Translating Natural Language Questions into CIDOC-CRM SPARQL Queries to Access Cultural Heritage Knowledge Bases

DAVIDE VARAGNOLO, Department of Informatics, University of Évora, Évora, Portugal and NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal
DORA MELO, Coimbra Business School (ISCAC), Polytechnic University of Coimbra, Coimbra, Portugal, CEOS.PP Coimbra, Polytechnic University of Coimbra, Coimbra, Portugal and NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal
IRENE PIMENTA RODRIGUES, Department of Informatics, University of Évora, Évora, Portugal and NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal

To explore information on the semantic web, SPARQL queries or DL-queries are suitable tools. However, users interested in exploring the content of such knowledge bases often find it challenging to employ formal query languages, as this requires familiarity with the target domain's representation model. To address these challenges, a question-answering system that automatically translates natural language questions into SPARQL queries, over the Smithsonian American Art Museum CIDOC-CRM representation is presented. The proposed approach uses an ontology, named Query Ontology, defined to represent the natural language concepts and relations specific to the question's domain. This system's architecture uses a traditional natural language processing symbolic approach, with a pipeline of modules for the syntactic, semantic, and pragmatic analysis. An evaluation of the proposed system is presented and shows very promising results.

CCS Concepts: • **Information systems** → **Question answering**; • **Computing methodologies** → **Discourse, dialogue and pragmatics**; **Ontology engineering**;

Additional Key Words and Phrases: datasets, SPARQL queries, CIDOC-CRM representation, SAAM's knowledge base

ACM Reference format:
Davide Varagnolo, Dora Melo, and Irene Pimenta Rodrigues. 2025. Translating Natural Language Questions into CIDOC-CRM SPARQL Queries to Access Cultural Heritage Knowledge Bases. *ACM J. Comput. Cult. Herit.* 18, 2, Article 21 (April 2025), 28 pages.
https://doi.org/10.1145/3715156

All authors contributed equally to this research.
This work is supported by NOVA LINCS ref. UIDB/04516/2020 (https://doi.org/10.54499/UIDB/04516/2020) and ref. UIDP/04516/2020 (https://doi.org/10.54499/UIDP/04516/2020) with the financial support of FCT.IP.
Authors' Contact Information: Davide Varagnolo, NOVA LINCS, Department of Informatics, University of Évora, Évora, Portugal and NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal; e-mail: d.varagnolo@studenti.unipi.it; Dora Melo (corresponding author), Coimbra Business School (ISCAC), Polytechnic University of Coimbra, Coimbra, Portugal, CEOS.PP Coimbra, Polytechnic University of Coimbra, Coimbra, Portugal and NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal; e-mail: dmelo@iscac.pt; Irene Pimenta Rodrigues, Department of Informatics, University of Évora, Évora, Portugal and NOVA Laboratory for Computer Science and Informatics (NOVA LINCS), Caparica, Portugal; e-mail: ipr@uevora.pt.

This work is licensed under Creative Commons Attribution International 4.0.

© 2025 Copyright held by the owner/author(s).
ACM 1556-4711/2025/4-ART21
https://doi.org/10.1145/3715156

1 Introduction

The representation of cultural heritage information in an ontology, such as **International Committee for Documentation Conceptual Reference Model (CIDOC-CRM)**, which was developed for museums by the CIDOC of the International Council of Museums [11, 18], enables new searches using dedicated interfaces or using a specific query language, such as SPARQL. The CIDOC-CRM Ontology, an OWL2 ontology,[1] is also used to represent other domains, such as national archives information [19, 20, 35].

To explore information on the semantic web, such as CIDOC-CRM or DBPedia knowledge bases, SPARQL queries or DL-Queries are adequate tools to use. Users interested in exploring the content of such knowledge bases find it difficult to use a formal query language such as SPARQL. To ease this task, platforms like Sparnatural[2] [7] and RDF Explorer[3] [36, 37] were developed allowing users to explore an RDF Knowledge Graph by building SPARQL queries intuitively. However, exploring RDF Knowledge Graphs still requires users to understand the underlying information representation model and the ontology classes and properties used to structure it. Therefore, to improve user interaction with a knowledge base, such as those built on CIDOC-CRM, an automatic translation system that converts natural language questions into SPARQL queries is proposed. As an application domain of the proposed system, the **Smithsonian American Art Museum (SAAM)**[4] knowledge base is used. The information about the digital collection of artefacts and artists in the American Art Museum is represented using the CIDOC-CRM ontology.[5] Answers to natural language questions within SAAM domain can be obtained by executing the SPARQL query generated by the system on the SAAM's SPARQL endpoint.[6]

The proposed system follows a traditional natural language processing symbolic approach, namely syntactic analysis, semantic analysis, and pragmatic interpretation. The syntactic analysis uses a dependency parser to analyze the natural language question, Stanza [25]. The semantic analysis uses a simplified **Discourse Representation Structure (DRS)** [13], the partial DRS. Pragmatic interpretation is achieved by mapping the partial DRS to the Query Ontology as a **Constraint Satisfaction Problem (CSP)**. The Query Ontology is defined to represent natural language concepts and relations specific to the question's domain. Additionally, class and property annotations were added to represent specific vocabulary information and syntactic role preferences. Semantic web rules were incorporated to evaluate the adequacy of the question representation. This strategy enables multiple interpretations for each question. Selecting the best interpretation is addressed as a multi-objective problem, considering lexical, syntactic, and semantic information to evaluate potential solutions.

The proposed system can be integrated into a user interface for natural language question-answering within the SAAM domain. This interface can be designed to accept natural language questions and display output as a SPARQL query, SPARQL query results, or a natural language answer derived from the SPARQL query results. In addition, the user interface could present the intermediate steps, such as the syntactic parse, the partial DRS, and the Query-Ontology solution, helping the users to understand the natural language question system interpretation.

A set of 50 SAAM-domain questions was used to evaluate the proposed system's performance. This evaluation helped identify and resolve issues in syntactic and semantic interpretation of natural language questions, while also highlighting the approach's potential. Due to the complexity of artefact names, a simple gazetteer-based **Named Entity Recognition (NER)** system was developed. Additionally, a 5,000-question dataset, based on 10 different template types proposed in [10], was used to assess NER performance and the system's ability to recognize names without using NER. Results indicate the effectiveness of using NER, achieving 100% precision compared to approximately 98% precision for artist names and 50% for artefact names without NER.

[1] https://www.w3.org/TR/owl2-rdf-based-semantics/
[2] https://sparnatural.eu/
[3] https://www.rdfexplorer.org/
[4] https://americanart.si.edu/
[5] https://triplydb.com/smithsonian/american-art-museum
[6] https://triplydb.com/smithsonian/american-art-museum/sparql

The remainder of this article is organized as follows. Section 2 presents an overview of relevant related work. Section 3 presents the proposed approach architecture on how a natural language question is transformed into a SPARQL query representation. The process of transforming the natural language input question into its DRSs variants, based on syntactic analysis and dependencies tree, is explained in Section 4. The mapping between the partial DRS and the Query Ontology is explained in Section 5. The Query Ontology, which serves as a middle layer to adequately interpret the vocabulary used in the input question and the knowledge base, is detailed in Section 6. Afterward, Section 7 explains the methodology used to choose the best semantic interpretation solution. In Section 8, the transformation process of the semantic interpretation solution, as a Query Ontology representation, into the corresponding SPARQL query is presented. The evaluation of the proposed question-answering system is detailed in Section 9. Finally, in Section 10, conclusions and future work are drawn.

2 Related Work

The SAAM,[7] as the inaugural collection of American art in the United States, houses one of the world's most extensive and diverse compilations of American art. Spanning from the colonial era to the present, its collection unravels America's intricate artistic and cultural narrative. With over 7,000 represented artists, the museum takes a pioneering role in identifying and acquiring significant facets of American visual culture. This encompasses photography, modern folk and self-taught art, African American art, Latino art, and even video games. The museum boasts the largest collection of New Deal art, alongside remarkable holdings of contemporary craft, American impressionist paintings, and masterpieces from the Gilded Age. Additionally, SAAM maintains six comprehensive online research databases, encompassing over half a million records. Among these databases is the Inventories of American Painting and Sculpture, cataloguing more than 400,000 artworks found in public and private collections worldwide.

Since 2014, the SAAM has consistently released information on the collections as **Linked Open Data (LOD)** [32, 33], aiming to enhance the discoverability of its collections by sharing metadata in a machine-readable linked data format. By offering this metadata under an open licence, the SAAM's goal is to facilitate greater accessibility and encourage creative utilization of the available artwork information in various applications, including search and other innovative uses. Additionally, SAAM offers an online platform[8] enabling users to access and query the SAAM dataset. This platform provides various tools, including a dynamic interface browser, elastic search, and a SPARQL endpoint, enhancing the flexibility and efficiency of exploring the dataset. The SAAM has employed the CIDOC-CRM to model the concepts and relationships that exist within its collection [6].

CIDOC-CRM provides a standardized framework for defining and structuring the concepts and relationships fundamental to documenting cultural heritage. The CIDOC-CRM, recognized as the most extensive cultural heritage ontology, consists of 81 entity types (classes), or concepts, that can be connected through 160 relationships (properties).[9] By employing these classes and properties, the CIDOC-CRM offers a structured framework for describing the complex relationships between objects and the events shaping their histories, such as creation, modification, acquisition, and exhibition. The CIDOC-CRM model is event-based, while museum data is object-based, i.e., consisting of discrete fields of information, such as artist, title, and date. The SAAM information model utilizes only 19 classes and 23 properties from an older version of CIDOC-CRM. Some of these classes and properties were dropped in the latest versions, such as the class `E82 Actor Appellation`. A CIDOC-CRM modelation must comply with the main principles of the CIDOC-CRM model[10] [11, 34]. Such modulation can

[7] https://americanart.si.edu/
[8] https://triplydb.com/smithsonian/american-art-museum
[9] CIDOC-CRM version 7.1.3, February 2024. https://cidoc-crm.org/html/cidoc_crm_v7.1.3.html
[10] CIDOC-CRM version 7 and its RDF Schema expression.

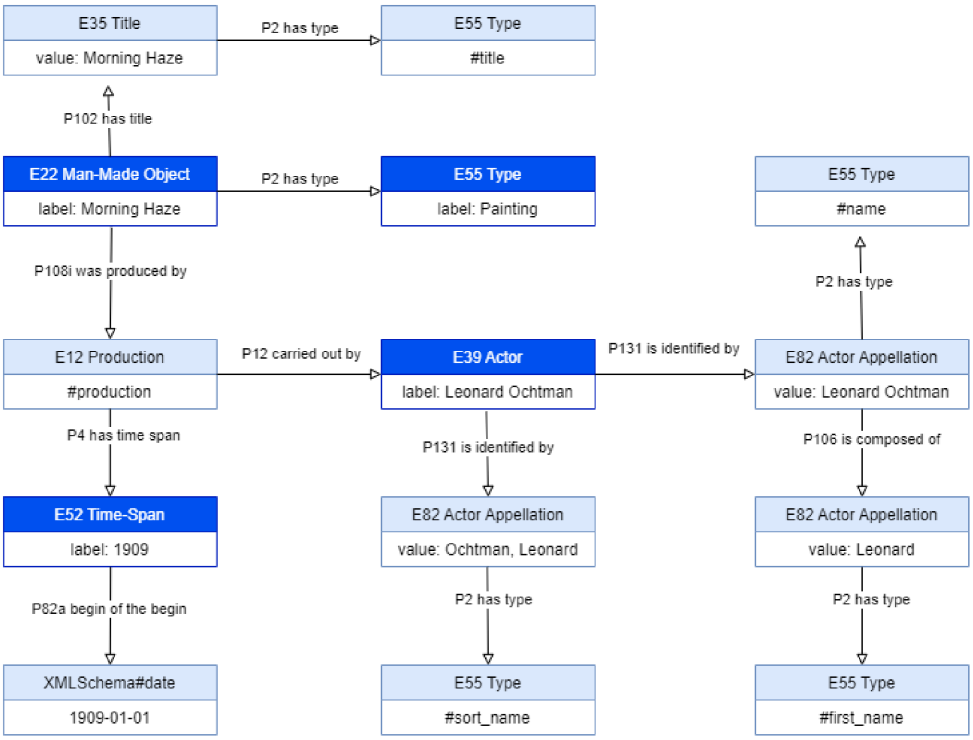

Fig. 1. SAAM representation of the information in the sentence "The Morning Haze was painted by Leonard Ochtman in 1909."

be complex and difficult for common users to understand and apply in the definition of SPARQL queries. For instance, consider the sentence

Sentence 1. *"The Morning Haze was painted by Leonard Ochtman in 1909."*

The information conveyed by this sentence within SAAM model is shown in Figure 1. The artefact "Morning Haze" is represented by an instance of the class `E22 Man-Made Object`, and the author "Leonard Ochtman" is represented as an instance of the class `E39 Actor`. A production event, represented by the class `E12 Production`, links the artefact and the author, indicating that the author created the artefact. The artefact is connected to the production event through the property `P108i was produced by`, while the author is linked to it through the property `P108i was produced by`. The creation date of the artefact is also associated with the production event. This date is represented as a time-span, using the class `E52 Time-Span`, and connected to the production event via the property `P4 has time span`. Finally, the type of artefact is represented using the class `E55 Type`. Additionally, author names are represented in more detail, using the class `E82 Actor Appellation` and `E55 Type`.

To search for information within the SAAM knowledge base, SAAM SPARQL endpoint can be utilized. SPARQL[11] is the standard query language and protocol for LOD on the web and for RDF triplestores that enables users to query information from knowledge bases mapped to RDF, such as OWL knowledge bases. Querying such knowledge bases using SPARQL queries is difficult and complex, even for experts. In addition to the syntax and

[11] https://www.w3.org/TR/rdf-sparql-query/

semantics of the SPARQL language, it is also necessary to know the information representation model of the knowledge base. Developing a system capable of translating natural language questions into SPARQL queries can significantly expand access to the SAAM knowledge base. This would benefit a wide range of users, including those unfamiliar with the SAAM CIDOC-CRM model or the SPARQL language.

To address these challenges, there has been a significant increase in researchers' focus on automating the translation of natural language questions into SPARQL queries in the past decade. Various approaches have been introduced, such as the work presented in [3, 4, 28, 30, 41], where the translation process is divided into distinct, independent layers. One layer classifies the question type (single-factual, etc.), another constructs a syntactic representation of the natural language question, a third applies rule-based techniques to generate query patterns, and the final layer maps the pattern contents to target ontology instance classes and properties. Some of these systems [3, 4] compute multiple SPARQL queries and employ a ranking methodology to select the best answer. These systems are typically designed to target ontologies like DBPedia or Yago, which have simpler representations compared to the CIDOC-CRM ontology. In an event-centric ontology like CIDOC-CRM, directly mapping patterns to the ontology may not be straightforward and often requires the creation of intermediate class instances to represent the underlying CIDOC-CRM events implied by the natural language questions. For example, the sentence 1 would require the introduction of a `E12 Production` instance to connect the author and artefact.

In recent years, there has been a notable rise in the adoption of machine learning techniques for addressing these challenges. In [2, 14, 22, 27, 40], language models are employed to identify language features within input queries. To effectively generate complex SPARQL queries, these models require extensive annotated datasets to learn intricate query patterns. While datasets are available for DBPedia that correlate natural language questions with SPARQL queries for specific languages, a substantial gap persists in dataset availability for various ontology domains and languages, impeding the training and optimization of large language models. The radius-based question-answering pipeline system [10] consists of several steps: first, natural language named entities are identified and mapped to corresponding ontology entities. Then, subgraphs with a radius of 1–4 are generated for each recognized entity. These subgraphs are subsequently converted into text, before the Roberta system [16] is employed to extract the answer. The system's performance is evaluated on a single-entity factoid question dataset, built by the authors, using the SAAM CIDOC-CRM ontology domain. This dataset is also employed to assess the performance of the proposed approach in the current paper.

Ontologies have been proposed as a method for interpreting questions within the context of knowledge bases, as seen in [1, 12, 17, 21, 31, 39]. This approach is adopted in the present work, where a processing pipeline, featuring a natural language understanding module, and an ontology for domain-specific knowledge is employed to develop components for a generic question-answering system over OWL knowledge bases. Specifically, to evaluate the performance of the proposed approach, the focus is on the SAAMs dataset, based on a CIDOC-CRM OWL2 representation. Employing an ontology as an intermediate layer to represent the semantics of natural language domain questions facilitates a clear definition of mapping rules to the target ontology. Additionally, it permits the establishment of criteria to prioritize the final SPARQL query translation. These criteria can be based on syntactic, semantic, and domain-specific information.

The natural language processing module utilizes a cutting-edge statistical English parser, namely the Universal Dependencies parser—Stanza,[12] and the semantic representation of the question takes the form of a simplified DRS [8, 13]. In [15], a Universal Discourse Representation Structure Parsing utilizing a Transformer architecture is introduced. However, this neural network-based tool is currently unavailable for use. Alternatively, other noteworthy semantic parsers are presented in [9, 26, 42].

[12]Stanza library https://stanfordnlp.github.io/stanza/

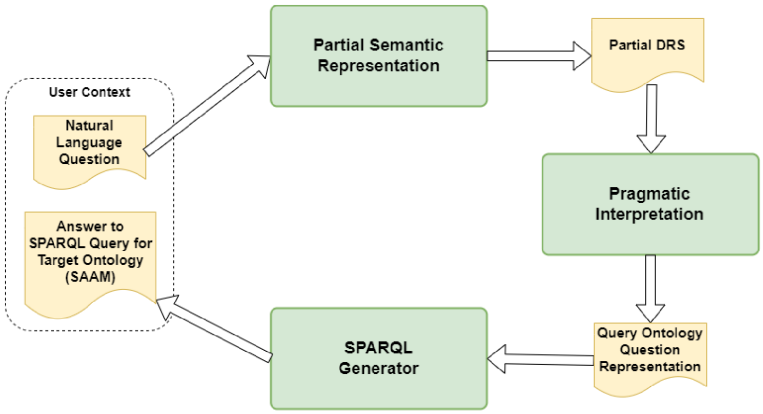

Fig. 2. Question answering architecture.

3 Question-Answering Architecture

The approach underlying the transformation of a natural language question into a SPARQL query representation involves applying several transformation steps to the representation. These steps begin with the input natural language question and end with the corresponding output SPARQL query representation. The transformation process passes through intermediate syntactic and semantic representations, as well as a matching query ontology representation. For this purpose, the proposed methodology comprises three main modules (see Figure 2): Partial Semantic Representation, Pragmatic Interpretation, and SPARQL Generator.

The Partial Semantic Representation module provides a new representation of the input natural language question, derived from its syntactic, semantic, and universal dependency analysis.

The transformation of the input natural language question into the new representation is performed in two steps. First, a dependency parser, namely Stanza,[13] is applied to the question. The Stanza parser analyzes the linguistic structure of the question and generates a parser tree. Then, this parser tree is transformed into a set of partial DRSs using the DRS process.

The DRS process, which is explained with further details in Section 4.1, converts the question parser tree into a representation that captures the meaning of the question. This is accomplished by organizing the information in the parser tree into a structured format that represents the discourse relations and the logical structure of the question.

It is important to note that a single syntactic analysis can sometimes lead to multiple DRSs. This can occur due to linguistic phenomena such as pp-attachment, where different syntactic analyses can result in different interpretations of the question. The Stanza parser selects the "best" analysis based on the corpus it was trained on. However, it is possible to recover other analyses during the partial semantic analysis if they are relevant.

The Partial Semantic Representation module is designed to be language-independent and domain-independent. The Stanza parser is available for many languages, allowing the module to analyze questions in multiple languages. Additionally, the DRS process uses Universal Dependencies Tags, which are defined uniformly for all languages, ensuring consistency in the analysis across different languages.

[13] https://stanfordnlp.github.io/stanza/

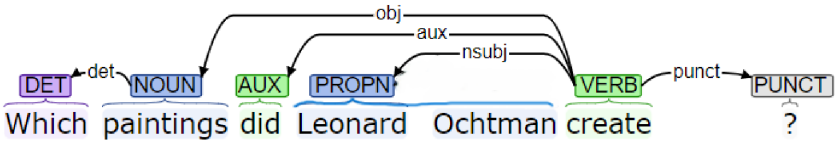

Fig. 3. Dependency tree for the question "Which paintings did Leonard Ochtman create?"

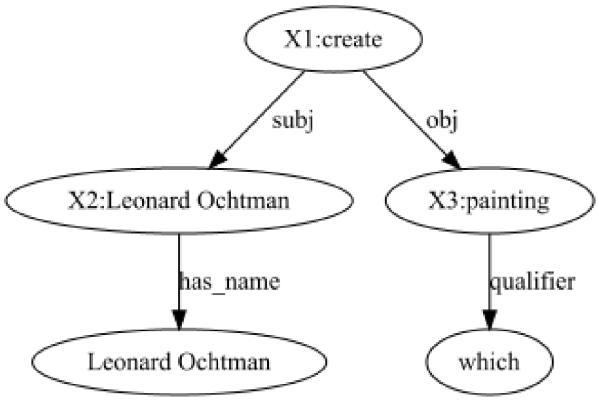

Fig. 4. DRS generated from the dependency tree.

As an illustrative example, consider the following question.

Question 1. *"Which paintings did Leonard Ochtman create?"*

The dependency tree of the illustrative Question 1 is shown in Figure 3 and, based on this dependency tree, the Partial Semantic Representation module generates a DRS, as shown in Figure 4.

After the Partial Semantic Representations of the question are obtained, the Pragmatic Interpretation module aims to rewrite those representations into a set of potential alternative meanings for the input question within the specific domain context. See Section 5.2 for further details. This process employs an ontology-based domain representation, the Query Ontology (detailed in Section 6), in conjunction with a multi-objective optimization approach to determine the best interpretation of the question within the domain ontology's context. Query Ontology is defined to represent natural language concepts specifically related to the target domain, encompassing only those subjects that can be queried about the information contained within the target knowledge base. Further details are presented in Section 7. Moving on to the illustrative question example, Figure 5 presents the solution generated by this module.

Finally, a SPARQL Query Builder is applied to the Semantic Query Representation solution, generating the corresponding SPARQL query representation. Further details are provided in Section 8. The SPARQL query 1 corresponds to the representation of the illustrative Question 1 obtained by this builder. Using the available SAAM SPARQL endpoint, the presented SPARQL query facilitates the search for the answers to the initial natural language question within the provided CIDOC-CRM Ontology Population of the Art Collection.

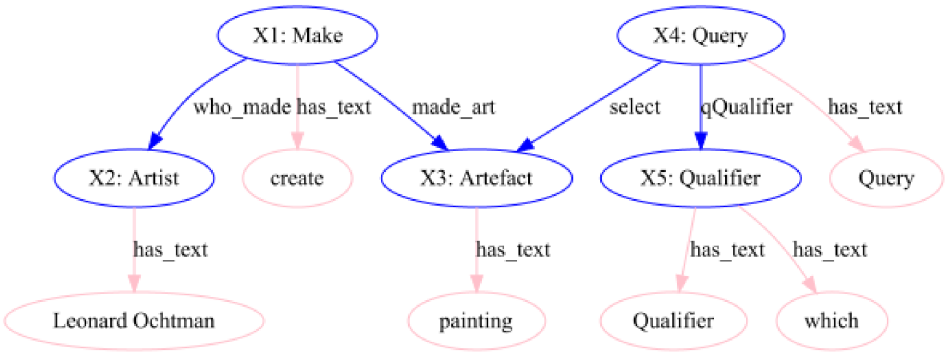

Fig. 5. Solution of the question "Which paintings did Leonard Ochtman create?"

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX cidoc: <http://www.cidoc-crm.org/cidoc-crm/>

SELECT  DISTINCT  ?snameArtefact3 ?Artefact3 ?snameArtist2 ?Artist2
WHERE {
    ?Artefact3 cidoc:P108i_was_produced_by ?Make1 .
    ?Artefact3 rdfs:label ?snameArtefact3 .
    ?Artefact3 cidoc:P2_has_type ?type .
    ?type rdfs:label ?ltype .
    Filter(REGEX(?ltype, "painting","i")) .
    ?Make1 cidoc:P14_carried_out_by ?Artist2 .
    ?Artist2 rdfs:label ?snameArtist2 .
    {{?Artist2 cidoc:P131_is_identified_by ?nameArtist2 .
    ?nameArtist2 rdf:value ?snameArtist2 .}
    UNION
    {?Artist2 rdfs:label ?snameArtist2 .}}
    filter(regex(?snameArtist2,"Leonard Ochtman","i")) .
}
```

Listing 1. SPARQL Query representation of the question "Which paintings did Leonard Ochtman create?"

4 Partial Semantic Representation

The Partial Semantic Representation module, shown in Figure 6, comprises an optional NER module, a syntactic analysis module, and a partial semantic analysis module.

Artwork titles (names) can be challenging for syntactic parsers to identify as proper nouns due to their potential to form coherent syntactic units. For example, the artwork title "Expulsion from the Garden" might be analyzed as a noun ("Expulsion") modified by a prepositional phrase ("from the Garden"). Given the finite and readily available nature of Artwork titles and Author names, a Gazetteer containing these names can be constructed. This Gazetteer could be employed in a straightforward NER process to identify and replace name occurrences with predefined tokens that are easily recognized as proper nouns by the syntactic analyzer. After the syntactic analysis, the original names are restored in the final sentence syntactic representation.

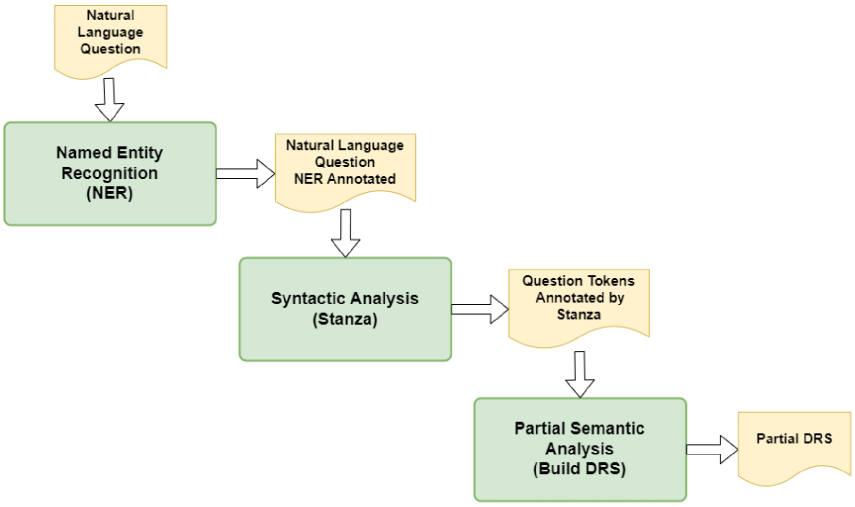

Fig. 6. Partial semantic representation architecture.

In the illustrative example Question 1, the NER module identifies the occurrence of "Leonard Ochtman" within the Gazetteer and replaces it with the predefined token "John," as presented in 2. This modified question is subsequently processed by the Syntactic Analysis module.

Question 2. *"Which paintings did John create?"*

```
[{"id": 1, "text": "Which", "lemma": "which", "upos": "DET",
            "feats": "PronType=Int", "head": 2, "deprel": "det"},
{"id": 2, "text": "paintings", "lemma": "painting", "upos": "NOUN",
            "feats": "Number=Plur", "head": 5, "deprel": "obj"},
{"id": 3, "text": "did", "lemma": "do", "upos": "AUX",
            "feats": "Mood=Ind|Number=Plur|Person=3|Tense=Past|VerbForm=Fin",
            "head": 5, "deprel": "aux", "misc": null},
{"id": 4, "text": "Leonard Ochtman", "lemma": "Leonard Ochtman", "upos": "PROPN",
            "head": 5, "deprel": "nsubj", "misc": null},
{"id": 5, "text": "create", "lemma": "create", "upos": "VERB",
            "feats": "VerbForm=Inf", "head": 0, "deprel": "root", "misc": null},
{"id": 6, "text": "?", "lemma": "?", "upos": "PUNCT",  "head": 5,
            "deprel": "punct", "misc": null}]
```

Listing 2. List of tokens annotated by Stanza of the question "Which paintings did Leonard Ochtman create?"

The syntactic analysis is done by Stanza [24], a Universal Dependencies Parser.[14]

The output of the Stanza analysis, after substituting the placeholder "John" with the original name, is visualized in Figure 3. The corresponding list of Stanza-annotated tokens is provided in Listing 2.

The list of annotated tokens is the input of the module Partial Semantic Analysis. This module builds a DRS with an algorithm based on Discourse Representation Theory [13] adapted to dependencies parser. Currently, it is restricted to questions (sentences) with determiners that give rise to existentially quantified variables, and conditionals and other complex discourse phenomena [29] are not considered.

[14]Stanza library (https://stanfordnlp.github.io/stanza/) is a Python natural language analysis package, which contains a collection of tools for the linguistic analysis of many human languages. Stanza includes a multilingual dependency parsing module that builds a tree structure from natural language sentence words which represents the syntactic dependency relations between words.

4.1 DRS Builder from Dependency Parser

A DRS is a set of discourse referents and a set of conditions, which are relations on the discourse referents [13]. The resulting DRSs are called Partial DRS since the conditions on discourse referents are not semantically interpreted. For instance, a noun that is an action agent remains as the subject of a verb, in the partial DRS.

A partial DRS is obtained by

—A noun gives rise to a new discourse referent.
—A verb also gives rise to a new discourse referent representing the event or action.
—The dependency annotations subject, object or indirect object of a verb, and verb modifiers, such as propositional phrases, adjectives and adverbs, define the names of the conditions between discourse referents.

The information associated with each discourse referent includes a variable, a lemma, and a determiner (maybe null). A condition is defined as having as name the syntactic role, and as discourse referents the token and the head token discourse referents variables or, in the case of verb modifiers, the name is composed of the lemma (ex: oblique) and the preposition lemma. In the case of noun modifiers (`nmod`) the condition name is the preposition lemma. Figure 4 presents the partial DRS of the list of tokens listed in 2.

According to the partial DRS definition:

(1) `token id=2, lemma="painting", upos= "NOUN", head: 5, deprel= "obj"`
New discourse referent: `X3 - painting - which`
New conditions: `qualifier(X3, "Which")`, `obj(X1,X3)`
Token `id=1` is consumed.
(2) `id= 4, lemma= "Leonard Ochtman", "upos": "PROPN", "head": 5, deprel: "nsubj"`
New discourse referent: `X2 - Leonard Ochtman - null`
New conditions: `subj(X1,X2)`, `has_name(X1,"Leonard Ochtman")`
(3) `id= 5, lemma="create", upos= "VERB", head= 0, deprel="root"`
New discourse referent: `X1 - create - null`
New conditions: none
Tokens `id=3` and `id=5` are consumed.

Since Stanza generates only one sentence parsing and some sentences may have other possible analyses, in the partial DRS builder, other partial DRSs can be obtained by considering that: the existence of nominal obliques (`obl`), nominal modifiers (`nmod`), appositional modifiers (`appos`), and propositions such as `of`, can have their head moved to another sentence token, the head of the current head (`pp-attachment` movement).

The semantic interpretation of a partial DRS is the attribution of an Ontology Class to each discourse referent and the attribution of an Ontology Property, an object property or a data property, to each condition. This interpretation, the Pragmatic Interpretation, is done by mapping the partial DRS on the Query Ontology.

5 Pragmatic Interpretation

The Pragmatic Interpretation module is presented in Figure 7. The input of this module is a sentence's partial DRS that the CSP solver component transforms into a set of new partial DRSs, where a class of the Query Ontology is coherently assigned to each discourse referent, see Section 5.1. Then, this set of DRSs is transformed into a new set of DRSs, where, in each DRS, each condition is replaced by a coherent ontology property by the Ontology Content-Matching component. The resulting DRSs are full DRSs, each one a semantic representation of the initial sentence. Since a sentence can have a large number of possible semantic representations, the two components objective function calculus and multi-objective problem solving are responsible for choosing the best semantic representation of the question sentence.

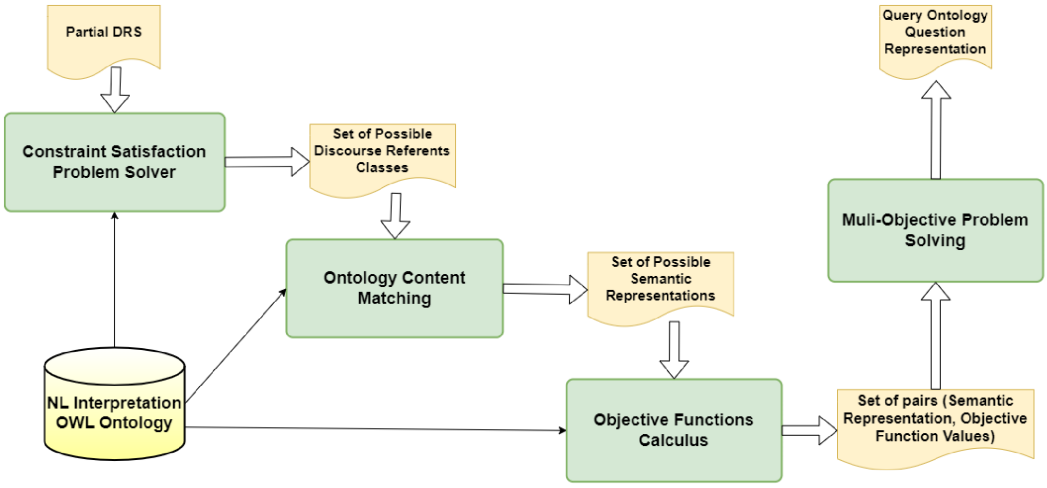

Fig. 7. Pragmatic interpretation architecture.

5.1 CSP Solver

To assign a coherent Ontology class to each partial DRS discourse referent as a CSP, the problem is defined by

(1) the set of variables $X = \{X_1, X_2, ..., X_n\}$, each X_i is a discourse referent;
(2) the set of domain variable values $D = \{D_1, D_2, ..., D_n\}$, each D_i is an ontology class;
(3) the set of constraints $C = (C_{1_1} \vee C_{1_2} \vee ...) \wedge ... \wedge (C_{m_1} \vee C_{m_2} \vee ..)$, established by the m conditions in the partial representation and the object properties in the ontology.
For each condition C_i from discourse referent X_j to discourse referent X_k in the DRS, the conjunction of the following restrictions is added:
For each object property, n, in the ontology with domain D_l and range D_m, the disjunction of the constraint $C_{i_n} = (X_j = D_l \wedge X_k = D_m)$ is added.

A solution is an evaluation that is consistent and that is complete. Consistency is verified when the solution does not violate any of the constraints and completeness is verified when the solution includes all the variables. Such evaluation is said to solve the CSP.

The Java package `CP-SAT Solver`[15] is used to represent and solve the CSP.

Considering N, the number of Query Ontology classes, and D, the number of discourse referents in a question's partial DRS, the time complexity of solving the CSP is $O(N^D))$. The maximum number of potential solutions is bounded by N^D.

For illustrative example Question 1, the number of solutions generated by applying this module is 559, which is lower than the maximum of $3^21 = 9{,}261$ for this case. Here, 21 represents the number of Query Ontology classes and 3 the number of discourse referents in the partial DRS (Figure 4).

To complete the Query Ontology solutions, the next step consists of assigning an ontology object property, or data property, to each DRS condition.

5.2 Ontology Content-Matching

The input of the Ontology Content-Matching component is a set of DRSs, where each discourse referent has an ontology class assigned. In each DRS, each two argument condition $C_k = (X_i, P_m, X_j)$ is assigned to a coherent

[15] https://developers.google.com/optimization/cp

object property of the ontology, taking into account that the class of X_i is the domain class and the class of X_j is the range class of the object property. When there is more than one ontology property with the same domain and range, the DRS is duplicated with the property variants. The process for interpreting the one argument conditions will assign one ontology data property to the condition constrained by the class of the discourse referent of the condition.

The output of this module is a set of DRS, each one is an ontology representation of the question. The best ontology question representation is chosen in the next modules of the Pragmatic Interpretation. The Ontology Content-Matching module has a linear time complexity based on the number of conditions in the DRS, since the number of object properties in the ontology is constant.

6 Query Ontology

The Query Ontology is designed to enable the representation of the natural language concepts conveyed by the questions used to query the SAAM knowledge base.

As mentioned, the SAAM knowledge base has information on artworks and artists, namely the authorship of the artworks. SAAM also has some information associated with artists such as their birth date, death place, nationality, or biography. And, it associates information to artworks such as the type (painting, sculpture, etc.), the material used, the dimension, how was obtained, and so on. In Figure 8 the classes of the Query Ontology are displayed. These classes represent the concepts present in the users' natural language questions: Action, Date, Place, Object, and Concept. The classes `Query` and `Qualifier` are used to represent the question structure, namely the question focus.

The class `Object` has the sub-classes `Artist` and `Artefact`, which are used to represent the concepts conveyed by noun phrases in natural language. The class `Action` has the sub-classes `Make` and `Give`, which are actions usually conveyed by verb phrases.

Figure 5 presents the final semantic representation of the illustrative example, Question 1. In this Figure, the discourse referent `X1`, representing the verb create, is assigned to the class `Make`; the discourse referent `X2`, representing the proper name "`Leonard Ochtman`," is assigned to class `Artist`; and the discourse referent `X3`, representing the noun "`paintings`," is assigned to class `Artefact`. The discourse referents `X4` and `X5` are added to define the sentence as a question.

Since some of the classes in the Query Ontology, such as `Object`, are defined to organize the knowledge domain and will never be used to be assigned to a discourse referent, the class `LanguageModel` was introduced to model the classes that can be used to interpret the class of a discourse referent. Figure 9 presents the Query Ontology with the class `LanguageModel`.

Query Ontology object properties are links between classes and should reflect the meaning of relationships between natural language expressions. For instance, there must be a link between `Artist` and `Artefact` due to the natural language expression "`[A] author of [B]`." Since there is no inverse expression in English to such meaning, the inverse link (property) should not be considered in the Query Ontology. In Table 1, the object properties related to the class `Artist` are presented. This table displays the object property name, its class domain, and class range, and an example of a natural language question where the property must be used.

The object property `birthdate_of` is used to link a special date, `Birthdate`, to an Artist. The use of this property reflects the fact that in English it is not possible to link a general date to an `Artist`. The expression "`the year of Leonard Ochtman`" is awkward and requires a lot of context to interpret it as referring to the date of birth. In the SAAM's domain, the expression will be ambiguous and the Query Ontology does not represent it. The `BirthDate` is a class that represents "the date of the birth." The classes `BirthPlace`, `DeathDate`, and `DeathPlace` and the object properties `birthplace_of`, `deathdate_of`, and `deathplace_of` are defined due to the same phenomena as in `BirthDate` and `birthdate_of`.

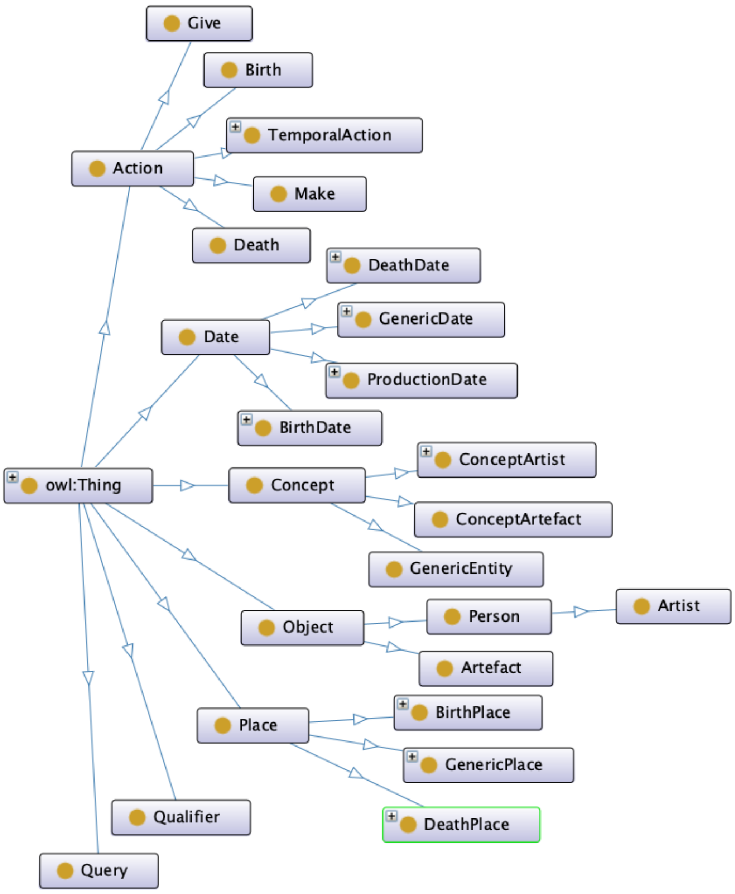

Fig. 8. Query Ontology classes.

Table 1. Query Ontology Object Properties Related to the Class `Artist`

Object-Property Name	Domain	Range	Use
author_of	Artist	Artefact	Who is the author of Morning Haze?
birthdate_of	BirthDate	Artist	What was the birthdate of Leonard Ochtman?
birthplace_of	BirthPlace	Artist	What was the birthplace of Leonard Ochtman?
deathdate_of	DeathDate	Artist	What was the deathdate of Leonard Ochtman?
deathplace_of	DeathPlace	Artist	Where was the deathplace of Leonard Ochtman?
who_made	Make	Artist	Who painted the Morning Haze?
who_born	Born	Artist	Who was born in 1854?
who_die	Die	Artist	When did Leonard Ochtman die?
conc_of_Artist	ConceptArtist	Artist	What was the nationality of Leonard Ochtman?

The object property `who_make` links an `Action Make` to an `Artist`. This property represents that the artist is the agent of the action to make. The properties `who_born` and `who_die` link an `Action`, `Born`, or `Die` to an `Artist`, reflecting that an artist is the patient of the action in each of the properties.

The object property `conc_of_Artist` links a `ConceptArtist` to an `Artist`. The class `ConceptArtist` is used to represent artist properties (attributes) within SAAM knowledge base, which are conveyed by noun phrases

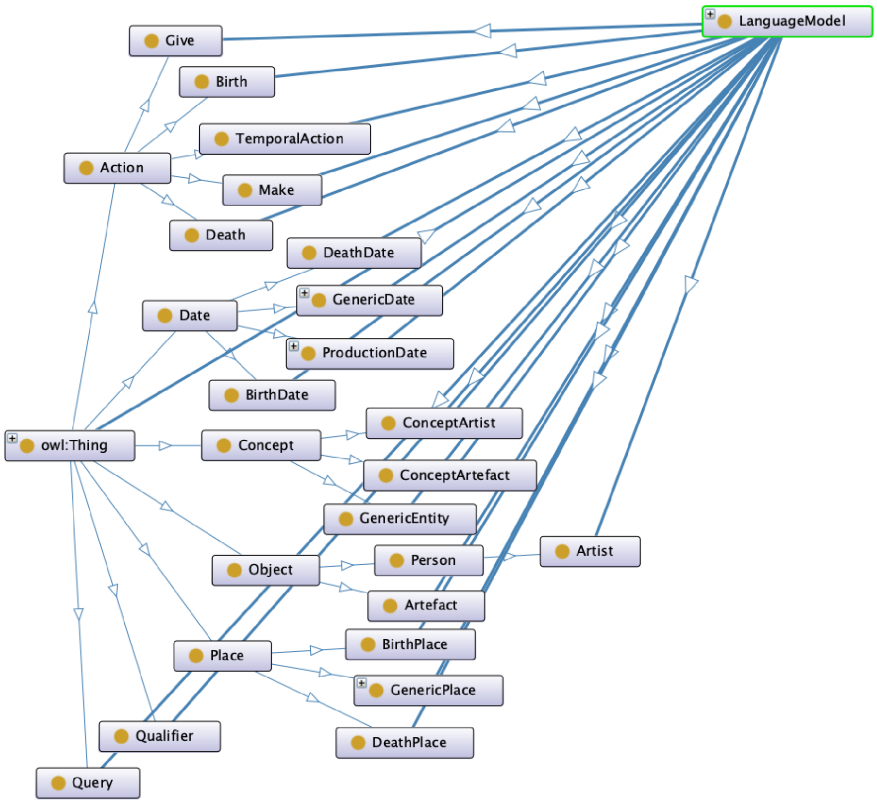

Fig. 9. Query Ontology classes in the language model.

Table 2. Query Ontology Object Properties Related to the Class `Artefact`

Object-Property Name	Domain	Range	Use
author_of	Artist	Artefact	What was the birthplace of the creator of the Morning Haze?
made_art	Make	Artefact	Who painted the Morning Haze?
give_art	Give	Artefact	Who donated the Morning Haze?
production_of_Artefact	Production	Artefact	When was the production of Morning Haze completed?
timeperiodArtefact	TemporalAction	Artefact	When was the Morning Haze completed?
conc_of_Artefact	ConceptArtefact	Artefact	What is the size of the Morning Haze?

such as "the biography" and "the nationality." To avoid the proliferation of classes, this class represents more than one concept of the SAAM knowledge base.

Regarding the concept `Artfact`, Table 2 displays its related Query Ontology Object Properties.

The object property `author_of` links an artist to an artefact, representing the fact that the `Artist` was the author of the `Artefact`. The object properties `made_art` and `give_art` link the `Action`, `Make` or `Give`, to an `Artefact` reflecting that the artefact is the patient of the action. The property `production_of_Artefact` links a `Production` to an `Artefact`. A production in SAAM knowledge base is the same as the `Action Make`, but this class `Production` is defined in the Query Ontology to facilitate the recognition of the nominalization of the action `Make` in particular to represent the noun modifiers (e.g., production of [A]). The `timeperiodArtefact` object property links a `TemporalAction` to an `Artefact`. The class `TemporalAction` represents actions such

Table 3. Query Ontology Object Properties Related to the Class Action

Object-Property Name	Domain	Range	Use
action_date	Action	Date	When was the Morning Haze created?
action_place	Action	Place	Who was born in Zonnemaire?

Table 4. Query Ontology Object Properties Related to the Sub-Classes of Action

Object-Property Name	Domain	Range	Use
who_die	Die	Artist	When died Leonard Ochtman?
who_born	Born	Artist	Who was born in 1854?
give_art	Give	Artefact	Who gave the Morning Haze?
give_to	Give	GenericEntity	Who gave the Honeymoon Motel to the museum?
who_give	Give	GenericEntity	Who gave the Morning Haze?
made_art	Make	Artefact	Who painted the Morning Haze?
who_made	Make	Artist	Who painted the Morning Haze?
makeConc	Make	ConceptArtefact	What are the sculptures that are made of lead?
timeperiodArtefact	TemporalAction	Artefact	When did Morning Haze start?
timeperiodProduction	TemporalAction	Production	When did the production of Morning Haze start?

Table 5. Query Ontology Object Properties to Structure a Query

Object-Property Name	Domain	Range
Select	Query	Cselect
qQualifer	Query	Qualifier

as “to start” or “to end.” These properties enable the representation of the time limits of the production of an artefact. Finally, the conc_of_Artefact object property links a ConceptArtefact to an Artefact. The class ConceptArtefact is used to represent the artefact properties (attributes) in SAAM that are conveyed by noun phrases such as “the size” and “the material.”

The object properties that have as domain the class Action are presented in Table 3. The temporal sentences and location modifiers are linked to an Action by these object properties. These object properties can have as domain any sub-class of Action, such as Make, Born, Die, and Give.

Regarding the sub-classes of Action, Table 4 presents the object properties that have those sub-classes as domains.

The actions Die and Born have only one object property to link to the Artist, which is the patient. Those verbs have no other arguments except temporal or location modifiers. The action Give has the object property, who_give, to represent the agent, the property give_art, to represent the patient, and the property give_to to represent the recipient.

The object property timeperiodArtefact links a TemporalAction to an Artefact and the object property timeperiodProduction links the temporal action to a production, both as patients of the action.

The object properties presented in Table 5 are used to structure a Query.

The object property select links a Query to a Cselect. The class Cselect groups the classes that can be the answer to a query: Object, Date, Place, and Concept. The object property qQualifer links a Query to a Qualifier, which is the interrogative used in the question.

When a question DRS does not have an instance of Query, as in the case of the question DRS in Figure 4, a new Query instance must be created and an instance of Cselect should be linked by the object property select. To decide the adequate Cselect to be linked to Query, the best candidate is the Object that has a data property qualifier. The following semantic rule is used to correct the DRSs that do not have defined a Query.

Table 6. Query Ontology Class Annotations

Class	Sub-Class	Annotation
Birth	Action	bear; born
Death	Action	death; die
Give	Action	give; offer; obtain; donate
Make	Action	create; make; paint; author
TemporalAction	Action	begin; start; end; finish; complete; ...
ConceptArtefact	Concept	art type; material; size; type
ConceptArtist	Concept	biography; nationality
GenericEntity	Concept	museum
BirthDate	Date	birthdate; birth date
DeathPlace	Place	deathplace; death place
Artist	Person	creator; author
Artefact	Object	painting; sculpture

```
qualifier(A,Nql) -> new(Q,Query), select(Q,A),
                    new(Ql,Qualifier), has_name(Ql,Nql),
                    qQualifier(Q,Qql).
```

If the object property `qualifier` is absent, a new `Query` instance is created without the `select` and `qQualifier` properties. This occurs when a natural language question is of Boolean type.

Regarding the illustrative example, its DRS presented in Figure 5 has a `Query` and a `Qualifier` obtained as a result of the application of this rule.

7 Choosing the Best Question Interpretation

As mentioned, given a DRS, a semantic interpretation of the question in the domain-specific ontology, Query Ontology, is defined as: the assigning of an ontology class to each discourse referent; the assigning of an ontology object property to each condition; and the assigning of an ontology data property to each one argument condition, such as "`has_text`," "`has_name`," "`has_value`," or "`qualifier`."

Some of these assignments are more adequate than others, which corresponds to a more adequate semantic interpretation of the question. The adequacy of a question's semantic representation can be evaluated on different dimensions such as lexical, syntactic, or semantic. The lexical evaluation measures the adequacy of the vocabulary. This can be done by measuring the correctness of the attribution of a class to a discourse referent.

Ontology Class Annotations enable the calculus of the lexical adequacy of the assignment of a class to a discourse referent. Each `ModelLanguage` Class of the Query Ontology has the appropriate vocabulary in the class annotations. In Table 6, a subset of the Query Ontology classes and their annotations is displayed.

The vocabulary associated with each class was done manually, but it can be imported automatically from domain-controlled vocabularies, or lexical databases such as WordNet [5].

In each semantic representation question, the number of classes assigned to a discourse referent, that have a lemma equal to a class annotation, are counted, `NumberClassesOK`. In Figure 5, the semantic representation displayed has `NumberClassesOK` = 2, since the class `Make` of the referent `X1` has the lemma of the referent `X1`, `create`, in its annotations; and the class `Artefact` of the referent `X3` has the lemma of `X3`, `painting`, in its annotations.

The value of `NumberClassesOk` is used to choose the best interpretations of a question. Considering that higher values of `NumberClassesOk` result in more adequate representations in terms of lexical interpretation, i.e., the vocabulary is well interpreted.

The syntactic dimension of a question representation is considered in the interpretation of the DRSs conditions, i.e., the selection of the object property to link two discourse referents. To enable the evaluation of the syntactic

Table 7. Query Ontology Object Properties Annotations

Object Property	Domain	Range	Annotation
who_made	Make	Artist	subj; by:obl; obl:pass
made_art	Make	Artist	obj; subj:pass
action_date	Action	Date	in; obl_in
action_place	Action	Place	obl_in; obl_at

Table 8. Query Ontology Data Properties

Data Property	Domain	Range
entitesOK	owl:Thing	xsd:int
entitesOK	owl:Thing	xsd:int

adequacy of a semantic representation, the Query Ontology object properties have annotations describing the conditions that have a better interpretation of the property.

Table 7 presents some examples of the object properties annotations. The annotations are established according to the Stanza dependency relations with some uniformity performed by the partial DRS Builder. The annotations on the property `who_made` indicate that the discourse referent of the object property range should be the subject of an active voice sentence (`subj`) or the object of a passive voice sentence that Stanza annotates with the dependency relation `obl:pass` or `by:obl`.

To evaluate the syntactic adequacy of a question interpretation, the annotations of the object property assigned to each DRS condition are checked for a match with the condition name. The total number of matches, `NumberPropertiesOK`, is the value used to evaluate the syntactic adequacy of the question's semantic representation.

The partial DRS, in Figure 4, with the interpretation presented in Figure 5, has `NumberPropertiesOK` = 2, since the condition "`subj(X1,X2)`" is assigned to "`who_made`" that has the "`subj`" annotation, and the condition "`obj(X1,X3)`" is assigned to "`made_art`," which has "`obj`" in its annotations.

The value of `NumberPropertiesOk` is also used to choose the best interpretations of a question. Considering that higher values of `NumberPropertiesOk` result in more adequate representations in terms of syntactic constraints, which means that the syntactic structure is better interpreted.

To evaluate the semantic adequacy of a question interpretation, data properties were also created and are presented in Table 8.

These data properties are used to evaluate each semantic interpretation. The ontology can model representation preferences by defining semantic web rule language[16] rules. The Query Ontology instances `entity_ok` and `entity_nok` are inferred by the rules constructed to define semantic preferences on the semantic representations of a question.

When multiple Query Ontology solutions exist for a given natural language question about the SAAM knowledge base, the preference between interpretations can be determined using the `entitiesOK` and `entitiesNOK` predicates.

Consider the following adequate question about SAAM knowledge base:

Question 3. *"What artworks did John Henry Brown make with watercolours?"*

In the Query Ontology interpretation of Question 3, solutions identifying "John Henry Brown" as the author of watercolor artworks should be prioritized over those that do not.

[16] https://www.w3.org/Submission/SWRL/

To state the preference of this kind of interpretation, the following two rules can be added to the Query Ontology:

```
who_made(?a,?o1), made_art(?a,?o2), make_conc(?a,?o3) -> entity_ok(?a, "2"^^xsd:int)
made_art(?a,?o1), make_conc(?a,?o2) -> entity\_ok(?a, "1"^^xsd:int)
```

For the solution of the Question 3, the sum of the values of `entity_ok` is 3.

Regarding the SAAM domain knowledge base, some interpretations should be avoided, such as assigning a place to a production of an artwork, since that information is not present in the knowledge base. The following rule uses the data property `entity_nok` to penalize such interpretation.

```
Make(?a), action_place(?a,?p) -> entity_nok(?a, "-1"^^xsd:int),entity_nok(?p, "-1"^^xsd:int)
```

This rule infers `entity_nok(?Make,-1)` and `entity_nok(?Place,-1)` when a solution links an action `Make` to a `Place`.

Now, consider the question:

Question 4. *"What materials are used in Morning Haze?"*

The verb "to use" can have the class `Make` assigned, which is the intended meaning. "Morning Haze" can have assigned either the class `Place` or `Artefact`, since both classes can be linked to `Make` by an object property, `action_place` or `made_art`, given that the preposition "in" belongs to the annotations of both object properties. The above rule enables the selection of the desired interpretation, "Morning Haze" is the name of an `Artefact`, by penalizing the interpretation where it is considered as a name of a `Place`.

For each interpretation, the sum of the values for the data properties `entity_ok` and `entity_nok` are assigned to the variables `NumberEntitiesOK` and `NumberEntitiesNOk`, respectively. These variables are used to choose the best question interpretation that is semantically more adequate.

7.1 Objective Functions Calculus

The question interpretation in the domain-specific ontology can be seen as a multi-objective optimization problem, where the weighting semantic, syntactic, and lexical rules define the set of objective functions.

The objective functions are the values of the previously introduced variables, as follows:

- —Lexical—`NumberClassesOK`
- —Syntactic—`NumberPropertiesOK`
- —Semantic—`NumberEntitiesOk` and `NumberEntitiesNOK`

For each question's semantic representation, a DRS solution, the calculation of the corresponding values of the objective functions is performed, and the maximum value of each objective function value is identified.

Consider S the set of all solutions. To obtain the set of best DRS solutions, S_{best}, the following steps are performed:

1 Calculate the maximum of all 4 measures in S
2 i=4
3 Repeat
  $S_{best} \leftarrow$ all $s \in S$ such that s has i measures equal to maximums
  i= i-1
  Until $S_{best} \neq \emptyset$
4 If # $S_{best} > 1$ and $S \neq S_{best}$
  Goto **1**

If the $\#S_{best}$ is greater than 1, one of the DRS in the set is chosen to be translated into SPARQL.

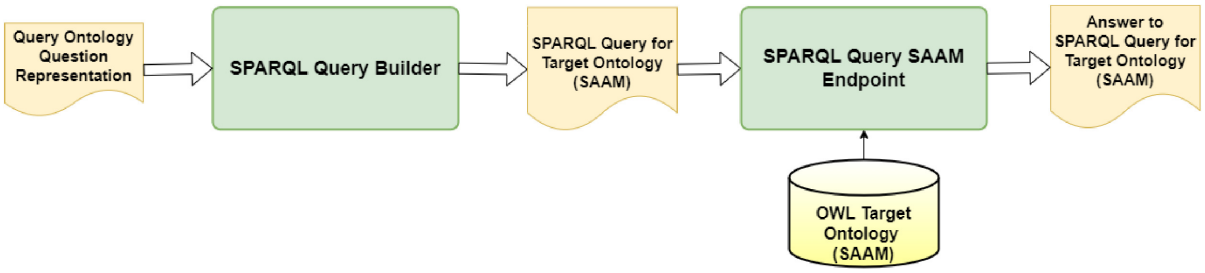

Fig. 10. SPARQL query generation architecture.

This process can be improved by giving preference to some aspects of the DRS solution, such as those that are more adequate with respect to semantic, syntactic structure and the question's vocabulary (lexical, syntactic, or semantic). It is also possible to analyze the results of the SPARQL query and exclude the DRS whose SPARQL query answers are empty.

In the Section 9, the system evaluation is done by choosing one (the first one) from the set of best solutions.

8 Generating a SPARQL Query from a Query Ontology Representation

The proposed approach aims to achieve a SPARQL Query for a given natural language question, capable of querying a target Ontology Population, specifically the SAAM knowledge base. The SPARQL Query Generation module involves transforming the specific-domain ontology solution of the natural language question into a SPARQL Query format within the context of the SAAM knowledge base, and follows the steps illustrated in Figure 10.

The generic SPARQL Query[17] formats are as follows.

```
SELECT DISTINCT <select>
WHERE {
    <body>
}
```

or

```
ASK {
        <body>
}
```

The Query Ontology solution for a natural language question is initially converted into a SPARQL query using a set of mapping description rules. These rules map elements of the Query Ontology solution to corresponding SPARQL syntax and structure within the SAAM Ontology. The type of SPARQL query (SELECT or ASK) is determined by the presence of the `select` property in the Query Ontology solution. If present, a SELECT query is generated; otherwise, an ASK query is created.

For SELECT queries, the initial variable representing the range instance of the `select` property is augmented with a variable for the instance's name, if applicable (e.g., artefacts have names but artefact types do not). To enrich the query results, additional variables are included for instances and corresponding names of all `has_name` property instances within the Query Ontology solution.

As mentioned, the SAAM Ontology population adheres to the CIDOC-CRM representation, an OWL2 Ontology. In this domain, a set of specific classes, object properties, and data properties were chosen from the CIDOC-CRM ontology to describe the attributes of the SAAM artefacts. This allows for defining an OWL2 representation of the

[17] https://www.w3.org/TR/rdf-sparql-query/

Table 9. Mapping Description Rules from Query Ontology Population into SAAM CIDOC-CRM Representation

#Rule	Property	Relation	CIDOC-CRM Triple
1	who_made	?Make1 who_made ?Artist2	?Make1 cidoc:P14_carried_out_by ?Artist2 .
2	made_art	?Make1 made_art ?Artefact3	?Artefact3 cidoc:P108i_was_produced_by ?Make1 . ?Artefact3 rdfs:label ?snameArtefact3 .
3	has_text	?Artefact3[a] has_text ?textArtefact3[b]	?Artefact3 cidoc:P2_has_type ?typeArtefact3 . ?typeArtefact3 rdfs:label ?ttypeArtefact3 . Filter(REGEX(?ttypeArtefact3, ?textArtefact3, "i")) .
4	has_name	?Artist2 has_name ?tNameArtist2	?Artist2 rdfs:label ?snameArtist2 . {{?Artist2 cidoc:P131_is_identified_by ?nameArtist2 . ?nameArtist2 rdf:value ?snameArtist2 .} UNION {?Artist2 rdfs:label ?snameArtist2 .}} filter(regex(?snameArtist2,?tNameArtist2, "i")) .

[a]?Artefact3 ∈ Artefact.
[b]?textArtefact3 ∈ {"paint," "painting," "sculpture," "collage," "decorative," "drawing," "graphic," "photography," "silhouette"}.

actual existing objects in SAAM. After thoroughly understanding such representation, a set of mapping description rules was manually defined. These rules reflect, for each individual in the Query Ontology, its representation in the SAAM CIDOC-CRM representation or the corresponding part of the SPARQL scheme, as explained in Section 6.

The proposed mapping description rules for the given illustrative example (Question 1) are outlined in Table 9. In particular, given the Query Ontology property `who_made`, with domain `Make` and range `Artist`, two variables are defined to handle the domain and range individuals of the Query Ontology solution, aiming to identify artists in the SAAM population associated with certain productions. To achieve this, SAAM representation chose the CIDOC-CRM property `cidoc:P14_carried_out_by`, where the domain is `Production` and the range is `Actor`. Consequently, the CIDOC-CRM representation of? `Make1 who_made? Artist2` is expressed as? `Make1 cidoc:P14_carried_out_by? Artist2`. This mapping facilitates the identification of artists responsible for specific productions within the SAAM knowledge base. The mapping description rule #3 is defined when there exists a data property `has_text` applied to an individual of the class `Artfact`, where the text value is the term used in the natural language question that is associated with the type of the artefact, as in paint, or sculpture, and so on.

Once the SPARQL Query is generated, it is executed using the SPARQL Query SAAM endpoint to validate the consistency and accuracy of the SPARQL Query solution. This ensures that the generated SPARQL Query effectively represents the initial natural language question and retrieves the desired information from the SAAM ontology.

The Query Ontology is independent of the SAAM ontology representation, which is based on CIDOC-CRM. However, the mapping description rules within the SPARQL Generator are dependent on the specific SAAM ontology. Consequently, while the Query Ontology can be reused with different SAAM ontologies (such as DBPedia instead of CIDOC-CRM), the mapping description rules must be adapted accordingly for each new SAAM ontology representation.

9 Query-Answer System Evaluation

The NL-Questions dataset,[18] consisting of a sample of 50 natural language questions, their corresponding SPARQL queries and SAAM answers, was manually built to evaluate the proposed question-answering system.

[18]https://github.com/dvaragnolo/NLP-QA-BENCHMARK

Table 10. NL-Questions Dataset

ID	Question	Evaluation	Comment
1	Who is Darryl Abraham?	Correct	
2	What is Honeymoon Motel?	Correct	
3	Which paintings did Leonard Ochtman create?	Correct	
4	What was the birthdate of Leonard Ochtman?	Correct	
5	What was the birthplace of Leonard Ochtman?	Correct	
6	What was the deathdate of Leonard Ochtman?	Correct	
7	When did Leonard Ochtman die?	Correct	
8	When died Leonard Ochtman?	Correct	
9	Where was the deathplace of Leonard Ochtman?	Correct	
10	What was the nationality of Leonard Ochtman?	Correct	
11	Who painted the Morning Haze?	Correct	
12	Who donated the Morning Haze?	Correct	
13	Who gave the Morning Haze?	Correct	
14	When was the production of Morning Haze completed?	Correct	
15	When was the Morning Haze completed?	Correct	
16	When did the production of Morning Haze start?	Correct	
17	When did Morning Haze start?	Correct	
18	Who is the author of Morning Haze?	Correct	
19	When was the Morning Haze created?	Correct	
20	What was the birthplace of the creator of the Morning Haze?	Correct	
21	What is the size of the Morning Haze?	Correct	
22	Who gave the Honeymoon Motel to the museum?	Correct	
23	What are the sculptures that are made of lead?	Correct	
24	Who was born in 1854?	Correct	
25	Who was born in Zonnemaire?	Correct	
26	Who are the authors of the Morning Haze?	Correct	
27	Which are the authors of Morning Haze?	Correct	
28	What is the place where Leonard Ochtman was born?	Correct	
29	What sculptures are made of lead?	Correct	
30	What sculptures are made of lead material?	Incorrect	Incorrect Partial DRS
31	Who made artworks with lead and wood?	Correct	
32	Who made Morning Haze and Summer Morning?	Incorrect	Incorrect Partial DRS
33	Which painters have died in Amsterdam?	Correct	
34	What things are made by the guy Leonard Ochtman?	Correct	
35	What artefacts are made of lead?	Correct	
36	What artefacts are made by the artist Leonard Ochtman?	Correct	
37	When did the production of Morning Haze end?	Incorrect	Incorrect Stanza analysis
38	What authors died in Madrid in 1660?	Correct	
39	When did the creation of Morning Haze start?	Correct	
40	When was the production of Morning Haze ended?	Correct	
41	When did Leonard Ochtman paint Morning Haze?	Correct	
42	Where did the painter of Morning Haze born?	Correct	
43	Which painter died at the birthplace of Leonard Ochtman?	Correct	
44	Who gave sculptures made by Leonard Ochtman?	Correct	
45	What are the artefacts produced at the deathdate of Leonard Ochtman?	Incorrect	Fails to find the best solution
46	Who was born between 1950 and 1970?	Correct	
47	Who was born before 1950?	Correct	
48	Who was born in 1950 or in 1970?	Incorrect	Incorrect Partial DRS
49	What artefacts were made after 2000?	Correct	
50	Where did Mary go in 1900?	Incorrect	Fails to find the best solution

The 50 questions listed in Table 10 were manually defined by the authors to comprehensively cover the diverse information within SAAM. These questions encompass a wide range of topics, including artist details (e.g., birthdate, birthplace), artefact information (e.g., title, date of production, artist), and exhibit variations in vocabulary, syntax, and length to simulate potential user queries.

During the evaluation process, a new dataset, NLQ-Results is automatically generated, containing the following information for each question in the NL-Questions dataset: the output of the dependency parser, the partial DRS of the best solution, the chosen best solution itself, the generated SPARQL query, the corresponding SPARQL query answer, and a comparison result between the manually curated answer and the system's generated answer.

The NLQ-Results are used to analyze and evaluate the performance of the various modules across several dimensions. These dimensions include:

— *Correctness*: Does the system's answer match the manually curated answer?
— *Dependency Parsing*: Was the dependency parsing performed by Stanza successful and accurate?
— *Partial DRS*: Is the constructed partial DRS logically sound and accurate?
— *DRS Best Solution*: Is the chosen best solution among the DRS candidates the correct one?
— *SPARQL Query*: Is the generated SPARQL query syntactically and semantically correct?

This set of questions contains 44 (forty-four) questions correctly written in English, which have an adequate interpretation and the SPARQL generated is correct, i.e., obtains the user intended answer. There are 6 (six) questions where the system fails to obtain the correct SPARQL.

(1) "*When did the production of Morning Haze end?*"
This question is an example of a sentence that its dependency parser is not correct.
The token "end" is mistakenly tagged as a noun, hindering the creation of a valid DRS and potentially impacting subsequent steps in the process.
(2) "*Who made Morning Haze and Summer Morning?*"
In this question the partial DRS module is unable to obtain an adequate representation.
The current partial DRS module is able to represent some questions with coordination in their syntactic structures, but is not yet complete.
For instance, the question "*Who made artworks with lead and wood?*" has an adequate semantic representation since the coordination is identified on the prepositional phrase.
This example question seems similar to the above one, but to be adequately interpreted requires that the partial DRS represents two events of the class `Make`, i.e., two productions of an artefact made by the same artist or by two different artists. However, currently, the partial DRS builder is not yet complete to do it. Further improvements are required in the partial DRS module to handle this type of questions.
(3) "*What sculptures are made of lead material?*"
In this question, the partial DRS module is also unable to obtain an adequate representation.
The interpretation of "lead material" is not correct since the two words are kept together, and the intended interpretation should consider just the term "lead."
However, the sentence "*What sculptures are made of lead?*", having the same intended meaning, is well processed. Additionally, a sentence like "*What artefacts are made by the artist Leonard?*" is also well interpreted since the partial DRS module is able to interpret the expression "the artist Leonard Ochtman."
(4) "*What are the artefacts produced at the deathdate of Leonard Ochtman?*"
The system is unable to choose the right solution from the set of DRS's solutions. Defining a similar question where "at" is replaced by "in" obtains the right interpretation. The multicriteria optimization problem defined to obtain the best solution is sensitive to many aspects of the solution, such as the syntactic structure of the question and the semantic evaluation of the solutions. When using the preposition "at" to specify a date, the sentence syntactic structure is not preferred, since the annotations of the objective property `action_date` does not include the proposition "at." The question with the preposition "in" obtains the right solution because this preposition is included in the annotations of the object property.
Consider the question "*What are the artefacts produced at 1900?*", the system is capable of choosing the best solution even with the proposition "at." This behavior is due to the combination of the values of the other criteria that work better with smaller sentences.

Table 11. Evaluation Results of the NL-Questions Dataset

Dimension	No. of Failed Questions	Comment
Correctness	6	44 questions are correct.
Dependency Parsing	1	49 questions were parsed successfully by Stanza
Partial DRS	3	46 questions had a correct partial DRS
DRS Best Solution	2	44 questions had a correct Query Ontology Representation
SPARQL Query	0	44 questions had a correct SPARQL Query in CIDOC

Table 12. A Sample of the 5,000 Question's Dataset

Question Set	Question Example
Q1	Which is the art type of Morning Haze?
Q2	What was the material used in the Morning Haze?
Q3	Who gave the Morning Haze to the museum?
Q4	Who is the creator of Morning Haze?
Q5	Which is the birth place of Leonard Ochtman?
Q6	When the production of Morning Haze started?
Q7	When the production of Morning Haze ended?
Q8	Which is the nationality of the creator of Morning Haze?
Q9	Which is the birth place of the creator of Morning Haze?
Q10	Which year died the creator of Morning Haze?

(5) "*Where did Mary go in 1900?*"
The set of solutions for this question does not include the intended meaning of the sentence. The ontology has no representation for the action "to go." The system chooses an interpretation of the question and generates a SPARQL query that is quite different from the sentence's meaning.
(6) "*Who was born in 1950 or in 1970?*"
This question has an or coordination similar to the and coordination of the previous question "*Who made Morning Haze and Summer Morning?*," for which the question the partial DRS module fails to produce an adequate representation.

A question is correctly analyzed by the system when the generated SPARQL results match the corresponding answer in the manual dataset.

The evaluation results in the NL-Questions dataset are summarized in Table 11.

The query-answer system was also evaluated using 5,000 questions (single-entity factoid questions), from the dataset proposed in [10].[19] The 5,000 question's dataset contains 10 different types of questions, each type has 500 variations defined by changing the names of the Artists (Q5) or the names of the Artefacts (Q1, Q2, Q3, Q4, Q6, Q7, Q8, Q9, Q10).

Table 12 presents an example of the 10 different types of questions.

For the above 10 questions, the proposed question-answer system is able to obtain the correct SPARQL question in SAAM CIDOC-CRM representation.

The evaluation of the 5,000 questions is done to assess the performance of the NER system, implemented using a gazetteer composed of all the names of SAAM artworks and artists. Table 13 presents the evaluation obtained in each question of the 500 questions. The system achieved its best performance without using NER on the question set Q5. This set comprised artist names, for which Stanza's dependency parser achieved successful results in most cases. When the NER is used, all the questions are associated with a correct SPARQL query.

[19] https://github.com/NicolaiGoon/CIDOC-QA-BENCHMARK/

Table 13. Evaluation Results of the 5,000 Question's Dataset

Question Type	% Correct Answers with NER	% Correct Answers without NER
Q1 (500 examples)	100%	44.0%
Q2 (500 examples)	100%	44.8%
Q3 (500 examples)	100%	54.4%
Q4 (500 examples)	100%	55.6%
Q5 (500 examples)	100%	98.0%
Q6 (500 examples)	100%	49.4%
Q7 (500 examples)	100%	53.8%
Q8 (500 examples)	100%	46.4%
Q9 (500 examples)	100%	44.8%
Q10 (500 examples)	100%	56%

Regarding system efficiency, the Pragmatic Interpretation module has a time complexity of $O(N^D))$, where N is the number of Query Ontology classes and D is the number of discourse referents in a question's partial DRS, determined by the CSP solver.

Ontology Content-Matching time complexity is proportional to the number of CSP solutions (M), representing possible class assignments to discourse referents. Objective function calculus time complexity is proportional to the number of Ontology Content-Matching solutions ($K \times M$), where K is the count of Query Ontology properties for each partial DRS condition solution.

Multi-objective problem solving time complexity is proportional to KM, the number of potential semantic representations of the natural language question.

The Partial Semantic Representation and SPARQL Generator modules rely on external services, which affects the execution time of the entire process from the natural language question to the SPARQL query answer. The execution time of the external services, Stanza and the SPARQL endpoint, depends not only on the time complexity of their specific processes but also on the network connection. In Partial Semantic Representation, the NER step has a time complexity proportional to the product of the number of tokens in the natural language question and the number of Gazetteer entries. Stanza parsing is an external service. DRS Builder time complexity is proportional to the number of tokens in the question's Stanza parse. The time complexity of the SPARQL query builder step is proportional to the number of ontology property instances in the solution. The SPARQL Query SAAM endpoint step relies on an external service.

Considering the illustrative example Question 1, the number of question tokens is 7, the number of Stanza tokens is 6, the number of referents is $D = 3$, the number of Query Ontology solutions is 559 ($M = 3^{21} = 9,261$), and the number of Query Ontology properties for each partial DRS condition solution is $K = 559$. The average execution time for this question on a standard personal computer is 3 seconds. In both datasets, the maximum number of discourse referents in a question is 5. For these questions, the average execution time is 22 seconds, as the maximum number of solutions (13,430) is below the potential $M = 5^{21}$. Across the 50 questions, the system takes an average of 339 seconds to process all of them. The shortest execution time for a single question is 1.3 seconds, while the longest is 31.7 seconds. These execution times could be significantly improved by using a higher-performance computer.

The evaluation results of the 5,000 questions conducted by the authors of [10] define, for each question, a set of tokens for the golden answer and a set of tokens for the predicted answer. Precision and recall metrics are calculated based on these token sets. The authors report an F1-score of 64.5% when a perfect entity is detected (correctly recognized and linked to its URI) in 78.4% (3,920) of the total questions. When applying NER, these results are significantly lower than those obtained by the proposed strategy in the current paper, which achieves 100% of correct answers that correspond to an F1-score of 100%. The proposed system's NER achieves 100% precision in recognizing named entities across 5,000 questions. These 5,000 questions are divided into 10 groups,

where each group follows the same question pattern. Within each group, the 500 questions differ only in the named entities. As a result, the 10 groups of questions are well represented in the Query Ontology whenever Stanza successfully recognizes the named entities. Since the NER accurately identifies the named entities, the system achieves correct syntactic parsing, partial DRS, Query Ontology representations, SPARQL query generation, and the final answer. In contrast, the compared system employs different strategies (e.g., Large Language Models) to recognize named entities and to obtain the answer. However, even when named entities are correctly recognized in 78.4% of the questions, it may still fail to provide the correct answer.

In the 5,000 questions analyzed, the authors of [10] report an F1-score of 51.4%. As this metric is calculated based on the number of tokens in the gold standard and predicted answers, it is not directly comparable to the 54.6% of correct answers achieved by the proposed system in this article when NER is not applied.

Regarding state-of-the-art systems employing Large Language Models to convert natural language questions into SPARQL queries, the best performers report F1-scores of 85% [38] and 54.79% [23] within the DBLP domain. These systems necessitate extensive datasets for task-specific fine-tuning. While such datasets exist for domains like DBLP, DBPedia, and Yago, provided by competitions such as the QALD Challenges, no comparable datasets are currently available for the CIDOC-CRM representation.

This study of the system behavior shows that the strategy proposed is adequate to obtain a tool, to help users consulting a knowledge base represented in CIDOC-CRM, such as SAAM's. The users can pose a natural language question and obtain the SPARQL query in the SAAM CIDOC-CRM representation.

Further improvements can be made on the system's partial DRS builder to accept questions with other discourse phenomena, such as coordination. But at this point, the proposed tool besides helping users, also provides an explanation for the final SPARQL Query, the question Query Ontology representation.

10 Conclusions and Future Work

A question-answering system was proposed to translate natural language questions into SPARQL queries for the CIDOC-CRM representation of SAAM's, to facilitate the search by users, even for those without SPARQL technical knowledge. The proposed question-answering system can be integrated into a user interface for the SAAM knowledge base, enabling users to pose natural language questions (spoken or written). The interface can display answers in a table format or a natural language text generated from the table content. Additionally, intermediate processing steps, such as syntactic parsing, Query Ontology representation, and the generated SPARQL query, can be exhibited to help the user understand the question processing steps. A dialogue system could also be incorporated into the interface to address ambiguities or information gaps in user queries.

The proposed approach is a novel one since a Concept-Context Ontology domain, the Query Ontology, is used to represent the user question's concepts. The proposed ontology is then mapped to the SAAM CIDOC-CRM representation. This system uses a traditional natural language processing symbolic approach, i.e., a dependency parser to analyze the natural language question, Stanza, and a simplified DRS to be interpreted in the proposed Query Ontology.

The Query Ontology is enriched with classes and properties annotations to add specific vocabulary information and syntactic role preferences, and with semantic web rules to evaluate the adequacy of the question representation.

The question's interpretation is obtained by matching the DRS initial representation on the Query Ontology. With this strategy, each question can have many interpretations, and choosing the best solution is resolved as a multi-objective problem, where the objective values are obtained for each solution using lexical, syntactic, and semantic information.

An evaluation of the various modules of the proposed system was presented and showed very promising results. The use of a NER, with the support of SAAM's artist and artefact name's gazetteer, allowed to improve the results and obtain 100% precision in an independent dataset.

The proposed question-answering system is language-independent, except for the annotations on the query Ontology that are language dependent. To adapt this system to a new domain, the Query Ontology must be designed to represent the new domain questions concepts. A new set of migration rules must be written to transform the classes and properties of the solution into the classes and properties of the target ontology of the new domain.

In future work, to overcome the time-consuming in obtaining an answer, a dataset with the questions and SPARQL queries obtained with the proposed system will be built, in order to be used in the development of a SPARQL Query Representation Generation using a language model such as BERT fine-tuned in this task.

References

[1] Mattia Atzeni and Maurizio Atzori. 2018. Translating natural language to code: An unsupervised ontology-based approach. In *2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE '18)*, 1–8. DOI: https://doi.org/10.1109/AIKE.2018.00009

[2] Manuel Alejandro Borroto, Francesco Ricca, and Bernardo Cuteri. 2021. A system for translating natural language questions into SPARQL queries with neural networks: Preliminary results. In *Proceedings of the 29th Italian Symposium on Advanced Database Systems (SEBD '21)*, Vol. 2994, 226–234.

[3] Dennis Diefenbach, Kamal Singh, and Pierre Maret. 2018. WDAqua-core1: A question answering service for RDF knowledge bases. In *Companion Proceedings of the Web Conference 2018 (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1087–1091. DOI: https://doi.org/10.1145/3184558.3191541

[4] Eleftherios Dimitrakis, Konstantinos Sgontzos, Michalis Mountantonakis, and Yannis Tzitzikas. 2020. Enabling efficient question answering over hundreds of linked datasets. In *Information Search, Integration, and Personalization*. Giorgos Flouris, Dominique Laurent, Dimitris Plexousakis, Nicolas Spyratos, and Yuzuru Tanaka (Eds.), Springer International Publishing, Cham, 3–17.

[5] Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

[6] Eleanor E. Fink. 2018. American Art Collaborative (AAC) Linked Open Data (LOD) Initiative: Overview and recommendations for good practices. Smithsonian American Art Museum.

[7] Thomas Francart. 2023. Sparnatural: A visual knowledge graph exploration tool. In *The Semantic Web: ESWC 2023 Satellite Events*. Catia Pesquita, Hala Skaf-Molli, Vasilis Efthymiou, Sabrina Kirrane, Axel Ngonga, Diego Collarana, Renato Cerqueira, Mehwish Alam, Cassia Trojahn, and Sven Hertling (Eds.), Springer Nature, Cham, 11–15.

[8] Bart Geurts, David I. Beaver, and Emar Maier. 2020. Discourse representation theory. In *The Stanford Encyclopedia of Philosophy* (Spring 2020 ed.). Edward N. Zalta (Ed.), Metaphysics Research Lab, Stanford University.

[9] Matthew Gotham and D. Haug. 2018. Glue semantics for universal dependencies. In *Proceedings of the LFG '18 Conference University of Vienna*. Miriam Butt and Tracy Holloway King (Eds.), CSLI Publications, 208–226.

[10] Nikos Gounakis, Michalis Mountantonakis, and Yannis Tzitzikas. 2023. Evaluating a radius-based pipeline for question answering over cultural (CIDOC-CRM based) knowledge graphs. In *Proceedings of the 34th ACM Conference on Hypertext and Social Media (HT '23)*. ACM, New York, NY, Article 24, 10 pages. DOI: https://doi.org/10.1145/3603163.3609067

[11] ICOM/CIDOC. 2020. *Definition of the CIDOC Conceptual Reference Model* (7.0.1 ed.). ICOM/CRM Special Interest Group.

[12] Mehdi Jabalameli, Mohammadali Nematbakhsh, and Ahmad Zaeri. 2020. Ontology-lexicon–based question answering over linked data. *Electronics and Telecommunications Research Institute* 42, 2 (2020), 239–246. DOI: https://doi.org/10.4218/etrij.2018-0312

[13] Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic: Introduction to Model Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers, Dordrecht

[14] Shiqi Liang, Kurt Stockinger, Tarcisio Mendes de Farias, Maria Anisimova, and Manuel Gil. 2021. Querying knowledge graphs in natural language. *Journal of Big Data* 8, 3 (2021).

[15] Jiangming Liu, Shay B. Cohen, Mirella Lapata, and Johan Bos. 2021. Universal discourse representation structure parsing. *Computational Linguistics* 47, 2 (05 2021), 445–476. DOI: https://doi.org/10.1162/coli_a_00406

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692. Retrieved from https://arxiv.org/abs/1907.11692

[17] Georgios Meditskos, Efstratios Kontopoulos, Stefanos Vrochidis, and Ioannis Kompatsiaris. 2020. Converness: Ontology-driven conversational awareness and context understanding in multimodal dialogue systems. *Expert Systems* 37, 1 (2020), e12378.

[18] Carlo Meghini and Martin Doerr. 2018. A first-order logic expression of the CIDOC conceptual reference model. *International Journal of Metadata, Semantics and Ontologies* 13, 2 (2018), 131–149.

[19] Dora Melo, Irene Pimenta Rodrigues, and Inês Koch. 2020. Knowledge discovery from ISAD, digital archive data, into ArchOnto, a CIDOC-CRM based linked model. In *Proceedings of the 12th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K '20), KEOD*. INSTICC, SciTePress, 197–204. DOI: https://doi.org/10.5220/0010134101970204

[20] Dora Melo, Irene Pimenta Rodrigues, and Davide Varagnolo. 2023. A strategy for archives metadata representation on CIDOC-CRM and knowledge discovery. *Semantic Web* 14, 3 (2023), 553–584. DOI: https://doi.org/10.3233/SW-222798

[21] David Milward and Martin Beveridge. 2003. Ontology-based dialogue systems. In *Proceedings of the 3rd Workshop on Knowledge and Reasoning in Practical Dialogue Systems (IJCAI '03)*, 9–18.

[22] Divyansh Shankar Mishra, Abhinav Agarwal, B. P. Swathi, and K. C. Akshay. 2022. Natural language query formalization to SPARQL for querying knowledge bases using Rasa. *Progress in Artificial Intelligence* 11 (2022), 193–206.

[23] Reham Omar, Ishika Dhall, Panos Kalnis, and Essam Mansour. 2023. A universal question-answering platform for knowledge graphs. *Proceedings of the ACM on Management of Data* 1, 1 (May 2023), Article 57, 25 pages. DOI: https://doi.org/10.1145/3588911

[24] Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Brussels, Belgium, 160–170.

[25] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Retrieved from https://nlp.stanford.edu/pubs/qi2020stanza.pdf

[26] Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. 2016. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics* 4 (2016), 127–140. DOI: https://doi.org/10.1162/tacl_a_00088

[27] Md Rashad Al Hasan Rony, Uttam Kumar, Roman Teucher, Liubov Kovriguina, and Jens Lehmann. 2022. SGPT: A generative approach for SPARQL query generation from natural language questions. *IEEE Access* 10 (2022), 70712–70723. DOI: https://doi.org/10.1109/ACCESS.2022.3188714

[28] Sharmela Shaik, Prathyusha Kanakam, S. Mahaboob Hussain, and D. Suryanarayana. 2016. Transforming natural language query to SPARQL for semantic information retrieval. *International Journal of Engineering Trends and Technology* 41, 7 (2016), 347–350.

[29] João Quirino Silva, Dora Melo, Irene Pimenta Rodrigues, João Costa Seco, Carla Ferreira, and Joana Parreira. 2021. An ontology based task oriented dialogue. In *Proceedings of the 13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K '21)*. David Aveiro, Jan L. G. Dietz, and Joaquim Filipe (Eds.), KEOD, Vol. 2, SCITEPRESS, 96–107. DOI: https://doi.org/10.5220/0010711900003064

[30] Nadine Steinmetz, Ann-Katrin Arning, and Kai-Uwe Sattler. 2019. From natural language questions to SPARQL queries: A pattern-based approach. In *Datenbanksysteme für Business, Technologie und Web*, 289–308. DOI: https://doi.org/10.18420/btw2019-18

[31] Giorgos Stoilos, Szymon Wartak, Damir Juric, Jonathan Moore, and Mohammad Khodadadi. 2019. An ontology-based interactive system for understanding user queries. In *European Semantic Web Conference*. Springer, 330–345.

[32] Pedro Szekely, Craig A. Knoblock, Fengyu Yang, Eleanor E. Fink, Shubham Gupta, Rachel Allen, and Georgina Goodlander. 2014. Publishing the data of the Smithsonian American Art Museum to the linked data cloud. *International Journal of Humanities and Arts Computing* 8, Supplement (2014), 152–166. DOI: https://doi.org/10.3366/ijhac.2014.0104

[33] Pedro Szekely, Craig A. Knoblock, Fengyu Yang, Xuming Zhu, Eleanor E. Fink, Rachel Allen, and Georgina Goodlander. 2013. Connecting the Smithsonian American Art Museum to the linked data cloud. In *The Semantic Web: Semantics and Big Data*. Philipp Cimiano, Oscar Corcho, Valentina Presutti, Laura Hollink, and Sebastian Rudolph (Eds.), Springer, Berlin, 593–607.

[34] Maria Theodoridou, George Bruseker, Maria Daskalaki, and Martin Doerr. 2016. Methodological tips for mappings to CIDOC CRM. In *Proceedings of the 44th Computer Applications and Quantitative Methods in Archaeology Conference (CAA '16 OSLO Exploring Oceans of Data)*, 89–102.

[35] Davide Varagnolo, Guilherme Antas, Mariana Ramos, Sara Amaral, Dora Melo, and Irene Pimenta Rodrigues. 2022. Evaluating and exploring text fields information extraction into CIDOC-CRM. In *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - KEOD*. INSTICC, SciTePress, 177–184. DOI: https://doi.org/10.5220/0011550700003335

[36] Hernán Vargas, Carlos Buil-Aranda, Aidan Hogan, and Claudia López. 2019. RDF explorer: A visual SPARQL query builder. In *The Semantic Web (ISWC '19)*. Chiara Ghidini, Olaf Hartig, Maria Maleshkova, Vojtěch Svátek, Isabel Cruz, Aidan Hogan, Jie Song, Maxime Lefrançois, and FabienGandon (Eds.), Springer International Publishing, Cham, 647–663.

[37] Hernán Vargas, Carlos Buil-Aranda, Aidan Hogan, and Claudia López. 2020. A user interface for exploring and querying knowledge graphs (extended abstract). In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI '20)*. Christian Bessiere (Ed.), International Joint Conferences on Artificial Intelligence Organization, 4785–4789. DOI: https://doi.org/10.24963/ijcai.2020/666

[38] Ruijie Wang, Zhiruo Zhang, Luca Rossetto, Florian Ruosch, and Abraham Bernstein. 2023. NLQxform: A language model-based question to SPARQL transformer. In *Joint Proceedings of Scholarly QALD 2023 and SemREC 2023 Co-Located with 22nd International Semantic Web Conference ISWC 2023*. Debayan Banerjee, Ricardo Usbeck, Nandana Mihindukulasooriya, Gunjan Singh, Raghava Mutharaju, and Pavan Kapanipathi (Eds.), CEUR Workshop Proceedings, Vol. 3592. CEUR-WS.org. Retrieved from https://ceur-ws.org/Vol-3592/paper2.pdf

[39] Michael Wessel, Girish Acharya, James Carpenter, and Min Yin. 2019. OntoVPA—An ontology-based dialogue management system for virtual personal assistants. In *Advanced Social Interaction with Agents*. Springer, 219–233.

[40] Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. 2021. Neural machine translating from natural language to SPARQL. *Future Generation Computer Systems* 117 (2021), 510–519. DOI: https://doi.org/10.1016/j.future.2020.12.013

[41] Neli Zlatareva and Devansh Amin. 2021. Natural language to SPARQL query builder for semantic web applications. *Journal of Machine Intelligence and Data Science* 2 (2021), 44–53. DOI: https://doi.org/10.11159/jmids.2021.006

[42] Zdeněk Žabokrtský, Daniel Zeman, and Magda Ševčíková. 2020. Sentence meaning representations across languages: What can we learn from existing frameworks? *Computational Linguistics* 46, 3 (11 2020), 605–665. DOI: https://doi.org/10.1162/coli_a_00385

Received 11 April 2024; revised 18 October 2024; accepted 30 December 2024