolio to support this architecture.



Top Layer: AI Agent Creation (OPAL)

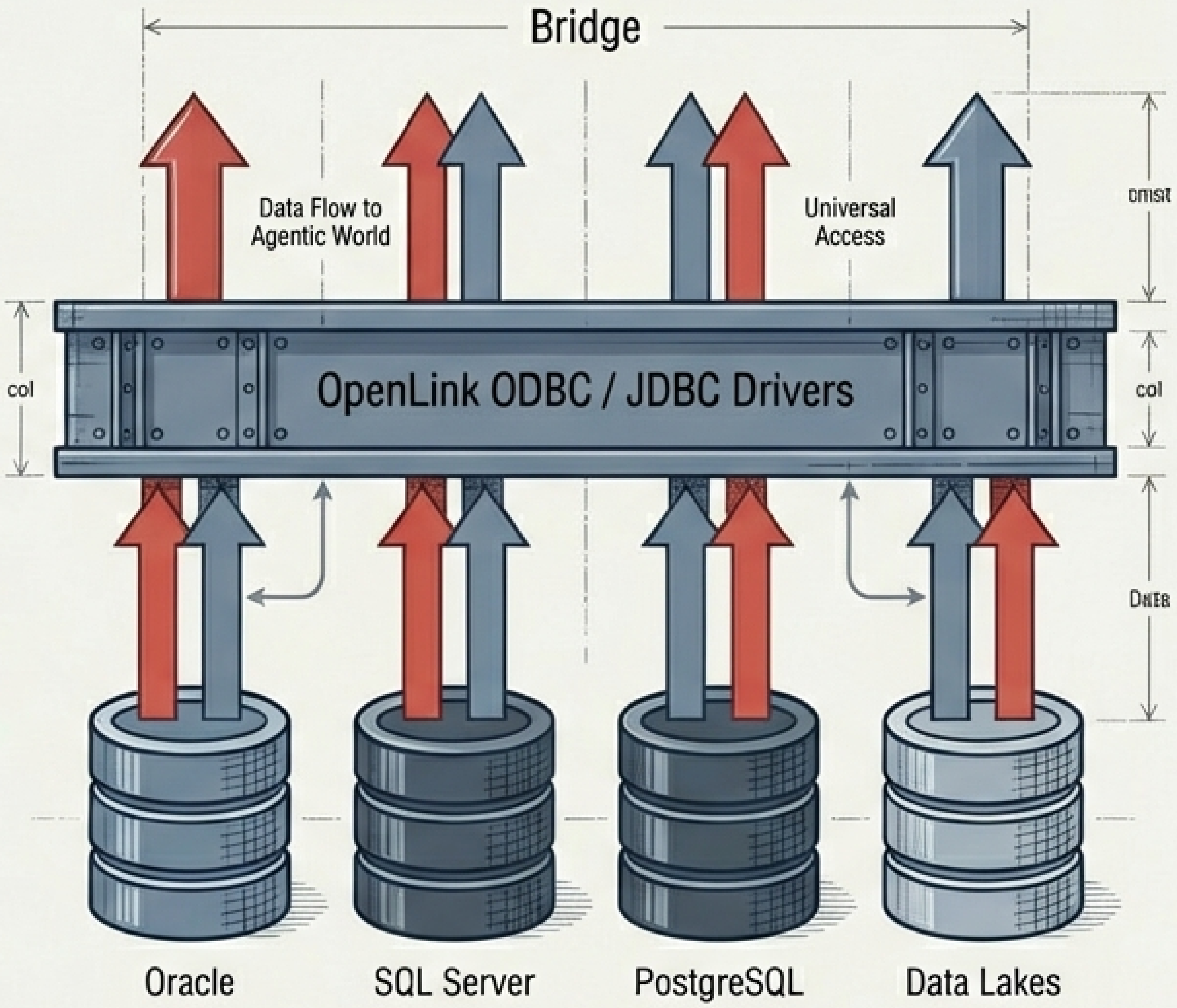
Middle Layer: Semantic Harmonisation (Virtuoso)

Foundation Layer: Data Access (Drivers)

OpenLink Vertically Integrated Stack
AI
Data Flow
Semantic Integration
Data Flow
APIs
HTTP
OpenLink Vertically Integrated Stack
Modular Architecture

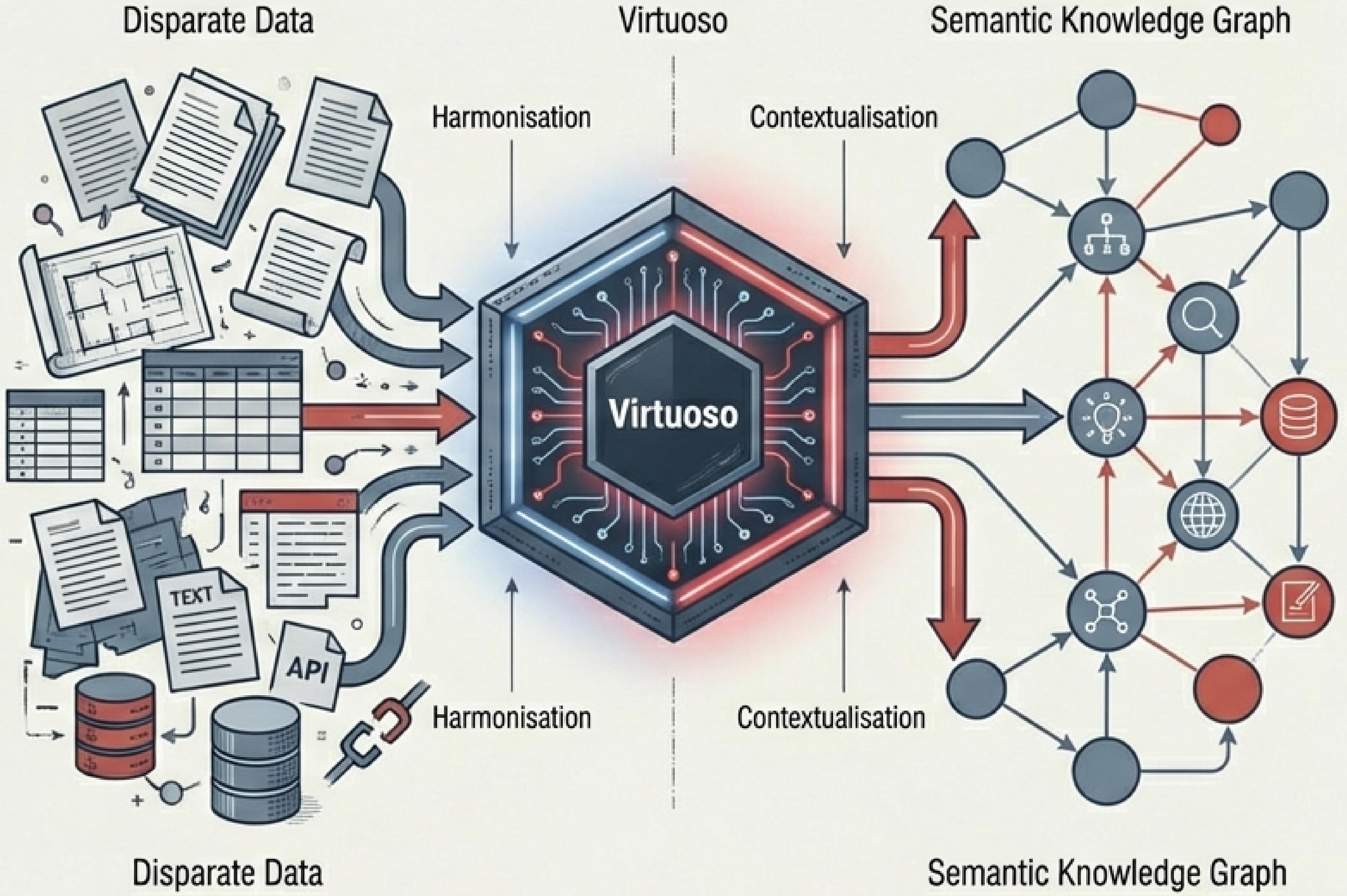
The Foundation Layer: Universal Data Access.

Before agents can act, they need access. OpenLink provides high-performance, secure ODBC and JDBC drivers. These drivers connect the new agentic world to existing DBMS-based systems of record, data warehouses, and data lakes, ensuring accessibility without ripping and replacing infrastructure.



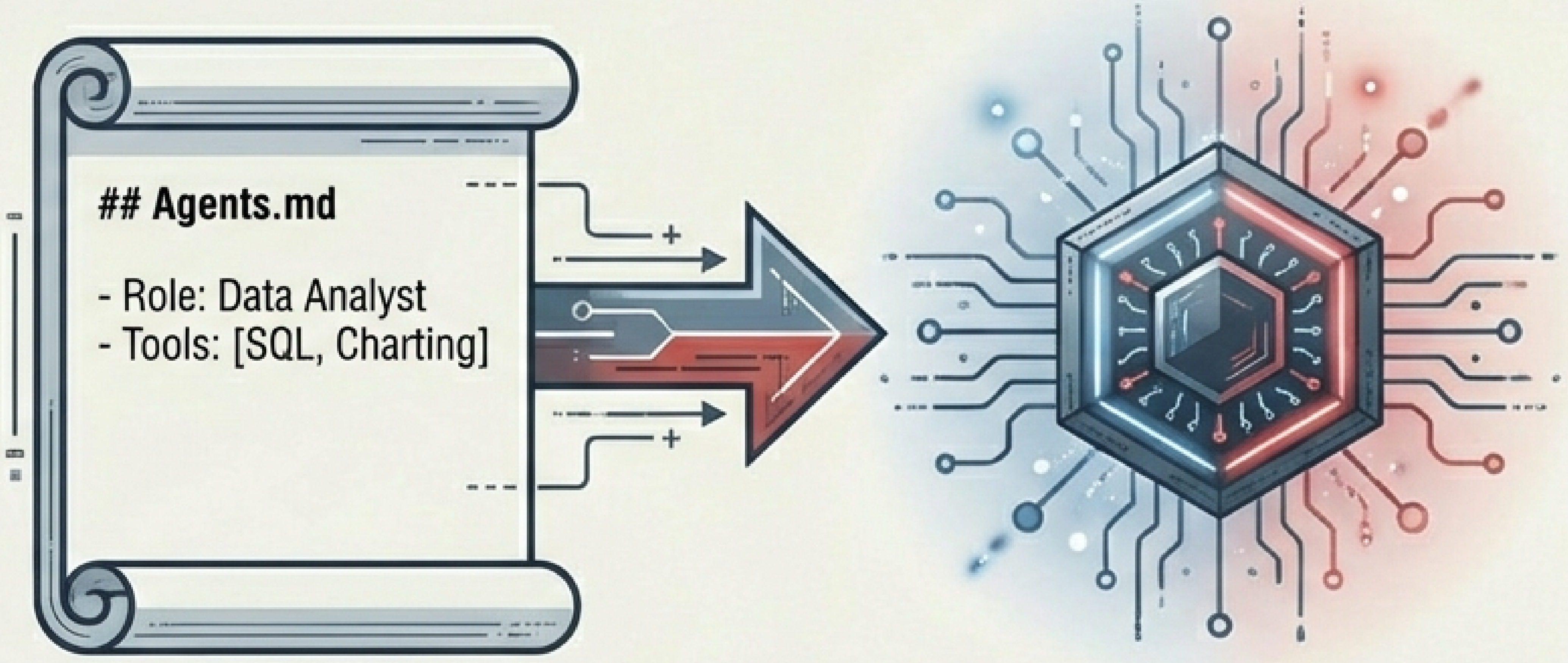
The Middle Layer: Virtuoso and Semantic Harmonisation.

Virtuoso serves as the semantic harmonisation layer. It loosely couples disparate Data Spaces—whet SQL databases, Knowledge Graphs, filesystems, or APIs. It provides the context, ensuring that when an agent accesses data, it understands the *meaning* and relationships within that data.



The Top Layer: OPAL and the Power of Markdown.

The OpenLink AI Layer (OPAL) provides tooling for creating powerful AI agents using simple Agents.md formats. It integrates tools via native stored procedures, RDF-based service descriptions, and OpenAPI.



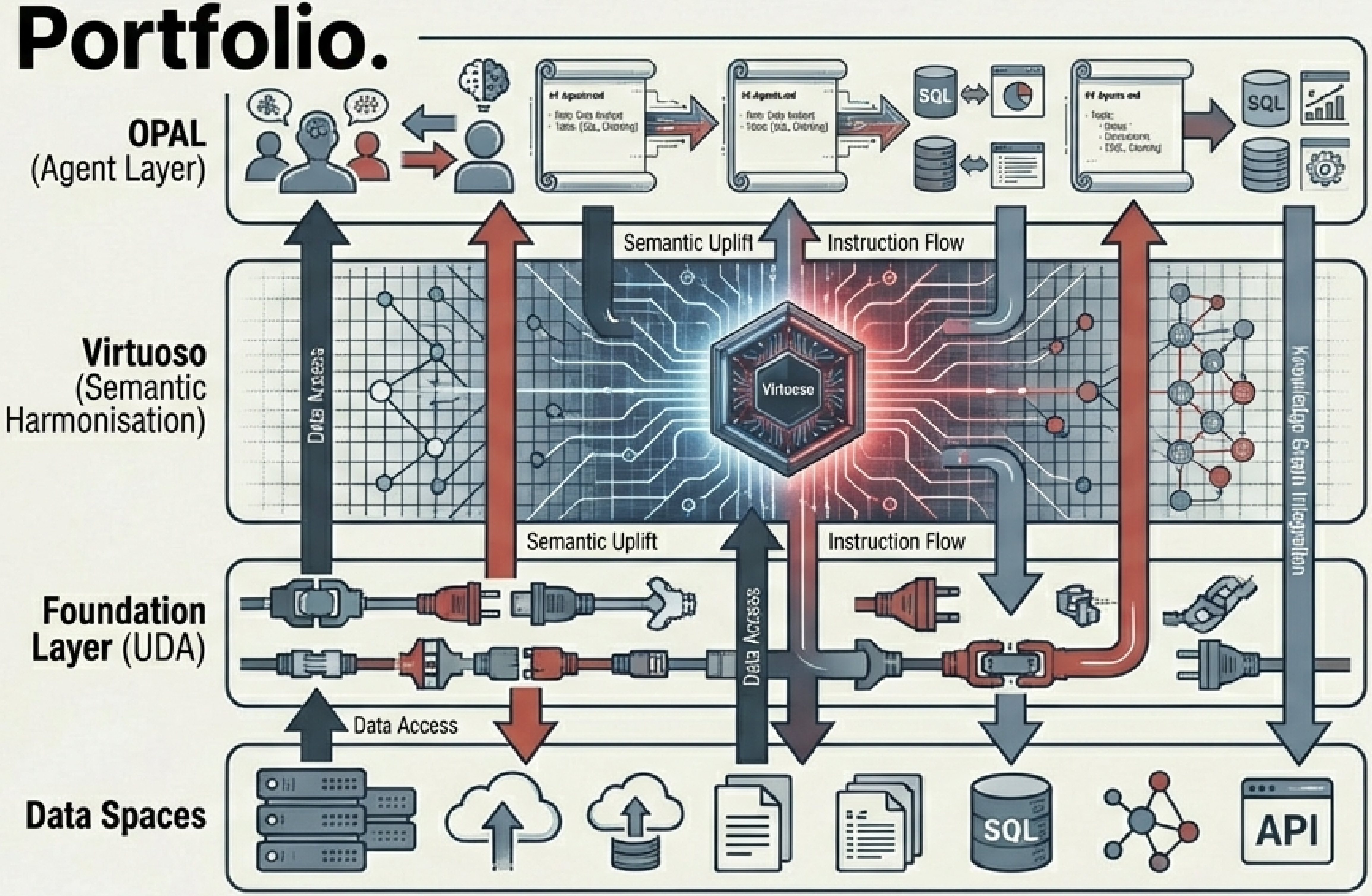
```
## Agents.md

- Role: Data Analyst
- Tools: [SQL, Charting]
```

NotebookLM

The Complete Architecture: An Integrated Portfolio.

The infrastructure required to capture the Trillion-Dollar Agentic opportunity.



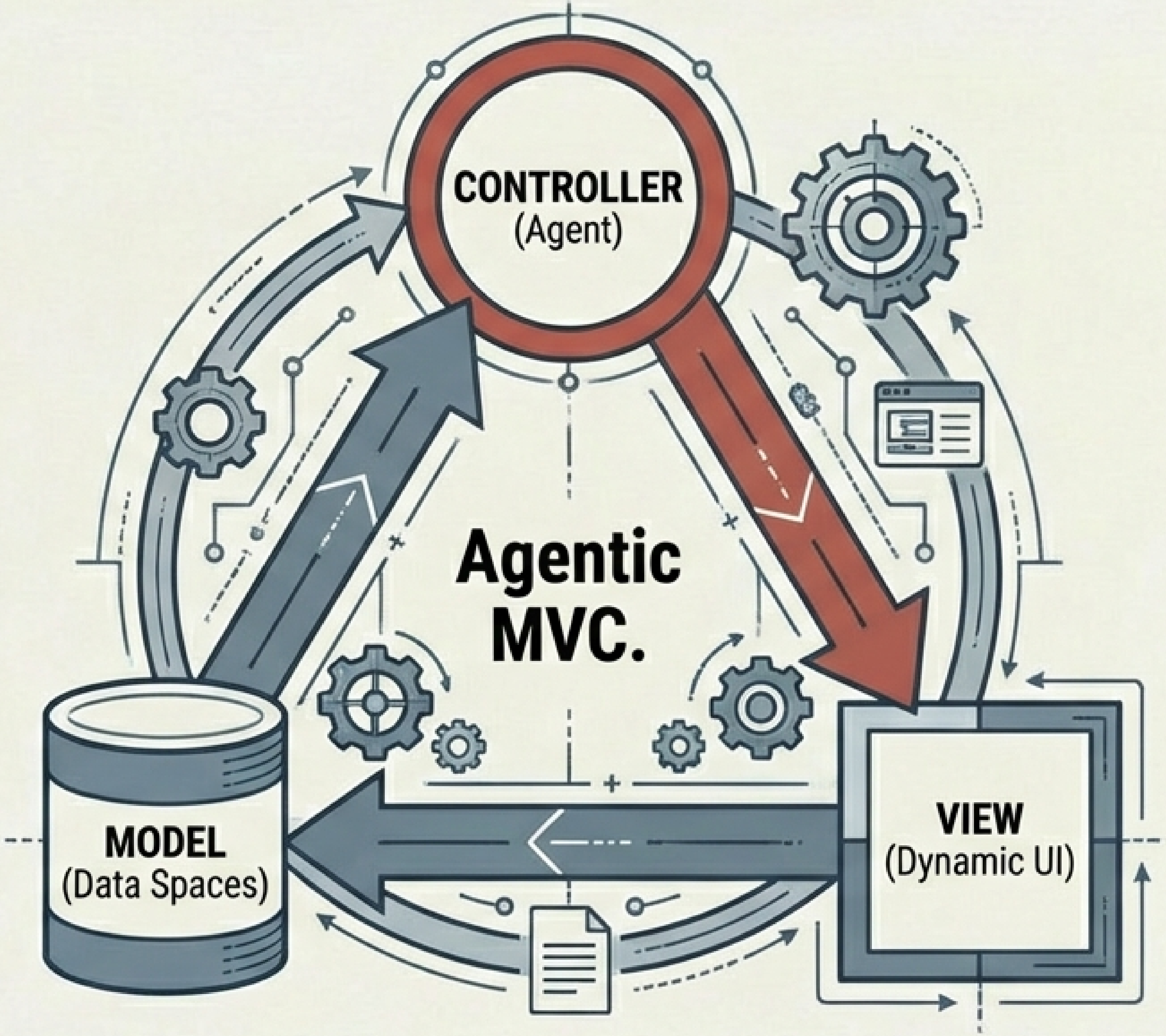
A New Software Architecture: Agentic MVC.

The familiar Model-View-Controller (MVC) pattern is evolving for the Agentic era:

Controller -> The Agent: The orchestrator of workflows.

View -> Dynamic: Generated or adapted in real-time by agents.

Model -> Loosely Coupled Data Spaces: Integrated via semantic harmonisation.


CONTROLLER
(Agent)
Agentic
MVC.
MODEL
(Data Spaces)
VIEW
(Dynamic UI)