Leveraging Large Language Models for Ontology Requirements Engineering

Yihang Zhao

March 31, 2025

Leveraging Large Language Models for Ontology Requirements Engineering

Yihang Zhao[1[0009−0009−2436−8145]]

King's College London, London, UK
yihang.zhao@kcl.ac.uk

Abstract. Ontologies are essential for structuring domain knowledge, enabling shared understanding to address the challenges of exponential web data growth. Ontology Engineering (OE) has evolved into a collaborative, community-driven practice, with Ontology Requirements Engineering (ORE) providing a systematic framework for capturing, documenting, and validating requirements to support ontology development, evaluation, and maintenance. However, ORE still relies on manual techniques such as brainstorming, interviews, and spreadsheets, making the process resource-intensive. Recent advances in Large Language Models (LLMs) present new opportunities to support ORE tasks. Existing studies highlight their potential in ontology user story generation, as well as competency questions (CQs) generation and retrofitting. However, LLM-based ORE frameworks are still in their early stages and lack structured guidance across the full ORE workflow. Therefore, this research aims to bridge the gap by investigating how ORE tasks can be potentially supported by LLMs and developing the conversational agent OntoChat to integrate LLMs for assisting users in these tasks. In this paper, we present preliminary findings on how LLMs can potentially support ORE based on the first year of this research.

Keywords: Large Language Models · LLMs · Ontology Engineering · Requirements Engineering · Competency Questions · User Stories

1 Introduction/Motivation

Ontology originated as a philosophical concept referring to the study of being [18]. In information science, it organizes domain knowledge into structured models by defining entities and their relationships, creating a shared understanding that enables automated reasoning, data integration, and semantic search. This helps address data overload, ambiguity, and system interoperability as the web evolves [15].

Ontology Engineering (OE) involves designing and managing ontologies [15]. Early approaches require experts to manually define ontologies using formal languages such as RDF, RDFS, and OWL [21, 9, 6, 3]. These centralized methods limit scalability. Modern OE adopts collaborative, community-driven approaches, allowing contributors with diverse skills from different locations to build and refine ontologies asynchronously [31, 33, 24, 26]. This approach makes ontology development more inclusive and adaptive to evolving requirements.

Ontology requirements are mainly represented as ontology user stories [7, 37, 38] and competency questions (CQs) [22, 25, 26, 34]. Ontology user stories define a typical user's persona, their goal, and the gap the ontology aims to fill. These stories are first converted into CQs, which are single-sentence natural language queries, and then transformed into structured queries like SPARQL that the ontology should answer. CQs can also be generated from alternative reference materials or processes or retrofitted from existing ontologies. Ontology requirements also include non-functional requirements such as ontological system constraints, performance criteria, interoperability considerations, etc.

Ontology Requirements Engineering (ORE) [19] provides a structured approach to defining, documenting, and validating these requirements, producing an Ontology Requirements Specification Document (ORSD) [33, 13]. The ORSD encapsulates the ontology's objectives, domain coverage, envisioned applications, CQs, etc. This artifact forms the foundation for ontology development and refinement, ensuring a shared understanding of its expected capabilities among contributors.

Traditional ORE methods, as specified in METHONTOLOGY [13] and NeOn [33], etc., rely on manual collaboration techniques such as brainstorming, structured interviews, and spreadsheets among knowledge engineers, ontology engineers, and domain experts to collect and refine requirements. These methods are resource-intensive [37], requiring skilled personnel to understand ontology user stories, CQs, and ORSD, as well as significant time for interviews, coordination, and data consolidation. These challenges highlight the need for automated approaches to streamline workflows.

In recent years, large language models (LLMs) have been increasingly adopted in ORE due to their ability to understand and generate text efficiently [23]. Existing research has explored LLM applications in ORE, including requirements elicitation (ontology user story generation [7, 37, 38] and CQs generation [10, 1] and retrofitting [37, 2]) and requirements analysis (CQs filtration [37, 1, 10]).

However, LLM-based ORE frameworks remain in their early stages, and structured guidance across the full ORE workflow, including requirements elicitation, analysis, documentation, validation, and management, is still lacking. Given that LLM applications in Software Requirements Engineering (SRE) are more mature [4], the first goal of this research is to bridge the gap by investigating how LLMs support SRE tasks and mapping ORE tasks to their SRE counterparts to assess LLMs' potential in ORE. Additionally, an early prototype of an LLM-integrated conversational framework, OntoChat [37, 38], built on the Infer, Design, Create, and Analyse (IDEA) framework [7], has provided initial support for ORE. The second goal of this study is to explore how OntoChat can evolve to further support ORE tasks with user satisfaction.

The remainder of the paper is structured as follows. Section 2 introduces the state of the art. Section 3 presents our research questions, problem statements, and contributions. Section 4 describes our intended methodology, followed by the evaluation approach in Section 5. Finally, Section 6 reports the early findings from the first year of research.

2 State of the Art

Conventional collaborative OE methodologies enable asynchronous collaboration among contributors with diverse expertise in decentralized environments, with each methodology focusing on different aspects [14]. For example, METHONTOLOGY [13] and Ontology Development 101 [22] focus on providing guidelines for defining ontology terms, constructing hierarchies, and refining structures using competency questions. DILIGENT [36], RapidOWL [5], and NeOn [33] focus on managing ontology evolution through structured decision-making, community feedback, and change tracking. SAMOD [24] focuses on guiding test-driven development for modular, reusable ontologies. Ontology Maturing [8] and HCOME [20] focus on formalizing informal knowledge into structured ontologies. Dogma-Mess [29, 32] focuses on ensuring accountability by assigning different roles to the most suitable tasks. However, these methodologies require extensive expert coordination and resources, limiting scalability in time-sensitive projects.

In conventional ORE, ontology requirements are often expressed as ontology user stories and CQs. Ontology user stories provide a user-friendly way to define a typical user's persona (name, age, occupation, skills, interests), user goal (the ontology's intended purpose), and scenarios (how the goal is currently achieved, highlighting gaps). The Polifonia Project [1] provides collaborative guidance where contributors submit and organize user stories into persona-based folders [7], accessible on GitHub [2]. GitHub's version control enables iterative refinement through collective feedback, though challenges include the lack of real-time guidance, variations in quality, and ongoing maintenance. CQs are single-sentence natural language queries the ontology should answer [22, 25, 26, 30, 34]. Gruninger and Fox [17], Ontology 101 [22], Rao et al. [27], and The "pay-as-you-go" methodology [30] provide guidance for CQ development, including structured questions and card sorting techniques. Beyond user stories and CQs, ontology requirements also include references for terminology, reusable external ontologies, system interface constraints, performance constraints, etc. All relevant requirements are documented into the Ontology Requirements Specification Document (ORSD) [33, 13], promoting a shared understanding of the ontology's expected capabilities among contributors. However, this process remains manual, complex, and requires extensive coordination among contributors and ontology engineers to ensure accuracy and consistency.

LLMs have introduced automated and semi-automated approaches for ORE. For ontology user story generation, an early prototype of OntoChat [37, 38] explores LLMs' potential in supporting Human-GenAI collaborative generation through guided elicitation and iterative refinement. However, the lack of user experience design results in inefficient user interactions. For CQs generation and

[1] https://polifonia-project.eu/
[2] https://github.com/polifonia-project/stories

retrofitting, approaches vary based on the knowledge resources used in prompts. OntoChat [37] generates CQs from ontology user stories and refines them by splitting non-atomic CQs and abstracting named entities, though it struggles with ambiguous story contents. AgOCQs [2] and NeOn-GPT [12] utilize domain-specific texts and controlled templates for semantically aligned CQ generation, with AgOCQs further filtering by removing duplicates and meaningless questions through semantic grouping. However, both methods struggle with generalization across diverse or sparsely documented domains. RevOnt [10] and RETROFIT-CQs [1] retrofit CQs from existing knowledge graphs (KGs) and filter redundancy through paraphrase detection, but their effectiveness depends on well-structured KGs.

3 Problem Statement and Contributions

LLM-based ORE frameworks are still in their early stages and lack structured guidance across the full ORE workflow. This research aims to explore the development of an LLM-integrated conversational agent to potentially support ORE through a user-centred approach.

- **RQ1: How can ORE tasks be potentially supported by LLMs?** To explore this, we plan to conduct a systematic literature review (SLR) to identify tasks in SRE that are already supported by LLMs. By mapping ORE tasks with their SRE counterparts, we aim to explore how LLMs can potentially be leveraged to support ORE.
- **RQ2: How can OntoChat be developed to provide end-user expected support at each ORE task?** An early prototype of an LLM-integrated conversational framework, OntoChat [37, 38], built on the Infer, Design, Create, and Analyse (IDEA) framework [7], has demonstrated the potential to support ORE. However, preliminary user evaluations indicate that end-users unfamiliar with ORE or prompting strategies often struggle to craft effective prompts at different interaction stages. To address this, we plan to implement participatory prompting [28, 11] with ontology engineers, combining contextual inquiry and participatory design with researcher-mediated interactions using the GPT-4o interface. During this process, we will identify user interaction challenges in LLM-supported ORE workflows and support users in iteratively refining prompting strategies through researcher-guided engagement to mitigate them. Once user satisfaction is achieved with the generated outcomes, we will determine the most effective prompting strategies that contributed to this satisfaction, generalize them into reusable prompt templates, and integrate them into OntoChat to support end-user interaction. Finally, we will assess the usefulness of supports provided OntoChat within the semantic web community.
- **RQ3: How useful are the supports provided by OntoChat?** We plan to evaluate OntoChat's support usefulness using concurrent and stimulated

retrospective think-aloud sessions [35], cross-validated with researcher observation checklists and Likert scale questionnaires. This evaluation will uncover both strengths and limitations of OntoChat's support provided. Based on these insights, we will propose new design opportunities for integrating LLMs into future ORE workflows to mitigate the identified challenges.

4 Research Methodology and Approach

We outline the planned methodologies for answering each of the RQs.

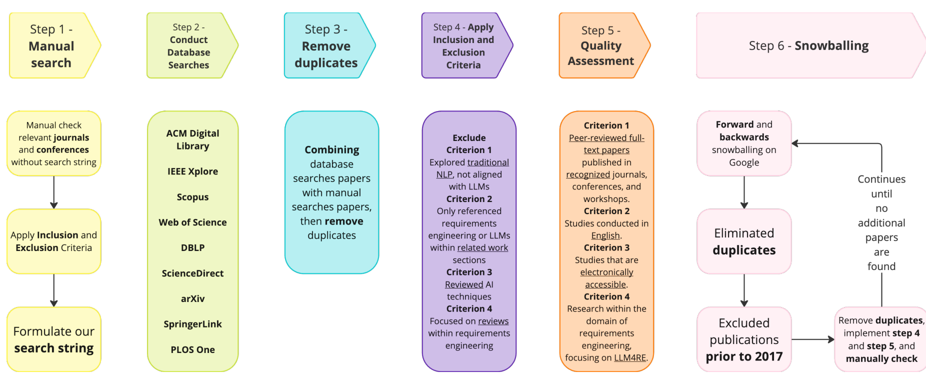

Fig. 1. SLR Pipeline for LLMs Applications in Requirements Engineering (LLM4RE)

To answer RQ1, we plan to conduct two main steps: (1) *SLR for LLM4RE* (Figure 1), where we will develop a taxonomy of LLM applications in SRE by first examining the sources of input data used for requirements processing, then identifying the types of LLM models applied, followed by mapping their roles in different SRE tasks. We will further analyse their performance, evaluation metrics, and real-world challenges to assess their effectiveness and limitations. (2) *Mapping Study*, building on the findings from the SLR, we will align ORE tasks with their SRE counterparts by comparing their objectives, input structures, and task dependencies to assess how LLMs can potentially support ORE.

For each ORE task identified as potentially supported by LLMs, we need to further investigate user challenges and expectations for LLM assistance during these tasks in RQ2.

To answer RQ2, we plan to conduct two main steps: (1) *Conduct Participatory Prompting* [28, 11], a user-centric method for exploring GenAI opportunities in ORE tasks. It involves researcher-mediated interactions with functional GenAI systems, allowing users to engage with "actually existing AI" (AEAI) while researchers gather insights. This ensures user feedback reflects AI's real capabilities and helps identify interaction challenges and support needs in ORE workflows. This process consists of three sub-steps: (1.1) *Query Initiation*, where the user reads instructions on ORE tasks and drafts an initial query; (1.2) *Prompt Refinement*, where the researcher applies pre-identified prompting strategies, submits the refined query to the GPT-4o interface, retrieves a response, and iterates based on user feedback until satisfaction is achieved; and (1.3) *Template Creation*, where the researcher generates a reusable prompt template based on the final refined prompt. This iterative process continues until all tasks are completed. (2) *Develop OntoChat*, integrating user-expected support identified in participatory prompting for ORE tasks. OntoChat will provide prompt templates that users can edit to support effective interaction with LLMs. The development process includes three phases: (2.1) *Interface Design*, including conversation window, a prompt templates library, and a prompt editor (user input box) to facilitate user interaction with OntoChat. (2.2) *System Architecture*, defining the workflow for Human-OntoChat collaboration in each ORE task; and (2.3) *Technical Implementation*, building and deploying OntoChat for publicly testing.

After building OntoChat, we will evaluate its usefulness for each ORE task in RQ3 using think-aloud protocols [35].

5 Evaluation/Evaluation Plan

To answer RQ3, the usefulness of OntoChat's supports will be assessed along two dimensions: usability and utility [16]. To evaluate *usability*, we will examine (EQ1.1) *User Understanding*, assessing whether users comprehend the purpose of prompt templates, (EQ1.2) *Ease of Locating*, determining how easily users find appropriate templates in the library, and (EQ1.3) *Ease of Using*, evaluating the intuitiveness of editing templates. To assess *utility*, we will analyse (EQ2.1) *Template Library*, measuring the extent to which the library provides templates that align with user needs at each ORE task stage, (EQ2.2) *Prompt Template*, assessing the extent to which the editable content within templates meets user needs, and (EQ2.3) *OntoChat Response*, assessing the degree to which OntoChat's generated outputs meet user expectations. Additionally, we will collect feedback on (1) *users' inclination to adopt provided templates*, (2) *the perceived relevance of generated outputs to their ORE workflows*, and (3) *the usefulness of these outputs for ontology construction*, all measured on a Likert scale to capture varying levels of satisfaction.

The user evaluation methods we plan to use follow a structured three-step process, combining think-aloud protocols [35] and a post-task questionnaire. (1) *Concurrent Think-Aloud (CTA) Session*: Participants interact with OntoChat without prior instructions, engaging in ORE tasks while verbalizing their thoughts. Researchers use observation checklists to track user actions, including template selection, editing, and submission, alongside encountered challenges. (2) *Stimulated Retrospective Think-Aloud (SRTA) Session*: Participants reflect on their experiences with OntoChat, responding to open-ended questions about the effectiveness and limitations of the provided support. (3) *Post-Task Questionnaire*: A Likert-scale questionnaire quantifies user perceptions of OntoChat's usefulness, supplemented by open-ended questions on user adoption, relevance to

ORE workflows, and potential feature improvements. (4) *Data Analysis*: Qualitative data will be analysed using thematic coding to identify key themes, while quantitative data will be processed using statistical methods, with bar charts visualizing findings. Data triangulation will be applied to identify convergence points between qualitative themes and quantitative results, uncovering user interaction challenges.

6 Results

We now present the early findings from the SLR and mapping study in RQ1. Through the SLR, we identify LLM applications in SRE, and in the mapping study, we align ORE tasks with SRE activities based on input data types and task objectives. Following this, we analyse how LLMs are applied in SRE to derive insights into their potential applications in ORE. As a result, we identify ten ORE tasks that LLMs can potentially support, as illustrated in Table 1.

Table 1. Potential Support of LLMs in Ontology RE Tasks

ORE Activities	ORE Tasks	How LLMs Can support
Requirements elicitation	Ontology user story generation	(1) *Knowledge elicitation*: LLMs guide users through a structured workflow to gather ontology user story components, including personas, goals, and scenarios, by posing elicitation questions and providing example answers to support user responses. (2) *Story refinement*: LLMs structure collected inputs into a formal story format, suggest missing necessary details, and iteratively refine the story based on user feedback until satisfaction is achieved.
	CQs generation	LLMs utilizes ontology user stories, or domain-specific texts to generate CQs.
	CQs retrofitting	LLMs retrofit new CQs for ontologies that lack explicitly published CQs.
Requirements analysis	Extract glossary of terminology	LLMs extract domain-specific concepts, relationships, and attributes from requirements sources such as ontology user stories, CQs, interview scripts, and additional documentation to establish a structured vocabulary.
	CQs filtration	LLMs refine CQs by applying redundancy checks, relevance filtering (paraphrase detection and semantic grouping), splitting complex queries into atomic ones, and abstracting named entities.
Requirements documentation	User story to ORSD conversion	LLMs generate the purpose, scope, objectives, domain coverage, and granularity of the ontology from ontology user stories to create ORSD.
Requirements validation	Requirements validation	(1) *Evaluate CQs*: LLMs evaluate CQs for correctness, completeness, consistency, verifiability, understandability, ambiguity, conciseness, realism, modifiability, and traceability, providing justifications for each assessment. (2) *Evaluate Ontology User Stories*: LLMs assess ontology user stories for well-formedness, realism, and correlation, offering explanations for their evaluations.
	Ontology testing support	(1) *SPARQL-Free Ontology Evaluation*: LLMs verbalize an OWL ontology by documenting its classes, properties, named entities, and relationships, and support users evaluating the verbalized ontology for coverage and correctness by prompting CQs. (2) *SPARQL-based ontology evaluation*: LLMs convert CQs into SPARQL queries to retrieve answers from ontology, validating its correctness and completeness.
Requirements management	Trace link establishment	LLMs generate semantic connections between interview scripts, conversation history, user stories, CQs, conceptual models, and other OE artifacts. These links improve consistency, maintainability, and traceability by aligning related concepts across ontology development artifacts.

As this review is still ongoing, the above list is not exhaustive. Once completed, the next step is to refine the taxonomy and analyse how to develop OntoChat to support these tasks (RQ2) and evaluate its usefulness (RQ3).

7 Conclusions/Lessons Learned

This work presents early-stage (first-year) PhD research on the open problem of theorizing and developing methods and tools for integrating LLMs to support ORE. The initial findings from RQ1 highlight LLM potential in ontology user story and CQs elicitation, analysis, documentation, validation, and management. However, limitations persist, as mapping results are subjective, requiring expert reviews and pilot studies for validation. Future work should focus on identifying effective prompting strategies to enhance user interactions with LLMs for specific ORE tasks (RQ2) and evaluating their usefulness (RQ3). Our aim is to advance LLM-assisted ORE, reducing resource-intensive efforts and enabling broader accessibility to ontology development across various domains.

Acknowledgments. This research is supervised by Prof. Elena Simperl and Dr. Albert Meroño Peñuela, with additional support from Dr. Valentina Tamma and Dr. Jacopo de Berardinis. This study has received multiple sources of funding, including co-funding from the MuseIT project under grant agreement No. 101061441 as part of the European Union's Horizon 2021-2027 research and innovation programme, as well as funding from SIEMENS AG and the Technical University of Munich, Institute for Advanced Study, Germany.

References

1. Alharbi, R., Tamma, V., Grasso, F., Payne, T.: An experiment in retrofitting competency questions for existing ontologies. In: Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing. pp. 1650–1658 (2024)
2. Antia, M.J., Keet, C.M.: Automating the generation of competency questions for ontologies with agocqs. In: Iberoamerican Knowledge Graphs and Semantic Web Conference. pp. 213–227. Springer (2023)
3. Antoniou, G., Harmelen, F.v.: Web ontology language: Owl. Handbook on ontologies pp. 91–110 (2009)
4. Arora, C., Grundy, J., Abdelrazek, M.: Advancing requirements engineering through generative ai: Assessing the role of llms. In: Generative AI for Effective Software Development, pp. 129–148. Springer (2024)
5. Auer, S., Herre, H.: Rapidowl—an agile knowledge engineering methodology. In: International Andrei Ershov Memorial Conference on Perspectives of System Informatics. pp. 424–430. Springer (2006)
6. Bechhofer, S., Van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A., et al.: Owl web ontology language reference. W3C recommendation **10**(2), 1–53 (2004)
7. de Berardinis, J., Carriero, V.A., Jain, N., Lazzari, N., Meroño-Peñuela, A., Poltronieri, A., Presutti, V.: The polifonia ontology network: Building a semantic backbone for musical heritage. In: International Semantic Web Conference. pp. 302–322. Springer (2023)
8. Braun, S., Schmidt, A.P., Walter, A., Nagypal, G., Zacharias, V.: Ontology maturing: a collaborative web 2.0 approach to ontology engineering. In: Ckc. Citeseer (2007)

9. Brickley, D., Guha, R.: Resource description framework schema (rdfs). W3C Proposed Recommendation **3** (1999)
10. Ciroku, F., de Berardinis, J., Kim, J., Meroño-Peñuela, A., Presutti, V., Simperl, E.: Revont: Reverse engineering of competency questions from knowledge graphs via language models. Journal of Web Semantics p. 100822 (2024)
11. Drosos, I., Sarkar, A., Xu, X., Negreanu, C., Rintel, S., Tankelevitch, L.: "it's like a rubber duck that talks back": Understanding generative ai-assisted data analysis workflows through a participatory prompting study. In: Proceedings of the 3rd Annual Meeting of the Symposium on Human-Computer Interaction for Work. pp. 1–21 (2024)
12. Fathallah, N., Das, A., Giorgis, S.D., Poltronieri, A., Haase, P., Kovriguina, L.: Neon-gpt: a large language model-powered pipeline for ontology learning. In: European Semantic Web Conference. pp. 36–50. Springer (2024)
13. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: Methontology: from ontological art towards ontological engineering (1997)
14. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontology tools. Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web pp. 293–362 (2004)
15. Gómez-Pérez, A., Fernández-López, M., Corcho, O.: Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. Springer Science & Business Media (2006)
16. Group, N.N.: Usability 101: Introduction to usability. https://www.nngroup.com/articles/usability-101-introduction-to-usability/, accessed: 2025-02-01
17. Grüninger, M., Fox, M.S.: The role of competency questions in enterprise engineering. In: Benchmarking—Theory and practice, pp. 22–31. Springer (1995)
18. He, Y.: Language models for ontology engineering. Ph.D. thesis, University of Oxford (2024)
19. Kendall, E.F., McGuinness, D.L.: Ontology engineering. Morgan & Claypool Publishers (2019)
20. Kotis, K., Vouros, G.A.: Human-centered ontology engineering: The hcome methodology. Knowledge and Information Systems **10**, 109–131 (2006)
21. Lassila, O.: Resource description framework (rdf) model and syntax (1997)
22. Noy, N.F., McGuinness, D.L., et al.: Ontology development 101: A guide to creating your first ontology (2001)
23. Pan, J.Z., Razniewski, S., Kalo, J.C., Singhania, S., Chen, J., Dietze, S., Jabeen, H., Omeliyanenko, J., Zhang, W., Lissandrini, M., et al.: Large language models and knowledge graphs: Opportunities and challenges. arXiv preprint arXiv:2308.06374 (2023)
24. Peroni, S.: Samod: an agile methodology for the development of ontologies. In: Proceedings of the 13th OWL: Experiences and Directions Workshop and 5th OWL reasoner evaluation workshop (OWLED-ORE 2016). pp. 1–14 (2016)
25. Poveda-Villalón, M., Fernández-Izquierdo, A., Fernández-López, M., García-Castro, R.: Lot: An industrial oriented ontology engineering framework. Engineering Applications of Artificial Intelligence **111**, 104755 (2022)
26. Presutti, V., Daga, E., Gangemi, A., Blomqvist, E.: extreme design with content ontology design patterns. In: Proc. Workshop on Ontology Patterns. pp. 83–97 (2009)
27. Rao, L., Reichgelt, H., Osei-Bryson, K.M.: Knowledge elicitation techniques for deriving competency questions for ontologies. In: ICEIS (3-2). pp. 105–110 (2008)

28. Sarkar, A., Drosos, I., Deline, R., Gordon, A.D., Negreanu, C., Rintel, S., Williams, J., Zorn, B.: Participatory prompting: a user-centric research method for eliciting ai assistance opportunities in knowledge workflows. arXiv preprint arXiv:2312.16633 (2023)
29. Schärfe, H., Hitzler, P., Ohrstrom, P.: Conceptual Structures: Inspiration and Application: 14th International Conference on Conceptual Structures, ICCS 2006, Aalborg, Denmark, July 16-21, 2006, Proceedings, vol. 4068. Springer (2006)
30. Sequeda, J.F., Briggs, W.J., Miranker, D.P., Heideman, W.P.: A pay-as-you-go methodology to design and build enterprise knowledge graphs from relational databases. In: The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II 18. pp. 526–545. Springer (2019)
31. Simperl, E., Luczak-Rösch, M.: Collaborative ontology engineering: a survey. The Knowledge Engineering Review **29**(1), 101–131 (2014)
32. Spyns, P., Tang, Y., Meersman, R.: A model theory inspired collaborative ontology engineering methodology. Journal of Applied Ontology **4**(1) (2007)
33. Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernández-López, M.: The neon methodology for ontology engineering. In: Ontology engineering in a networked world, pp. 9–34. Springer (2011)
34. Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernandez-Lopez, M.: The neon methodology framework: A scenario-based methodology for ontology development. Applied ontology **10**(2), 107–145 (2015)
35. Van Den Haak, M., De Jong, M., Jan Schellens, P.: Retrospective vs. concurrent think-aloud protocols: testing the usability of an online library catalogue. Behaviour & information technology **22**(5), 339–351 (2003)
36. Vrandečić, D., Pinto, S., Tempich, C., Sure, Y.: The diligent knowledge processes. Journal of Knowledge Management **9**(5), 85–96 (2005)
37. Zhang, B., Carriero, V.A., Schreiberhuber, K., Tsaneva, S., González, L.S., Kim, J., de Berardinis, J.: Ontochat: a framework for conversational ontology engineering using language models. In: European Semantic Web Conference. pp. 102–121. Springer (2024)
38. Zhao, Y., Zhang, B., Hu, X., Ouyang, S., Kim, J., Jain, N., de Berardinis, J., Meroño-Peñuela, A., Simperl, E.: Improving ontology requirements engineering with ontochat and participatory prompting. In: Proceedings of the AAAI Symposium Series. vol. 4, pp. 253–257 (2024)