Leveraging Meta-Modelling Language for Ontology Structuring and Validation

Zekeri Adams[1,*], Martin Homola[1], Ján Kľuka[1] and Vojtěch Svátek[2]

[1]Comenius University Bratislava, Mlynská dolina, 842 48 Bratislava, Slovakia
[2]Prague University of Business and Economics, Nám. W. Churchilla 1938/4, 130 67 Praha 3, Czech Republic

Abstract

Knowledge representation (KR) is foundational to AI, providing the structures that enable systems to reason and interpret complex domains, powering applications like expert systems and intelligent agents. Yet, traditional KR methods struggle with the growing complexity in fields like biomedicine, finance, and cybersecurity, where interdependencies and rapid change challenge accurate representation. Meta-modeling offers a flexible solution with reusable frameworks that improve adaptability. Ontological meta-modeling, in particular, provides hierarchical structures for more abstract, adaptable models. However, current frameworks often lack standardized formal structures and strong theoretical foundations, limiting practical adoption. Issues in validation and consistency further constrain scalability in real-world settings.

Our research addresses these gaps by formalizing the PURO meta-modeling language, focusing on theoretical rigor and verification techniques for consistency. We also present use cases in Wikidata to showcase enhanced expressiveness and reliability in representing complex, dynamic knowledge domains.

Keywords

Meta-modeling, ontology, PURO, MLT

1. Introduction

Meta-modeling is a methodology that represents the structure and relationships of models at a higher level of abstraction, enabling the creation of frameworks applicable across various domains. In ontologies, it provides an abstract schema that defines the rules and relationships governing elements, ensuring a standardized and reusable approach to modeling knowledge. This is essential for structuring complex domains effectively [1, 2] . It defines how classes, properties, and individuals interact within a domain, specifying relationships and constraints to maintain consistency, scalability, and adaptability. This is particularly important in fields where entity relationships are constantly evolving [3, 4].

A key benefit of meta-modeling is its ability to enhance ontology coherence. By introducing well-defined meta-level schemas, it avoids redundancy and conflicting definitions, ensuring a coherent representation of knowledge. It also systematically defines class interrelations, such as inheritance and composition, reflecting real-world structures more accurately [5]. This coherence is critical for maintaining the reliability of the represented knowledge.

Although numerous meta-modeling languages have been proposed, many lack a formal logical foundation. One work in this direction is the multi-level modelling theory, MLT [6]. This theory addresses the modeling of domains that involve multiple classification levels. It is designed to formally characterize the nature of these classification levels and define the structural relationships both within entities of the same level and between entities at different levels. MLT is grounded in the concepts of types and individuals. In the theory, types are viewed as predicative entities (e.g., "Person", "Organization",

EKAW 2024: EKAW 2024 Workshops, Tutorials, Posters and Demos, 24th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2024), November 26-28, 2024, Amsterdam, The Netherlands.
*Corresponding author.
†These authors contributed equally.
zekeri.adams@fmph.uniba.sk (Z. Adams); homola@fmph.uniba.sk (M. Homola); kluka@fmph.uniba.sk (J. Kľuka); svatek@vse.cz (V. Svátek)
0009-0006-3413-2409 (Z. Adams); 0000-0001-6384-9771 (M. Homola); 0000-0002-3406-3574 (J. Kľuka); 0000-0002-2256-2982 (V. Svátek)

"Product") that can apply to multiple entities, including other types. When a type t applies to an entity e, we say that e is an instance of t. Conversely, individuals are entities that cannot have instances, such as “John”, “an apple”, or “my cellphone”. MLT also accounts for the stratification of types into various orders, such as individual, first-order types, second-order types, and so on. This framework is axiomatized in first-order logic, where both individuals and types are quantified, forming the entities in the theory. A fundamental aspect of MLT is the instance of relation, which connects entities to the types they instantiate [7, 6].

Fonseca et.al. [6], applied the theory to the well-known knowledge base, Wikidata, which structures items using RDF (subject-predicate-object) format. Wikidata’s item hierarchy is inherently multi-leveled, utilizing the instance of (P31) and subclass of (P279) relations to stratify items into different classification levels. The classification of items in Wikidata includes individual entities, fixed-order classes, and variable-order classes, classes with orders ranging from the first to higher levels. However, the study [6] revealed that despite this classification system, multi-level anti-patterns (they involve classification of entities as individuals or classes of some order conflicting with instantiation and specialization relationships among them) persist. These multi-level anti-patterns underscoring the challenges of maintaining a coherent multi-level classification system. Although, Fonseca et.al. [6], identified anti-patterns in Wikidata, they were unable to provide a visualization of these patterns beyond the syntactic errors revealed by the logical axioms. Additionally, there was no validation of the logical axioms or the detected syntactic errors within Wikidata.

Our study focuses on the meta-modeling language PURO [8], and its formalization in first-order logic. PURO provides a robust framework for structuring and understanding ontologies, combining logical formalization with a graphical tool, the PURO Modeler, for ontology visualization. In this work, we validate the satisfiability of the PURO formalization using the widely recognized theorem prover, Vampire [9]. Furthermore, we explore the practical application of the language with its graphical tool within Wikidata, a prominent knowledge base, to analyze and identify potential structural issues and the validation of its structure.

2. PURO Language

PURO [8] is a graphical language designed to simplify the creation of OWL ontologies [10] and OntoUML models [11]. It provides a foundational set of modeling elements, such as objects and their associated types, relationships, and quantitatively valued attributes. One of the distinctive features of PURO is its support for multi-level types, allowing for the hierarchical organization of types across different abstraction levels, which enhances the expressiveness of ontological models. Additionally, PURO supports relations of arbitrary arity, enabling the modeling of more complex relationships between entities that involve multiple participants.

Ontologies developed using the PURO language are constructed through the PURO Modeler [10], a web graphical tool that facilitates the visualization and structuring of ontological elements. The PURO Modeler allows users to interactively build and manipulate ontologies, with the primitives shown in Figure 1. This visual approach not only enhances the user experience but also ensures that the logical structure of the ontology remains clear and accessible, making it easier to model complex domains.

A PURO ontological background model establishes two fundamental distinctions: one between ontological particulars and universals, and another between objects, relationships, and valuations. These distinctions result in the creation of six basic PURO terms. They include:

- B-object represents specific objects, often real-world entities, which can be tangible (like people, animals, or objects) or intangible (such as concepts, events, or processes). This concept parallels the notion of individuals in traditional models.
- B-type represents a universal entity whose instances are entities belonging to the same concept or sharing a common property. B-types generalize the concept of classes and cover qualities (e.g., red color), although there may be slight ontological differences. B-types are multilevel (that is, B-type 1, B-type 2, B-type 3, and so on).

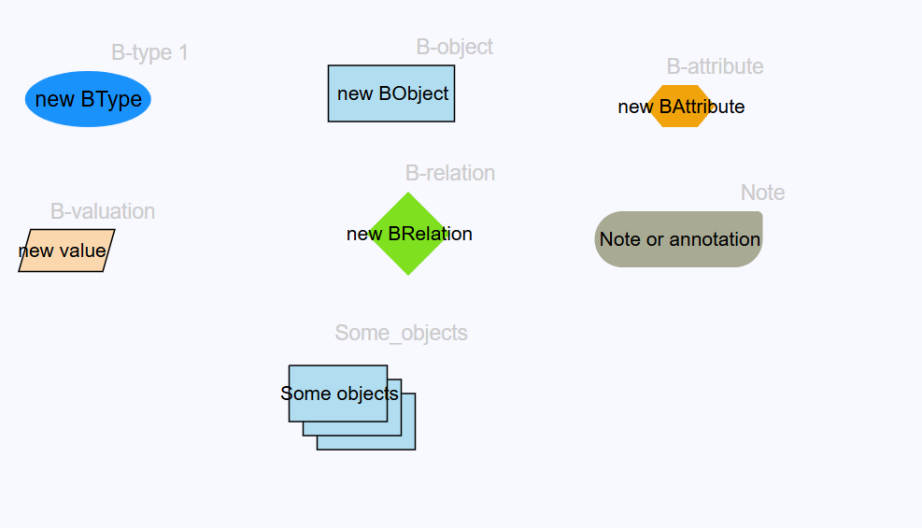

Figure 1: Puro primitives.

- B-relationship denotes a specific relationship between two or more entities, such as a musical work being composed by a composer, or an object being produced by a producer. It also includes relationships like an object being of a certain type or one type of goods being a special case of another.
- B-relation refers to a conceptual relation, serving as the universal counterpart of B-relationship. It resembles the more restricted ontological notion of an object property, with instances of B-relations being B-relationships.
- B-valuation involves the assignment of a quantitative data value to an entity, typically a B-object. Unlike B-relationships, B-valuations are always binary, with their second participants being quantitative data values. This aligns with the notion of data property assertion in regular ontologies.
- B-attribute represents a universal consisting of valuations of the same quantitative property. They are akin to the regular notion of data properties and attributes in ER schemas.

These primitives serve as the foundational predicates in the formalization of the PURO language within first-order logic. Also, the binary predicate instanceOf is employed to signify that a particular entity belongs to the extension of a universal, and the infix binary predicate : (colon) associates an entity with a PURO term by (1a).

$$x : t \rightarrow \mathsf{Entity}(x) \wedge \mathsf{PuroTerm}(t) \tag{1a}$$

$$\mathsf{instanceOf}(x, y) \rightarrow \mathsf{Entity}(x) \wedge \mathsf{Entity}(y) \tag{1b}$$

$$\exists x \, (\mathsf{instanceOf}(x, y) \rightarrow y : \mathsf{Universal}) \tag{1c}$$

Axiom (1b) specifies that the instanceOf property is used to establish a relationship between two entities. Axiom (1c) further clarifies that an entity x is considered an instance of an entity y that is associated with a Universal—meaning y can be a B-type, B-relation, or B-attribute. By using the instanceOf property, we create a clear and formal structure for representing hierarchical relationships within the ontology, ensuring that each entity is accurately classified according to its designated type. An important consequence of Axioms (1b) and (1c) is that particulars do not themselves have instances, reinforcing a structured hierarchy where instances relate only to Universals and not to other particulars:

$$x : \mathsf{Particular} \rightarrow \neg \exists y \, \mathsf{instanceOf}(x, y) \tag{2}$$

3. Formal verification of PURO axiomatization and use case

A central objective of our research is the exploration of PURO axiomatization, which has proven effective in supporting ontology development and structuring. Validating this framework is crucial to ensure both its logical consistency and the satisfiability of the models built upon it. Logical consistency ensures that the models are free from contradictions, while satisfiability guarantees that the models can be interpreted without violating any imposed constraints.

To facilitate this validation, we encoded the PURO axiomatization using the TPTP syntax [12]. For example, we encode (1a), (1b) and (1c) in TPTP syntax, below:

```
fof(colon_sorts, axiom, ![X,T]: (colon(X, Y) => (entity(X) & puroterm(T)))).
fof(instof_sorts, axiom, ![X,Y]: (instanceof(X, Y) => (entity(X) & entity(Y)))).
fof(has_instance_universal, axiom,
    ![Y]: (?[X]: instanceof(X, Y) => colon(Y, universal))).
```

We then used the first-order logic theorem prover Vampire to verify the satisfiability of PURO's axiomatization. The result obtained was satisfiable, validating the correctness of the axiomatization.

We apply the PURO modeler to real-world examples and validated their satisfiability using Vampire. Each tested example proved to be satisfiable, highlighting the robustness of the PURO framework in real-world ontology development. This validation supports the use of PURO for building ontologies that can handle intricate relationships while remaining logically sound.

To further assess the practical applicability of our approach, we extended our investigation to Wikidata, a large-scale collaborative knowledge base. Using SPARQL queries, we identified structural inconsistencies in Wikidata, highlighted in Figure 2, where the PURO modeler points out specific errors. The SPARQL Query shown below targets entities $?x$ and $?y$, both acting as instances of the first-order class Q104086571 (directly or through specializations) with $?x$ instantiating $?y$.

```
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
SELECT DISTINCT ?x ?y WHERE {
  ?x wdt:P31/(wdt:P279*) wd:Q104086571.
  ?y wdt:P31/(wdt:P279*) wd:Q104086571.
  ?x wdt:P31/(wdt:P279*) ?y.
}
```

A similar SPARQL query was executed for the second-order class Q24017414 in place of Q104086571, and in both cases, a significant number of inconsistencies were identified.

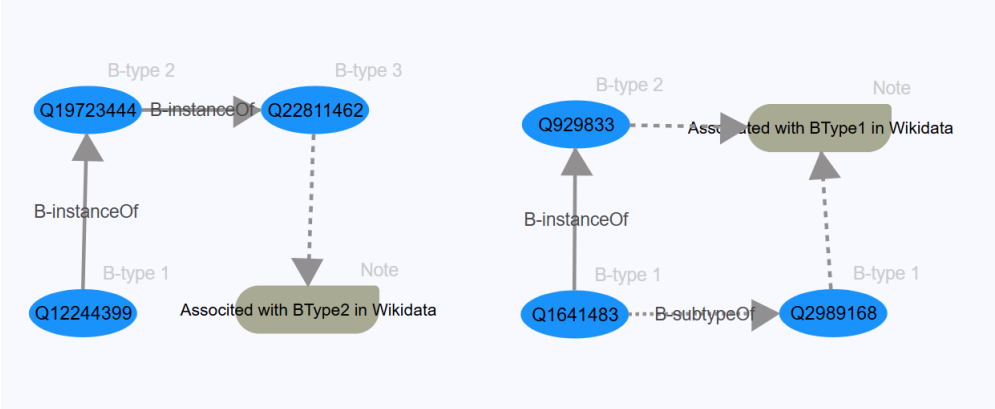

Figure 2: Incoherence in Wikidata

Two examples of the identified inconsistencies are depicted in Figure 2. On the left side of the figure, the entity *iPhone 15 Pro (Q12244399)*, which is categorized as a first-order class, is marked as an instance of *Cell phone model (Q19723444)*. However, *Cell phone model (Q19723444)*, itself an instance of *Type of*

manufactured goods (Q22811462), is classified as a second-order class (B-type 2). This violates constraint (3a), as *Type of manufactured goods (Q22811462)* is designated as a second-order class (B-type 2) in Wikidata, creating a conflict.

On the right side of Figure 2, the entity *Hypotrichosis (Q1641483)*, a subclass of *Hair disease (Q2989168)*, is incorrectly shown as an instance of *Rare disease (Q929833)*. This contradicts constraint (3b), as *Rare disease (Q929833)* is categorized as a B-type 1 class, leading to further inconsistency.

$$\neg\exists x(x\colon \text{B-type}_1 \wedge x\colon \text{B-type}_2) \tag{3a}$$

$$\neg\exists x(x\colon \text{B-type}_2 \wedge x\colon \text{B-type}_3) \tag{3b}$$

$$\forall x, y\ (\text{instanceOf}(x, y) \wedge y\colon \text{B-type}_2 \rightarrow x\colon \text{B-type}_1) \tag{3c}$$

$$\forall x, y\ (\text{instanceOf}(x, y) \wedge y\colon \text{B-type}_3 \rightarrow x\colon \text{B-type}_2) \tag{3d}$$

This analysis revealed the challenges of maintaining coherence in large, evolving datasets and highlighted the importance of using formal tools like PURO modeler to detect and address these issues, ensuring more reliable and consistent data structures.

We also extended our formal verification to the models depicted in Figure 2. The model on the left of Figure 2 was encoded using PURO axioms, as shown in (4a) and (4b) in TPTP syntax, and tested in Vampire. This resulted in a refutation, indicating inconsistencies or unsatisfiability within its logical structure. Similarly, the model on the right of Figure 2 was encoded using PURO axioms, also transformed into TPTP syntax. The same result, a refutation, was obtained, confirming the presence of inconsistencies.

$$\text{Q12244399}\colon \text{B-type}_1 \wedge \text{Q19723444}\colon \text{B-type}_2 \wedge \text{instanceOf}(\text{Q12244399}, \text{Q19723444}) \tag{4a}$$

$$\text{Q19723444}\colon \text{B-type}_2 \wedge \text{Q22811462}\colon \text{B-type}_2 \wedge \text{instanceOf}(\text{Q19723444}, \text{Q22811462}) \tag{4b}$$

These refutations reinforce the critical role of thorough logical analysis in ontology modeling, as even well-designed models can reveal hidden flaws when subjected to rigorous testing.

The inconsistencies identified in these models suggest areas that require further refinement, such as the item *Type of manufactured goods (Q22811462)* should be properly categorized as a third-order class (B-type 3) and the item *Rare disease (Q929833)* should be properly categorized as a second-order class (B-type 2). This is necessary to achieve logical soundness.

4. Conclusion

Our work highlights the crucial role of the PURO Modeler in structuring and visualizing ontologies, enhancing the clarity and organization of complex models. By offering a clear framework for understanding hierarchical relationships, the PURO Modeler improves the effectiveness of ontology design. Additionally, we emphasize the importance of formal verification tools like Vampire in detecting and resolving logical inconsistencies, ensuring the integrity of ontologies built with the PURO language.

Our experiments with PURO, which revealed some incoherences in Wikidata, stress the need for rigorous logical validation in knowledge representation. Addressing these issues enhances both the reliability and effectiveness of ontologies, making them better suited for applications in AI, knowledge management, and semantic web technologies.

The combination of PURO's flexible structure and Vampire's verification power provides a scalable and robust foundation for building consistent ontologies that can adapt to evolving domains. This synergy ensures that ontology-driven technologies remain reliable as they grow to support increasingly complex applications of knowledge representation.

Acknowledgments

This work was supported by the Slovak Republic under SRDA grants nos. APVV-23-0292 (DyMAX) and APVV-20-0353 (APEColT). Vojtěch Svátek has been partially supported by the EU's Horizon Europe grant no. 101058682 (Onto-DESIDE).

References

[1] A. Gangemi, Ontology design patterns for semantic web content, in: Proceedings of the International Semantic Web Conference (ISWC), 2005, pp. 262–276.

[2] N. Guarino, D. Oberle, S. Staab, What is an ontology?, in: Handbook on Ontologies, Springer, 2009.

[3] N. F. Noy, D. L. McGuinness, Ontology Development 101: A Guide to Creating Your First Ontology, Technical Report, Stanford Knowledge Systems Laboratory, 2001.

[4] S. Staab, R. Studer, Handbook on Ontologies, Springer, 2009.

[5] J. Euzenat, P. Shvaiko, Ontology Matching, Springer, 2013.

[6] C. M. Fonseca, J. P. Almeida, G. Guizzardi, V. A. Almeida, Multi-level conceptual modeling: Theory, language, and application, 2016.

[7] V. Almeida, J. P. Almeida, Toward a well-founded theory for multi-level conceptual modeling (2015).

[8] V. Svátek, M. Homola, J. Kľuka, M. Vacura, Mapping structural design patterns in OWL to ontological background models, in: Proceedings of the Seventh International Conference on Knowledge Capture (K-CAP '13), ACM, New York, NY, USA, 2013, pp. 117–120. doi:10.1145/2479832.2479847.

[9] L. Kovács, A. Voronkov, First-order theorem proving and vampire, in: N. Sharygina, H. Veith (Eds.), Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings, volume 8044 of *Lecture Notes in Computer Science*, Springer, 2013, pp. 1–35. URL: https://doi.org/10.1007/978-3-642-39799-8_1.

[10] M. Dudáš, V. Svátek, M. Vacura, O. Zamazal, Starting ontology development by visually modeling an example situation - a user study, in: VOILA@ISWC, volume 1704 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2016, pp. 114–119.

[11] M. Dudáš, T. Morkus, V. Svátek, T. P. Sales, G. Guizzardi, Kickstarting ontouml modeling from PURO instance-level examples, in: Procs. EKAW 2020 Posters and Demonstrations Session, volume 2751 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 36–40.

[12] G. Sutcliffe, The TPTP Problem Library and Associated Infrastructure. From CNF to TH0, TPTP v6.4.0, Journal of Automated Reasoning 59 (2017) 483–502.