A Framework for Semantic Identifier Resolution and LLM Grounding in Engineering Systems

Rustam Mehmandarov[1], Arild Waaler[1]

[1] *Department of Informatics, University of Oslo, Norway*

Abstract

The digital transformation of engineering systems demands scalable and precise identifier management. Information about a single asset is often fragmented across numerous systems and organizational boundaries, with each context using its own identifiers. This paper addresses this challenge by introducing a formal framework for semantic reference. We build on prior work by introducing two key concepts: **reference contexts**, which formalize the boundary conditions for identifier interpretation, and **public models**, which serve as curated, shared layers for anchoring reference. We define reference equality as the symmetric, transitive closure over typed proxy relations that link identifiers across these contexts. Finally, we demonstrate how this semantic infrastructure provides a foundational component for a new generation of industrial systems, including providing essential grounding for Large Language Model (LLM) workflows. This approach bridges the gap between human-readable descriptors and machine-readable identifiers, aligns with modern architectural principles like Data Mesh, and supports hybrid reasoning in industrial knowledge systems.

Keywords

Identifier Management, Semantic Integration, Large Language Models, Reference Context, Data Mesh, Knowledge Graphs, Engineering Systems, Asset Management, Data Integration, Software Interoperability

1. Introduction

Industrial systems rely on vast quantities of information, scattered across tools, organizations, and lifecycle stages. This information is fragmented across databases, models, and documents, each using its own identifiers to refer to system elements like equipment or functional locations [1]. Resolving these identifiers across contexts is a prerequisite for automation and digital traceability, but current practices are heavily reliant on manual, error-prone intervention.

Recent work has highlighted semantic identifier management as a cornerstone of digital transformation [1, 2]. Prior approaches have introduced identifier classification schemes and model-based mapping services. Yet, a principled and formal account of reference semantics—linking identifiers, models, and human interpretation across heterogeneous systems—has been lacking.

Large Language Models (LLMs) are increasingly used as natural language interfaces to engineering data. They are now employed to query structured sources and assist engineers in technical tasks. However, LLMs lack a native notion of reference. They process text without grounding it in the structured semantics of the systems they support, resulting in ambiguity, hallucinations, and a lack of verifiable traceability in their outputs [3].

This paper addresses both challenges—semantic reference and LLM grounding—by introducing two interlinked concepts:

- **Reference contexts**, which formalize the use of identifiers under specific boundary conditions.
- **Public models**, which serve as curated referential layers for deriving shared descriptors.

We demonstrate how these structures enable robust semantic reference resolution and provide the necessary anchoring points for LLM workflows. Our approach extends prior work by focusing on logic-based interpretation and integration into hybrid, knowledge-based systems.

SGKi 2025: Scaling Knowledge Graphs for Industry Workshop, co-located with SEMANTiCS'25: International Conference on Semantic Systems, September 3–5, 2025, Vienna, Austria

rustamm@ifi.uio.no (R. Mehmandarov); arild@ifi.uio.no (A. Waaler)

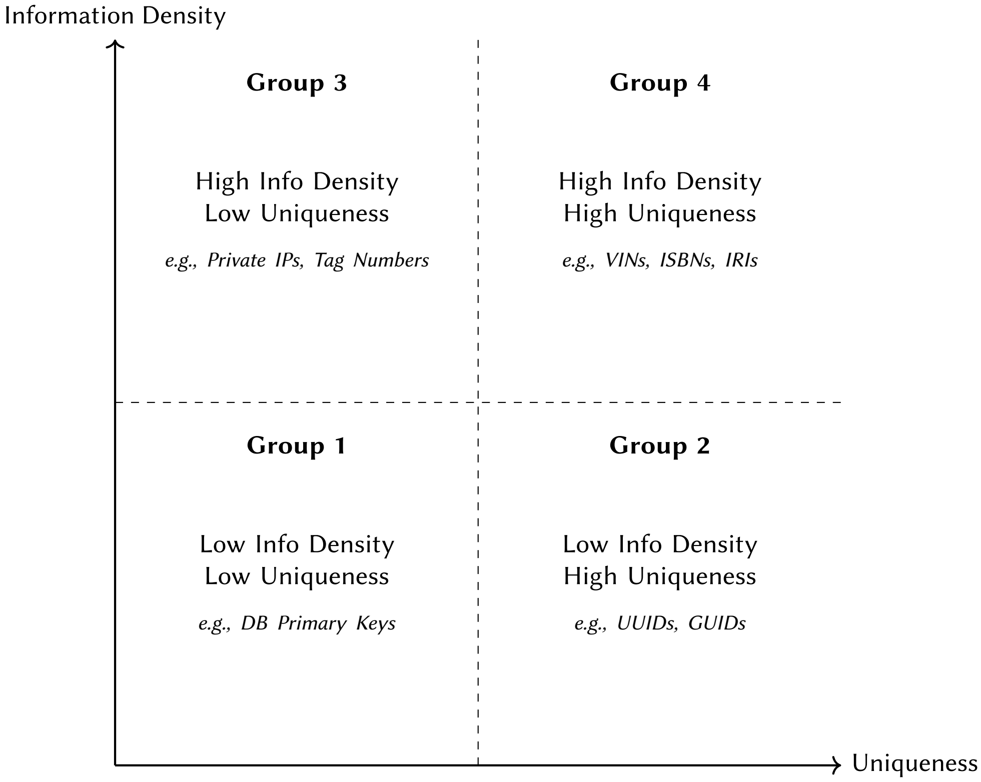
2. Background and Related Work

2.1. The Identifier Fragmentation Problem

In large-scale engineering projects, information about a single asset is scattered across numerous applications. Each system assigns its own identifier based on its specific context, domain, or lifecycle phase. The challenge lies in the heterogeneity of these identifiers—not only in syntax, but in structure and semantics.

To better understand the diversity of identifier types encountered in industrial systems, we introduce a two-dimensional classification scheme based on *information density* and *uniqueness*. This classification helps reveal structural patterns that motivate the need for explicit, semantics-based identifier resolution. The resulting taxonomy can be visualized in four quadrants, as shown in Figure 1.

Figure 1: The Four-Quadrant Classification of Identifier Systems.
Each quadrant represents a different trade-off, from simple, context-specific database keys (Group 1) to globally unique and semantically rich identifiers like Vehicle Identification Numbers (Group 4).



This classification reveals a fundamental tension: highly unique identifiers (Groups 1 and 2) are often opaque to humans, while interpretable descriptors (Groups 3 and 4) tend to be ambiguous or reused. As a result, resolving identifiers across systems cannot rely on lexical matching alone. Instead, it requires *explicit, declarative links* between identifiers—regardless of their type or quadrant—which is precisely what our framework provides through proxy relations and reference equality.

Simple lexical or structural matching techniques are often insufficient [4], as the information subsets in different systems may have little overlap. Consequently, organizations rely on manual mapping by subject matter experts [1], a process that is neither scalable nor sustainable. This challenge aligns with the goals of the **Data Mesh** paradigm, which seeks to enable data interoperability in a decentralized ecosystem while maintaining domain-oriented ownership [5].

2.2. Advances in Semantic Integration

Prior work has aimed to solve this fragmentation using model-based integration and semantic technologies. The approaches in [1, 2] distinguish between human-readable *descriptors* (typically Groups 3 and 4) and system-level *identifiers* (Groups 1 and 2), proposing:

- A modular, plug-in-based web service for identifier resolution that uses domain-specific rules to perform semantic lifting and lookup.
- A classification of identifier systems by their function and lifecycle governance, along with a proposal for layered model views tied to specific information aspects and actor roles.

These articles advocate for a shift from document-centric practices to model-driven practices, aligning with standards like **ISO/IEC 81346** [6]. However, they do not provide a formal semantics of reference or a general theory for cross-context resolution.

2.3. The LLM Grounding Challenge

LLMs are powerful tools for natural language tasks, but lack the mechanisms to handle identifier semantics reliably. Without a formal model of reference, they cannot distinguish between identifiers that are homonyms (same string, different meaning) or synonyms (different strings, same meaning). This is a critical failure point in technical applications, especially in Retrieval-Augmented Generation (RAG) pipelines, where the model's output quality is directly tied to its ability to interpret retrieved, and potentially ambiguous, information [7].

Recent research has focused on grounding LLMs by enabling them to use external tools and APIs, a capability now widely available in major models, often under the name "function calling". Frameworks like ReAct [8] interleave reasoning and action steps, while models like Toolformer [9] are fine-tuned to decide when and how to call external services. While these "tool-using" LLMs represent a significant advance, they presuppose the existence of reliable, well-defined APIs. They do not, in themselves, solve the underlying semantic interoperability problem when the tools and data sources they query suffer from the identifier fragmentation described in Section 2.1. Our work provides the missing semantic resolution layer required for such tool-using LLMs to function reliably in heterogeneous industrial environments.

3. Reference Contexts and Semantic Identifier Resolution

Engineering and industrial information systems rely on large numbers of identifiers to reference assets, components, activities, and locations. However, these identifiers are rarely global or semantically transparent. They are typically defined and interpreted within bounded domains—such as organizational databases, contractor systems, or design environments—each governed by local conventions. To manage semantic interoperability in this setting, we introduce the concept of a *reference context* as a formal unit of identifier interpretation.

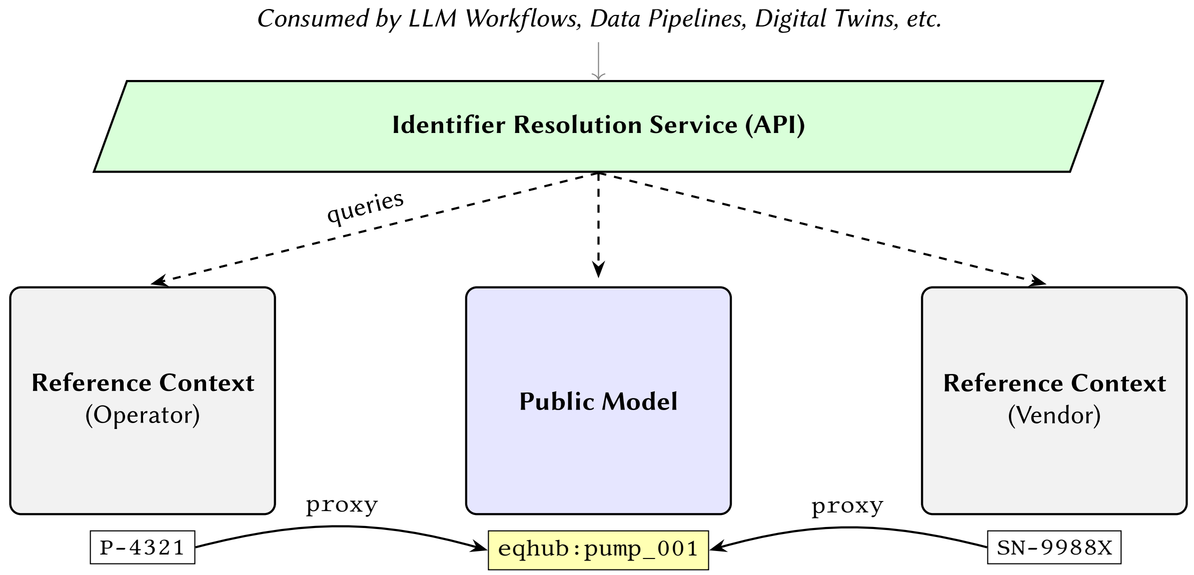
A **reference context** R defines a local referential environment. It specifies a disjoint set of identifiers $\mathcal{I}_R$, partitioned into two categories:

- $\mathcal{D}_R$: a set of *descriptors*, such as tag numbers or internal model labels, whose meaning depends on implicit or context-specific knowledge;
- $\mathcal{A}_R$: a set of *direct references*, such as database primary keys, serial numbers, or ontology URIs, whose interpretation is grounded in a formal or empirical reference base.

In the terminology of our framework, descriptors (typically from Groups 3 and 4) belong to $\mathcal{D}_R$, while system-level keys and global identifiers (Groups 1 and 2) typically serve as direct references in $\mathcal{A}_R$. Our resolution relation $\mathcal{R}_R$ captures the mapping between these two classes within each reference context. To relate these two sets, each reference context may define a **resolution relation**

$$\mathcal{R}_R \subseteq \mathcal{D}_R \times \mathcal{A}_R,$$

Figure 2: The Semantic Resolution Architecture.
Local identifiers within distinct **Reference Contexts** (e.g., Operator, Vendor) are linked via curated **Proxy Relations** to a canonical entity in the shared **Public Model**. The **Identifier Resolution Service** acts as a unified API layer, enabling external systems like LLMs or data pipelines to query this federated structure and resolve references across domains.



which expresses how descriptors can be resolved to direct references. Importantly, this relation is neither total nor functional in general. A descriptor may resolve to multiple direct references (e.g., due to modeling ambiguity or distributed realization), or to none at all (e.g., if it represents a design placeholder or a failed lookup). Likewise, a single direct reference may be associated with multiple descriptors originating from different naming conventions. This many-to-many structure reflects real-world ambiguity and design abstraction. It enables the framework to support both extensional identifiers (grounded in specific artifacts) and intensional ones (defined through schema, queries, or roles).

To support interoperability across multiple reference contexts, we introduce the notion of a **public model**. A public model is a curated referential layer hosted by a recognized authority—such as an industry consortium like POSC Caesar Association (PCA) [10], standards body, or shared data platform—that spans several reference contexts. It defines:

- A set of referable elements, each assigned a globally stable identifier (e.g., `pca:pump_001`);
- A set of *proxy relations* linking these public identifiers to identifiers drawn from individual reference contexts;
- A resolution API or service interface that enables identifier translation, cross-context querying, and semantic reference checks.

A **proxy relation** captures a semantic link between identifiers drawn from different reference contexts. In our framework, this relation is used to assert that two identifiers are different representations or views of the *same underlying system element.* We denote by $\mathcal{I}$ the global set of all identifiers:

$$\mathcal{I} := \bigcup_{R} \mathcal{I}_R.$$

The proxy relation is a binary relation over $\mathcal{I}$:

$$\texttt{proxy} \subseteq \mathcal{I} \times \mathcal{I}.$$

An assertion $(i_1, i_2) \in \texttt{proxy}$ states that identifier i_1 serves as a proxy for identifier i_2. These links are explicitly declared and curated as part of the public model. They may be typed to express their interpretation—for example, indicating that a functional capability is fulfilled by an implementation, or that one identifier administratively mirrors another. The complete set of reference contexts and proxy relations can thus be naturally represented as a knowledge graph, where identifiers are nodes and proxy relations are typed edges.

From this relation, we define **reference equality** as the symmetric, transitive closure of proxy links:

$$\texttt{refEq}(i,j) \iff i \equiv j \text{ under } \texttt{proxy}^*_{\text{sym}}.$$

This means that two identifiers are considered reference-equal if they are connected through a chain of proxy assertions, regardless of their origin or naming convention. Reference equality provides a declarative, traceable, and explainable basis for establishing semantic equivalence across heterogeneous systems and naming schemes.

By organizing identifiers into structured reference contexts and linking them via public proxy models, this framework provides a robust foundation for semantic resolution. It enables alignment between human-oriented labels and machine-level identifiers, supports cross-organizational data flows, and provides a formal anchoring mechanism for integrating symbolic reasoning with LLM-based workflows.

4. Application to LLM Workflows

Our framework provides the semantic scaffolding needed to ground LLM operations in verifiable data.

By linking identifiers in natural language to elements in a public model, we create **semantic anchor points**. This allows an LLM to disambiguate references correctly. This is particularly effective in RAG pipelines [7], where retrieved documents can be annotated with public descriptors to resolve ambiguity before generation.

We propose an identifier resolution service with three primary functions:

1. `resolve(id, context)`: Returns a corresponding public identifier (if one exists) and the traceable proxy path.
2. `areEqual(id1, id2)`: Returns true if a proxy chain exists between the two identifiers.
3. `getRelated(public_id, role)`: Returns all known context-specific identifiers linked to the given public identifier in the role of vendor.

This service equips LLMs with the ability to validate references, disambiguate terms, and generate text that is demonstrably grounded in a formal, external model.

5. Case Study: Multi-Context Reference to a Pump

Consider a pump referenced across multiple contexts in an engineering project, as detailed in Table 1.

Table 1
Identifier Contexts for a Shared Pump Asset.

Context ID	Role	Identifier Type	Example ID
C_1	Operator	Descriptor	`P-4321`
C_2	EPC Contractor	Descriptor	`EJ101A`
C_3	Vendor	Serial Number	`SN-9988X`
C_4	Public Model	Public ID	`eqhub:pump_001`

The public model (C_4) declares proxy links from `eqhub:pump_001` to the other three identifiers. By computing the symmetric transitive closure, a system can deduce that these identifiers belong to the same reference-equal group via proxy chains.

An LLM-powered assistant receives the prompt: *"Generate a maintenance summary for P-4321, including key specs from the vendor datasheet."*

A standard RAG pipeline, lacking semantic grounding, would first retrieve documents based on the string "P-4321". This initial retrieval is likely to be ambiguous, potentially returning datasheets for assets with the same tag number but located in different plants or projects.

Using the resolution service, the LLM workflow executes these steps:

1. **Call:** `resolve("P-4321", context=operator)`
   Returns: `eqhub:pump_001`.
2. **Call:** `getRelated("eqhub:pump_001", role=vendor)`
   Returns: `SN-9988X` and a link to the vendor's documentation.

The LLM can now reliably retrieve the correct vendor datasheet using the serial number and generate an accurate, traceable summary.

6. Discussion

6.1. Architectural Alignment and Framework Implications

The proposed framework complements existing industry standards. Public models can be exposed through **Asset Administration Shell (AAS)** interfaces [11], and the layered approach aligns with the **RAMI 4.0** architecture [12]. By enabling federated governance while ensuring interoperability, the model directly supports the core principles of a **Data Mesh** [5], as shown in Table 2.

Table 2
Alignment of the Proposed Framework with Data Mesh Principles.

Data Mesh Principle	How the Framework Addresses It
1. Domain-Oriented Ownership	The framework uses **Reference Contexts** to formalize domain-specific identifier systems. Each domain (e.g., operator, vendor) retains full ownership and control over its internal identifiers, avoiding the need for a single, centralized master ID.
2. Data as a Product	Domains treat their identifier mappings as a discoverable and usable 'product'. They publish **Proxy Relations** to a shared **Public Model**, which acts as a clean, interoperable interface. The **Identifier Resolution Service** serves as the API for consuming this data product.
3. Self-Serve Data Platform	The proposed architecture is a core component of a self-serve platform. The platform hosts the **Public Models** and the graph of **Proxy Relations**, while the **Identifier Resolution Service** provides a universal, self-serve tool that enables all domains to easily create and consume mappings.
4. Federated Computational Governance	Governance is **federated**, not centralized. Domains collaborate to define proxy links and agree on public models. This governance is also **computational**: reference equality is not just a policy but is actively computed and enforced by the system through the transitive closure of the proxy relations in the graph. The quadrant classification provides a conceptual backdrop for our resolution model: it helps frame the diversity of identifiers as not merely syntactic variation but as structural variation that must be reconciled through formally declared relations. This approach enables semantic alignment across systems without requiring a universal identifier scheme, by instead relying on explicitly declared proxy chains.

This approach addresses the identifier bottleneck by replacing the need for total harmonization with incremental, proxy-based resolution. It is important to note that while we emphasize the application for LLM grounding, the resolution service is a general-purpose integration component that can be consumed directly by any automated system, such as data integration pipelines or digital twin platforms, that requires robust, cross-context identifier lookups. For LLMs, it provides the structured grounding required for reliable and safe operation in technical domains. Open challenges include the formal governance of proxy relations, ensuring the scalability of resolution services, and developing methods to fine-tune LLMs on reference-aware architectures.

The framework proposed in this paper also complements practices in Model-Based Systems Engineering (MBSE), particularly those grounded in standards such as ISO/IEC/IEEE 42010 [13] and modeling languages like SysML [14]. These approaches focus on the structural, behavioral, and architectural aspects of system design, typically within a well-defined viewpoint framework. However, they often treat identifier semantics—how system elements are named, referenced, and reconciled across tools or phases—as external to the core modeling formalism.

Our framework addresses this gap. It introduces a semantic infrastructure for identifier resolution that is orthogonal to structural modeling concerns but essential for enabling lifecycle traceability, model integration, and cross-organizational interoperability. In this sense, the proposed framework can serve as a semantic backbone to MBSE environments by providing formally grounded mechanisms for managing reference equality, declaring proxy relationships, and grounding engineering artifacts across the lifecycle.

Our framework sketches a roadmap for a new generation of hybrid, reference-aware industrial systems. This vision begins with establishing the core semantic infrastructure, which is then used to ground LLM-based assistants, making them safe and reliable for technical work. The roadmap culminates in a virtuous cycle, as described in Section 6.2, where these grounded LLMs are, in turn, employed to accelerate the curation and expansion of the very knowledge base they rely on. This creates a scalable, self-improving ecosystem that bridges the gap between static engineering knowledge and dynamic AI capabilities.

6.2. The Role of LLMs in Automating Proxy Relation Curation

While this paper focuses on LLMs as consumers of a pre-existing semantic infrastructure, they can also serve as powerful tools to automate and scale the creation of this infrastructure itself. The curation of `proxy` relations is currently a knowledge-intensive task reliant on subject matter experts. We propose that LLMs can accelerate this process in a human-in-the-loop workflow:

- **Candidate Generation:** An LLM can be tasked to read heterogeneous sources, such as technical datasheets, piping and instrumentation diagrams (P&IDs), and maintenance logs. By identifying co-occurrence patterns and understanding document context, the LLM can propose potential proxy links. For example, it might suggest that a tag number from a P&ID and a serial number from a vendor manual likely refer to the same asset.
- **Expert Validation:** These LLM-generated candidates are then presented to a human domain expert for validation. This shifts the expert's role from manual, painstaking discovery to efficient verification, dramatically increasing the rate at which the proxy graph can be populated.

This semi-automated approach leverages the pattern-matching capabilities of LLMs to tackle the scaling problem in knowledge graph construction, while using human expertise to mitigate the risk of hallucinations and ensure the accuracy of the final mappings.

7. Conclusion

We introduced a semantic infrastructure for identifier management built on reference contexts, public models, and typed proxy relations. This framework allows reference equality to be established through curated proxy chains, enabling traceable and verifiable cross-context reasoning over identifier relations. Crucially, we have shown how this formal approach aligns with modern IT architecture principles such as Data Mesh, providing a practical path for implementation in decentralized enterprise ecosystems. By integrating these mechanisms with LLM workflows, we provide a robust solution for semantic grounding, supporting safe and reliable AI-assisted engineering.

While our examples are drawn from engineering systems, the underlying principles are widely applicable. Similar challenges of identifier fragmentation and context-dependent interpretation arise in domains such as healthcare, finance, supply chains, and public administration. We believe that the proposed framework can serve as a general foundation for reference-aware integration in any domain where semantic consistency across heterogeneous systems is required.

Future work will proceed along several key paths:

- Formalizing the resolution logics to support a wider range of typed proxy relations.
- Designing and implementing a scalable prototype of the distributed lookup service.
- Conducting a comprehensive evaluation of the framework, both quantitative (on resolution accuracy and performance) and qualitative (with domain experts to assess usability and impact).
- Integrating the identifier resolution service directly into RAG pipelines as a formal grounding component.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] R. Mehmandarov, A. Waaler, D. Cameron, R. Fjellheim, T. B. Pettersen, A semantic approach to identifier management in engineering systems, in: 2021 IEEE International Conference on Big Data (Big Data), IEEE, 2021, pp. 4613–4616. doi:10.1109/BigData52589.2021.9671515.

[2] R. Mehmandarov, D. Hovland, T. Saltvedt, A. Waaler, Towards addressing requirements to identification posed by the digital transformation, in: Proceedings of the International Workshop on Semantic Industrial Information Modelling (SemIIM'22), CEUR-WS.org, 2022. URL: http://ceur-ws.org/Vol-3355/identifier.pdf.

[3] E. M. Bender, T. Gebru, A. McMillan-Major, S. Shmitchell, On the dangers of stochastic parrots: Can language models be too big?, in: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, FAccT '21, ACM, 2021, pp. 610–623. doi:10.1145/3442188.3445922.

[4] J. Euzenat, P. Shvaiko, Ontology Matching, Springer-Verlag, 2013. doi:10.1007/978-3-642-38721-0.

[5] Z. Dehghani, Data Mesh: Delivering Data-Driven Value at Scale, O'Reilly Media, 2022.

[6] International Organization for Standardization, Industrial systems, installations and equipment and industrial products — Structuring principles and reference designations — Part 1: Basic rules, Standard ISO/IEC 81346-1:2022, ISO, 2022. URL: https://www.iso.org/standard/82229.html.

[7] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive nlp tasks, in: H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, H.-T. Lin (Eds.), Advances in Neural Information Processing Systems 33 (NeurIPS 2020), 2020, pp. 9459–9474.

[8] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, Y. Cao, React: Synergizing reasoning and acting in language models, in: International Conference on Learning Representations (ICLR), 2023.

[9] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, T. Scialom, Toolformer: Language models can teach themselves to use tools, Advances in Neural Information Processing Systems 36 (2023) 68539–68551.

[10] POSC Caesar Association, About pca, https://www.posccaesar.org/about/, 2023. Accessed: July 31, 2025.

[11] Plattform Industrie 4.0, Details of the asset administration shell - part 1, https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part1_V3.html, 2020. Accessed: July 31, 2025.

[12] K. Schweichhart, Reference Architectural Model Industrie 4.0 (RAMI 4.0), 2017. URL: https://ec.europa.eu/futurium/en/system/files/ged/a2-schweichhart-reference_architectural_model_industrie_4.0_rami_4.0.pdf, accessed: July 31, 2025.

[13] ISO/IEC/IEEE, Systems and software engineering — Architecture description, Standard 42010:2011, International Organization for Standardization, 2011.

[14] Object Management Group, OMG Systems Modeling Language (OMG SysML®) Version 1.6, https://www.omg.org/spec/SysML/1.6/PDF, 2019. Accessed: July 31, 2025.