SymphonyAI

WHITE PAPER

Unified Data Model: Applying Generative AI for Data Unification

INDUSTRIAL

Contents

Executive Summary

Industrial operations generate vast amounts of sensor and operational data from diverse sources. Standardizing this data through a common namespace is critical for interoperability, advanced analytics, and improved decision-making processes. According to industry reports, over 80% of industrial data remains unanalyzed due to inconsistent and inscrutable data naming practices. Research from McKinsey & Company suggests that companies leveraging advanced AI for data integration can unlock up to $1 trillion in global economic value annually. Key business outcomes of this approach include enhanced operational resilience and improved decision-making capabilities, fostering scalable business growth and sustainability.

Generative Agentic AI, supported by a fine-tuned small language model (SLM), can transform data analysis and processing at a massive scale. These models have the potential to create a unified data model with acceptable accuracy using minimal training and limited human involvement. Achieving data analysis and processing at scale has been a challenge for "traditional" rule-based or other AI-based approaches.

Introduction

Companies face significant challenges in unifying fragmented data despite major investments in data management infrastructure. Fragmented data estates often result in wasted time for domain and technical experts and millions of dollars spent without achieving scalable business value through a unified data layer. According to a recent McKinsey report, 60% of data integration projects fail due to a lack of cohesive strategy and execution.

60%

of data integration projects fail due to a lack of cohesive strategy and execution, according to a recent McKinsey report

However, applying AI in the right quantities with a structured methodology can effectively address these challenges. While AI is not a silver bullet, it offers significant leaps at every stage of the data unification journey, reducing costs and project timelines by several magnitudes. A Generative AI-based approach significantly outperforms traditional rule-based mapping models by minimizing the effort and time required to build unified data models. Realizing this potential requires a comprehensive strategy and a meticulous execution plan.

This white paper covers one such approach, illustrating how generative AI can drive automated sensor tag mapping and create a unified data structure. The provided example targets a midstream oil and gas (O&G) use case, but the underlying approach and processes also apply to adjacent industrial verticals with similar challenges.

Problem Statement Overview: Midstream Oil & Gas Data Ecosystem Overview

The midstream oil and gas industry involves complex pipeline networks, numerous flow stations, and diverse data-generating assets like flow meters and compressors. Data from various sources, including SCADA systems, enterprise historians, and operational logs, often remain siloed due to mismatches in data structures, nomenclatures, and formats.

McKinsey's research indicates that unifying such data can boost operational efficiency by up to 30%. However, traditional methods often fall short due to technical limitations, limited collaboration between data scientists and domain experts, and a lack of standardization protocols. Overcoming these challenges necessitates an AI-driven approach for more effective data unification, which has been proven to offer an optimal solution with minimal expenditure and effort compared to the traditional rule-based data unification methods.

30%

boost in operational efficiency can be achieved by unifying such data, according to McKinsey's research.

Problem Statement Detail: Data Fragmentation

A typical industrial facility or ecosystem contains the following disparate data types, data formats, and data sources (machine-generated data is referred to as “tags”):

Data Types	Data Formats	Data Sources	Possible Data Naming Conflicts & Issues
Operational Data	Time-series, event-based data	SCADA, DCS, Historian, IoT Platforms	Inconsistent tag names, duplicate sensor IDs, unclear unit labels, overlapping alert codes
Measurement and Metering Data	Relational databases	Measurement Systems, ERP	Inconsistent naming for meters, redundant asset IDs, unclear data labels
Environmental and Regulatory Data	Documents, text files, time-series	Environmental Monitoring Systems, ERP	Inconsistent report names, file duplication, ambiguous environmental metric labels
Maintenance and Asset Data	Relational databases, digital documents	CMMS, ERP, Asset Management Systems	Inconsistent asset IDs, varying maintenance task descriptions, duplicate part codes

These diverse data types are collected at various points across enterprise operations. However, differences in data formats, inconsistent tagging, fragmented storage systems, and many other data naming conventions and anomalies often hinder effective data unification strategies.

Inconsistent data naming conventions can lead to inefficiencies and financial losses for these facilities. The World Economic Forum highlights that manufacturers spend 10-30% of revenue addressing data quality issues, with poor data costing an average of $12.9 million annually.

10-30%

of revenue is spent on addressing issues related to data quality

McKinsey & Company notes that data processing and cleanup can consume more than half of an analytics team's time, limiting scalability and frustrating employees. These insights underscore the financial and operational risks associated with poor data naming practices in the manufacturing sector.

Proposed Solution: Generative AI-Driven Sensor Tag Mapping Framework

The proposed solution covers three key phases:

Design & Develop: Covering the complete data unification algorithmic flow, details of each step within the flow, and the specific suggestive algorithms to apply at each step with a bias towards Small Language Model (SLM) for its superiority over other specific-purpose NLP models in language inference, its ability to carry context across wide and large dataset, and its reduced demand for hyper-parameter handling.

Deploy: Covering the key aspects of where to deploy the unification model(s) and the cost/benefit.

Execute: Covering key aspects of interacting with the data in the execution phase in three significant ways: one-time bulk load, multiple bulk loads with consideration for delta, and dynamic continuous loading process.

Data Unification Process

Design & Develop

- Algorithmic Flow for Data Unification

Deploy

- On-the Edge Infrastructure
- On / near the Asset
- Commonization Layer
- Analytics Layer

Execute

- Bulk
- Delta
- Dynamic Real-time

While all three phases must be addressed, this whitepaper focuses on applying generative agentic AI to the design and develop phase of the unification process.

Design & Develop: Algorithm Flow

The scope of the algorithm flow starts with acquiring the data from the source systems, creating the unified data model, and providing input to a knowledge graph. The knowledge graph is a neural network-based real-time digital representation of an operational ecosystem where end-users can interact with it to draw data-driven insights, measure outcomes, and perform actions, enabling the graph to proactively and reciprocatively learn and encode human knowledge. This document does not extend to the design of the knowledge graph or higher-level AI-enabled end-user functions.

The algorithm flow is not a novel proposal but restates the proven standard process that contains five broad steps. The details within each step outline the opportunity to apply agentic AI and SLM to replace the cumbersome, labor-intensive, and failure-prone traditional rule-based approach and simplify the complex discriminative AI approach.

1. **Automated Data Ingestion and Normalization:** Collect, parse, and standardize sensor tag data by accounting for various formats, ensuring seamless integration and accurate data representation.

2. **Semantic Tag Mapping:** Align tags to a unified structure with ontology techniques, contextual embeddings, and domain-specific models for dynamic tag interpretation.

3. **Anomaly Detection and Correction:** Detect and resolve tag conflicts, ensuring data consistency and reducing manual intervention.

4. **Tag Recommendation:** The assets (or equipment), processes, and human interaction with the equipment generate extensive time-series data of various formats. Such data offers a forensic footprint of operational activities and provides an excellent opportunity for AI-powered analytics tools to create data-driven insights.

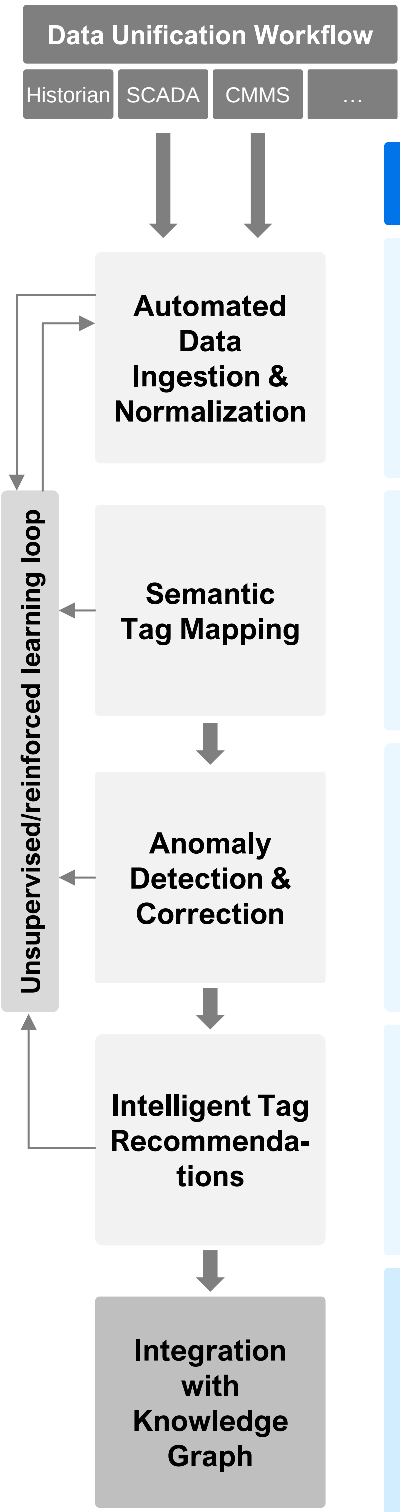
5. **Integration with Knowledge Graphs:** Link tags to knowledge graphs, enabling semantic interoperability and advanced data analysis.

The analysis below illustrates how the unification workflow is managed across three separate methods:

a) Traditional method

b) Discriminative AI models

c) Agentic AI with SLM



Offers Advanced Language Inference e.g., LLaMA 3.2 1B, LLaMA 7B, YOLO

	Traditional Method	**Discriminative AI Models**	**GenAI L(S)LM**
Automated Data Ingestion & Normalization	Manual parsing, hardcoded schema recognition, and static rules for standardization	NLP models like spaCy for parsing, ML for schema recognition, and transformers for standardization	Named Entity Recognition (NER), schema adaptation using contextual embeddings
Semantic Tag Mapping	Static look-up tables, hardcoded mappings, and manually managed dictionaries	Ontology alignment using Word2Vec, contextual embeddings with BERT, and domain-specific models	Ontology alignment and contextual embeddings for industry-specific tags
Anomaly Detection & Correction	Rule-based conflict resolution and manual data validation	Anomaly detection using Isolation Forest, DBSCAN, and Autoencoders for conflict resolution	Vision Model (YOLO): Detect anomalies in visual pipeline data; LLaMA-2 7B for text-based anomaly tagging
Intelligent Tag Recommenda-tions	Hardcoded tag recommendations based on historical data and static rules	Tag recommendations using collaborative filtering, XGBoost, and predictive neural models	Predictive text generation for tag recommendations and mapping predictions
Integration with Knowledge Graph	Manual database linking, relational models, and static hierarchies	Graph Neural Networks (GNNs), embeddings like TransE, and automated knowledge graph linking	Lightweight entity linking for knowledge graphs; Vision Model (CLIP) for visual tag correlations

Step 1: Automated Data Ingestion & Normalization

1 **Data Context:** Ingesting and transforming data into standardized formats with recognized schemas to prepare the ingested data for parsing

a. **Traditional Method:** In traditional systems, data ingestion involves manual ETL (Extract, Transform, Load) pipelines. Engineers write scripts to parse data and create mapping rules based on pre-defined dictionaries. Schema recognition is rule-based, requiring significant manual intervention.

b. **AI-Driven Approach:** AI-powered ingestion automates tag parsing using NLP models. Machine learning algorithms recognize schemas dynamically and adapt without requiring manual reprogramming. Tags like "FlowMeter_01" and "FM_Main" are parsed automatically and standardized.

c. **Agentic AI with SLM Approach:** Use SLM to parse sensor tags by analyzing patterns and inferring relationships between tag structures (e.g., Flow_Rate_Sensor and FRS as equivalent). Implement agentic AI to monitor incoming data streams, adapt to real-time schema variations, and automatically generate updated normalization rules.

- Example: A pipeline receives data in formats like JSON and CSV. SLM identifies column headers like FlowRate and Flow_R as semantically related, while agentic AI updates the schema to handle a new tag format, ensuring no disruptions to ongoing processes.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI
Setup Effort	High, Manual	Moderate, Automated	Low, Self-Adaptive
Adaptability	Limited, Static	High, Dynamic	Very High, Real-Time
Scalability	Low, Labor-Intensive	High, Data-Specific	Unlimited, Seamless

2 **Tag Parsing:** Process incoming sensor tags from SCADA systems, enterprise historians, and other sources.

a. **Traditional Method:** Parsing relies on static mapping scripts based on fixed rules. Any deviation from expected formats requires manual correction.

b. **AI-Driven Approach:** Using contextual embeddings, AI models extract relevant components from tags. Tags such as "Compressor_Temperature" and "Comp_Temp" are understood as equivalent through semantic parsing.

c. **Agentic AI with SLM Approach:** Use lightweight models for semantic understanding and add an agentic layer to resolve parsing ambiguities in real-time. For instance, when encountering an unknown tag like "Cmp_Tmp," the model queries data dictionaries or metadata APIs, applies similarity scoring to identify likely matches, and auto-corrects based on confidence thresholds. The agent can also log unresolved cases for future training updates, ensuring dynamic adaptability with minimal manual oversight.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI
Parsing Flexibility	Fixed Rules Only	Contextual Understanding	Contextual & Real-Time
Maintenance Effort	High, Constant Updates	Minimal, Self-Adapting	Very Low, Proactive Updating
Data Accuracy	Moderate	High with Continuous Learning	Very High, Dynamic Validation

3 **Schema Recognition:** Adapt to different schemas used by data systems like OSIsoft PI or AspenTech IP 21. Schema variations are recognized and resolved automatically.

a. **Traditional Method:** Manual inspection and definition of data models are required. Engineers create mappings and transformation rules for each data source.

b. **AI-Driven Approach:** Machine learning models automatically recognize and adapt to schema variations. Models trained on industry datasets detect inconsistencies in schema definitions from systems like OSIsoft PI or AspenTech IP 21.

c. **Agentic AI with SLM Approach:** This approach handles schema detection by analyzing structural patterns in metadata and applying pre-trained embeddings for industry-specific systems. The agentic component continuously monitors for schema anomalies, automatically suggesting mappings or transformations based on learned rules. For example, discrepancies in data models from OSIsoft PI and AspenTech IP 21 are resolved by dynamically generating crosswalks, validated against live data, and refined through user feedback loops. This ensures schema integration is both fast and precise.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI
Recognition Speed	Slow, Manual Inspection	Instant, Automated	Instant, Real-Time Monitoring
Model Adaptability	Static Rules	Adaptive, Learning-Based	Highly Adaptive, Proactive Updates
Operational Impact	Delays in Integration	Real-Time Processing	Seamless, Continuous Optimization

4

Data Standardization: Clean up inconsistent tags such as "Comp_Temp" and "CompressorTemperature," standardizing them into a unified format.

a. **Traditional Method:** Engineers create standardization scripts using pre-defined dictionaries. This process is prone to human error and requires frequent updates.

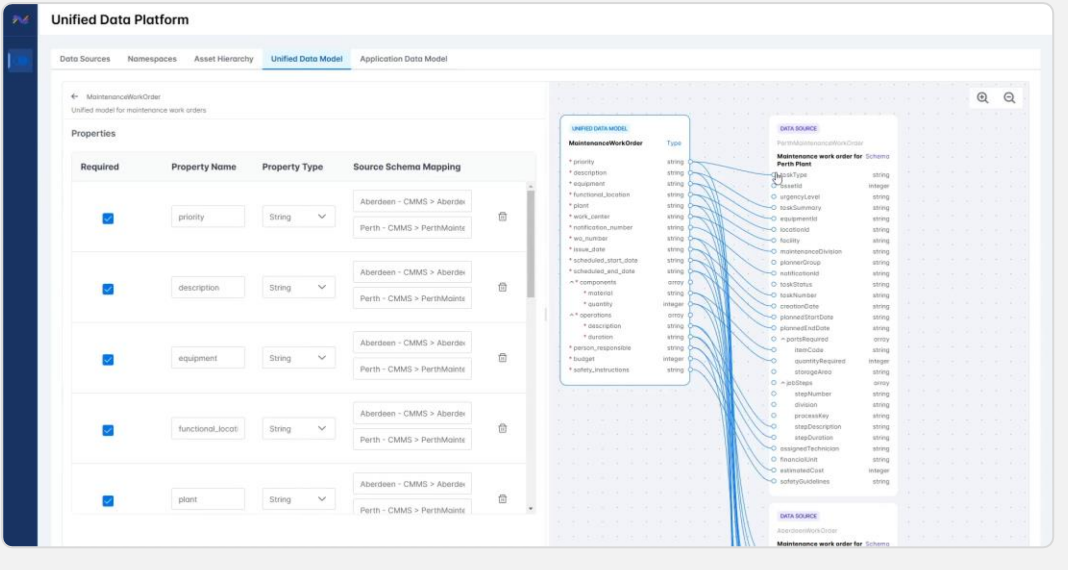
b. **AI-Driven Approach:** NLP models standardize tags by understanding abbreviations, noise, and spelling variations. For instance, "Compressor _Temperature" and "Cmp_Temp" are normalized into a unified format.

c. **Agentic AI with SLM Approach:** An SLM analyzes tag inconsistencies using pre-trained embeddings, identifying semantic relationships even in complex cases. For example, tags like "Cmp_Temp_Main01" and "Compressor.Temp.Zone1" are mapped by parsing structural patterns and context. Agentic AI monitors tag streams, detects deviations (e.g., "CmprZone_T01"), applies similarity scoring, and auto-generates normalization rules. It validates against metadata hierarchies and historical mappings, ensuring real-time standardization with minimal manual oversight.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI
Standardization Speed	Slow, Manual Updates	Fast, Automated	Instant, Real-Time Adaptation
Error-Prone Nature	High Due to Manual Work	Low with AI Validation	Minimal, Proactive Correction
Maintenance Effort	High	Minimal, Self-Learning	Negligible, Fully Autonomous

Step 2: Semantic Tag mapping

This process assigns tags to a unified namespace by applying contextual understanding and ontology-based alignment.



- **Traditional Approach:** Manual Ontology Matching: Data engineers manually map sensor tags like "Flow_Rate" or "Temp_Sensor" from different vendor systems to a common data schema. Fixed Rules and Dictionaries: Predefined dictionaries for terms like "Pressure_Sensor" vs. "Pres_Sens" are created. Static Domain Models: Models based on known industry standards (e.g., API standards) are periodically updated manually.

- **AI-Driven Approach:** Ontology Matching: AI maps sensor tags from multiple vendors (e.g., "FlowRate," "Flow_R," "FR_01") to a unified schema based on learned relationships. Contextual Understanding: Using historical sensor logs, NLP models like BERT identify "Pressure_Sensor" and "Pres_Sens" as synonyms. Domain-Specific Tag Libraries: Custom AI models trained on midstream operational datasets ensure accurate mapping across pipeline assets.

- **Agentic AI with SLM Approach:** An SLM integrates contextual embeddings to map tags across disparate vendor systems, while Agentic AI dynamically aligns tags with a unified namespace. For example, when handling complex tags like "ZoneA.FlowRate.Pipe01" and "Flow_P01.Zone1," SLMs parse structural hierarchies, infer equivalencies, and map them to a shared ontology. Agentic AI continuously monitors for new tags (e.g., "PipeFlow_Zone_A01"), applies probabilistic scoring for alignment, and updates ontologies in real time. This ensures seamless integration and reduces manual dependency for complex, domain-specific systems.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI Approach
Mapping Accuracy	Moderate, Rule-Based	High, Contextual Understanding	Very High, Real-Time Contextual Alignment
Adaptability	Limited, Static Models	Adaptive with Retraining	Highly Adaptive, Continuous Ontology Updates
Maintenance Effort	High, Manual Updates	Low, Self-Learning	Minimal, Proactive, and Autonomous

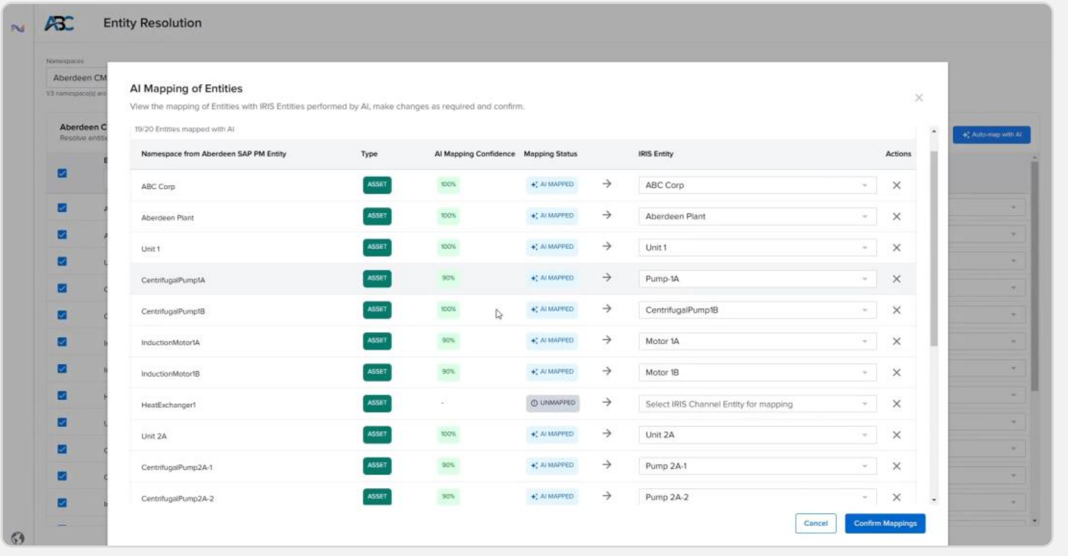
Step 3: Anomaly Detection and Correction

This process includes a range of conflict resolutions and analysis o data rules.

- **Traditional Approach:** Manual Conflict Resolution: Engineers compare readings from multiple sensors when flow rates from different sources do not align. Predefined Data Rules such as: "Flow rate > 0 during operation" are checked in periodic audits. Periodic Audits: Monthly data reviews identify missing or incorrect sensor readings.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI Approach
Efficiency	Manual, Time-Intensive	Automated, Real-Time Monitoring	Instant, Proactive Detection and Resolution
Accuracy	Moderate, Rule-Based	High, Contextual Analysis	Very High, Context-Aware, and Dynamic
Scalability	Limited by Manual Processes	Scalable with Structured Data	Unlimited, Adaptive to Complex Environments
Conflict Resolution	Manual Checks and Audits	Automated with ML Algorithms	Autonomous, Probabilistic, and Adaptive
Contextual Understanding	Limited to Static Rules	Learns from Historical and Operational Data	Real-Time, Domain-Specific Context Parsing
Domain Adaptability	Frequent Manual Updates for New Tags	Adaptive with Retraining	Fully Adaptive, Continuous Ontology Updates
Cost of Implementation	High Due to Manual Effort	High Initial AI Investment	Moderate, Low Ongoing Costs Due to Automation

Step 4: Intelligent Tag Recommendation



Traditional Approach:

- Manual Tag Creation: Engineers manually create new tags based on system expansion or operational changes.
- Periodic Reviews: Scheduled audits to identify missing tags or areas needing new measurements.
- Expert Consultation: Domain experts are consulted to determine appropriate tag naming and placement.

AI-Driven Approach:

- Pattern Recognition: ML models identify recurring patterns in existing tags to suggest new ones.
- Predictive Analytics: AI analyzes operational data to recommend tags for potential blind spots.
- Automated Suggestions: NLP models generate tag names based on learned conventions and contextual understanding.

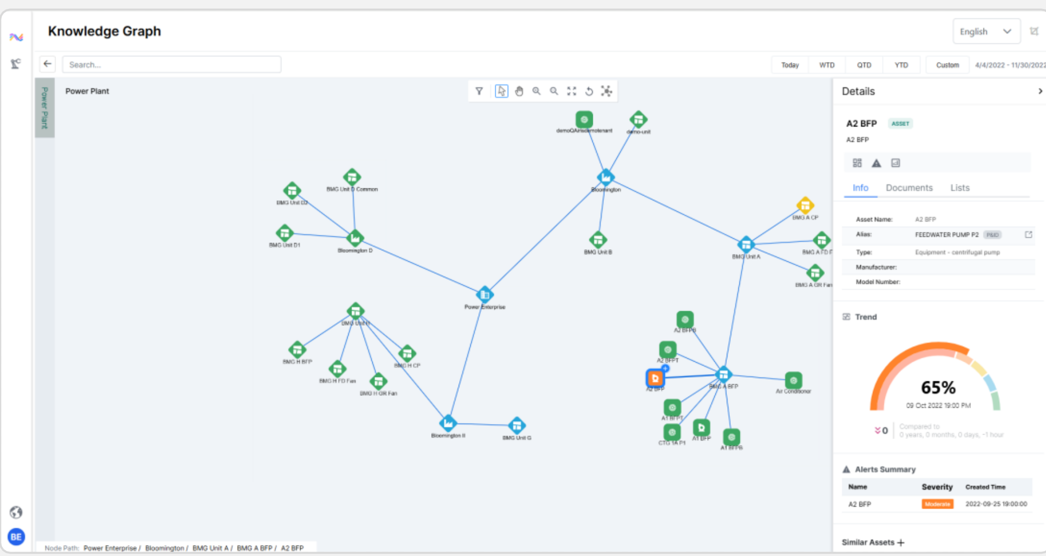
Agentic AI with SLM Approach:

- Semantic Analysis: SLM analyzes existing tag relationships and operational context to identify gaps.
- Dynamic Recommendation: Agentic AI proactively suggests new tags based on evolving system configurations and operational patterns.
- Contextual Integration: Recommendations are automatically aligned with the existing tag hierarchy and naming conventions.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI Approach
Recommendation Quality	Moderate, Experience-Based	High, Data-Driven	Very High, Context-Aware, and Proactive
Scalability	Limited, Manual Process	Good, Automated Suggestions	Excellent, Self-Improving System
Time Efficiency	Low, Time-Consuming	High, Rapid Suggestions	Very High, Real-Time Recommendations

The outcome will be a unified data model covering the schema, entity, tag, attribute, and value layers. Once this unification is built, the system must undergo a test and validation phase to assess the model's accuracy, predictability, stability, and repeatability. The final step is to create a knowledge graph.

Step 5: Integration with Knowledge Graphs



Traditional Approach:

- Manual Graph Construction: Engineers manually create relationships between tags and entities.
- Static Relationships: Connections between data points are predefined and infrequently updated.
- Limited Scope: Graphs typically focus on direct, known relationships.

AI-Driven Approach:

- Automated Graph Building: ML algorithms construct initial graphs based on tag relationships and operational data.
- Dynamic Updates: Graphs are periodically updated using new data and detected patterns.
- Inference Capabilities: AI models can infer indirect relationships between entities.

Agentic AI with SLM Approach:

- Semantic Graph Construction: SLM builds rich, contextual relationships between tags, entities, and operational concepts.
- Real-Time Graph Evolution: Agentic AI continuously updates the graph structure as new data and relationships emerge.
- Proactive Insight Generation: The system autonomously identifies and adds new nodes and edges for emerging operational patterns.

Criteria	Traditional Approach	AI-Driven Approach	SLM + Agentic AI Approach
Graph Complexity	Low, Basic Relationships	High, Multi-Dimensional	Very High, Contextual, and Temporal
Adaptability	Limited, Manual Updates	Good, Periodic Updates	Excellent, Continuous Real-Time Evolution
Insight Generation	Basic, Predefined Queries	Advanced Pattern Recognition	Comprehensive, Proactive, and Predictive

Midstream Oil & Gas Example

The following example outlines how data from a midstream Oil & Gas operator can be contextualized into a unified data model using SLM + Agentic AI.

Scenario Overview

The midstream operation consists of 32 Programmable Logic Controllers (PLCs) categorized as follows:

- **10 PLCs:** Follow a consistent naming convention (e.g., ZoneA.FlowRate_01).
- **15 PLCs:** Use a different naming convention (e.g., Flow_P01.Zone1).
- **7 PLCs:** Use non-semantic tags with no clear naming logic, relying on data flow patterns for identification (e.g., Tag001, SensorX).

  Additionally:

- SCADA systems and historians contain modified tag names over time (e.g., FlowRate renamed to FR_Main_01).

Data unification must address conflicts, anomalies, and misaligned naming conventions while contextualizing this diverse dataset into a unified namespace. Sample equipment and data include:

Equipment Type	Tag Examples (Class 1 PLCs)	Tag Examples (Class 2 PLCs)	Tag Examples (Class 3 PLCs)	Tag Examples (SCADA)	Tag Examples (Historian)	Notes
Flow Meters	AreaA.FlowRate.S1, AreaB.Flow.S2	FR_A1, FRate_B2	T001, F003	FlowRate_Area1, FRate_ZoneA_1	FR1, FlowZone2	Class 1 uses structured tags, Class 2 uses abbreviations, and Class 3 lacks semantic meaning. SCADA renames tags, and the historian truncates them.
Pressure Sensors	AreaA.Pressure.S1, AreaB.Pressure	PRS_A1, Press_B2	P002, P004	Pressure_Area1, Pres_Zone1	P1, Pres_Main	Class 3 uses non-semantic tags (P002) inferred through context. Historian truncates Class 2 tags further (PRS_A1 → P1).

Temperature Sensors	AreaB.TempSensor.S2, Temp.AreaC	TMP_B2, Temp_C3	T002, T005	Temp_AreaB_Sensor2, Temp_Zone2	T2, Temp_B2	SCADA renames tags systematically; the historian compresses Class 2 and Class 3 tags (T002 → T2).
Vibration Monitors	AreaC.Vibration.S3, Vib.AreaB	VBRT_A3, Vib_C3	V003, V006	Vibration_AreaC_Sensor3, Vib_Zone3	V3, Vibration_Zone1	Class 3 tags like V003 lack context and require inference from dependency relationships.
Compressors	Comp_001.Pressure, Comp_A1.Temp	Comp_A1, Comp_T2	T002_Comp, P003_Comp	Comp_Area1_Pressure, Comp_Temp_Zone1	C1P, Comp Zone1.FlowRate	Multi-parameter data adds complexity. Historian truncates compressor tags (T002_Comp → C1P).
Pumps	Pump_A.FlowRate, Pump_B.Pressure	PumpZone2.FRate, Pump2.Press	P001, P005	Pump_Area1_FlowRate, Pump_Zone1	P1F, Pump01.Main	Class 3 tags require inference for contextualization; SCADA renaming and historian compression complicate alignment.
Valves	Valve_A.Position, Valve_B.Temp	ValveZone2.Temp, Valve_B2.Pos	V001, V007	Valve_Area1_Position, Valve_Zone1	V1, Valve1_Status	Class 3 tags like V001 are ambiguous. Historian truncates SCADA tags further (Valve_Area1_Position → V1).
Turbines	Turbine_A.Power, Turbine_B.RPM	TurbineZone1.Temp, Turbine_B2.Power	T002, T008	Turbine_Area1_Power, Turbine_Zone1	T1P, Turbine1.RPM	Historian uses highly compressed tags (T002 → T1P). Turbine data involves multi-parameter information like power, RPM, and

Step 1: Automated Data Ingestion and Normalization

Input

- Raw data streams from PLCs (Class 1, 2, and 3), SCADA, and historian systems in mixed formats (JSON, CSV, XML).
- Metadata inconsistencies: Some PLCs include unit and location metadata, while others lack it entirely.

Applied Process

- Multi-format Parsing:
  - Libraries like pandas and xml.etree are used to extract schema details. Tags like FR_A1 and P002 are structured into rows with metadata attributes (e.g., time, value, units).
- SLM Tokenization:
  - SLM models tokenize tags (e.g., splitting AreaA.FlowRate.S1 into components like AreaA, FlowRate, S1). This creates embeddings capturing semantic relationships.
- Regex-Based Normalization:
  - Patterns like FR → FlowRate are matched using predefined rules. SLM embeddings handle ambiguous abbreviations like PRS or T001 by comparing tag similarity across PLC classes.

Output

- A normalized data structure with consistent tag formats across PLCs, SCADA, and historian systems.
- Example: FR_A1, F003, FRate_A1, and FlowRate.ZoneA1 normalized to FlowRate_Area1.

Human Involvement

- Initial setup: Domain experts define common patterns (FR → FlowRate).
- Minimal validation: Human intervention is needed only for outliers or ambiguous tags not handled by embeddings.

Benefit of SLM + Agentic AI

- Self-learning embeddings reduce reliance on hardcoded patterns, enabling dynamic adaptation to new tag formats.

Step 2: Semantic Tag Mapping

Input

- Normalized tags from Step 1 with partial metadata and inferred relationships.
- Examples: FlowRate_Area1, P002, Temp.ZoneA.

Applied Process

- Ontology Alignment:
  - SLM embeddings match tags to a predefined ontology (e.g., FlowRate_Main, Pressure_Main) using cosine similarity scoring.
- Probabilistic Expansion:
  - Bayesian models infer relationships for ambiguous tags (e.g., P002 mapped to Pressure.Zone1 based on location and flow dependencies).
- Dependency Graph Construction:
  - A DAG is created for Class 3 PLCs, linking T001 to upstream FlowRate_Area1 based on operational patterns.

Output

- Unified tag mappings enriched with semantic and operational context.
- Example: T001 linked to Temp.Sensor.Area1 using flow dependencies and metadata.

Human Involvement

- Ontology refinement: Domain experts periodically validate ontology mappings.
- Conflict resolution: Human intervention is needed for new equipment types or regions with no historical data.

Benefit of SLM + Agentic AI

- Dynamic, self-adaptive mapping minimizes manual efforts for ambiguous or evolving tag formats.

Step 2.1: Schema Recognition

Input

- Tag mappings and schemas from SCADA and historian systems.
- Example: SCADA schema has FlowRate.Area1, historian stores as FR1.

Applied Process

- Schema Matching:
  - SLM embeddings match fields across schemas (e.g., FlowRate.Area1 → FR1).
- Metadata Validation:
  - Metadata attributes (units, timestamp format) are cross-validated to ensure alignment.
- Dynamic Updates:
  - Agentic AI detects schema changes in real-time and adjusts mappings (e.g., new tag FRate.ZoneA1 automatically added).

Output

- A unified schema integrating SCADA and historian data.
- Example: Pressure.Area1 (SCADA) and P1 (historian) are unified as Pressure_Main.

Human Involvement

- Minimal validation: Experts verify critical schema relationships initially.
- Real-time schema changes are handled autonomously by Agentic AI.

Benefit of SLM + Agentic AI

- Continuous schema monitoring eliminates manual rework during schema changes.

Step 3: Anomaly Detection and Correction

Input

- Unified schema with real-time sensor data.
- Example: FlowRate_Area1 reports 120 GPM in SCADA and 80 GPM in historian.

Applied Process

- Anomaly Detection:
- SLM embeddings detect deviations in real time by comparing current data to historical patterns.

Probabilistic Correction

- Agentic AI applies calibration corrections or flags anomalies for human review.

Feedback Loop

- Unresolved conflicts are logged, and corrections are incorporated into future inference models.

Output

- Anomaly-free, consistent data streams.
- Example: Calibration drift corrected, aligning FlowRate_Area1 to 100 GPM.

Human Involvement

- Validation: Experts verify flagged anomalies and update system rules as necessary.

Benefit of SLM + Agentic AI

- Real-time correction reduces downtime and improves data reliability with minimal human oversight.

Step 4: Intelligent Tag Recommendation

Input

- Unified schema with contextualized tags from previous steps.
- Example: FlowRate_Area1, Pressure_Main, Temp.Sensor.Area1.

Applied Process

- Semantic Analysis:
  - SLM embeddings analyze existing tag relationships and operational context.
- Pattern Recognition:
  - Agentic AI identifies recurring patterns in tag usage and relationships across different PLCs and areas.
- Predictive Modeling:
  - Machine learning models predict likely new tags based on historical data and current system configuration.

Output

- Suggested new tags or modifications to existing tags.
- Example: Recommendation to add "Efficiency_Compressor1" based on relationships between pressure, flow rate, and temperature tags.

Human Involvement

- Approval: Experts review and approve suggested tags before implementation.
- Feedback: Engineers provide input on the relevance and accuracy of recommendations.

Benefit of SLM + Agentic AI

- Proactive tag management reduces manual effort in identifying and creating new tags, ensuring consistent naming conventions and improving overall system organization.

Step 5: Integration with Knowledge Graphs

Input

- Contextualized, anomaly-free tags and unified schema.

Applied Process

- Graph Construction:
  - Tags are linked as nodes, and semantic/operational relationships (e.g., FlowRate → Pressure) are captured as edges.
- Graph Enrichment:
  - Agentic AI dynamically adds new nodes and edges for emerging relationships (e.g., linking Vibration.Zone1 to compressor failure patterns).

Output

- A knowledge graph enabling advanced analytics and decision-making.
- Example: Compressor anomalies linked to downstream pressure drops and upstream vibration signals.

Human Involvement

- Initial graph design: Experts define core relationships.
- Minimal ongoing input: Agentic AI autonomously updates the graph.

Benefit of Agentic AI with SLM

- Scalable, self-adaptive graph construction accelerates actionable insights without manual data preparation.

Key Outcomes

- Unified and contextualized tags for diverse equipment and systems.
- Real-time anomaly detection and resolution improve operational efficiency.
- Knowledge graph integration enables predictive analytics and advanced decision-making with minimal human intervention.

Source Data Example

Class 1 PLCs (Consistent Naming)

Tag	Value	Units	Timestamp	Notes
AreaA. FlowRate.S1	120	GPM	2025-01-08T10:00:00Z	Consistent naming, easily understood.
AreaA. Pressure.S1	15	PSI	2025-01-08T10:00:00Z	Clear and well-documented metadata.

Class 2 PLCs (Alternative / Region Specific Naming)

Tag	Value	Units	Timestamp	Notes
FR_Zone1		GPM	2025-01-08T10:00:00Z	Abbreviated naming requires mapping.
PRS_Zone1		PSI	2025-01-08T10:00:00Z	Region-based shorthand adds ambiguity.

Class 3 PLCs (Non-semantic Tags)

Tag	Value	Units	Timestamp	Notes
T0003		GPM	2025-01-08T10:00:00Z	Non-semantic, lacks metadata context.
P0007		PSI	2025-01-08T10:00:00Z	Requires inference from relationships.

SCADA system

Tag	Value	Units	Timestamp	Notes
Flow.A1. Zone1	120	GPM	2025-01-08T10:00:00Z	Modified naming introduces conflicts.
Pressure. Zone1.Main	14.5	PSI	2025-01-08T10:00:00Z	Updated metadata available.

Historian System

Tag	Value	Units	Timestamp	Notes
F1_Zone1	120	GPM	2025-01-08T10:00:00Z	Truncated tags lose semantic meaning.
P1_Main	15	PSI	2025-01-08T10:00:00Z	Compressed data creates ambiguity.

Unified Data Model

Unified Tag	Value	Units	Timestamp	Aliases from Source Data	Contextual Relationships
FlowRate_ Main	120	GPM	2025-01-08T10:00:00Z	AreaA.FlowRate .S1, FR_Zone1, T0003, Flow.A1.Zone1, F1_Zone1	Connected to Pressure_Main and Valve_Zone1.
Pressure_ Main	15	PSI	2025-01-08T10:00:00Z	AreaA.Pressure. S1, PRS_Zone1, P0007, Pressure.Zone1. Main, P1_Main	Connected to FlowRate_Main and Compressor_1.

Conclusion

Agentic AI, supported by a fine-tuned small language model (SLM) approach, offers significant advantages in data unification for complex operational environments like midstream oil and gas. Traditional data integration projects often fail after an initially motivated start due to fragmented, subjective, and sometimes conflicting domain expertise. As these projects drag on, the opportunity cost of involving SMEs and data experts becomes evident to executive management. Agentic AI with an SLM addresses this challenge by breaking free from "static inertia," generating an initial baseline of data groupings directly from unstructured data. This enables experts to collaborate more cohesively, building on data-driven insights rather than relying solely on subjective knowledge. Additionally, this approach bypasses the need for rigid rule-based models, focusing instead on semantics and knowledge-driven correlations to create an adaptive meta-model.

Since operational data is continuously generated, with new fragments and anomalies emerging regularly, deploying AI-driven models at the right operational points ensures ongoing learning and adaptation. Properly designed, these models can outperform static rule-based systems by learning from evolving data patterns, making data unification scalable and sustainable.

References

World Economic Forum
McKinsey & Company