IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 September 2024, accepted 15 October 2024, date of publication 1 November 2024,
date of current version 18 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3489889

RESEARCH ARTICLE

Influence of Event Specialization Strategy on Some Aspects of Natural Language Querying Interfaces to Ontologies

RITA BUTKIENĖ[1], ALGIRDAS ŠUKYS[1], LINAS ABLONSKIS[1], AND RIMANTAS BUTLERIS[2]
[1]Department of Information Systems, Faculty of Informatics, Kaunas University of Technology, 44249 Kaunas, Lithuania
[2]Centre of Information Systems Design Technologies, Faculty of Informatics, Kaunas University of Technology, 44249 Kaunas, Lithuania
Corresponding author: Rita Butkienė (rita.butkiene@ktu.lt)

ABSTRACT The performance characteristics and certain maintenance aspects of a natural language querying (NLQ) interface depend on how the data model is conceptualized. One of the areas where alternative conceptualizations are available is events and their specialization. The concept of event specialization is already known from event extraction methods, which allows for a more precise description of the events identified in a text. In the context of NLQ interfaces, event specialization allows narrower or broader questions. This study investigates how the choice of event specialization strategy in OWL (Web Ontology Language) ontologies affects the complexity and performance-related aspects of the NLQ interface to ontologies. In this paper, we present four event specialization strategies and investigate how they impact the size of the ontology schema and vocabulary of the NLQ interface, the performance of querying and data import, the size of the semantic repository, and the complexity of SPARQL queries. We discuss the strengths and weaknesses of each approach and present recommendations on determining the best one for the needs of NLQ interface end-users and developers.

INDEX TERMS Natural language query (NLQ) interface, event ontology, event representation, n-ary relation, event specialization strategy, OWL, SPARQL, SBVR.

I. INTRODUCTION

For decades, the desire to enable users to access information from knowledge bases by querying in natural language has driven a wide range of research in many fields, such as natural language processing, computer science, data science, and user experience (UX) design. These studies have led to the development of natural language query (NLQ) interfaces, which can understand questions and user intent and produce queries (e.g., SQL and SPARQL) that can be executed on the underlying knowledge base. Simultaneously, the NLQ interface must ensure usability and maintenance characteristics. For example, it should help formulate questions and make it easy to understand which questions can and cannot be answered. In addition, performance characteristics such as query execution speed and data import time must be acceptable. One of the things that these characteristics depend on is the underlying data model, the influence of which we investigate in this study.

The NLQ interface implements the information retrieval (IR) [1] process of an information system (Fig. 1) and ensures that users obtain the information they seek from the knowledge base where it is stored. Studies show the advantages of this type of interface, which allows the hiding of the complexity of the linked data, ontologies, and formal languages from the user, expresses complex information needs intuitively, and, in principle, in their language offers a user-friendly option to query ontology-based knowledge sources.

In this study, we treat the semantic database as the knowledge base, where facts extracted from news portals, such as what occurred, where and when it took place, and who was involved, are stored. Data are added to such a system through an information extraction (IE) process that identifies and

The associate editor coordinating the review of this manuscript and approving it for publication was Mansoor Ahmed.

extracts specific facts from written texts or speech transcripts and converts them into structured representations [2].

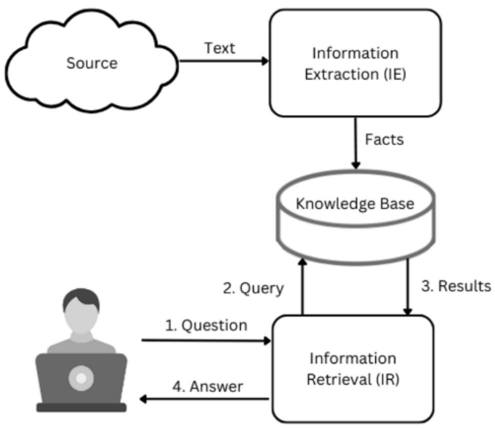


FIGURE 1. Information retrieval and Information extraction processes.

The IR and IE processes are closely related, and the quality of the IE process directly affects the IR process. Poor IE performance can lead to inaccurate or incomplete IR results. Moreover, how the extracted data are structured – specifically, the conceptual model used – can affect various operational and maintenance characteristics (e.g., query performance, storage requirements, etc.) of the IR process and the overall information system.

Facts extracted from news portal texts have specific event types and structures. When populating a knowledge base with new facts, the representation of the extracted events should correspond to the ontology schema used for knowledge storage. A widely used event representation is an n-ary relation pattern that has been utilized in various event annotation and ontology schemas, such as Event Ontology (EO) [3] and Simple Event Model (SEM) [4].

Schemas of existing event ontologies provide a framework and essential concepts for representing events and can be extended and tailored to specific domains or task needs. According to a review on ontologies on occurrents [5], event classification is a significant consideration when modeling events. Using different characteristics of events as classification criteria, a modeler can specialize events and construct a taxonomy of events relevant to domain or task solving. Therefore, one decision that a modeler must make when designing a conceptual model for event representation is the choice of an appropriate specialization strategy. An event specialization strategy is a structured approach to creating data models, knowledge graphs, or ontologies to precisely define and categorize events based on their specific characteristics, participants, and contextual information. Such a strategy aims to ensure that events are represented in a detailed, unambiguous, and organized way, capturing their particularities and enabling systems to process them correctly in various contexts.

Event classification and specialization have been used in decades of studies on the automated extraction of various events. According to Guan [6], these studies initially focused on defining the concept of an event and its structure [7], [8], [9], [10]. Later, with the launch of the Automatic Content Extraction (ACE) program [11], automated event extraction tasks from the text were solved. As a result, standards for annotating events in text emerged, as well as annotated datasets for evaluating event extraction methods [12]. The specialization of events, or defining their types, was applied when creating these datasets. In IE tasks, specialization enables a more detailed description of an event and its meaning. In addition, knowing that events of different types differ in structure (i.e., to what objects an event can be associated with) can contribute to event detection algorithms.

For IR tasks, the specialization of events allows the querying of specific events. Consider the following question: *What did the president say to the Prime Minister?*. This question concerns the event of a type saying. The specialization of this event can be based on various characteristics. For example, an event of saying type can be specialized according to the manner in which it is spoken, thus enabling the user to ask and receive answers to questions with a more specialized meaning, such as *What the president confidently said to the Prime Minister?*. Creating the possibility for a user to ask questions with broader or narrower meanings requires appropriate IE and IR implementation.

This study does not focus on event extraction methods but rather on how extracted events are conceptualized and stored as ontology individuals in a semantic database. Although the n-ary relation pattern is the most widely used when conceptualizing events, different specialization strategies lead to specific variants of the event representation (data model, schema). The choice of strategy determines not only how events will be represented, but also how much effort (time) will be required to develop it, what resources will be required to store it, and what will be the efficiency of query execution.

When developing the NLQ interface [13], [14] we already faced the challenge of choosing a model for event representation. The decisions made were promising, but the development experience and the obtained results [15] motivated us to study more widely and deeply the influence of the choice of the event specialization strategy in the development of NLQ interfaces of a similar architecture. The main reason for undertaking the research was that the implementation of our chosen NLQ interface concept requires considerable effort in the development of an event ontology schema and associated vocabulary. It was also unclear how changing the event representation model will influence the operating characteristics of the developed system; even then the built-in solution was performing queries more slowly than we expected. Since we did not find enough research to help resolve these uncertainties, we decided to investigate how the differences between alternative event specialization strategies affect the implementation, maintenance, continuous development, and properties of the NLQ interface and semantic search system.

We posed the following research questions:

- RQ1. *How does the choice of specialization strategy affect the ontology schema regarding the cost of modification?*
  If the event ontology schema is developed incrementally, then its initial variants will include only a part of the types of events or certain properties of the event. Later, you may need to modify the schema by removing, adjusting, or adding new event types or their properties. The time required to make changes depends on how many concepts must be specified in the schema for one event type. This number of concepts depends both on the type of event itself and on the selected event specialization strategy.
- RQ2. *How does the choice of specialization strategy affect the size of the semantic database and data insertion?*
  Different schemes for a specialized event lead to different numbers of triplets representing the same event. Some of those triplets may be derived and may take time to derive. Therefore, before choosing a specialization strategy, it is necessary to take into account the development possibilities of data storage resources and the time required to insert data into the semantic database.
- RQ3. *How does the choice of specialization strategy affect the performance of SPARQL queries?*
  Since applying different specialization strategies results in different event specialization schemas, semantic queries are formulated differently to answer the same question. The differences in the triple patterns of the semantic query can significantly affect the query execution.

To answer these questions, we conducted a study, which we present in the following structure. Section II explains the background of this study and related work. Section III presents the event specialization strategies used in our study and their experimental investigations. Section IV provides a summary and analysis of the results and provides recommendations for NLQ interface developers. Section V discusses the limitations of this study. Finally, Section VI concludes the paper.

II. BACKGROUND AND RELATED WORK

In this section, we first present the main components and principles of the NLQ interface. We then present a literature review of how events are conceptualized and represented, underlining the importance of event specialization. Finally, we provide an overview of studies such as ours that investigate the influence of event representation choices.

A. NATURAL LANGUAGE QUERY INTERFACES

1) INTRODUCTION TO NLQ INTERFACES

NLQ interfaces allow users unfamiliar with query languages to search in knowledge bases. This type of system accepts a natural language question from a user, transforms it into a knowledge base query, and provides results. Depending on how a question is analyzed and interpreted, two architecture types are used to develop such systems: rule-based and machine learning-based [16]. Rule-based systems use a set of manually created rules that imitate the manner in which humans interpret and understand questions. Such systems have the advantage of understanding and answering complex questions. However, they allow less freedom in formulating questions and struggle to understand the nuances of the language [17].

Machine learning-based techniques use statistical methods. By analyzing the training data, they built their knowledge to analyze questions and produce queries. The major challenge in building such a system is the lack of training data, which often must be created manually. Machine learning-based systems have the advantage of understanding more of the nuances of the language and allow questions to be formulated more freely. However, they usually struggle to answer complex questions. Moreover, machine learning-based systems cannot explain how questions are interpreted, which may reduce trust in such systems [17].

There are also hybrid methods that exploit the strengths of both rule-based and machine learning methods. In such methods, certain steps (e.g., understanding the data model and linking with a question, natural language understanding, keyword mapping) of rule-based methods are executed using machine-learning algorithms.

2) RULE-BASED NLQ INTERFACES

The research presented in this paper focuses on rule-based approaches. We present the diagram in Fig. 2 to demonstrate the principle of operation of these systems.

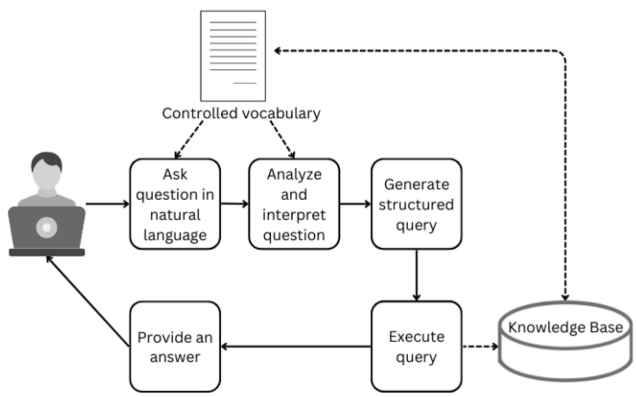


FIGURE 2. **The structure of rule-based NLQ system.**

A question written by a user is further analyzed by following these steps [17]: (1) entities mentioned in a question and the relationships between them are identified; (2) the identified entities and relationships are linked to the corresponding structures in a knowledge base; (3) a query is generated. The analysis of questions and translation to queries is performed using the semantic representations of the underlying knowledge base. It is essentially a controlled vocabulary (often referred to as a lexicon) that matches the semantics of the stored data and holds mappings between entities identified in a question and the structure of the knowledge base. Multiple works (e.g., [18], [19]) show that this process is almost straightforward when questions can be mapped directly to binary relations of the knowledge base. However, in more complex cases, semantic and vocabulary gaps between a question and a knowledge base can occur and have to be addressed [20], [21], [22], [23].

A semantic or structural gap means that a question does not correspond to the structure of the knowledge base. For example, when the knowledge base uses n-ary relations and does not have relations between entities used in questions. In such a case, the system should be able to find the correct mapping to generate a query corresponding to the structure of the knowledge base. A vocabulary gap occurs when a question contains words that do not match the vocabulary of a knowledge base (e.g., class, property names, etc.). This gap can be addressed by enriching the vocabulary with synonyms.

As summarized in [17], vocabularies used in NLQ systems can have different semantic expressiveness, ranging from a simple inverted index (e.g., Precis [24], QUICK [25]) containing terms from a subject area to a taxonomy (e.g., NaLIR [26]) or ontology (e.g., SODA [27], ATHENA [28], ATHENA++ [29], BELA [30], GeoQA [31]). The more semantically rich a vocabulary, the more complex the questions the system can interpret and answer. For example, taxonomy allows questions to be formulated using abstract or specialized terms. Ontology additionally allows the inference of facts that are not explicitly expressed in a knowledge base, such as relations between entities, that are used for questioning.

3) NLQ INTERFACE BASED ON SBVR VOCABULARY

In this research, we used the NLQ interface based on vocabulary specified using the Semantics of Business Vocabulary and Business Rules (SBVR) standard [32]. It allows the specification of a human-readable vocabulary whose meaning is also represented using logical formulations and is understandable for a computer. SBVR vocabulary has semantic expressiveness similar to that of the OWL (Web Ontology Language) ontology [13]. SBVR definitions can be used to close the semantic gap by expressing the language formulations used in questions in a manner that corresponds to the structure of the knowledge base. Another advantage of the SBVR vocabulary is the possibility of defining synonyms and avoiding them in the ontology schema.

The aforementioned NLQ interface is part of the SBVR-based semantic search framework that was developed to enable semantic content processing and search over Lithuanian language texts on the World Wide Web [14]. This framework consists of tightly coupled information extraction and retrieval modules. The IE module performs preprocessing linguistic and semantic annotation of text documents extracted from the web. It primarily aims to capture and conceptualize events and entities participating in or related to these events. The events and entities extracted from the text documents are written to the semantic repository as ontology instances.

The applicability of the framework was evaluated by performing a case study on the Lithuanian news corpus [15] and annotating what people have said. Therefore, we constructed an event ontology, corresponding SBVR vocabulary, and IE module to recognize events in the text. Although the case study demonstrated promising results, one disadvantage of the developed system was the low performance of SPARQL queries. We hypothesized that this might be due to the chosen structure of the ontology, which led to the research presented in this paper.

B. EVENT REPRESENTATION

1) EVENT DEFINITION

The concept of an event has been analyzed for a long time, but there is no universally agreed-upon definition or representation of it [33]. The most common definitions describe events as "things that occur or happen or take place". More specific event definitions consider the application or task to be solved and, therefore, emphasize specific characteristics of an event that are relevant to that domain. For example, Automatic Content Extraction (ACE) English Annotation Guidelines for Events [11] defines an event as "a specific occurrence involving participants" and specifies which participants should be added. Hence, an event in the ACE guidelines is defined as a structure containing several arguments with different roles. This representation is consistent with the neo-Davidsonian representation (also called reification) of event semantics, which is most common in computational linguistics, databases, and ontologies.

2) NEO-DAVIDSONIAN REPRESENTATION

Neo-Davidsonian representation, born out of the work of Davidson [10] and Parsons [34], represents events as n-ary relations, where an event is conceptualized as an entity having relations with other entities involved. The number and type of entities included in the representation depend on the type of event and the granularity of the information collected regarding an event. Therefore, specifying an event type and entities associated with an event has become a common practice in conceptual event modeling.

3) EVENT ANNOTATION IN DATASETS FOR IE

Analysis of existing datasets for creating event extraction models shows that event type, subtype, or both are specified when annotating corpora. For example, the ACE and ERE [12] datasets annotate eight event types and 33 event subtypes. The RichERE [35] dataset annotates nine types and 38 event subtypes, and the TimeML [36], ECB+ [37], and RED [38] datasets annotate six, seven, and four event types, respectively. Different types of events have different structures. For example, according to [11], the event-type

TRANSPORT has six participant slots, whereas BE-BORN has only one. Perhaps the most comprehensive event structuring was done in the FrameNet project [39]. Its knowledge base identifies over 1200 semantic frames (i.e., event types), indicating their structural elements, relationships with other frames (inheritance, usage, etc.), and lexical units.

4) REPRESENTATION OF EXTRACTED EVENTS

In the IE context, the specification of the event type and its structure indicates what types of events and what details about the event a machine can be trained or programmed to recognize and extract (for example, from ACE-2005 dataset [11] you can extract events of 8 types). From the viewpoint of IR, the specification of the event type and its structure indicates the search capability, that is, the type and specificity of questions that can be asked about events. When developing an NLQ system, where the IR relies on the results of the IE, consideration should be given to how the extracted information about events is represented and stored.

With the advent of the ability to represent knowledge of real-world entities in a machine-readable format, IE results have been stored in knowledge graphs. For example, Rospocher et al. [40] proposed constructing an event-centric knowledge graph to represent events extracted from text. For this purpose, a Simple Event Model (SEM) ontology [4] was used, where the event representation corresponds to a neo-Davidsonian approach to event representation using n-ary relations. S. Gottschalk and E. Demidova developed a multilingual event-centric temporal knowledge graph (EventKG) by extracting historical events from several large-scale sources. The EventKG scheme is built using SEM, which allows you to retrieve detailed information about certain types of events: temporal information, location, related entities.

5) REPRESENTING EVENTS IN ONTOLOGY

Because our study is limited to finding information about events from semantic databases, we review how an event is represented in ontologies. One part of the ontologies that specify the notion of an event is the upper level and is independent of the domain (e.g., BFO [41], UFO [42], DOLCE [43], SUMO [44]). The other part is designed to address specific event-related tasks (Event Ontology [3], Simple Event Model [4], Event Model F [45], and LODE [46]). The suitability of these ontologies for intended event-related tasks was evaluated in [47], [48], and [49]. The following section discusses the most used ontologies for storing information about events and how they represent event specialization.

6) MOST USED EVENT ONTOLOGIES

Event ontology (EO) [3] is one of the most prominent ontologies for modeling events. This ontology contains a class *event* related to the *agent*, *place*, *factors*, *products*, and *time*. Events are represented as reified n-ary relationships. Although an event can be composed of subevents, the ontology does not directly provide the capability to specialize events by type.

A Simple Event Model (SEM) [4] achieves interoperability with datasets from various domains. This model has four core classes: *Event*, *Actor*, *Place*, and *Time*. Events can be specialized by using the *EventType* class. To make ontology more flexible and capable of storing data from various Internet sources, the authors of SEM have attempted to define a minimum number of restrictions, such as disjoints and functional properties. This results in the use of more SPARQL graph patterns to achieve more efficient reasoning. The ontology also uses external vocabularies for names of places and determining the types of individual concepts.

LODE [46] is an ontology created to store historical events. Specifically, the ontology is restricted to only including events that happen over a limited time and that have been reported as events by some agent, for example, a historian or journalist. LODE has a class *Event* and properties that define the location, time of occurrence, and agents involved. This ontology does not directly support event specialization.

Event Model F [45] is based on DOLCE+Dns Ultralite (simplified version of DOLCE [43] and DnS upper ontologies). This model supports the mereological, causal, and correlational structural relationships between events. The Event Model F was created for interoperability in distributed event-based systems. It contains seven patterns. The participation pattern enables the definition of the participants in an event. The *EventType* class was used differently than in the SEM. It classifies an Event by describing how it should be interpreted or executed (e.g., *DescribedEvent*, *Composite*, *Component*, etc.).

7) SUMMARY

All event ontologies use event reification and model an event as a distinct concept associated with concepts such as participants, location, and time. Some ontologies allow representation of the granularity of events (i.e., sub-events) and dependencies between events. However, only the SEM and Event Model F ontologies provide the apparent possibility of specifying the event type and, thus, specializing events. A significant advantage of these ontologies is that they can easily be extended and adapted to different domains or events. In addition, the structure of these ontologies is compatible with schemes for annotating events in texts, which makes it easier to store the events extracted from texts.

C. RELATED STUDIES

In this section, we analyze studies that investigate and compare ontology design solutions and ways to represent specialization.

1) STUDIES COMPARING MODELING CHOICES OF N-ARY RELATIONS OR SPECIALIZATION STRATEGIES

The first group of studies, directly related to our work, compares the modeling choices of n-ary relations or specialization strategies. To the best of our knowledge, two such studies have been conducted.

Gangemi and Presutti [50] analyzed design patterns to represent n-ary relations in OWL. They compared ontology design patterns in terms of both the quantitative and qualitative characteristics. Sample ontologies involving each pattern were investigated with respect to expressivity, time taken for consistency checking and classification, intuitiveness of representation, usability, robustness, and interoperability.

Hammar [51] explored the effects of strategies for object property specialization in ontology design patterns. The author classified specialization strategies used in design pattern-based and non-design pattern-based ontologies. In addition to the popularity of strategies, this study also investigated how strategy selection affects querying, reasoning performance, and usability from an ontology engineer's perspective. The experiments showed that executing reasoning tasks on datasets adhering to the property-oriented strategy was faster than on the same datasets using ontologies adhering to the class-oriented strategy.

2) STUDIES COMPARING METADATA ATTACHMENT METHODS FOR RDF

Another group of similar studies compares methods to attach metadata for RDF (Resource Description Framework) statements. Unlike event ontologies, where relations are modeled around an event class, the central entity is an RDF statement. For example, in RDF reification, an entity represents an entire triple so that additional triples can be added to describe the reified triple. Some studies have performed experimental comparisons of how the choice of these methods affects the query speed, usability, and other aspects.

Das et al. [52] compared named graph, sub-property, and reification-based models for storing property graphs as RDF in the Oracle Database. Although the focus is on query performance, the authors also analyzed these models in terms of storage cost and SPARQL query formulation.

Hernandez et al. [53] investigated the SPARQL performance of Wikidata using standard RDF reification, n-ary relations, singleton properties, and named graph models. Experiments were conducted on four RDF storage engines. In [54], this work was extended by comparing Wikidata querying in the RDF store, relational, and graph databases. Frey et al. [55] revised and reproduced the latter study and presented a qualitative and quantitative comparison of different RDF-based metadata representation models in three RDF stores: Blazegraph, Virtuoso, and Stardog.

Orlandi et al. [56] provided a benchmark for comparing three popular RDF reification models (i.e., standard RDF reification, singleton property, and RDF-star) in terms of querying performance, storage efficiency, and usability.

Iglesias-Molina et al. [57] investigated how the choice of the RDF reification model (i.e., standard RDF reification, n-ary relations, RDF-star, and qualifiers) impacts knowledge exploration, systematic querying, and knowledge graph embedding. The authors state that none of the models is the most suitable for all scenarios and provide insights into which model is better suited for a given situation.

3) SUMMARY

The reviewed studies investigated how the choice of modeling solution influences certain quantitative and qualitative aspects. However, no comprehensive study has been found that focused on the choice of specialization method and its influence on aspects relevant to developing an NLQ interface for ontology, such as querying and data import performance, vocabulary size, and storage requirements.

III. EVALUATION OF EVENT SPECIALIZATION STRATEGIES

In this section, we explore how selecting an event specialization strategy influences various aspects of the NLQ interface. We do this by analyzing and experimenting with an ontology schema to store event data. The event in our example represents a situation where a person has said something.

We define and investigate four specialization strategies, which later, together with corresponding ontology schemas, are described in detail:

1. Individual-oriented (S1);
2. Class-oriented using OWL definitions (S2);
3. Class-oriented (S3);
4. Property-oriented (S4).

To obtain an overall picture of the effects of specialization strategies, we performed a multi-dimensional comparison based on the following six parameters:

1. the size of ontology schema;
2. the size of vocabulary;
3. the performance of querying;
4. the performance of data import operations;
5. the size of the semantic repository;
6. the query size.

In Section A, we present three ontology schema variants for representing alternative event specialization strategies and compare the complexity of the schemas in terms of the number of OWL constructs. To make it easier to understand the semantics of the elements of the scheme, we also present analogous elements from the FrameNet project.

In Section B, we present the vocabulary for each schema variant and compare their complexities in terms of the number of vocabulary concepts.

In Section C, we explore how alternative specialization strategies influence the size of the semantic repository and performance of data imports.

Finally, in Section D, we investigate how different modeling choices affect the performance of SPARQL queries.

In both previous sections, we aim to identify performance trends by experimenting with repositories containing different numbers of individuals.

A. THE INFLUENCE OF SPECIALIZATION STRATEGY ON EVENT ONTOLOGY SCHEMA

This section presents the ontology schema used in the study. We did not use any of the existing event ontologies presented in Section II-B because we needed additional features to link the ontology with a vocabulary; instead, we only took some of their design solutions. The main idea is event reification to support the involvement and participation of different objects and circumstances.

The ontology schema was designed by considering the questions that should be answered. Users can ask what happened, when it happened, or who was involved in the event. They can also be interested in different types of events, such as political, cultural, sports, etc. Therefore, user's questions can be of different complexity and specificity. The complexity of questions varies by involving a different number of participants (i.e., objects) or data properties for an event, and its specificity varies by involving a different number of event specializations. Below we describe the types of questions of varying complexity and specificity that we considered when developing the ontology schema.

- A question involves the two objects of an event. For example, in the question *What did a person talk about?*, a person and a substance (i.e., a text of what was said) are involved;
- A question involves two objects and one data property. For example, in the question *What did a person talk about in a given year?* a year is a third property, in addition to a person and substance;
- A question involves a specialized event. For example, in the question *What did a person confirm?*, we are interested in talking specialized by its type, that is, confirmation;
- A question involves a specialized event and a data property. For example, *What did a person confirm by 2023?*;
- A question involves an event that is specialized according to two criteria. For example, in the question *What did a person emotionally confirm?*, we are interested in talking specialized by type and manner of speaking;
- A question involves an event specialized according to two criteria and a data property. For example, *What did a person emotionally confirm by 2023?*.

These questions were used to investigate the size and performance of SPARQL queries in Section III-0. Below, we introduce additional design requirements to make the schema more versatile and adaptable to different real-life applications.

- Answers to the questions should provide a document (i.e., a web page from which an event was crawled) where an event was found, together with the date of the document;
- The number of event specializations should not be limited;
- Additional properties of the event (e.g., time, location, etc.) should be supported;
- The schema should be adaptable to different events (e.g., meetings and conference participation).

Fig. 3 presents an ontology schema with its basic classes and relations. The main class is *talking* (i.e., a specific type of event), which is related to the class *agent* to determine who was talking and the *substance* to store a text of what was said. In the FrameNet project, these classes correspond to *Statement*, *Speaker*, and *Message*. A type of event can be assigned to a *talking* to support specialization. All classes inherit the relation to the class *document* from the class *object* to indicate in which document an individual of a particular class was recognized when annotating. To create a more versatile schema that can be easily extended in the future, we also included more general classes (i.e., *state_of_affair* and *agent*).

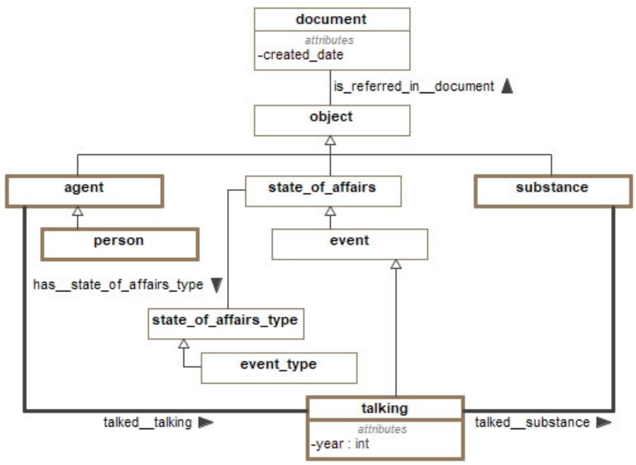


FIGURE 3. The general part of the ontology schema.

Note that we define the names of the object properties by combining a verb and the name of the property's range class. This naming convention allows for uniquely identifying object properties when the same verbs have to be used for different object properties. The ontology schema also has RDF labels to specify corresponding vocabulary concepts for classes and object properties and to link the ontology and the vocabulary. However, we do not display RDF labels to avoid too many details in the schema.

Further, we demonstrate how we extended the schema, resulting in three variants of the ontology schema, as shown in Fig. 4 to 6. The variants presented are expressively the same, allowing us to answer the same questions. To save space, we omitted the general part of the schema and showed only the specific parts of each variant.

1) THE FIRST SCHEMA VARIANT (V1)

The first schema variant (Fig. 4 employs the approach used in the SEM [4], where the *EventType* class is used. It also matches the ''values as sets of individuals'' pattern described in [58].

In this schema variant, we use the class *talking_type* to represent specialization of talking by its type (e.g., agreement, negation, complaint, etc.) and the class *talking_manner* to represent specialization by manner (e.g., emotional, fast, polite, etc.). Note, that the *Manner* element is also present in the FrameNet project, where it has the same meaning.

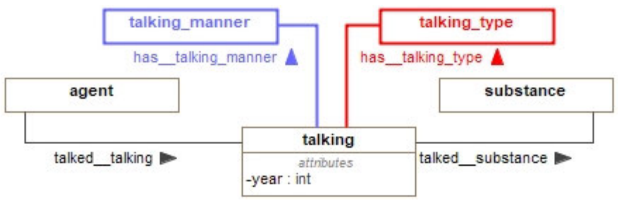


FIGURE 4. The first schema variant (V1).

2) THE SECOND SCHEMA VARIANT (V2)

The second schema variant is shown in Fig. 5. It is based on the approach used in *Event Ontology* [3] to classify events into subclasses.

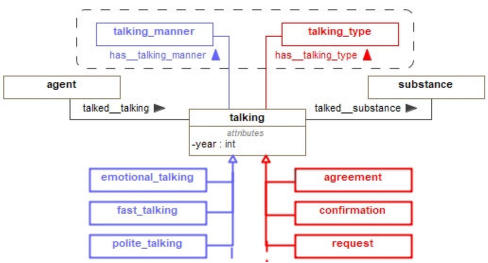


FIGURE 5. The second schema variant (V2).

3) THE THIRD SCHEMA VARIANT (V3)

The third schema variant (Fig. 6) uses property specialization instead of class specialization.

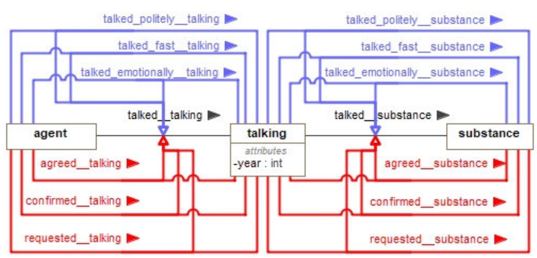


FIGURE 6. The third schema variant (V3).

In this schema variant, each property of the event is specialized into sub-properties for each chosen specialization criterion. In our example, two properties (*talked_talking*and *talked_substance*) are specialized according to two criteria (talking type and manner).

4) EVENT SPECIALIZATION STRATEGIES

We defined four event specialization strategies and denoted them by S. A particular specialization strategy is based on a specific schema variant.

The S1 strategy is based on schema variant V1 and is characterized by the fact that the event class is not directly specialized in the ontology schema. Specialization is implemented by assigning a special subtype value to an event individual from a set of subtype class individuals.

When applying the S2 strategy, the event class is specialized by defining event subclasses with OWL equivalence axioms formulated using property restrictions. In contrast, the S3 strategy, also based on schema variant V2, specifies event subclasses but does not use equivalence axioms.

The S4 strategy is based on the schema variant V3 and uses sub-properties to implement event specialization.

In the following, to maintain the relationship between specialization strategies and schema variants, we will refer to them by adding the schema variant, for example, S1(V1). The specialization strategies are presented in Table 1.

TABLE 1. Specialization strategies and schema variants.

Specialization strategy		Ontology schema	Rule set
S1	Individual-oriented	V1	RDFS
S2	Class-oriented using OWL definitions	V2	OWL-Horst
S3	Class-oriented	V2	RDFS
S4	Property-oriented	V3	RDFS

Note that strategies S2 and S3 use the same schema variant but require different rulesets. In the case of S2, subclasses are inferred from the explicitly defined relations with *Event-Type*(i.e., *talking_type* or *talking_manner*) individuals, using equivalence axioms. For example, the individual of *talking* can be classified as an *agreement* if its properties satisfy the conditions of the following axiom (written in Manchester syntax):

$$\textit{Equivalent to} : \textit{talking AND}$$
$$(\textit{has\_\_talking\_type VALUE agreement\_talking\_type})$$

For these axioms to work, the repository in which the specialization strategy is implemented should use a richer ruleset, OWL-Horst.

In the case of the S3 strategy, subclasses are assigned explicitly, and the RDFS ruleset is enough.

5) THE EVALUATION OF THE COMPLEXITY OF THE SCHEMAS

In this section, we investigate how specialization strategies influence the complexity of an event ontology schema. We calculated the relative complexity score S_{ONT} to compare the schema complexity in terms of the number of constructs (i.e., classes, subclasses, etc.). First, we derive an individual complexity measure, C_{ONT} for every specialization strategy *Sx*. It represents the number of ontology elements required using a given number of specializations N. For example, when specializing the individuals of the talking class by type and manner of talking, the value of N is 2. After deriving the C values, we mapped them into a relative score R in the interval [1;5] using Equation (1).

$$R = 6 - \left(\frac{C - C_{min}}{C_{max} - C_{min}} * 4 + 1\right) \tag{1}$$

A relative value of *1* indicates the worst (highest) complexity, and *5* indicates the best (lowest) complexity. Table 2 presents the complexity estimations C_{ONT} and relative scores R_{ONT}.

TABLE 2. Complexity estimation of ontology schemas.

Ontology elements	Specialization strategy			
	S1(V1)	S2(V2)	S3(V2)	S4(V3)
Classes	N	N		
Subclasses		N	N	
Object properties	N	N		
Sub-object properties				2N
Individuals	N	N		
Axioms		N		
C_{ONT}	3N	5N	N	2N
$\boldsymbol{R_{ONT}}$	**3.0**	**1.0**	**5.0**	**4.0**

The table shows that specialization strategy S2(V2) requires the largest number of elements (five) for each specialization, including OWL axioms for deriving subclasses. In this strategy, an increased number of event specializations increases the number of elements in the ontology schema to a greater extent than in the other schema variants.

S1(V1) has an ontology schema with a similar S2(V2) structure. It also uses individual concepts, but does not use subclasses or axioms. Each specialization has three ontological elements.

At the other end of the spectrum, S3(V2) and S4(V3) are notable for their simplicity. The ontology schemas for these specialization strategies require only one or two elements for each specialization.

B. THE INFLUENCE OF EVENT SPECIALIZATION STRATEGY ON A VOCABULARY

In this section, we investigate how the selected specialization strategy influences vocabulary complexity. Minimizing the maintenance effort needs to be considered. We express the complexity in the number of SBVR vocabulary elements required for each specialization. We present the vocabulary for each ontology schema variant because they must be compatible.

Table 3 presents the general vocabulary. This part corresponds to the concepts specified in the general ontology schema. It also has definitions that bridge the structural gap between questions and the ontology schema. For example, the vocabulary contains a verb concept agent *talked* substance, which is defined as agent *talked* talking that *talked* substance, connecting classes *agent* and *substance* directly. This definition combines the two verb concepts used to form graph patterns in a SPARQL query [14].

Table 4 presents the vocabulary specific to the **schema variant V1**. It contains specialized verb concepts defined using the individual concepts of *talking_type* and *talking_manner* to support specialized questions.

Table 5 introduces the vocabulary specific to **schema variant V2**. Specialized questions were formulated using the event subclasses. Notably, the definitions in this variant are shorter, a distinct difference from variant V1, owing to the non-use of individuals of *talking_type* and *talking_manner*.

Table 6 presents the vocabulary specific to the **schema variant V3**. This vocabulary uses three verb concepts for each specialization and hierarchy of verb concepts.

TABLE 3. Fragment of the common part of SBVR vocabulary.

```
object
agent
 General_concept: object
person
 General_concept: agent
state_of_affairs
 General_concept: object
event
 General_concept: state_of_affairs
talking
 General_concept: event
substance
 General_concept: object
state_of_affairs_type
event_type
 General_concept: state_of_affairs_type
object is_referred_in document
agent talked talking
talking talked substance
agent talked substance
 Definition: agent talked talking that
                    talked substance
state_of_affairs has state_of_affairs_type
year
 General_concept: number
 Concept_type: role
talking talked_in year
  Concept_type:  property_association
```

TABLE 4. Fragment of SBVR vocabulary for the schema variant V1.

```
talking type
 General_concept: event type
talking has talking type
talking manner
 General_concept: event type
talking has talking manner

Confirmation talking type
 General_concept: talking type
agent confirmed substance
 General_concept: agent talked substance
 Definition: agent talked talking that
                   talked substance and
                   has talking type
                   Confirmation talking type

Emotional talking manner
 General_concept: talking manner
agent talked emotionally substance
 General_concept: agent talked substance
 Definition: agent talked talking that
                   talked substance and
                  has talking manner
                  Emotional talking manner
```

To compare the complexity of vocabularies, we used the same approach to compare the complexity of the ontology schemas. We derive an individual complexity measure, C_{VOC} for every specialization strategy. We then calculated

TABLE 5. Fragment of SBVR vocabulary for the schema variant V2.

```
confirmation
 General_concept: talking
agent confirmed substance
 General_concept: agent talked substance
 Definition: agent talked confirmation that
                   talked substance

emotional talking
 General_concept: talking
agent talked emotionally substance
 General_concept: agent talked substance
 Definition: agent talked emotional talking
                  that talked substance
```

TABLE 6. Fragment of SBVR vocabulary for the schema variant V3.

```
agent confirmed talking
 General_concept: agent talked talking
talking confirmed substance
 General_concept: talking talked substance
agent confirmed substance
 General_concept: agent talked substance
 Definition: agent confirmed talking that
                  confirmed substance
agent talked_emotionally_talking
 General_concept: agent talked talking
talking talked_emotionally_substance
 General_concept: talking talked substance
agent talked_emotionally_substance
 General_concept: agent talked substance
 Definition: agent talked_emotionally talking
                  that talked_emotionally substance
```

the relative complexity score R_{VOC} using a straightforward adaptation of formula (1) (replacing C with C_{VOC}).

A comparison is presented in Table 7. Each event specialization for strategy S1(V1) requires five SBVR elements: one noun concept, one verb concept, one compound definition, and one individual concept. This was the highest number; therefore, this variant had the lowest score of 1. This means that an increased number of event specializations in this variant increases the number of vocabulary elements relative to other schema variants.

Variants S2(V2) and S3(V2) used the same vocabulary with two elements for each specialization and had the highest score of 5. Variant S4(V3) is worse with four elements.

TABLE 7. Complexity estimation of SBVR vocabulary variants.

Vocabulary elements	Specialization strategy			
	S1(V1)	S2(V2)	S3(V2)	S4(V3)
Noun concepts	N	N	N	
Verb concepts	N			3N
Definitions	2N	N	N	N
Indiv. concepts	N			
C_{VOC}	5N	2N	2N	4N
S_{VOC}	**1.0**	**5.0**	**5.0**	**2.3**

C. THE INFLUENCE OF EVENT SPECIALIZATION STRATEGY ON A REPOSITORY SIZE AND THE PERFORMANCE OF DATA IMPORT OPERATIONS

In this section, we explore how the size of the semantic repository and ontology data import time depends on the event specialization strategy. To assess the dependence of performance on the amount of data, we created 20 repositories, that is, five repositories with 2, 4, 6, 8, and ten million individuals of class *talking* for each specialization strategy. We determined these numbers experimentally to allow the observation of trends in the performance. As the relationship between the performance and the number of individuals is linear, we decided that the largest repository with 10 million individuals was sufficient.

To store the repositories, we used the free version of the GraphDB semantic graph database running on a local machine with the following characteristics: Intel Core i-5-4570 CPU, 3.20 GHz, 16 GB RAM, and 500 GB SSD.

The data used in the experiment (i.e., individuals of classes *person*, *talking*, *substance*, etc.) were generated by simulating the annotation of a single document from a news portal. Hence, in a single iteration, from two to five individuals of class *talking* were generated. They were assigned a single instance of a class *person* with a name and surname chosen randomly from a dictionary. The type and manner of *talking* were also randomly set. Finally, all the generated individuals were assigned to an individual in the class *document*.

After creating and filling the repositories with the initial data, we began the experiments to measure the data import times. For each repository, we measured the time required to import an additional data set consisting of 100,000 individuals of class *talking* and their related individuals. The goal was to simulate a real-world scenario in which data are batch-appended to a repository containing a certain amount of data. We repeated each import procedure ten times and calculated the average import time. Before each iteration, we restored the repository to its initial state and executed warm-up queries to achieve the most realistic results. All resources required for running the experiments can be found at https://github.com/algirdassukys/event-specialization.

Fig. 7 presents the total starting number of triples in each repository. Repositories of specialization strategy S2(V2) stand out with the largest number of triples, attributed to their use of the OWL Horst ruleset, which leads to a higher number of inferred triples.

Fig. 8 shows how long it takes to import an additional 100,000 individuals of class *talking*. This is also the longest for S2(V2) because the OWL-Horst ruleset requires more time to run the inference process. It should be noted that S2(V2) and S3(V2) repositories utilize the schema variant V2, where subclasses are used. However, in S3(V2) repositories, the subclasses are explicitly set in the imported data, leading to a significant advantage, and an import process is approximately twice as fast as in S2(V2) repositories.

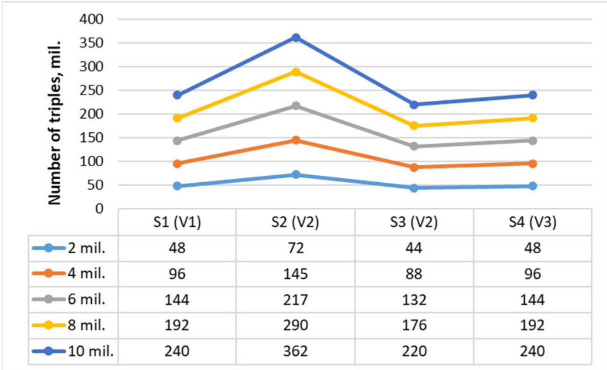


FIGURE 7. Total starting number of triples in repositories (mil).

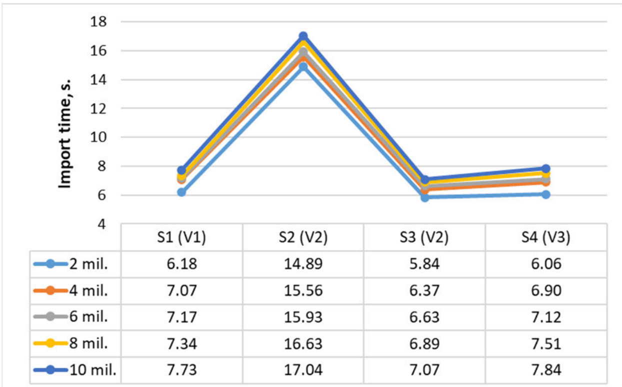


FIGURE 8. Import times of an additional 100,000 individuals of talking class (s).

Table 8 presents a quantitative comparison of the repositories based on their size in terms of triples. Initially, we selected the smallest repositories, S3(V2), to establish a baseline. We then derive the relative size of the remaining specialization strategies from this baseline. Using these values, we calculated the average relative size parameter E_{tr} and final relative size score R_{tr} for each specialization strategy using formula (1) (replacing C with E_{tr}).

TABLE 8. Relative repository size and score.

The number of individuals of the class talking, mil.	Specialization strategy			
	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2	1.1	1.6	<u>1.0</u>	1.1
4	1.1	1.6	<u>1.0</u>	1.1
6	1.1	1.6	<u>1.0</u>	1.1
8	1.1	1.6	<u>1.0</u>	1.1
10	1.1	1.6	<u>1.0</u>	1.1
E_{TR}	1.1	1.6	<u>1.0</u>	1.1
R_{TR}	**4.4**	**1.0**	**<u>5.0</u>**	**4.4**

Table 9 shows a quantitative comparison of repositories based on data import time performance. We calculated the numbers in this table and the final score, R_{imp} using the same logic as in the previous case.

TABLE 9. Relative data import time and score.

The number of individuals of the class talking, mil.	Specialization strategy			
	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2	1.1	2.5	<u>1.0</u>	1.0
4	1.1	2.4	<u>1.0</u>	1.1
6	1.1	2.4	<u>1.0</u>	1.1
8	1.1	2.4	<u>1.0</u>	1.1
10	1.1	2.4	<u>1.0</u>	1.1
E_{IMP}	1.1	2.4	<u>1.0</u>	1.1
R_{IMP}	**4.8**	**1.0**	**<u>5.0</u>**	**4.8**

D. THE INFLUENCE OF EVENT SPECIALIZATION STRATEGY ON THE SIZE AND THE PERFORMANCE OF SPARQL QUERIES

In this section, we explore how the selected event specialization strategy influences the performance and size of the SPARQL queries.

Table 10 provides a general template for the queries used in this experiment. The query results were ordered and limited to reflect their usage in the semantic search portal.

TABLE 10. General template of queries.

```
PREFIX talkings:
<http://semantika.lt/ns/Talkings#>;
PREFIX agents:
<http://semantika.lt/ns/Agents#>;
PREFIX semLT: <http://semantika.lt/ns/SemLT#>;
SELECT ?p ?s ?date
WHERE
{
        <question specific part goes in
        here>
        ?p rdf:type agents:person .
        ?p semLT:is_referred_in__document ?d .
        ?s semLT:is_referred_in__document ?d .
        ?d semLT:created_date ?date .
        ?p rdfs:label "Carl Cameron" .
}
ORDER BY DESC(?date)LIMIT 20
```

Tables 11-16 present triple patterns that are unique to a specific specialization strategy. These queries were written to identify what a particular person said in a certain manner and the document in which this information was found.

TABLE 11. Triple patterns for question (Q1) What did Carl Cameron say?.

Spec. strategy	Triple patterns
All	`?p talkings:talked__talking ?t .` `?t talkings:talked__substance ?s .`

Note that the triple patterns for answering questions Q1 and Q2 are the same for each specialization strategy. This is because these questions do not require the retrieval of

TABLE 12. Triple patterns for question (Q2) What did Carl Cameron say in 2023?.

Spec. strategy	Triple patterns
All	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t talkings:year "2023"^^xsd:int .

TABLE 13. Triple patterns for question (Q3) What did Carl Cameron confirmed?.

Spec. strategy	Triple patterns
S1(V1)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t talkings:has__talking_type talkings:confirmation_talking_type.
S2(V2), S3(V3)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t rdf:type talkings:confirmation .
S4V4)	?p talkings:confirmed__talking ?t . ?t talkings:confirmed__substance ?s .

TABLE 14. Triple patterns for question (Q4) What did Carl Cameron confirmed in 2023?.

Spec. strategy	Triple patterns
S1 (V1)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t talkings:has__talking_type talkings:confirmation_talking_type . ?t talkings:year "2023"^^xsd:int .
S2(V2), S3(V2)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t rdf:type talkings:confirmation . ?t talkings:year "2023"^^xsd:int .
S4(V3)	?p talkings:confirmed__talking ?t . ?t talkings:confirmed__substance ?s . ?t talkings:year "2023"^^xsd:int .

TABLE 15. Triple patterns for question (Q5) What did Carl Cameron emotionally confirmed in 2023?.

Spec. strategy	Triple patterns
S1(V1)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . talkings:has__talking_type talkings:confirmation_talking_type . ?t talkings:has__talking_manner talkings:emotional_talking_manner .
S2(V2), S3(V2)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t rdf:type talkings:confirmation . ?t rdf:type talkings:emotional_talking.
S4(V3)	?p talkings:confirmed__talking ?t . ?t talkings: confirmed__substance ?s . ?p talkings: talked_emotionally__talking ?t . ?t talkings: talked_emotionally__substance ?s .

specialized events. Queries for these questions examine the part of the ontology that is the same in all schemas. Other questions required specialization according to one criterion (Q3 and Q4) or two criteria (Q5 and Q6). The sets of triple patterns differ in these cases, because ontology schemas use different specialization strategies.

TABLE 16. Triple patterns for question (Q6) What did Carl Cameron emotionally confirmed in 2023?.

Spec. strategy	Triple patterns
S1(V1)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t talkings:has__talking_type talkings:confirmation_talking_type . ?t talkings:has__talking_manner talkings:emotional_talking_manner . ?t talkings:year "2023"^^xsd:int .
S2(V2), S3(V2)	?p talkings:talked__talking ?t . ?t talkings:talked__substance ?s . ?t rdf:type talkings:confirmation . ?t rdf:type talkings:emotional_talking. ?t talkings:year "2023"^^xsd:int .
S4(V3)	?p talkings:confirmed__talking ?t . ?t talkings:confirmed__substance ?s . ?p talkings: talked_emotionally__talking ?t . ?t talkings: talked_emotionally__substance ?s . ?t talkings:year "2023"^^xsd:int .

Further, we present an experiment to evaluate the performance of SPARQL query execution. The study was conducted as follows. First, warm-up queries are executed to fill the caches and achieve stable query execution times. Subsequently, a series of queries were executed multiple times, and the average execution times in milliseconds were calculated. The results are shown in Figs. 9–14.

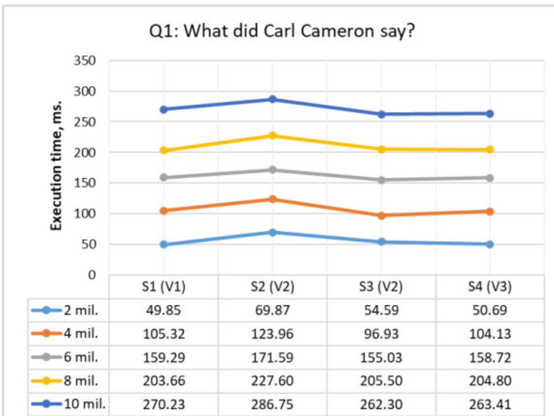
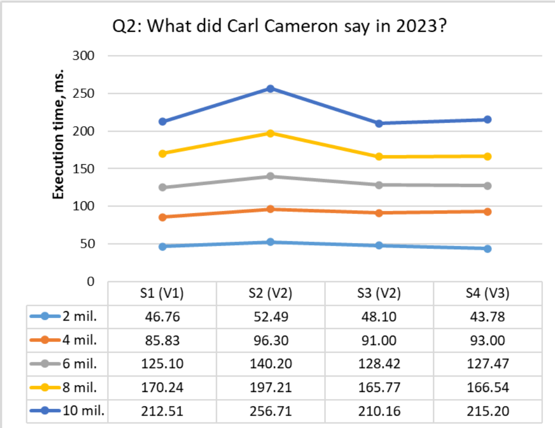


FIGURE 9. Average execution time of Q_1, ms.

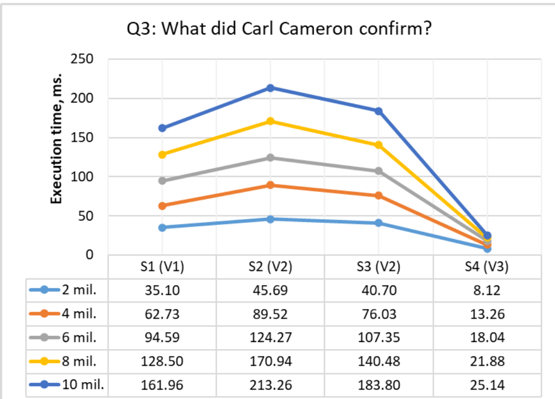
It can be observed that query execution speed is inversely proportional to the size of the repository. This is not surprising since the larger the repository, the larger the graph must be examined during query execution. The relation between the repository size and query execution time appears linear.

Queries Q1 and Q2 have similar execution times in the repositories of all specialization strategies, and queries Q3



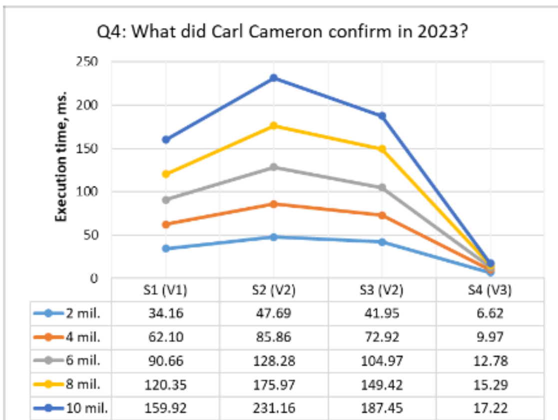
	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2 mil.	46.76	52.49	48.10	43.78
4 mil.	85.83	96.30	91.00	93.00
6 mil.	125.10	140.20	128.42	127.47
8 mil.	170.24	197.21	165.77	166.54
10 mil.	212.51	256.71	210.16	215.20

FIGURE 10. **Average execution time of Q_2, ms.**



	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2 mil.	35.10	45.69	40.70	8.12
4 mil.	62.73	89.52	76.03	13.26
6 mil.	94.59	124.27	107.35	18.04
8 mil.	128.50	170.94	140.48	21.88
10 mil.	161.96	213.26	183.80	25.14

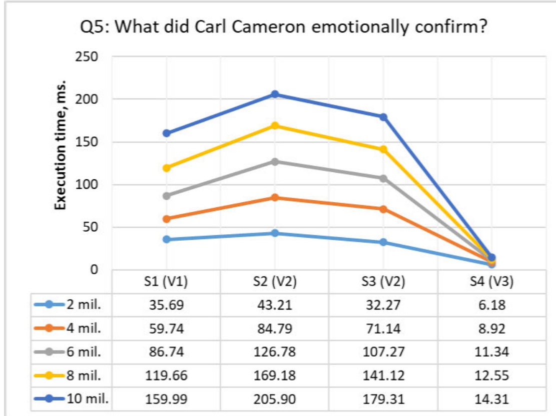
FIGURE 11. **Average execution time of Q_3, ms.**



	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2 mil.	34.16	47.69	41.95	6.62
4 mil.	62.10	85.86	72.92	9.97
6 mil.	90.66	128.28	104.97	12.78
8 mil.	120.35	175.97	149.42	15.29
10 mil.	159.92	231.16	187.45	17.22

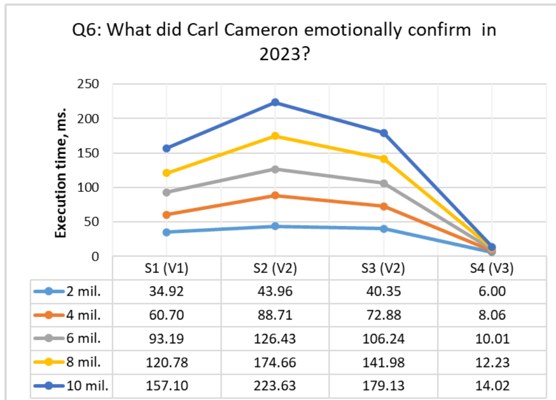
FIGURE 12. **Average execution time of Q_4, ms.**

to Q6 (i.e., where specialized events are queried) are significantly faster in the repositories of strategy S4. This can be explained by the fact that the S4 repositories use an object property specialization strategy. This strategy allows us to find specialized individuals in the *talking* class (i.e., *confirmation*, *emotional confirmation*) directly, without the need to examine all individuals of the *talking* class.



	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2 mil.	35.69	43.21	32.27	6.18
4 mil.	59.74	84.79	71.14	8.92
6 mil.	86.74	126.78	107.27	11.34
8 mil.	119.66	169.18	141.12	12.55
10 mil.	159.99	205.90	179.31	14.31

FIGURE 13. **Average execution time of Q_5, ms.**



	S1 (V1)	S2 (V2)	S3 (V2)	S4 (V3)
2 mil.	34.92	43.96	40.35	6.00
4 mil.	60.70	88.71	72.88	8.06
6 mil.	93.19	126.43	106.24	10.01
8 mil.	120.78	174.66	141.98	12.23
10 mil.	157.10	223.63	179.13	14.02

FIGURE 14. **Average execution time of Q_6, ms.**

To perform the quantitative comparison, we first selected the specialization strategies with the fastest queries in a particular repository size and assigned them a relative score E_Q of 1. For example, query Q1 is the fastest in the S1 strategy when the repository has 2 million individuals. The relative scores E_Q for other specialization strategies were calculated by dividing the average execution time by the average execution time of the fastest query. Finally, the E_Q values were averaged to derive the relative query execution time, E_{RQ} for each specialization strategy. The results of this comparison are listed in Table 17.

To derive a unified query performance score R_{RQ}, we calculated the performance score for each specialization strategy using Formula (1) (replacing C with E_{RQ}) with question-specific E_{RQ} values. We then averaged these values to derive R_{RQ}, as shown in Table 18.

IV. ANALYSIS OF THE EVALUATION RESULTS

Fig. 15 presents a visual summary of the experimental results as a multi-dimensional graph. Each polygon in the chart represents an evaluation of a particular specialization. The axes of the graph represent a relative comparison of various parameters affected by alternative event specialization strategy, and allow the choice of the best alternative for a reader's

TABLE 17. **Relative query execution time.**

The number of individuals of the class talking, mil.	Specialization strategy			
	S1(V1)	S2(V2)	S3(V2)	S4(V3)
Question Q1				
2	1.00	1.40	1.10	1.02
4	1.09	1.28	1.00	1.07
6	1.03	1.11	1.00	1.02
8	1.00	1.12	1.01	1.01
10	1.03	1.09	1.00	1.00
E_{RQ}	**1.03**	**1.20**	**1.02**	**1.02**
Question Q2				
2	1.07	1.20	1.10	1.00
4	1.00	1.12	1.06	1.08
6	1.00	1.12	1.03	1.02
8	1.03	1.19	1.00	1.00
10	1.01	1.22	1.00	1.02
E_{RQ}	**1.02**	**1.17**	**1.04**	**1.03**
Question Q3				
2	4.32	5.63	5.01	1.00
4	4.73	6.75	5.73	1.00
6	5.24	6.89	5.95	1.00
8	5.87	7.81	6.42	1.00
10	6.44	8.48	7.31	1.00
E_{RQ}	**5.32**	**7.11**	**6.09**	**1.00**
Question Q4				
2	5.16	7.20	6.34	1.00
4	6.23	8.61	7.31	1.00
6	7.09	10.04	8.21	1.00
8	7.87	11.51	9.77	1.00
10	9.29	13.42	10.89	1.00
E_{RQ}	**7.13**	**10.16**	**8.50**	**1.00**
Question Q5				
2	5.78	6.99	5.22	1.00
4	6.70	9.51	7.98	1.00
6	7.65	11.18	9.46	1.00
8	9.53	13.48	11.24	1.00
10	11.18	14.39	12.53	1.00
E_{RQ}	**8.17**	**11.11**	**9.29**	**1.00**
Question Q6				
2	5.82	7.33	6.73	1.00
4	7.53	11.01	9.04	1.00
6	9.31	12.63	10.61	1.00
8	9.88	14.28	11.61	1.00
10	11.21	15.95	12.78	1.00
E_{RQ}	**8.75**	**12.24**	**10.15**	**1.00**

situation. On each axis, one represents the most negative and five represents the most positive evaluations.

We measured these parameters to answer the research questions presented in the Section I. Table 19 links the experiment's results with the research questions and lists the parameters used to answer them.

Regarding the first research question RQ1, the individual-oriented specialization strategy S1(V1) describes the rules for bridging the semantic gap between questions and an ontology in a vocabulary. This requires the lengthiest vocabulary, leading to the highest vocabulary modification cost. By contrast, the S2(V2) strategy describes the rules for bridging the semantic gap in an ontology schema using derivation rules, making it more complex. Because the vocabulary in this strategy is freed from rules, it is much simpler.

TABLE 18. **Query performance score.**

Questions	Specialization strategy			
	S1(V1)	**S2(V2)**	**S3(V2)**	**S4(V3)**
Q1	4.8	1.0	5.0	4.9
Q2	5.0	1.0	4.6	4.9
Q3	2.2	1.0	1.7	5.0
Q4	2.3	1.0	1.7	5.0
Q5	2.2	1.0	1.7	5.0
Q6	2.2	1.0	1.7	5.0
$R_{RQ} = avg(Q1{:}Q6)$	**3.6**	**1.0**	**3.3**	**4.9**

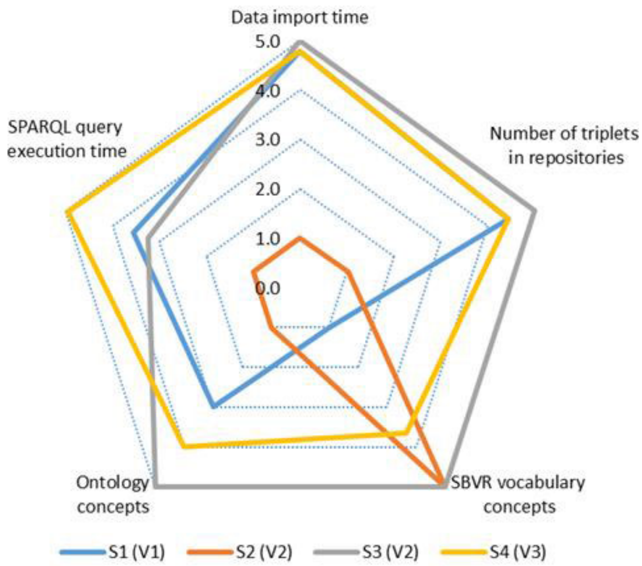


FIGURE 15. **A visual summary of experiment results.**

TABLE 19. **Linking results with research questions.**

Research question	Parameters measured	Relative evaluation of specialization strategies			
		S1(V1)	**S2(V2)**	**S3(V2)**	**S4(V3)**
RQ1	The cost of vocabulary modification (the number of vocabulary concepts)	very high	very low	very low	average
RQ1	The cost of ontology schema modification (the number of ontology concepts)	average	very high	very low	low
RQ2	The size of data (the number of triples in repositories)	small	very large	very small	small
RQ2	Data import time	very short	very long	very short	very short
RQ3	SPARQL query performance	fast	very slow	mode-rate	very fast

The S3(V2) strategy allows the simplest ontology schema (requires only one element to add for each event specialization). Its structure is also the least distant from the way questions are formulated, which is why this strategy also uses a relatively simpler vocabulary. For S4(V3), the ontology schema was also relatively simple, but the vocabulary required additional elements to bridge the semantic gap.

Regarding the second research question, all strategies except S2(V2) are relatively fast for data-import operations. This is because S2(V2) infers additional triples using derivation rules, significantly reducing performance and leading to a notable increase in the number of triples.

Regarding the third research question, query performance is similar in all strategies when specialization is not used in queries. However, when querying for specialized events, S4(V3) exhibited significantly higher performance. This is because triple patterns of SPARQL queries in this strategy map to sub-object properties, resulting in queries immediately finding specialized events without the need to examine all events in a repository.

At the other end of the spectrum, the lowest performance is in S2(V2), which can be explained by the fact that the repositories in this strategy are the largest; therefore, more triples need to be examined when executing a query.

The individual-oriented approach S1(V1) is recommended in most cases. It is especially desirable when the ontology schema is rarely changed because many new elements are needed in both the ontology schema and vocabulary.

Both strategies S2(V2) and S3(V2) use a class hierarchy. The experimental results show that classification using the derivation rules in S2(V2) does not offer any advantage, and this strategy should be avoided.

In summary, although the differences between some of the specialization strategies in some of the studied characteristics are not very large, none of the studied strategies performed best in all aspects studied. However, the results obtained provide some insights for NLQ interface developers. First, before choosing an event specialization strategy, we recommend developers consider the needs of the users of the NLQ interface being developed and clarify and prioritize the requirements for system development, maintenance, and query performance. Recommendations for choosing an event specialization strategy would be as follows:

- In the case of an obvious need to regularly modify the knowledge base of the NLQ interface system by expanding the ontology schema with new event types, the S3(V2) strategy would be the best choice. This strategy, in contrast to S1(V1) and S2(V2), requires the least effort and time to specify ontology concepts and SBVR vocabulary concepts. However, the S3(V2) strategy should not be chosen if query performance is the highest priority. Then an alternative could be strategy S4(V3), which does not cause difficulties when the ontology schema changes, but also ensures the highest performance of queries.
- If the knowledge base of the operated NLQ system needs to be constantly supplemented with actual data to ensure timely information about events to the user, then we would also suggest choosing the S3(V2) strategy. A good alternative would be S1(V1) or S4(V3) since the actual data import time generated by them is slightly longer than S3(V2). The S2(V2) strategy would be the worst choice.
- We recommend choosing the S3(V2) strategy if it is important for the owners of the NLQ system that the costs of data storage do not grow too quickly. These costs can increase as the knowledge base is constantly updated with new facts. Strategy S3(V2) (similarly S1(V1) and S4(V3)) is characterized by generating the lowest number of triplets for each specialized event fact (about 60% less than S2(V2)). Therefore, when filling the knowledge base with new facts, the need for storage resources will grow the slowest.
- If query performance is an essential feature that NLQ system developers must focus on, we recommend S4(V3) when considering an event specialization strategy. To our knowledge, the S4(V3) strategy is the least common specialization strategy. We observed the largest query performance increase using this specialization strategy when querying specialized events. Owing to the fundamental specifics of searching in a graph, the use of object property specializations allows for a significant reduction in query execution times when searching for events with a narrower meaning. In addition, this strategy, like S3(V2), is less distant from natural language formulations than S1(V1).

V. LIMITATIONS

This study has several limitations. First, it was conducted using a single type of RDF triple store, that uses a forward-chaining reasoning strategy or materialization and computes all inferred statements at the data load time. This increases the data load time to some extent but results in an improved query execution time compared to on-request inference. Choosing a triple store with a backward-chaining or hybrid reasoning strategy (e.g., OpenLink Virtuoso, Stardog, Jena TDB) will result in different queries and data import performance.

The use of SBVR vocabulary as a lexicon also limited this study. This is our unique solution, and in reality, it is more likely that a different form of lexicon will be used. In such a case, the results of vocabulary size evaluation will not be directly applicable. However, they allow an approximate assessment of how the lexicon size depends on the chosen schema. All other results were independent of lexicon form.

It is also worth noting that the n-ary relations in OWL can be represented differently. Our experimental ontologies are based on the *Situation* pattern (according to [50]) because it is one of the most commonly used in ontologies for representing events. Other patterns are beyond the scope of this study.

VI. SUMMARY AND CONCLUSION

We compared four strategies for event specialization in OWL ontologies and the impact of their choice on various aspects of the natural language interface. First, for each strategy, we presented ontology schemas and vocabularies needed for the natural language interface to bridge the semantic gap between the structure of ontology and natural language questions. We then presented experiments to investigate how the choice of these strategies affects the complexity of ontology schemas and vocabularies, the performance of querying and data import, and the size of semantic repositories. Experiments have shown that there is no single best strategy. We discussed the advantages and disadvantages of each strategy and make recommendations on the conditions under which a particular approach is most suitable. In summary, we conclude that an NLQ interface developer should consider the desired priority features of the NLQ being developed before deciding on an event specialization strategy.

In the future, we plan to compare the influence of specialization strategies using RDF repositories of other types based on a backward chain reasoning. We also plan to investigate the possibility of applying large language models to improve the performance of some components of the NLQ interface, such as natural language question interpretation.

REFERENCES

[1] C. Sammut and G. I. Webb, "Information retrieval," in *Encyclopedia of Machine Learning*. Boston, MA, USA: Springer, 2010, p. 548, doi: 10.1007/978-0-387-30164-8_403.

[2] H. Ji, "Information extraction," in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Ed. Boston, MA, USA: Springer, pp. 1476–1481, doi: 10.1007/978-0-387-39940-9_204.

[3] Y. Raimond and S. Abdallah, "The event ontology," Centre Digit. Music Queen Mary Univ. London, London, U.K., Tech. Rep., 2007. [Online]. Available: https://motools.sourceforge.net/event/event.html

[4] W. R. van Hage, V. Malaisé, R. Segers, L. Hollink, and G. Schreiber, "Design and use of the simple event model (SEM)," *J. Web Semantics*, vol. 9, no. 2, pp. 128–136, Jul. 2011.

[5] F. H. Rodrigues and M. Abel, "What to consider about events: A survey on the ontology of occurrents," *Appl. Ontol.*, vol. 14, no. 4, pp. 343–378, Nov. 2019, doi: 10.3233/ao-190217.

[6] S. Guan, X. Cheng, L. Bai, F. Zhang, Z. Li, Y. Zeng, X. Jin, and J. Guo, "What is event knowledge graph: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7569–7589, Jul. 2022.

[7] Z. Vendler, "Verbs and times," *Phil. Rev.*, vol. 66, no. 2, p. 143, Apr. 1957.

[8] A. P. D. Mourelatos, "Events, processes, and states," *Linguistics Philosophy*, vol. 2, no. 3, pp. 415–434, 1978.

[9] J. Pustejovsky, "The syntax of event structure," *Cognition*, vol. 41, nos. 1–3, pp. 47–81, Dec. 1991.

[10] D. Davidson, "The logical form of action sentences," in *The Logic of Decision and Action*. Pittsburgh, PA, USA: Univ. of Pittsburgh Press, 1967, pp. 81–95.

[11] Linguistic Data Consortium. (2005). *ACE (Automatic Content Extraction) English Annotation Guidelines for Events*. [Online]. Available: https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-events-guidelines-v5.4.3.pdf

[12] J. Aguilar, C. Beller, P. McNamee, B. V. Durme, S. Strassel, Z. Song, and J. Ellis, "A comparison of the events and relations across ACE, ERE, TAC-KBP, and framenet annotation standards," in *Proc. 2nd Workshop EVENTS, Definition, Detection, Coreference, Represent.*, 2014, pp. 45–53.

[13] J. Karpovič, G. Krikščiūnienė, L. Ablonskis, and L. Nemuraitė, "The comprehensive mapping of semantics of business vocabulary and bussiness rules (SBVR) to OWL 2 ontologies," *Inf. Technol. Control*, vol. 43, no. 3, pp. 289–302, Sep. 2014.

[14] A. Šukys, L. Nemuraitė, and R. Butkienė, "SBVR based natural language interface to ontologies," *Inf. Technol. Control*, vol. 46, no. 1, pp. 118–137, 2017.

[15] T. Vileiniškis, A. Šukys, and R. Butkiene, "An approach for semantic search over Lithuanian news website corpus," in *Proc. 7th Int. Joint Conf. Knowl. Discovery, Knowl. Eng. Knowl. Manage. (IC3K)*, vol. 1, Nov. 2015, pp. 57–66.

[16] F. Őzcan, A. Quamar, J. Sen, C. Lei, and V. Efthymiou, "State of the art and open challenges in natural language interfaces to data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2020, pp. 2629–2636.

[17] A. Quamar, V. Efthymiou, C. Lei, and F. Őzcan, "Natural language interfaces to data," *Found. Trends Databases*, vol. 11, no. 4, pp. 319–414, 2022.

[18] D. Damljanovic, V. Tablan, and K. Bontcheva, "A text-based query interface to owl ontologies," in *Proc. 6th Int. Conf. Lang. Resour. Eval.*, 2008, pp. 1–8.

[19] E. Kaufmann, A. Bernstein, and R. Zumstein, "Querix: A natural language interface to query ontologies based on clarification dialogs," in *Proc. In: 5th Int. Semantic web Conf. (ISWC 2006)*. Princeton, NJ, USA: Citeseer, 2006, pp. 980–981.

[20] D. Diefenbach, A. Both, K. Singh, and P. Maret, "Towards a question answering system over the semantic web," *Semantic Web*, vol. 11, no. 3, pp. 421–439, Apr. 2020.

[21] D. Diefenbach, V. Lopez, K. Singh, and P. Maret, "Core techniques of question answering systems over knowledge bases: A survey," *Knowl. Inf. Syst.*, vol. 55, no. 3, pp. 529–569, Jun. 2018.

[22] A. Freitas and E. Curry, "Natural language queries over heterogeneous linked data graphs: A distributional-compositional semantics approach," in *Proc. 19th Int. Conf. Intell. User Interfaces*, Feb. 2014, pp. 279–288.

[23] K. Höffner, S. Walter, E. Marx, R. Usbeck, J. Lehmann, and A.-C. N. Ngomo, "Survey on challenges of question answering in the semantic web," *Semantic Web*, vol. 8, no. 6, pp. 895–920, Aug. 2017.

[24] A. Simitsis, G. Koutrika, and Y. Ioannidis, "Précis: From unstructured keywords as queries to structured databases as answers," *VLDB J.*, vol. 17, no. 1, pp. 117–149, Nov. 2007.

[25] G. Zenz, X. Zhou, E. Minack, W. Siberski, and W. Nejdl, "From keywords to semantic queries incremental query construction on the semantic web," *J. Web Semantics*, vol. 7, no. 3, pp. 166–176, 2009.

[26] F. Li and H. V. Jagadish, "NaLIR: An interactive natural language interface for querying relational databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2014, pp. 709–712.

[27] L. Blunschi, C. Jossen, D. Kossman, M. Mori, and K. Stockinger, "SODA: Generating SQL for business users," 2012, *arXiv:1207.0134*.

[28] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. R. Mittal, and F. Özcan, "ATHENA: An ontology-driven system for natural language querying over relational data stores," *Proc. VLDB Endowment*, vol. 9, no. 12, pp. 1209–1220, Aug. 2016.

[29] J. Sen, C. Lei, A. Quamar, F. Özcan, V. Efthymiou, A. Dalmia, G. Stager, A. Mittal, D. Saha, and K. Sankaranarayanan, "ATHENA++: Natural language querying for complex nested SQL queries," *Proc. VLDB Endowment*, vol. 13, no. 12, pp. 2747–2759, Aug. 2020.

[30] S. Walter, C. Unger, P. Cimiano, and D. Bär, "Evaluation of a layered approach to question answering over linked data," in *Proc. 11th Int. Semantic Web Conf.*, Boston, MA, USA. Berlin, Germany: Springer, 2012, pp. 362–374.

[31] D. Punjani, K. Singh, A. Both, M. Koubarakis, I. Angelidis, K. Bereta, T. Beris, D. Bilidas, T. Ioannidis, N. Karalis, C. Lange, D. Pantazi, C. Papaloukas, and G. Stamoulis, "Template-based question answering over linked geospatial data," in *Proc. 12th Workshop Geographic Inf. Retr.*, Nov. 2018, pp. 1–10.

[32] (Oct. 2019). *Semantics of Bus. Vocabulary and Bus. Rules (SBVR)*. [Online]. Available: https://www.omg.org/spec/SBVR/1.5/PDF

[33] R. Piryani, N. Aussenac-Gilles, and N. Hernandez, "Comprehensive survey on ontologies about event," in *Proc. ESWC Workshops Semantic Methods Events Stories*, vol. 3443, May 2023, pp. 1–15.

[34] T. Parsons, "Events in the semantics of english: A study in subatomic semantics," MIT Press, Cambridge, MA, USA, Tech. Rep., p. 334, 1990.

[35] Z. Song, A. Bies, S. Strassel, T. Riese, J. Mott, J. Ellis, J. Wright, S. Kulick, N. Ryant, and X. Ma, "From light to rich ERE: Annotation of entities, relations, and events," in *Proc. The 3rd Workshop EVENTS, Definition, Detection, Coreference, Represent.*, 2015, pp. 89–98.

[36] J. Pustejovsky, J. M. Castano, R. Ingria, R. Sauri, R. J. Gaizauskas, A. Setzer, and D. R. Radev, "TimeML: Robust specification of event and temporal expressions in text," Rep., 2003, pp. 28–34.

[37] A. Cybulska and P. Vossen, "Guidelines for ECB+ annotation of events and their coreference," Netw. Inst., VU Amsterdam, Amsterdam, The Netherlands, Tech. Rep. NWR-2014-1, 2014.
[38] T. O'Gorman, K. Wright-Bettner, and M. Palmer, "Richer event description: Integrating event coreference with temporal, causal and bridging annotation," in *Proc. 2nd Workshop Comput. News Storylines (CNS)*, 2016, pp. 47–56.
[39] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The Berkeley FrameNet project," in *Proc. 17th Int. Conf. Comput. Linguistics*, vol. 1, 1998, p. 86.
[40] M. Rospocher, M. van Erp, P. Vossen, A. Fokkens, I. Aldabe, G. Rigau, A. Soroa, T. Ploeger, and T. Bogaard, "Building event-centric knowledge graphs from news," *J. Web Semantics*, vols. 37–38, pp. 132–151, Mar. 2016.
[41] J. N. Otte, J. Beverley, and A. Ruttenberg, "BFO: Basic formal ontology," *Applied ontology*, vol. 17, no. 1, pp. 17–43, 2022.
[42] G. Guizzardi, A. B. Benevides, C. M. Fonseca, D. Porello, J. P. A. Almeida, and T. P. Sales, "UFO: Unified foundational ontology," *Appl. Ontol.*, vol. 17, no. 1, pp. 167–210, Mar. 2022.
[43] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider, "Sweetening ontologies with DOLCE," in *Proc. Int. Conf. Knowl. Eng. Knowl. Manage.* Cham, Switzerland: Springer, 2002, pp. 166–181.
[44] I. Niles and A. Pease, "Towards a standard upper ontology," in *Proc. Int. Conf. Formal Ontol. Inf. Syst.*, Oct. 2001, pp. 2–9.
[45] A. Scherp, T. Franz, C. Saathoff, and S. Staab, "F–A model of events based on the foundational ontology dolce+DnS ultralight," in *Proc. 5th Int. Conf. Knowl. capture*, Sep. 2009, pp. 137–144.
[46] R. Shaw, R. Troncy, and L. Hardman, "LODE: Linking open descriptions of events," in *Proc. Asian Semantic Web Conference*. Cham, Switzerland: Springer, 2009, pp. 153–167.
[47] A. Ali, S. A. M. Noah, and L. Q. Zakaria, "Representation of event-based ontology models: A comparative study," *Int. J. Comput. Sci. Netw. Secur.*, vol. 22, no. 7, p. 147, 2002.
[48] I. Astrova, A. Koschel, J. Lukanowski, J. L. M. Martinez, V. Procenko, and M. Schaaf, "Ontologies for complex event processing," *Int. J. Comput., Electr., Autom., Control Inf. Eng.*, vol. 8, no. 5, pp. 695–705, 2014.
[49] C. Tzelepis, Z. Ma, V. Mezaris, B. Ionescu, I. Kompatsiaris, G. Boato, N. Sebe, and S. Yan, "Event-based media processing and analysis: A survey of the literature," *Image Vis. Comput.*, vol. 53, pp. 3–19, Sep. 2016.
[50] A. Gangemi and V. Presutti, "A multi-dimensional comparison of ontology design patterns for representing n-ary relations," in *Proc. Int. Conf. Current Trends Theory Pract. Comput. Sci.* Cham, Switzerland: Springer, 2013, pp. 86–105.
[51] K. Hammar, "Ontology design pattern property specialisation strategies," in *Proc. Int. Conf. Knowl. Eng. Knowl. Manag.* Cham, Switzerland: Springer, 2014, pp. 165–180.
[52] S. Das, J. Srinivasan, M. Perry, E. I. Chong, and J. Banerjee, "A tale of two graphs: Property graphs as RDF in Oracle," in *Proc. EDBT*, 2014, pp. 762–773.
[53] D. Hernández, A. Hogan, and M. Krötzsch, "Reifying RDF: What works well with wikidata?" in *Proc. SSWS@ ISWC*, 2015, pp. 32–47.
[54] D. Hernández, A. Hogan, C. Riveros, C. Rojas, and E. Zerega, "Querying wikidata: Comparing sparql, relational and graph databases," in *Proc. The Semantic Web-ISWC 2016: 15th Int. Semantic Web Conf.*, Kobe, Japan. Cham, Switzerland: Springer, 2016, pp. 88–103.
[55] J. Frey, K. Müller, S. Hellmann, E. Rahm, and M.-E. Vidal, "Evaluation of metadata representations in RDF stores," *Semantic Web*, vol. 10, no. 2, pp. 205–229, Jan. 2019.
[56] F. Orlandi, D. Graux, and D. O'Sullivan, "Benchmarking RDF metadata representations: Reification, singleton property and RDF," in *Proc. IEEE 15th Int. Conf. Semantic Comput. (ICSC)*, Jan. 2021, pp. 233–240.
[57] A. Iglesias-Molina, K. Ahrabian, F. Ilievski, J. Pujara, and O. Corcho, "Comparison of knowledge graph representations for consumer scenarios," in *Proc. Int. Semantic Web Conf.* Cham, Switzerland: Springer, Oct. 2023, pp. 271–289.
[58] A. Rector, "Representing specified values in OWL: Value partitions, and value sets, W3C working group 17," Tech. Rep., 2005.

RITA BUTKIENĖ received the B.Sc., M.Sc., and Ph.D. degrees from Kaunas University of Technology, in 1993, 1995, and 2002, respectively. She is currently a Full Professor with the Department of Information Systems, Kaunas University of Technology. Her research interests include information system engineering, ontologies, semantic technologies, databases, distributed ledger, information extraction, and retrieval.

ALGIRDAS ŠUKYS received the Ph.D. degree from Kaunas University of Technology, in 2017. He is currently a Teaching Assistant with the Department of Information Systems, Kaunas University of Technology. In addition to his teaching role, he is engaged in cutting-edge research at the Center of Information Systems Design Technologies. Since 2010, he has been an Analyst and a Developer for e-commerce and national forestry projects. His research interests include ontologies and semantic technologies. He was also interested in technologies and standards for specifying business vocabulary and rules.

LINAS ABLONSKIS received the Ph.D. degree from Kaunas University of Technology. He is currently a Teaching Assistant with the Department of Information Systems, Kaunas University of Technology. He is also engaged in cutting-edge research at the Center of Information Systems Design Technologies. His research interests include ontologies, semantic technologies, and the analysis of natural language texts.

RIMANTAS BUTLERIS received the Ph.D. degree in informatics engineering from Kaunas University of Technology, in 1988. He is currently the Head of the Center of Information Systems Design Technologies and a Full Professor with the Department of Information Systems, Kaunas University of Technology, Lithuania. During his career, he has authored or co-authored more than 170 research articles and participated in more than 20 national and international research projects. In many of those, he was a Leading Researcher. His research interests include requirements engineering, semantic technologies, and business rule modeling.