Development of the Lem Knowledge Graph: Implementation of LRMoo Ontology

Jakub Gomułka[2,*], Luiz do Valle Miranda[1], Szymon Piotr Kukulak[2], Krzysztof Kutt[1] and Grzegorz J. Nalepa[1]

[1]*Department of Human-Centered Artificial Intelligence, Institute of Applied Computer Science, Faculty of Physics, Astronomy and Applied Computer Science, Jagiellonian University, prof. Stanisława Łojasiewicza 11, 30-348 Kraków, Poland*
[2]*Faculty of Humanities, AGH University of Kraków, Czarnowiejska 36, 30-054, Kraków, Poland.*

Abstract

This article explores the conceptual and methodological decisions involved in the construction of LKG.core: a knowledge graph mapping the publications of Stanisław Lem, developed as part of the broader Lem Knowledge Graph project. The ontology used to structure the graph is LRMoo—an extension of CIDOC CRM designed to represent collections of literary works—slightly adapted to suit the needs of the project. Although the final form of LKG.core will result from the integration of several digital sources, this paper focuses on the first stage of the process, which makes use of Victor Yaznevich's database containing Lem's non-fiction publications in Polish and other languages.

Keywords

LRMoo, Cultural Heritage, Stanisław Lem, Linked Data, Knowledge Graph, Victor Yaznevich

1. Introduction

The increasing need for semantic interoperability between domains of cultural heritage has driven the development of formal ontologies that can accurately model complex bibliographic relationships [1]. Among these, LRMoo[1] (Object-orientated Library Reference Model) represents a significant advancement, providing a comprehensive framework for representing bibliographic information. Developed as an extension of CIDOC-CRM, it facilitates the interoperability of library resources with other cultural heritage institutions, including museums and archives.

Given LRMoo's multilayered abstraction for bibliographic records, confusion during modelling is prone to arise without well-defined modelling principles and clear guidance from previous projects. Explicit guidelines and examples for the use of LRMoo can be found in the ontology's documentation, with detailed information for each declared class and property. Further guidance can be found by aligning with the modelling decisions taken within other projects, such as in [2, 3, 4]. Although both sources are helpful towards achieving consistent and interoperable implementations of LRMoo, the context and the extent of existing systems' descriptions are often limited or tailored to specific use cases, such as medieval manuscripts.

Our use case is the construction of the Lem Knowledge Graph (LKG), a knowledge base that provides a comprehensive catalogue of Stanisław Lem's (1921–2006) fiction and non-fiction works[2]. The LKG project is divided into three modules: LKG.core, LKG.letters, and LKG.content. The first module forms the part of the graph that will represent information about individual texts by Lem, their various editions,

Proceedings of the Joint Ontology Workshops (JOWO) - Episode XI: The Sicilian Summer under the Etna, co-located with the 15th International Conference on Formal Ontology in Information Systems (FOIS 2025), September 8-9, 2025, Catania, Italy
*Corresponding author.
jgomulka@agh.edu.pl (J. Gomułka); luiz.dovallemiranda@doctoral.uj.edu.pl (L. d. V. Miranda); skukulak@agh.edu.pl (S. P. Kukulak); krzysztof.kutt@uj.edu.pl (K. Kutt); gjn@gjn.re (G. J. Nalepa)
0000-0002-9100-0334 (J. Gomułka); 0000-0003-1838-5693 (L. d. V. Miranda); 0009-0009-2701-6567 (S. P. Kukulak); 0000-0001-5453-9763 (K. Kutt); 0000-0002-8182-4225 (G. J. Nalepa)

[1]https://cidoc-crm.org/lrmoo
[2]See [5] for a detailed discussion of the challenges in cataloguing Lem's literary legacy and its alignment with the LRMoo model

and internal structure. The second will serve as a representation of a closely annotated collection of Lem's correspondence. The third module will gather information on the themes of the individual texts. LKG.core provides the foundation for the other two modules and is being developed first.

The use of semantic web technologies—and more precisely LRMoo—allows us to faithfully represent the intricate publication history, translations, and adaptations of Lem's works, while maintaining alignment with broader cultural heritage frameworks. The LKG aims not only to serve as a centralised and semantically enriched catalogue of Lem's literary production but also to act as a demonstrator of best practices for applying LRMoo in the context of author-centric digital humanities projects.

This paper presents the conceptual and methodological decisions involved in constructing the LKG.core, including the selection of LRMoo classes and properties, the alignment with CIDOC CRM principles, and the eventual necessity of creation of new subclasses. By documenting our approach and its rationale, we contribute to the growing body of practical applications of LRMoo and offer a reusable framework for similar projects in literary and bibliographic knowledge representation.

2. Related works

LRMoo is a CIDOC-CRM extension designed "to capture and represent the underlying semantics of bibliographic information and to facilitate the integration, mediation, and interchange of bibliographic and museum information" ([6], p. 40). LRMoo, previously known as FRBRoo[3], is based on the harmonisation of the FRBR model, a conceptual model for bibliographic records developed by the International Federation of Library Associations, and the CIDOC-CRM cultural heritage event-centric ontology.

The FRBR model—and consequently LRMoo—is structured around four core classes that represent different levels of abstraction in the representation of literary works: F1_Work, F2_Expression, F3_Manifestation, and F5_Item. Several linked data projects for cultural heritage have reused this structure to model specific collections or domains related to bibliographic information. [8] mentions the reuse of FRBRoo elements for a data model to trace the history and provenance of medieval and Renaissance manuscripts. However, no explicit examples of the use of LRMoo in the data set concerning have been given. In [2], the authors report on the reuse of FRBRoo to create an ontology for medieval manuscripts. In this case, subclasses have been created for different FRBRoo main classes with the aim of a more precise representation of the concerned objects. Properties such as Has Part have been reused to model the relation between instances of the subclasses.

Another recent work reporting the use of LRMoo is [3], where it is reused in the creation of an ontology about geographical works of the 6th-15th century. In this paper, an exhaustive description of the use of different components of CIDOC-CRM and LRMoo is present, where more specific information on the decision to use particular classes can be found. Similar presentations of the use of LRMoo can be found in other works such as [9, 10, 4].

Although previous work offers valuable insights for modelling practices in LRMoo-based projects, they also present notable limitations. Most authors tend to state their choice of classes without providing clear justifications for these decisions. This lack of explicit reasoning also extends to the introduction of subclasses, subproperties, or new elements within LRMoo ontologies, making it difficult to understand the underlying modelling principles. Furthermore, there is a strong bias in the existing literature toward projects focused on medieval manuscripts, with far fewer studies addressing the challenges of representing contemporary bibliographic information and editorial practices. To address these gaps, the following sections aim to provide a more detailed and principled approach to LRMoo modelling.

3. Adaptation of LRMoo to LKG

As noted above, the LRMoo model proposes four layers of instances related to literary works. At the first and most general level, represented by the class F1_Work, the focus is on the idea or concept

[3]Information on the transition from FRBRoo to LRMoo can be found in [7]

behind a particular work. This work can be a novel, a short story, a poem, or even the idea of a collection of different texts. Because of this, some F1 instances can be linked to other F1 instances through the property R67_has_part. However, it is important to note that not all parts of a larger whole should automatically be considered separate F1 instances. Only those parts that can be seen as independent works in their own right should qualify. Examples include individual stories in Stanisław Lem's collection *The Star Diaries*, separate dialogues from his *Dialogues*, and sections from his *Science Fiction and Futurology*, *The Philosophy of Chance*, or *Summa Technologiae*. In contrast, chapters of Lem's novels—for instance, his best-known work *Solaris*—would not be treated as separate F1 instances, as they lack subject-matter autonomy and do not constitute self-contained units.

At the more specific level of the F2_Expression class, we deal with representations of particular versions of the works. We follow an interpretation according to which these versions are linguistic in nature and therefore also qualify as instances of the CIDOC-CRM class E33_Linguistic_Object. This approach deliberately sets aside non-textual elements such as illustrations, which—although present in Lem's work—will be represented at a different stage of the project, in the LKG.content section.

By "particular linguistic version," we mean a specific sequence of characters arranged in a certain order. We apply a relatively flexible criterion for identity here: for example, if an editor of a new edition of a given work uses the same base material as a previous edition but makes minor updates to modernise spelling, punctuation, or even certain expressions (such as replacing "electronic brain" with "supercomputer"), we still treat it as the same F2 object. However, if the title is changed, paragraphs or entire sections are added or removed, or the text is translated into another language, we then consider it a new instance of F2. This new instance is linked to the original through the property R76_is_derivative_of.

Each specific version of a work realises an abstract idea of that work. This is modelled by the property R3_is_realised_in, which connects an F1 instance to an F2 instance. Since a single work may exist in many different versions—revised editions, translations, or adaptations—one F1 instance may be realised by multiple F2 instances.

Given that Lem relatively often made changes to the structure of his works when preparing new editions (as well as during translation processes), the various ways in which one F2 instance may be derived from another form an important aspect of the domain we are modelling. For this reason, we decided to introduce several new elements into our ontology: four subproperties of R76. These are

- S761_is_translation_of, which links a source version to its translation, assuming by default that the translation is faithful to the original;
- S762_is_altered_form_of, which connects a modified instance to its original, where the modification consists of replacing one sequence of characters with another;
- S763_is_reduced_form_of, which links a shortened version to the original;
- S7641_is_extended_form_of, which links an expanded version to the original.

These S76X properties are not mutually exclusive. In particular, it is possible to model a situation in which a later version of a work is simultaneously a shortened, extended, and modified translation of the earlier one. An alternative approach would involve qualifying R76 using the "Properties of Properties" modelling pattern and employing instances of the class E55_Type. However, the solution based on subproperties was chosen for the sake of simplicity in the resulting graph (the same applies to P14, discussed later in the text).

Individual F2 instances can also be linked to each other using the property R5_has_component. For example, in the case of a short story collection, the value of this property would be the individual stories; for a collection of essays, the sections; and for a novel, the chapters. An F2 instance that serves as a component of another F2 instance can itself have subcomponents—for example, a short story may be divided into parts, or a section of an essay collection may include individual essays. The R5 property is transitive, which means that these subcomponents are also considered components of the larger whole.

A different kind of relationship between two F2 instances is represented by the property R75_incorporates. This is used when one work contains another—for instance, when a short story

includes a poem that is itself a separate literary work, or when a novel includes a passage that functions independently as a non-fiction text. In such cases, the involved F2 instances are connected using the R75 property.

Instances of the F3_Manifestation class represent specific editions of a work. These are understood as physical or digital embodiments of particular linguistic versions of the work, which is modelled using the property R4_embodies that links F3 instances to F2 instances. Multiple F3 instances can embody the same F2 object—this includes later editions of a given work or collection, provided the text remains unchanged. Typically, though not necessarily in the case of older editions, the objects modelled by instances of F3 are assigned ISBN or ISSN numbers.

We follow the principle that texts written by Lem scholars, editors, or translators, often added to newer editions of Lem's works, although also embodied by the relevant F3 instances, are not considered part of the core F2 objects those editions represent. For this reason, more recent editions of Lem's works, even if they contain additional textual material (such as the 2008 Agora edition of *His Master's Voice*, which includes a section titled *Słownik terminów Lemowskich* (Eng: *A Dictionary of Lem's Terms*) that was taken from Wojciech Orliński's book [11], absent from previous editions), are still considered manifestations of the same object of F2 as the earlier ones, provided that no significant changes have been made.

F3 instances can form hierarchies because a specific edition may consist of two or more volumes. In such cases, the higher-level F3 object embodies a higher-level F2 instance, while the lower-level F3 objects representing individual volumes embody only those lower-level F2 instances that are actually included in each volume (a property that allows F3 instances to be combined into hierarchical structures is R71_has_part).

For example, the F3 object representing the 1996 Interart edition of *Dialogues* embodies the F2 instance that represents the version of *Dialogues* prepared by Lem for the second edition from 1972. This version consists of the author's foreword, the "Dialogues" section containing eight original dialogues, and two appendices. The two subordinate F3 objects representing the two volumes of the Interart edition each embody only those F2 instances that they contain. Thus, the F3 representation of Volume I embodies the F2 instances modelling the foreword and the "Dialogues" section, while the F3 representation of Volume II embodies the F2 instances modelling the two appendices, as well as the afterword by Jerzy Jarzębski.

The most concrete layer of representation is formed by the F5_Item class. Its instances are texts in the sense of specific physical objects such as published books, printouts, typescripts, manuscripts, or digital copies. Within LKG.core, instances of this class include both copies of books and journal editions relevant to the project (including digital versions, such as files), as well as preserved typescripts of Lem's works. F5 instances are linked to F3 instances through the property R7_exemplifies. Although we envisage utilising class F5 in LKG, this will only be implemented at later stages of the project and therefore falls outside the scope of this paper.

In addition to the four basic classes, the LRMoo ontology also includes three classes that are relevant to our work, which represent the processes involved in creating instances of the F1, F2, and F3 classes. These are, respectively: F27_Work_Creation, F28_Expression_Creation, and F30_Manifestation_Creation. All of these are subclasses of the CIDOC-CRM class E65_Creation. A fourth class in this category is F31_Performance, which represents acts of performing dramatic or stage works.

Instances of these classes make it possible to link textual objects and performances to the individuals or organisations responsible for their creation, such as authors, translators, publishers, journal editors, actors, or directors. F27 instances are linked to F1 instances via the property R16_created; F28 instances to F2 via R17_created; F30 instances to F3 via R24_created; and F31 instances to F1 via R80_performed.

Instances of the class E39_Actor—including its two subclasses, E21_Person (representing individual people) and F11_Corporate_Body (representing institutions)—are connected to instances of the three E65_Creation-related classes and to F31 through the CIDOC-CRM property P14_carried_out_by. Within LKG.core, we decided to introduce seven sub-properties of P14 in order to model the specific roles played by particular actors in the creation of objects. These are

- S141_composed_by — representing the act of conceiving the idea of a work (used with F27 instances);
- S142_written_by — representing the act of writing or adapting a specific linguistic version of a work (used with F28);
- S143_translated_by — representing the act of producing a translation of a text (used with F28);
- S144_edited_by — representing the act of editing a particular edition (used with F30);
- S145_published_by — representing the act of publishing a specific edition (used with F30);
- S146_performed_by — representing the act of performing a work (used with F31);
- S147_directed_by — representing the act of directing a performance (used with F31).

In addition to the previously mentioned classes E21, E39 and E65, as well as property P14, the LKG ontology also includes other elements taken directly from CIDOC-CRM. Most importantly, it is assumed that all objects of class F2 are also instances of E33_Linguistic_Object. This allows them to be linked—via the property P72_has_language—to instances of the class E56_Language. Each F2 object is also linked through the property P102_has_title to an instance of the class E35_Title. In the LKG ontology, these E35 instances are also treated as E33 objects, meaning they too can be linked to instances of E56_Language. Additionally, each E35 instance has the data property P190_has_symbolic_content, whose value is a character string representing the title—this is the text that will eventually be displayed in the user interface.

Other CIDOC-CRM classes used in the LKG.core ontology to store user interface content include E41_Appellation and its subclass E42_Identifier. Instances of these classes are linked to objects of type F2, F3, F11, and E21 using the property P1_is_identified_by. Furthermore, instances of E42 are specified using the property P2_has_type, with values from class E55_Type, indicating the kind of identifier used.

Finally, the class E52_Time-Span, together with the property P4_has_time-span and the data property P82_at_some_time_within, allows time of creation to be assigned to instances of subclasses of E65_Creation.

4. Assigning LRMoo levels of representation to various objects in LKG

The use of LRMoo for building collections of information on literary legacy requires a number of detailed decisions regarding how to interpret the roles of the various levels of representation. This is particularly relevant for a project such as LKG, whose main goal is to create a knowledge graph based on already collected and structured information about individual editions and their internal structure. The primary source of these data is the catalogue of Lem's non-fiction texts, compiled by Victor Yaznevich, which takes the form of a spreadsheet file with multiple tabs, each representing groups of editions in different languages. The direct printed representation of the content of this file is [12]. Later, the graph must be extended with information related to fiction. Fortunately, digitised sources are available for this purpose, including Vladimir Borisov's database and data collected by Wojciech Zemek. We already have these sources and have permission to integrate them into the graph.

The Yaznevich catalogue records individual editions of texts as well as their internal structure when applicable, for example, when they are divided into parts. In presenting this structure, the Lem scholar occasionally introduces certain simplifications. For example, in the case of the second edition of *Dialogues* from 1972, which is significantly expanded compared to the first edition from 1957, he identifies four main parts. However, the second of these—whose internal structure closely mirrors that of the entire first edition—is not further divided into subparts. Instead, a note in the relevant cell of the spreadsheet indicates that its content consists of eight parts, detailed in the description of the first edition.

It might seem that, since the entries in Yaznevich's catalogue refer to specific editions, the base layer of the database should consist of instances of the F3_Manifestation class. However, there is a reason why this should not be the case. The creators of the LRMoo standard state that "an F3 Manifestation is the outcome of a publication process where one or more F2 Expressions are prepared for public

dissemination" and that "an instance of F3 Manifestation typically incorporates one or more instances of F2 Expression representing a distinct logical content and all additional input by a publisher such as text layout and cover design" ([13], p. 25).

In the case of single-work publications, such as *Dialogues*, there does indeed appear to be an F3 instance corresponding directly to that edition. However, this is not the case with a work like "Dialog o wskrzeszaniu z atomów" (Eng: "A Dialogue on Resurrection from Atoms"), published in issue 3 of the 1957 volume of *Nowa Kultura*. In this case, the F3 instance is the entire issue of the journal, which potentially incorporates many F2 instances, including the one that is relevant to us: Lem's text. The manifestation level does not break down into the individual texts it contains, but rather into individual volumes in the case of multi-volume publications. In such situations, there is a parent F3 instance representing the full multivolume edition, as well as F3 instances for each individual volume, which are linked to the former via the R71_has_part property ([13], p. 49).

In addition, Yaznevich's data do not include texts by other authors. This means that the F3 instance corresponding to the 2008 edition of *Dialogues*—just like the F3 instance for issue 3 of *Nowa Kultura* from 1957—also incorporates multiple F2 instances. In this situation, it becomes clear that no single LRMoo level should directly mirror the structure of the Yaznevich catalogue. Instead, this structure should be distributed between both F2 and F3 instances.

Specifically, all catalogue records that represent separate publications (as long as they are not individual volumes of a larger whole) should serve as the basis for creating distinct F2 instances. In addition, records that represent parts or subparts of such publications should also give rise to F2 instances, which are connected to the relevant publication instances via the R5_has_component property. Records that indicate standalone books (which are systematically marked in the file) should also form the basis for creating F3 instances, linked to the corresponding F2 instances via the R4_embodies property. Records referring to individual volumes should lead to the creation of F3 instances for each of those volumes, as well as an F3 instance representing the complete multivolume edition. Although this overarching instance is not explicitly represented in the Yaznevich file, it can be identified automatically based on the available data. Only this parent F3 instance should have a corresponding F2 instance, since the division into volumes does not reflect a logical division of the text into parts.

Finally, records corresponding to publications in journals should generate the appropriate F2 instances, while F3 instances would correspond to the journal issues themselves, which can be reconstructed from the data fields in these records, as separate volumes.

5. Generating the first version of the graph

The first version of the graph is created based on the Yaznevich file: other sources will be involved on later stages. The initial process of generating the graph from the file is automated: the appropriate structures are produced by Python scripts. The content of the records makes it possible to generate not only instances of the classes F1, F2, and F3, but also E35_Title, F28_Expression_Creation, and F30_Manifestation_Creation. Furthermore, depending on whether an F3 instance represents a book edition or a journal issue, one or more instances of the class E42_Identifier are created. The main identifier for the F1, F2, and F3 objects is the record number of the Yaznevich file (referred to as "YID"). This number is an integer for top-level F2 instances, while for subordinate instances, it consists of the top-level instance number followed by the number of the subordinate instance, separated by a dot. For journal volumes, the identifiers include the issue number, the journal title, and the year of publication.

Instances of F1_Work are generated as follows: it is assumed that all instances of F2_Expression that have not been assigned the property R76_is_derivative_of (or any of its subproperties) should have a corresponding F1_Work instance. Furthermore, each F1_Work instance created on the basis of a non-derivative F2_Expression is linked to that F2 instance and all of its derivatives with the property R3_is_realised_in.

Top-level F3 objects representing multivolume editions of a single work, along with their corresponding F2 objects, are assigned a Yaznevich identifier that combines the numbers of individual volumes (for

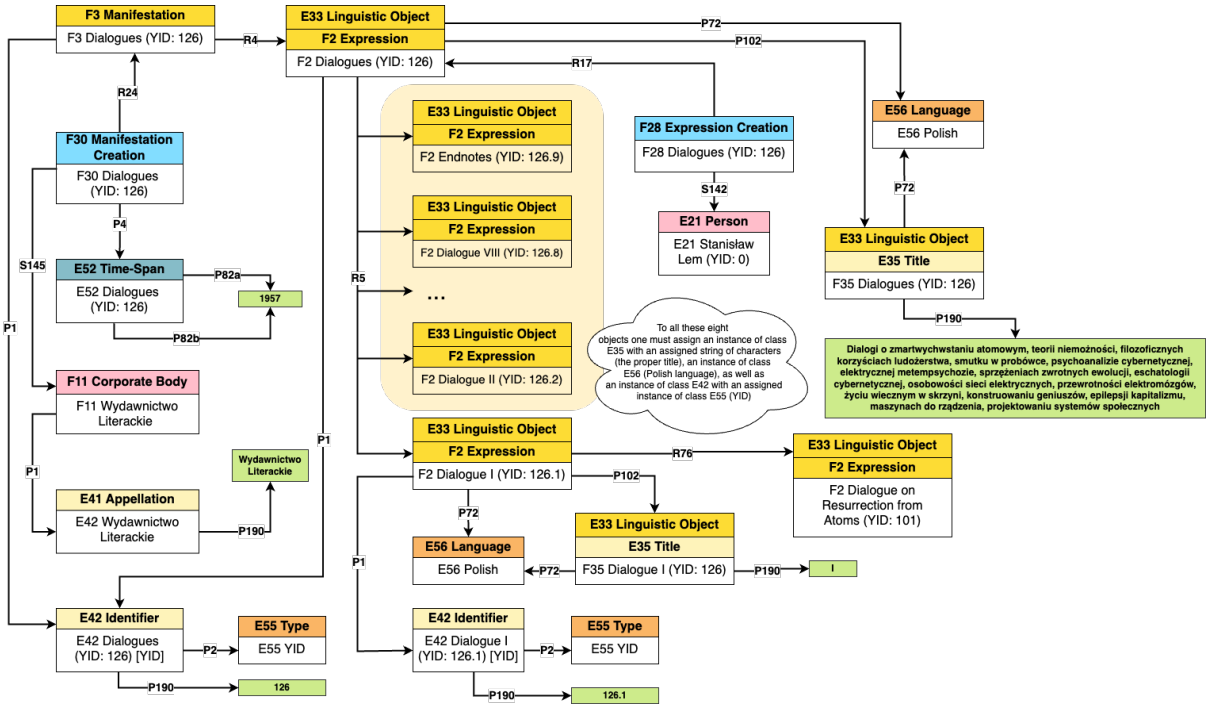

Figure 1: Creating the graph based on the Yaznevich file: the first standalone edition of the *Dialogues* (1957)

example, "1025/1026"). The type of identifier is specified by instances of the E55_Type class.

We assume that all instances of the F2 class are also instances of the E33_Linguistic_Object class. For this reason, both F2 and E35 objects are assigned instances of the E56_Language class. Since the first version of the graph is based on Polish editions, the E56 instance representing the Polish language is assigned to all F2 and E35 instances in the graph.

It is also known who is responsible for creating all the texts described in the graph. Therefore, all F28 objects are assigned the property S142_written_by, its value being an object of the E21_Person class representing Stanisław Lem. Meanwhile, instances of the F30 class are assigned E52_Time-Span objects that represent the years in which a volume or journal issue was published. They are also linked, through the S145_published_by property, to an object of the F11_Corporate_Body class representing the publishing house or the journal editorial team. The relevant information is also found in the Yaznevich file. Fig. 1 presents a target fragment of the first version of the graph, which may be fully derived from the content of the Yaznevich file.

6. Concluding remarks

The procedure outlined in the previous section enables the construction of only part of the initial version of the LKG.core graph. Not only does it lack information about individual publications of Lem's fiction, but the generated graph is also severely imprecise. In particular, not all relevant relations between different F2 instances are properly marked in the Yaznevich file. As a result, some F2 instances created as non-derivative actually represent derivative linguistic objects, which in turn leads to the erroneous creation of F1 instances that should not exist. Moreover, Yaznevich did not indicate which of the relations modelled by subproperties of R76 hold between F2 instances corresponding to different editions of the same text. Consequently, the method used to generate the graph automatically produces an F2 instance for each edition, regardless of whether any actual textual differences exist. All of this implies that manual correction of the graph, informed by expert knowledge, will be necessary at a later stage.

Nevertheless, even this preliminary version of the graph, generated automatically from the Yaznevich

file, already allows for the retrieval of valuable information about Lem's publications. Furthermore, the modeling decisions and accompanying rationale described in this article can serve as a reference for future projects, supporting more consistent and transparent applications of LRMoo.

Acknowledgments

This publication was funded by a flagship project "CHExRISH: Cultural Heritage Exploration and Retrieval with Intelligent Systems at Jagiellonian University" under the Strategic Programme Excellence Initiative at Jagiellonian University.

The research for this publication has been supported by a grant from the Priority Research Area DigiWorld under the Strategic Programme Excellence Initiative at Jagiellonian University.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-4o, GPT-4o-mini and DeepSeek-V3 in order to: Grammar and spelling check, Paraphrase and reword. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] E. Hyvönen, Using the semantic web in digital humanities: Shift from data publishing to data-analysis and serendipitous knowledge discovery, Semantic Web 11 (2020) 187 – 193. doi:10.3233/SW-190386.

[2] S. Barzaghi, M. Palmirani, S. Peroni, et al., Development of an ontology for modelling medieval manuscripts: The case of progetto irnerio, Umanistica Digitale 9 (2020) 117–140.

[3] V. Bartalesi, D. Metilli, N. Pratelli, P. Pontari, Towards a knowledge base of medieval and renaissance geographical latin works: The imago ontology, Digital Scholarship in the Humanities 37 (2022) 34–50.

[4] J. Rodwell, M. Whitelaw, From inception to interface: Ontologies, data modelling, and linked data for online exhibition-making, Parergon 41 (2024) 161 – 187. doi:10.1353/pgn.2024.a946929.

[5] L. do Valle Miranda, J. Gomułka, S. P. Kukulak, K. Kutt, G. J. Nalepa, Lrmoo as the conceptual model for the lem knowledge graph, in: Semantic Digital Humanities 2025: Proceedings of the Second International Workshop of Semantic Digital Humanities (SemDH 2025) co-located with the Extended Semantic Web Conference 2025 (ESWC 2025), 2025. URL: https://ceur-ws.org/Vol-4009/paper_14.pdf.

[6] V. Bartalesi, D. Metilli, N. Pratelli, P. Pontari, Towards a knowledge base of medieval and renaissance geographical latin works: The imago ontology, Digital Scholarship in the Humanities 37 (2021) 34–50. doi:10.1093/llc/fqab060.

[7] P. Riva, M. Zumer, Array(0x55ec518936d8), 2017. URL: https://library.ifla.org/id/eprint/2130/.

[8] T. Burrows, D. Emery, A. M. Fraas, E. Hyvönen, E. Ikkala, M. Koho, D. Lewis, A. Morrison, K. Page, L. Ransom, et al., Mapping manuscript migrations knowledge graph: data for tracing the history and provenance of medieval and renaissance manuscripts, Journal of Open Humanities Data 6 (2020).

[9] V. Bartalesi, N. Pratelli, C. Meghini, D. Metilli, G. Tomazzoli, L. M. Livraghi, M. Zaccarello, A formal representation of the divine comedy's primary sources: The hypermedia dante network ontology, Digital Scholarship in the Humanities 37 (2022) 630–643.

[10] F. Tomasi, F. Giovannetti, et al., The linked finding aid as a platform for textual research: The case study of the giuseppe raimondi archive, in: CEUR WORKSHOP PROCEEDINGS, volume 3019, 2021, pp. 104–113.

[11] W. Orliński, Co to są sepulki? : wszystko o Lemie, Społeczny Instytut Wydawniczy Znak, Kraków, 2007.

[12] V. Yaznevich, Stanislaw Lem Non Fiction. Bibliography and Personalities, Medisont, Minsk, 2021.
[13] I. L. W. Group, Lrmoo: Object-oriented definition and mapping from the ifla library reference model, 2024. URL: https://cidoc-crm.org/sites/default/files/LRMoo_V1.0.pdf, last accessed: 12 March 2025, CIDOC CRM.