Orange Research
Shape the future and value innovation

ModelDesk

Shifting Models Left

End-to-End Traceability as a Foundation for Knowledge Graphs as Data Products

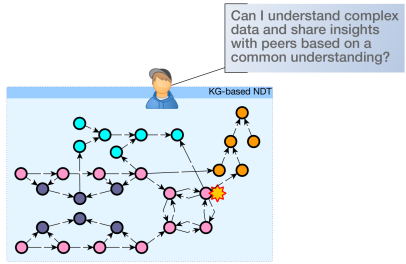
KGCW – ESWC 2026

Lionel Tailhardat, Joanna Balcerzak, Arij Elmajed, Pano Maria, Gabriel Fomi-NGameni

Orange & ModelDesk

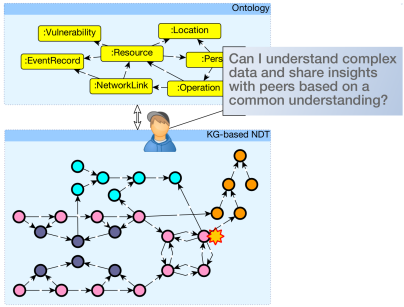
May 10, 2026

Context & Motivations – A Comprehensive View of KG-based Applications



Expectation	Multi-domain Knowledge Graph (KG) for all stakeholders
Pipeline	Consistent stitching of domain-specific subgraphs during Knowledge Graph Construction (KGC)
Support Eng.	Provenance calculus
Data Eng.	Semantic integration and KG lifecycle management
Data Owner	Data governance and data lineage

Context & Motivations – A Comprehensive View of KG-based Applications



Expectation Multi-domain Knowledge Graph (KG) for all stakeholders

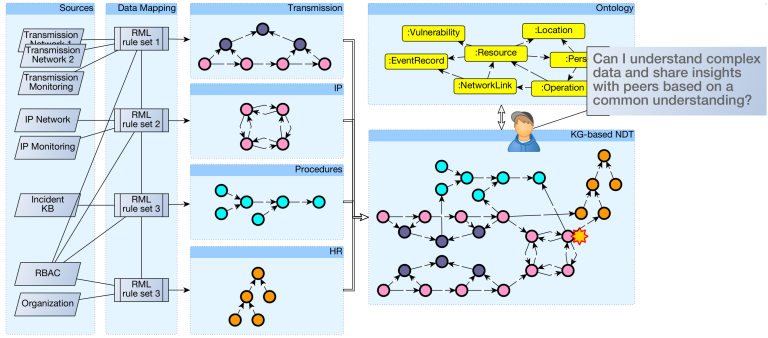
Pipeline Consistent stitching of domain-specific subgraphs during Knowledge Graph Construction (KGC)

Support Eng. Provenance calculus

Data Eng. Semantic integration and KG lifecycle management

Data Owner Data governance and data lineage

Context & Motivations – A Comprehensive View of KG-based Applications



Expectation Multi-domain Knowledge Graph (KG) for all stakeholders

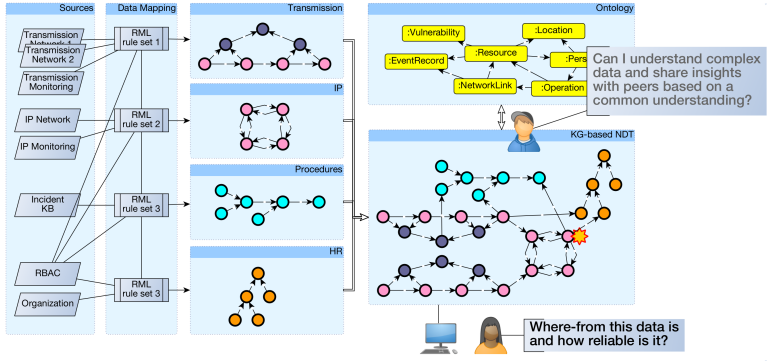
Pipeline Consistent stitching of domain-specific subgraphs during Knowledge Graph Construction (KGC)

Support Eng. Provenance calculus

Data Eng. Semantic integration and KG lifecycle management

Data Owner Data governance and data lineage

Context & Motivations – A Comprehensive View of KG-based Applications



Expectation Multi-domain Knowledge Graph (KG) for all stakeholders

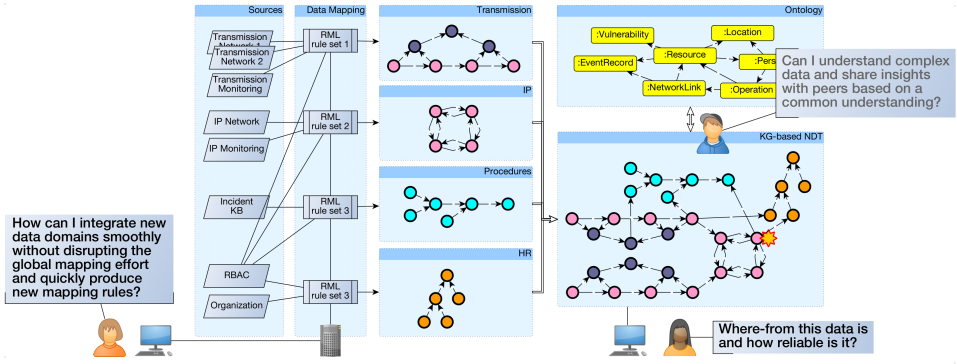
Pipeline Consistent stitching of domain-specific subgraphs during Knowledge Graph Construction (KGC)

Support Eng. Provenance calculus

Data Eng. Semantic integration and KG lifecycle management

Data Owner Data governance and data lineage

Context & Motivations – A Comprehensive View of KG-based Applications



Expectation Multi-domain Knowledge Graph (KG) for all stakeholders

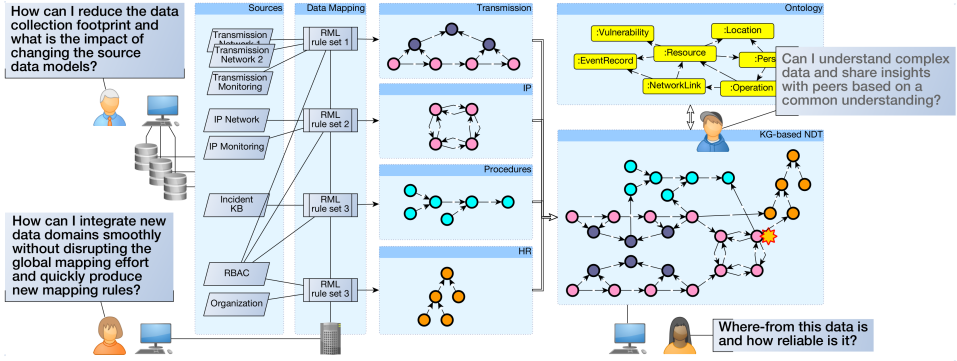
Pipeline Consistent stitching of domain-specific subgraphs during Knowledge Graph Construction (KGC)

Support Eng. Provenance calculus

Data Eng. Semantic integration and KG lifecycle management

Data Owner Data governance and data lineage

Context & Motivations – A Comprehensive View of KG-based Applications



Expectation Multi-domain Knowledge Graph (KG) for all stakeholders

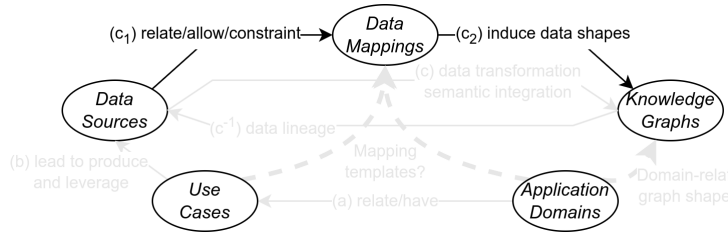
Pipeline Consistent stitching of domain-specific subgraphs during Knowledge Graph Construction (KGC)

Support Eng. Provenance calculus

Data Eng. Semantic integration and KG lifecycle management

Data Owner Data governance and data lineage

Problem Statement – Shifting Models Left



Semantic Gap Issue How to maintain semantic continuity from how data is structured (schemas) to its actual meaning in context (semantics)?

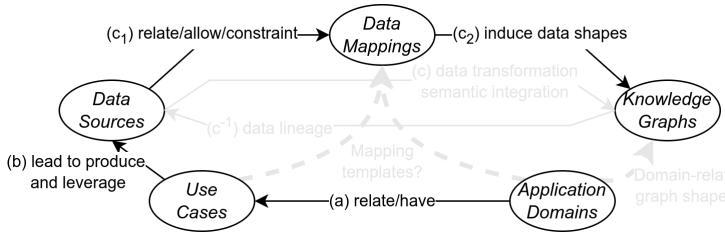
Multi-Domain KGC Pipelines Could there be a standard way to express transformations made to the data at the knowledge graph level (a.k.a. patching queries) and define how these relate to the upstream data mapping rules (e.g., RML-based)?

↪ KGs could be viewed as Data Products

i.e., a craftsmanship combination of data assets for a specific purpose.

↪ KGC processes and tools embody an (implicit) logic that we can rely on.

Problem Statement – Shifting Models Left



Semantic Gap Issue How to maintain semantic continuity from how data is structured (schemas) to its actual meaning in context (semantics)?

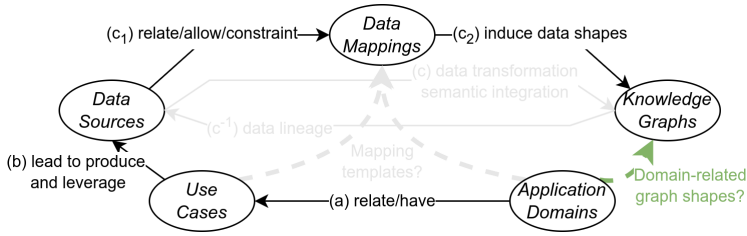
Multi-Domain KGC Pipelines Could there be a standard way to express transformations made to the data at the knowledge graph level (a.k.a. patching queries) and define how these relate to the upstream data mapping rules (e.g., RML-based)?

↪ KGs could be viewed as Data Products

i.e., a craftsmanship combination of data assets for a specific purpose.

↪ KGC processes and tools embody an (implicit) logic that we can rely on.

Problem Statement – Shifting Models Left



Semantic Gap Issue How to maintain semantic continuity from how data is structured (schemas) to its actual meaning in context (semantics)?

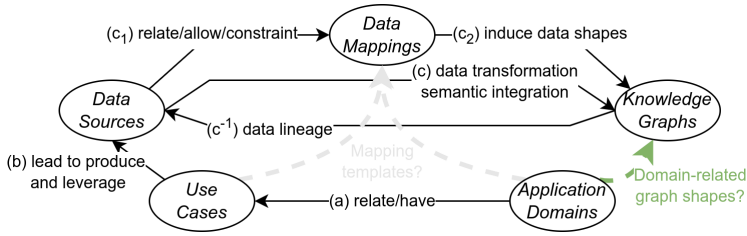
Multi-Domain KGC Pipelines Could there be a standard way to express transformations made to the data at the knowledge graph level (a.k.a. patching queries) and define how these relate to the upstream data mapping rules (e.g., RML-based)?

↪ KGs could be viewed as Data Products

i.e., a craftsmanship combination of data assets for a specific purpose.

↪ KGC processes and tools embody an (implicit) logic that we can rely on.

Problem Statement – Shifting Models Left



Semantic Gap Issue How to maintain semantic continuity from how data is structured (schemas) to its actual meaning in context (semantics)?

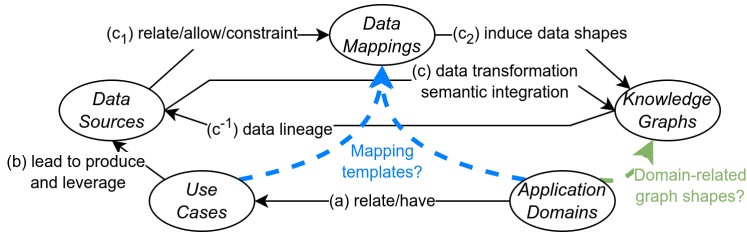
Multi-Domain KGC Pipelines Could there be a standard way to express transformations made to the data at the knowledge graph level (a.k.a. patching queries) and define how these relate to the upstream data mapping rules (e.g., RML-based)?

↪ KGs could be viewed as Data Products

i.e., a craftsmanship combination of data assets for a specific purpose.

↪ KGC processes and tools embody an (implicit) logic that we can rely on.

Problem Statement – Shifting Models Left



Semantic Gap Issue How to maintain semantic continuity from how data is structured (schemas) to its actual meaning in context (semantics)?

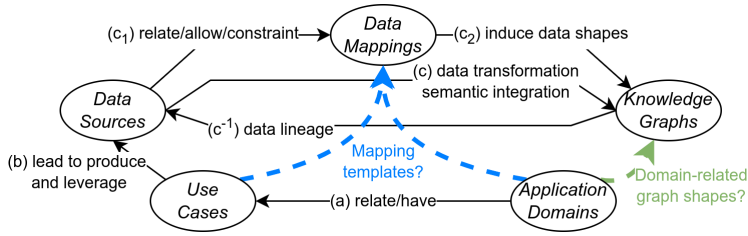
Multi-Domain KGC Pipelines Could there be a standard way to express transformations made to the data at the knowledge graph level (a.k.a. patching queries) and define how these relate to the upstream data mapping rules (e.g., RML-based)?

↪ KGs could be viewed as Data Products

i.e., a craftsmanship combination of data assets for a specific purpose.

↪ KGC processes and tools embody an (implicit) logic that we can rely on.

Problem Statement – Shifting Models Left



Semantic Gap Issue How to maintain semantic continuity from how data is structured (schemas) to its actual meaning in context (semantics)?

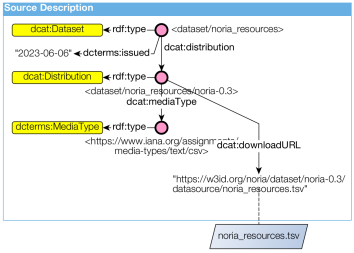
Multi-Domain KGC Pipelines Could there be a standard way to express transformations made to the data at the knowledge graph level (a.k.a. patching queries) and define how these relate to the upstream data mapping rules (e.g., RML-based)?

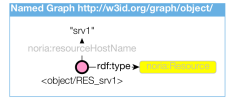
↪ KGs could be viewed as Data Products

i.e., a craftsmanship combination of data assets for a specific purpose.

↪ KGC processes and tools embody an (implicit) logic that we can rely on.

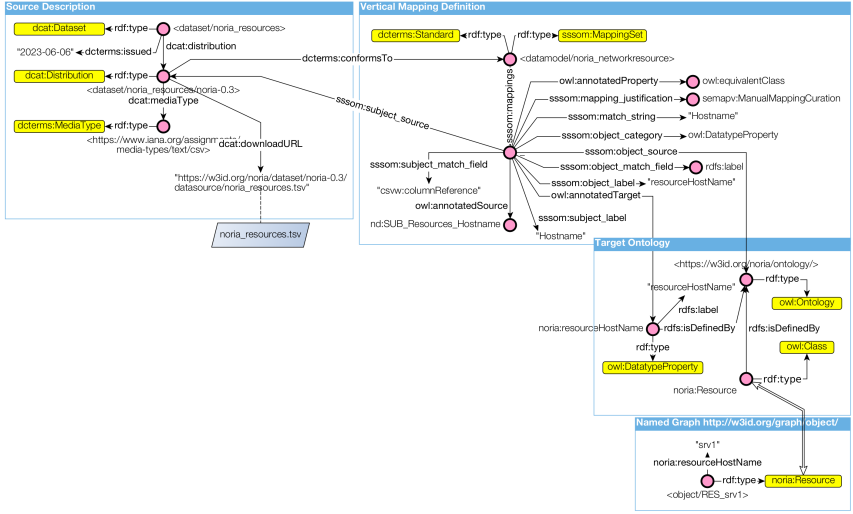
Could We Address This With Existing KGC-related Vocabularies?





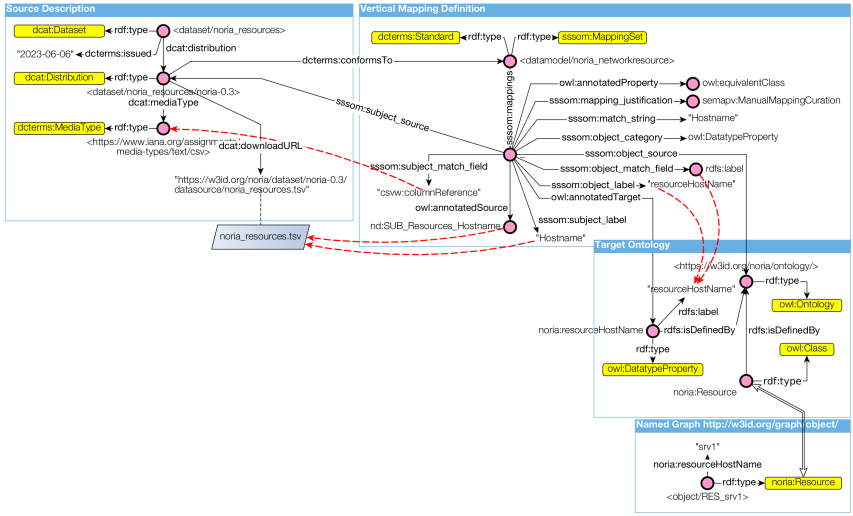
- ✗ Missing definitions across vocabularies to see lineage/provenance as graph traversal problem.
- ✗ Lack of standard and unified ways to describe data insertion and publishing.

Could We Address This With Existing KGC-related Vocabularies?



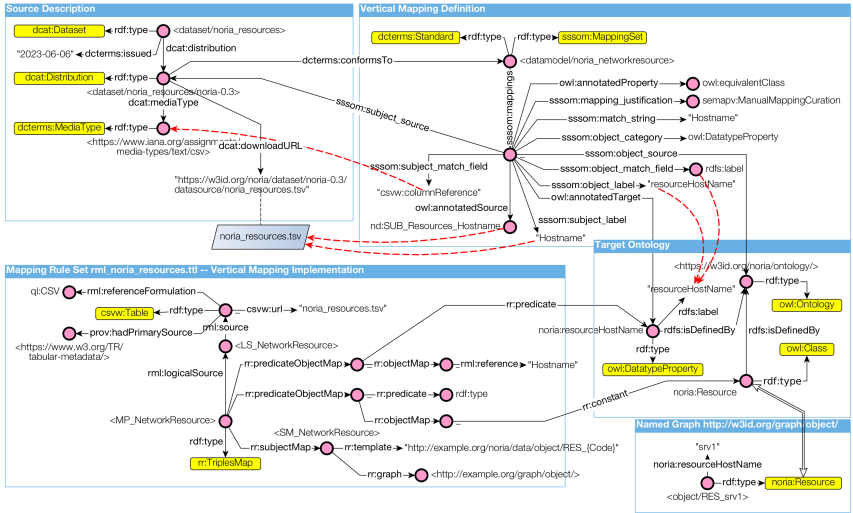
- ✗ Missing definitions across vocabularies to see lineage/provenance as graph traversal problem.
- ✗ Lack of standard and unified ways to describe data insertion and patching.

Could We Address This With Existing KGC-related Vocabularies?



- ✗ Missing definitions across vocabularies to see lineage/provenance as graph traversal problem.
- ✗ Lack of standard and unified ways to describe data insertion and patching.

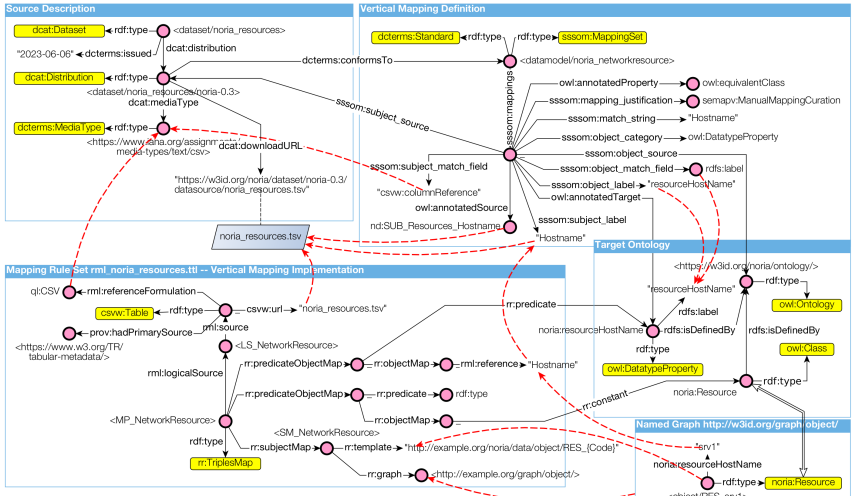
Could We Address This With Existing KGC-related Vocabularies?



✗ Missing definitions across vocabularies to see lineage/provenance as graph traversal problem.

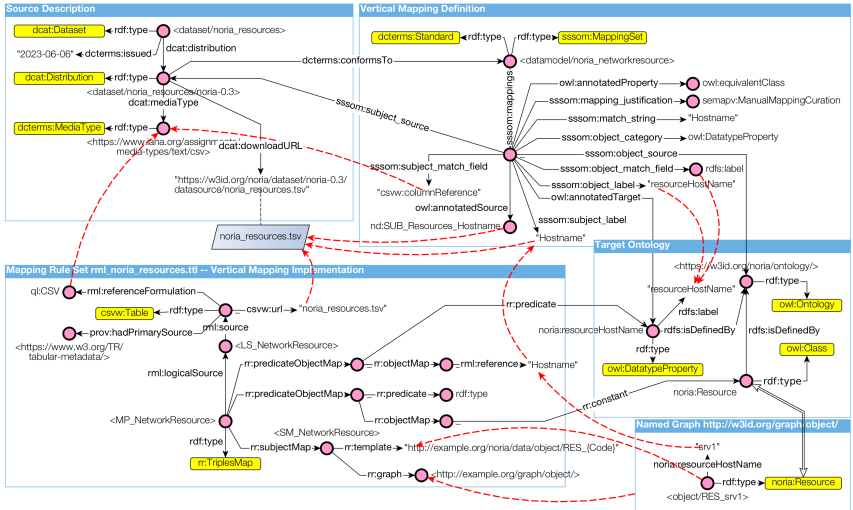
✗ Lack of standard and unified ways to describe data insertion and patching.

Could We Address This With Existing KGC-related Vocabularies?



- ✗ Missing definitions across vocabularies to see lineage/provenance as graph traversal problem.
- ✗ Lack of standard and unified ways to describe data insertion and patching.

Could We Address This With Existing KGC-related Vocabularies?



- ✗ Missing definitions across vocabularies to see lineage/provenance as graph traversal problem.
- ✗ Lack of standard and unified ways to describe data insertion and patching.

Could We Address This With Existing KGC-related Abstract Frameworks?

Methodology

- LOT4KG (Pernisch'25) Typical steps for knowledge engineering and KG lifecycle management.
- FAIR principles (Serra'25) Metadata infrastructure for traceability artifacts.

Category theory

- Multi-Model Data Transformation (Uotila'21) Data transformation as a lifting problem.
  $I_1 : C_1 \to \mathrm{Set}, I_2 : C_2 \to \mathrm{Set}, I_1 \to I_2 = (\mathrm{Rift}_{I_2} I_1 : C_1 \to C_2, \epsilon : I_2 \circ \mathrm{Rift}_{I_2} I_1 \Rightarrow I_1)$

Functional

- Knowledge Graph Ecosystems (KGE) (Geisler'25) KGs as a state machine.
  $\mathrm{KGE}_{t+1} = \sigma(\mathrm{lcs}, \mathrm{KGE}_t)$, with lcs a lifecycle step definition.

Algebraic

- Mapping Operators (Oo'25) Optimizing the execution plan of a KGC engine.
  $\Lambda_{\mathrm{output}} = \mathrm{Target}(\mathrm{Serialize}(\mathrm{Join}(\mathrm{Extend}(\mathrm{Project}(\mathrm{Source})))))$
- Provenance Semirings (Green'07) Generalization of the positive relational algebra to tagged-tuples relations.
  $t : U \to \mathbb{D}, R : \text{U-Tup} \to K, \text{K-relation} : \mathrm{supp}(R) = \{t | R(t) \neq 0\}$ is finite.
- Data Integration as Queries (Lenzerini'02) Source compatibility w.r.t. target data model.
  $g^B \supseteq \phi_S^C(\text{sound source}) : \forall \bar{x}, \phi_S(\bar{x}) \to g(\bar{x})$

✗ Lack of concrete and actionable tools (in some cases).
✗ Lack of a cohesive combination of these frameworks (each have different generalization levels).
↪ We need a (flexible) framework for understanding ...

Could We Address This With Existing KGC-related Abstract Frameworks?

Methodology

- LOT4KG (Pernisch'25) Typical steps for knowledge engineering and KG lifecycle management.
- FAIR principles (Serra'25) Metadata infrastructure for traceability artifacts.

Category theory

- Multi-Model Data Transformation (Uotila'21) Data transformation as a lifting problem.
  $I_1 : C_1 \to \text{Set}, I_2 : C_2 \to \text{Set}, I_1 \to I_2 = (\text{Rift}_{I_2} I_1 : C_1 \to C_2, \epsilon : I_2 \circ \text{Rift}_{I_2} I_1 \Rightarrow I_1)$

Functional

- Knowledge Graph Ecosystems (KGE) (Geisler'25) KGs as a state machine.
  $KGE_{t+1} = \sigma(lcs, KGE_t)$, with lcs a lifecycle step definition.

Algebraic

- Mapping Operators (Oo'25) Optimizing the execution plan of a KGC engine.
  $\Lambda_{output} = \text{Target}(\text{Serialize}(\text{Join}(\text{Extend}(\text{Project}(\text{Source})))))$
- Provenance Semirings (Green'07) Generalization of the positive relational algebra to tagged-tuples relations.
  $t : U \to \mathbb{D}, R : U\text{-Tup} \to K, K\text{-relation} : \text{supp}(R) = \{t | R(t) \neq 0\}$ is finite.
- Data Integration as Queries (Lenzerini'02) Source compatibility w.r.t. target data model.
  $g^B \supseteq \phi_S^C(\text{sound source}) : \forall \vec{x}, \phi_S(\vec{x}) \to g(\vec{x})$

✗ Lack of concrete and actionable tools (in some cases).
✗ Lack of a cohesive combination of these frameworks (each have different generalization levels).
↪ We need a (flexible) framework for understanding ...

Could We Address This With Existing KGC-related Abstract Frameworks?

Methodology

- LOT4KG (Pernisch'25) Typical steps for knowledge engineering and KG lifecycle management.
- FAIR principles (Serra'25) Metadata infrastructure for traceability artifacts.

Category theory

- Multi-Model Data Transformation (Uotila'21) Data transformation as a lifting problem.
  $I_1 : C_1 \to \text{Set}, I_2 : C_2 \to \text{Set}, I_1 \to I_2 = (\text{Rift}_{I_2} I_1 : C_1 \to C_2, \epsilon : I_2 \circ \text{Rift}_{I_2} I_1 \Rightarrow I_1)$

Functional

- Knowledge Graph Ecosystems (KGE) (Geisler'25) KGs as a state machine.
  $\text{KGE}_{t+1} = \sigma(\text{lcs}, \text{KGE}_t)$, with lcs a lifecycle step definition.

Algebraic

- Mapping Operators (Oo'25) Optimizing the execution plan of a KGC engine.
  $\Lambda_{\text{output}} = \text{Target}(\text{Serialize}(\text{Join}(\text{Extend}(\text{Project}(\text{Source})))))$
- Provenance Semirings (Green'07) Generalization of the positive relational algebra to tagged-tuples relations.
  $t : U \to \mathbb{D}, R : \text{U-Tup} \to K, \text{K-relation} : \text{supp}(R) = \{t | R(t) \neq 0\}$ is finite.
- Data Integration as Queries (Lenzerini'02) Source compatibility w.r.t. target data model.
  $g^B \supseteq \phi_S^C(\text{sound source}) : \forall \vec{x}, \phi_S(\vec{x}) \to g(\vec{x})$

✗ Lack of concrete and actionable tools (in some cases).
✗ Lack of a cohesive combination of these frameworks (each have different generalization levels).
↪ We need a (flexible) framework for understanding ...

Could We Address This With Existing KGC-related Abstract Frameworks?

Methodology

- LOT4KG (Pernisch'25) Typical steps for knowledge engineering and KG lifecycle management.
- FAIR principles (Serra'25) Metadata infrastructure for traceability artifacts.

Category theory

- Multi-Model Data Transformation (Uotila'21) Data transformation as a lifting problem.
  $I_1 : C_1 \to \text{Set}, I_2 : C_2 \to \text{Set}, I_1 \to I_2 = (\text{Rift}_{I_2} I_1 : C_1 \to C_2, \epsilon : I_2 \circ \text{Rift}_{I_2} I_1 \Rightarrow I_1)$

Functional

- Knowledge Graph Ecosystems (KGE) (Geisler'25) KGs as a state machine.
  $\text{KGE}_{t+1} = \sigma(\text{lcs}, \text{KGE}_t)$, with lcs a lifecycle step definition.

Algebraic

- Mapping Operators (Oo'25) Optimizing the execution plan of a KGC engine.
  $\Lambda_{\text{output}} = \text{Target}(\text{Serialize}(\text{Join}(\text{Extend}(\text{Project}(\text{Source})))))$
- Provenance Semirings (Green'07) Generalization of the positive relational algebra to tagged-tuples relations.
  $t : U \to \mathbb{D}, R : \text{U-Tup} \to K, \text{K-relation} : \text{supp}(R) = \{t | R(t) \neq 0\}$ is finite.
- Data Integration as Queries (Lenzerini'02) Source compatibility w.r.t. target data model.
  $g^B \supseteq \phi_S^C(\text{sound source}) : \forall \vec{x}, \phi_S(\vec{x}) \to g(\vec{x})$

✗ Lack of concrete and actionable tools (in some cases).
✗ Lack of a cohesive combination of these frameworks (each have different generalization levels).
↪ We need a (flexible) framework for understanding ...

Could We Address This With Existing KGC-related Abstract Frameworks?

Methodology

- LOT4KG (Pernisch'25) Typical steps for knowledge engineering and KG lifecycle management.
- FAIR principles (Serra'25) Metadata infrastructure for traceability artifacts.

Category theory

- Multi-Model Data Transformation (Uotila'21) Data transformation as a lifting problem.
  $\mathsf{I}_1 : \mathsf{C}_1 \to \mathsf{Set}, \mathsf{I}_2 : \mathsf{C}_2 \to \mathsf{Set}, \mathsf{I}_1 \to \mathsf{I}_2 = (\mathsf{Rift}_{\mathsf{I}_2}\mathsf{I}_1 : \mathsf{C}_1 \to \mathsf{C}_2, \epsilon : \mathsf{I}_2 \circ \mathsf{Rift}_{\mathsf{I}_2}\mathsf{I}_1 \Rightarrow \mathsf{I}_1)$

Functional

- Knowledge Graph Ecosystems (KGE) (Geisler'25) KGs as a state machine.
  $\mathsf{KGE}_{t+1} = \sigma(\mathsf{lcs}, \mathsf{KGE}_t)$, with lcs a lifecycle step definition.

Algebraic

- Mapping Operators (Oo'25) Optimizing the execution plan of a KGC engine.
  $\Lambda_{\text{output}} = \mathsf{Target}(\mathsf{Serialize}(\mathsf{Join}(\mathsf{Extend}(\mathsf{Project}(\mathsf{Source})))))$
- Provenance Semirings (Green'07) Generalization of the positive relational algebra to tagged-tuples relations.
  $\mathsf{t} : \mathsf{U} \to \mathbb{D}, \mathsf{R} : \mathsf{U\text{-}Tup} \to \mathsf{K}, \mathsf{K\text{-}relation} : \mathsf{supp}(\mathsf{R}) = \{\mathsf{t} | \mathsf{R}(\mathsf{t}) \neq 0\}$ is finite.
- Data Integration as Queries (Lenzerini'02) Source compatibility w.r.t. target data model.
  $\mathsf{g}^{\mathsf{B}} \supseteq \phi_{\mathsf{S}}^{\mathsf{C}}(\text{sound source}) : \forall \vec{\mathsf{x}}, \phi_{\mathsf{S}}(\vec{\mathsf{x}}) \to \mathsf{g}(\vec{\mathsf{x}})$

✗ Lack of concrete and actionable tools (in some cases).

✗ Lack of a cohesive combination of these frameworks (each have different generalization levels).

↪ We need a (flexible) framework for understanding ...

Vision – End-to-end Traceability as a Building Block

Hyp. 1 – End-to-End Traceability

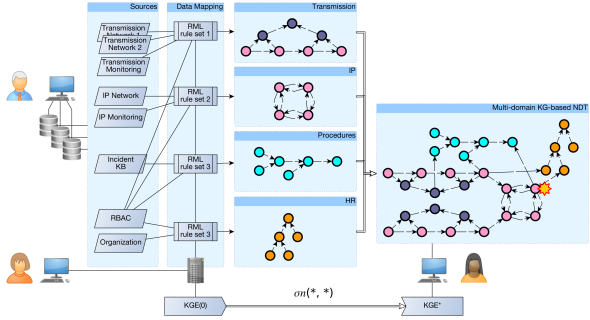
End-to-end traceability should be a fundamental requirement for designing a KGC pipeline, if not considering a KG-based application.

Hyp. 2 – End-to-End Semantic

End-to-end traceability should encompass a semantically analyzable description of source data and a formal description of how end users leverage the KG data, including application domain description and implicit/explicit requirements

Hyp. 3 – Two Axes Mapping

The same concepts and properties of an ontology are involved in an orthogonal yet complementary manner during the KGC process, depending on whether it involves data transformation or semantic integration.



$$S = \{mapping, patching, merging, pruning\}$$

$$lcs = \left(S, \langle P, Ro, C, Re, N\rangle\right)$$

$$KGE_{t+1} = \sigma(lcs, KGE_t)$$

$$\text{lineage} \equiv \sigma^n(*,*) \Leftrightarrow \forall t, \left[KGE_0, KGE_t\left[\bigcap\right]KGE_{t+1}, KGE^*\right] \neq \emptyset$$

Vision – End-to-end Traceability as a Building Block

Hyp. 1 – End-to-End Traceability

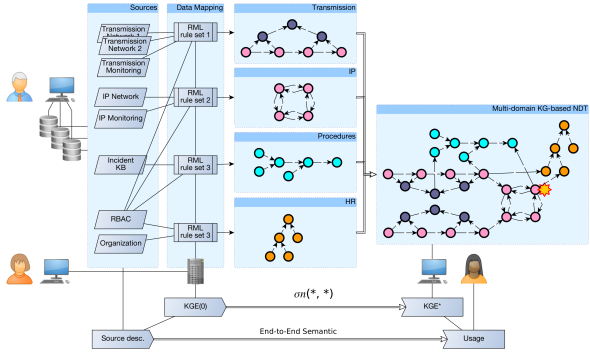
End-to-end traceability should be a fundamental requirement for designing a KGC pipeline, if not considering a KG-based application.

Hyp. 2 – End-to-End Semantic

End-to-end traceability should encompass a semantically analyzable description of source data and a formal description of how end users leverage the KG data, including application domain description and implicit/explicit requirements

Hyp. 3 – Two Axes Mapping

The same concepts and properties of an ontology are involved in an orthogonal yet complementary manner during the KGC process, depending on whether it involves data transformation or semantic integration.



Vision – End-to-end Traceability as a Building Block

Hyp. 1 – End-to-End Traceability

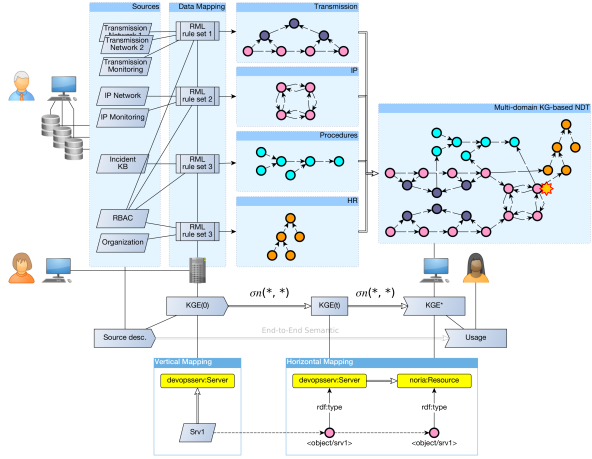
End-to-end traceability should be a fundamental requirement for designing a KGC pipeline, if not considering a KG-based application.

Hyp. 2 – End-to-End Semantic

End-to-end traceability should encompass a semantically analyzable description of source data and a formal description of how end users leverage the KG data, including application domain description and implicit/explicit requirements

Hyp. 3 – Two Axes Mapping

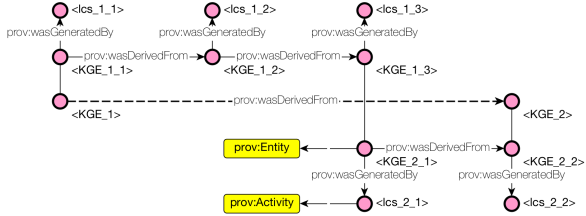
The same concepts and properties of an ontology are involved in an orthogonal yet complementary manner during the KGC process, depending on whether it involves data transformation or semantic integration.



Vision – Sketching a Research Agenda

Mathematical framework

- Algebraic analysis of data models (Green'07).
- Chaining or parallelization of data integration traces →



Formal proof of systems

- Seek a proof (Sergey'14) of the Hyp. 1 – 3 properties for concrete KGC pipeline / KG-based app instances →
- Faithful translation of the RML rules and SPARQL queries into the proof system.

Relationships with non-KG or legacy technologies

- Connecting `datacontract-cli` (Bitol'25) with the DPROD (Gioia'25) and LinkML-SSSOM (Matentzoglu'22) ecosystems.

↪ What do you think?

Vision – Sketching a Research Agenda

Mathematical framework

- Algebraic analysis of data models (Green'07).
- Chaining or parallelization of data integration traces →

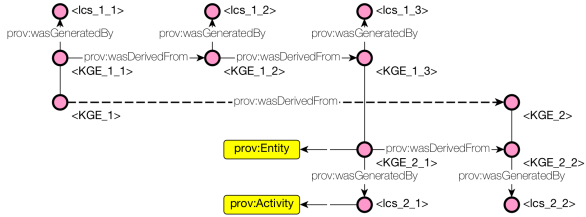
Formal proof of systems

- Seek a proof (Sergey'14) of the Hyp. 1 – 3 properties for concrete KGC pipeline / KG-based app instances →
- Faithful translation of the RML rules and SPARQL queries into the proof system.

Relationships with non-KG or legacy technologies

- Connecting `datacontract-cli` (Bitol'25) with the DPROD (Gioia'25) and LinkML-SSSOM (Matentzoglu'22) ecosystems.

→ What do you think?



$$\forall m_{[\text{MappingRule}]}, \forall src_{[\text{Source}]}, \forall t_{[\text{RDF triple}]} : \\ map(m, src) \Leftrightarrow (src, t) \in \mathcal{G}_{\text{prov}}$$

Completness proof → For any RML rule and any source data, demonstrate that if a triplet is generated, then there is a trace in the provenance graph. ←

Vision – Sketching a Research Agenda

Mathematical framework

- Algebraic analysis of data models (Green'07).
- Chaining or parallelization of data integration traces →

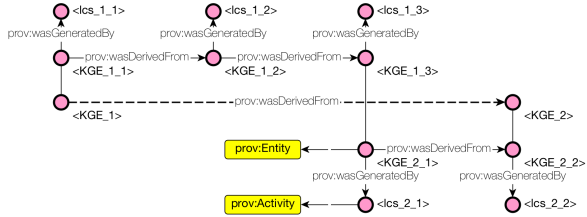
Formal proof of systems

- Seek a proof (Sergey'14) of the Hyp. 1 – 3 properties for concrete KGC pipeline / KG-based app instances →
- Faithful translation of the RML rules and SPARQL queries into the proof system.

Relationships with non-KG or legacy technologies

- Connecting `datacontract-cli` (Bitol'25) with the DPROD (Gioia'25) and LinkML-SSSOM (Matentzoglu'22) ecosystems.

→ What do you think?



$$\forall m_{[\text{MappingRule}]}, \forall src_{[\text{Source}]}, \forall t_{[\text{RDF triple}]} : \\ map(m, src) \Leftrightarrow (src, t) \in \mathcal{G}_{\text{prov}}$$

Completness proof → For any RML rule and any source data, demonstrate that if a triplet is generated, then there is a trace in the provenance graph. ←

Vision – Sketching a Research Agenda

Mathematical framework

- Algebraic analysis of data models (Green'07).
- Chaining or parallelization of data integration traces →

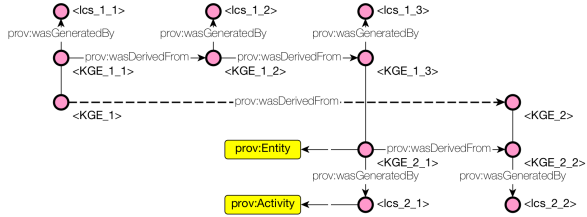
Formal proof of systems

- Seek a proof (Sergey'14) of the Hyp. 1 – 3 properties for concrete KGC pipeline / KG-based app instances →
- Faithful translation of the RML rules and SPARQL queries into the proof system.

Relationships with non-KG or legacy technologies

- Connecting `datacontract-cli` (Bitol'25) with the DPROD (Gioia'25) and LinkML-SSSOM (Matentzoglu'22) ecosystems.

↪ What do you think?



$$\forall m_{[\text{MappingRule}]}, \forall src_{[\text{Source}]}, \forall t_{[\text{RDF triple}]} : \\ map(m, src) \Leftrightarrow (src, t) \in \mathcal{G}_{\text{prov}}$$

Completness proof → For any RML rule and any source data, demonstrate that if a triplet is generated, then there is a trace in the provenance graph. ←

Summary & Future Work

Problem Facilitating the design and management of multi-domain KGC pipelines.

Intuitions Addressing the semantic gap issue by viewing KGs as data products and leveraging the implicit logic in KGC processes and tools.

Analysis Missing KGC-related vocabulary definitions, standard ways to describe data insertion and patching, and tooling/combination of existing abstract frameworks.

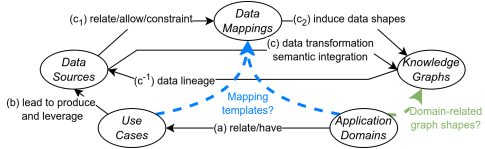
Vision Framework for understanding/analysis based on End-to-End Traceability, End-to-End Semantic, and Two Axes Mapping hypotheses.

Next Discussions and experiments on three complementary approaches – connecting/deepening mathematical frameworks, formal proofs of multi-domain KG-based systems, and integrating non-KG or legacy technologies.

Paper – KGCW @ ESWC 2026

Lionel Tailhardat, Joanna Balcerzak, Arij Elmajed, Pano Maria, Gabriel Fomi-NGameni.

Shifting Models Left: End-to-End Traceability as a Foundation for Knowledge Graphs as Data Products.



Bibliographical References

(Bitol'25) Bitol, "Open Data Contract Standard (ODCS)", LF AI & Data Foundation, 2025.

(Geisler'25) S. Geisler, et al. "From Genesis to Maturity: Managing Knowledge Graph Ecosystems Through Life Cycles", VLDB, 2025.

(Gioia'25) A. Gioia, et al. "Data Product Ontology (DPROD)", 2025.

(Green'07) T. J. Green, et al. "Provenance Semirings", ACM Symposium on Principles of Database Systems (PODS), 2007.

(Lenzerini'02) M. Lenzerini, "Data Integration: A Theoretical Perspective", ACM Symposium on Principles of Database Systems (PODS), 2002.

(Matentzoglu'22) N. Matentzoglu, et al. "A Simple Standard for Sharing Ontological Mappings (SSSOM)", 2022.

(Oo'25) S. M. Oo, et al. "Algebraic Mapping Operators for Knowledge Graph Generation", Semantic Web Journal, 2025.

(Pernisch'25) R. Pernisch, et al. "LOT4KG: A Joint Methodology for the Ontology and Knowledge Graph Lifecycle", 2025.

(Poveda'19) M. Poveda-Villalón, et al. "Linked Open Terms (LOT) Methodology", 2019.

(Sergey'14) I. Sergey, "Programs and Proofs: Mechanizing Mathematics with Dependent Types", 2014.

(Serra'25) A. Serra. "FAIR Principles for Knowledge Graphs in SSbD", Venice Training School, 2025.

(Uotila'21) V. Uotila, J. Lu. "A Formal Category Theoretical Framework for Multi-Model Data Transformations", Springer, 2021.