Mind Mapping and Spreadsheets in Collaborative Design of Knowledge Graphs

Dmitry Kudryavtsev[1](✉), Tatiana Gavrilova[1], Irina Leshcheva[1], Alena Begler[1],
Miroslav Kubelskiy[1], Olga Tushkanova[2]

[1] Graduate School of Management, St. Petersburg University, St. Petersburg, Russia
[2] InfoWings, St. Petersburg, Russia
{d.v.kudryavtsev, gavrilova, leshcheva}@gsom.pu.ru
{alena.begler, mirqube, tushkanova.on}@gmail.com

Abstract. The paper proposes a method for creating a knowledge graph focused on visualization, usage of spreadsheets and group work. A knowledge graph is a set of typed entities (with attributes) related to each other by relationships. Entity, attribute and relationship types are determined by a scheme called ontology. The proposed method was created within the project with an automotive company, which needed systematization, integration and re-use of knowledge about the assembly units of various cars within the company. The created knowledge graph has a complex structure of properties associated with one class and its subclasses, while there is a little number of ontology classes. The need for active involvement of experts in the creation of the knowledge graph determines the scope of the method, since they and their colleagues will use the system in the future (internal customers). The proposed method helped to create a pilot knowledge graph, which includes more than 50 assembly units and about 400 properties.

Keywords: knowledge graph; ontology; ontology engineering.

1 Introduction

A knowledge graph is a set of typed entities (with attributes) relating to one another by typed relationships. Types of entities and relationships are defined in schemas called ontologies [1]. “Ontology is a formal, explicit specification of a shared conceptualization” [2], which consists of a hierarchy of concepts of the subject domain, links between them and constraints that operate within the framework of this model. Ontology is constructed as a network consisting of concepts and connections between them. Connections can be of various types (for example, “is a”, “consists of”, “is executor of”, etc.). The term “ontology” overlaps with the term “knowledge graph”, since when an ontology is populated with instances it becomes a knowledge graph (or a knowledge base). However, the ontology primarily focuses on classes, properties and axioms (TBox), while knowledge graph pays more attention to instances (ABox).

A range of methodologies for ontology development, however, they are not directly applicable to the knowledge graph creation due to their focus on the classes structure and omission of population stage (description of instances). Ontology population, in contrast, is important for the knowledge graph development. Besides existing ontology engineering methodologies are not differentiated based on the type of ontology, but certain applications requires specific ontologies and there is a need for specialized methods. For example, faceted search systems, which are often used in ecommerce and knowledge management, requires light-weight ontologies with many properties of classes (faceted taxonomy). The development method for such ontology type will differ from the creation of heavy-weight ontology for reasoning purposes.

Thus, in line with design science research [3] the requirements for the knowledge graph development method were specified:

- The knowledge graph, the final result of the method application has a complex structure of properties (large amount, hierarchical organization) associated with one class and its subclasses;
- TBox (structure of properties mostly) and ABox (description of instances) are of equal importance;
- The main source of information for creating the knowledge graph is semi-structured data thus, it assumes manual development
- The knowledge graph development often involves experts and users, thus, its development should be collaborative at every stage of ontology development.

The purpose of the paper is to suggest and to describe the method of knowledge graph creation that meets these requirements.

The paper is structured as follows. The next section describes related work. Section 3 proposes for general description of the method for knowledge graph development, and the application and demonstration of the method for the automotive industry assembly units' case. Section 4 is dedicated to the conclusion and discussion of suggested method implication.

2 Related Work

Related work is carried out for ontology engineering methodologies, knowledge graph development methods and within construction of faceted taxonomies. We will briefly review these approaches with respect to the aforementioned characteristics of the applied method: iterations during the building process and the focus on the property structure. Additionally, visualization methods' position in the process is reviewed.

2.1 Ontology Engineering Methodologies

Existing methodologies are focused on ontology schema development, and pay less attention to ontology population (which is important for knowledge graph). Nevertheless, these methodologies constitute the core of knowledge graph development.

These methodologies may be split into universal and domain-specific. Domain-specific frameworks and methodologies for product ontologies [4] focus mainly on the result, rather than process of ontological engineering. None of them fits the aforementioned characteristics, thus, we will not review them in this paper.

Several universal methodologies for ontology engineering are widely disseminated. The most developed and complex universal methodology is the NeOn (www.neon-project.org), which integrates the achievements of previous methodologies [5]. NeOn methodology offers nine scenarios for ontology networks building and two types of processes for ontology development. While scenarios for ontology building are flexible, the development process guidelines are relatively firm and offer only two models: waterfall and iterative-incremental. None of these models allows moving back-and-forth between the development stages within the iteration.

This problem was solved in the Methontology [6] approach by adding the evolving lifecycle, which allows to come back to any phase of development to make changes. On-To-Knowledge [7] applies stricter restrictions on the iterations inside the ontology building and assumes them at the refinement and evaluation stages, which should form a cycle. Similar approach with iterations at the building phase was proposed by the [8] methodology for the ontology creation in SMEs. This methodology focuses on development time reduction and its stages are similar to the NeOn's waterfall model with the several short iterations in the ontology building phase. The same iterative process is applied by the [9] middle-out approach to the ontology terms definition, which allows to reach the balanced level of detalization and eXtreme Design [10] approach, which provides for short iterations and active customer involvement.

The majority of mentioned methodologies support collaborative style of ontology development. They are second generation methodologies focused on centralized development with the team members' role differentiation, and they play positive role in the corporate ontology development [11].

2.2 Knowledge Graph Development

Several attempts to classify methods of knowledge graph design and to create a methodology have been made. For example, [1] describes three tasks related to the knowledge graph usage: knowledge acquisition and integration, storage, and consumption. The first relates to creation of the knowledge graph. At this stage, ontology engineering methods can be applied. There are automatic and manual methods for creating knowledge graph. This paper is devoted to the manual one [12].

2.3 Methods for Construction of Faceted Taxonomies

The required knowledge graph can be used within faceted search system [13] and plays the role of faceted taxonomy. So, methods used for creating faceted taxonomies are applicable for our purposes. This process typically subdivides into facet term extraction and hierarchy construction steps in line with existing ontology development methods. The extraction can be done automatically, semi-automatically or manually [13]. Since we will use heterogeneous semi-structure data (suppliers' catalogs, expert knowledge) and high quality of knowledge graph is required, automatic and semi-automatic methods are not applicable.

2.4 Visualization Techniques Usage

During the ontology development, two visualization techniques were used for communication within the team: mind maps and tables. These formats were chosen for being the most easy-to-understand for domain experts and customers (developers of the system using the knowledge graph). Additionally, both of proved to be efficient as an ontology development support methods.

Mind maps are good for early stages of ontology engineering, as they do not involve any implicit semantics [14]. Additionally, they serve communication goals, as they may easily be understood and edited by domain experts. Nodes were transformed into properties in our methodology mind map; each new level determined subproperty relationships similar to the OntoEdit [14] methodology, where mind map concepts were linked to subConceptOf relationships as well as [15] subClassOf relationships.

Spreadsheets are often used for automatic and semi-automatic ontologies construction [16]. However, they may be used for manual construction as well. Specifically, spreadsheets may support two tasks. The first one is glossary creation, where spreadsheets ease synonyms search and terms connection [17]. The second one is ontology enrichment together with classes' structure identification [18]. Additional benefit of the spreadsheets is their automatically convertibility into ontology.

3 Method and its Application

The VITON (VIsual collecTive development of the ONtontological knowledge graph) method helps to create knowledge graphs with complex structure of properties under conditions specified in the requirement description (see Introduction).

At the top-level, the suggested method follows the logic of existing ontology engineering methodologies; particularly, the NeOn methodology was applied [5] as the basis for the development process with several special features added:

1. Focus on properties and their structure;
2. Usage of mind maps (high-level structure of properties, description of instances) in order to collaborate with experts, future users (internal customers) of the system, since they are not familiar with knowledge engineering methods and tools;
3. Usage of spreadsheets ("Properties – Instances" matrix) to structure properties (about 400), to analyze their usage in instances and to support collaboration within the development team;
4. Strong integration of ontology development with description of instances, where each instance description suggests new properties for the ontology.

The demo-graph building process followed Scenario 2 of the NeOn methodology process [5]. This scenario includes repeated use of the non-ontological resources stage in addition to the core scenario stages: requirement specification, search for candidate knowledge resources for repeated use, scheduling, and development. We reviewed these stages as a direct sequence, except for scheduling phase that took place after requirement specification. Two ontology support activities were performed simultaneously with the development process: knowledge acquisition and documentation. Knowledge acquisition was performed during the analysis of instances (including specification documents, data sheets, and interviews with experts). Documentation was made for different versions of the mind maps, spreadsheets, and ontologies (version control files), decisions (meeting notes, decision track document), and other sources in use (list of knowledge resources, wiki pages with references to instance-related information). The method was applied to make the demo-graph (about 50 instances). After initial graph development, employees of the enterprise will fill it up.

3.1 VITON Method description

This section gives domain-independent description of the method through a possible application scenario.

Ontology requirements specification was performed in close interaction with the customers (automotive company) and domain experts. The requirements main form was the use cases.

Unlike the NeOn methodology, where scheduling is performed after the search of candidate knowledge resources, our project time was limited to three months by the consumer. Thus, we defined scope instead of timeframes and established the team at the project planning stage. Project scope is described in the previous section. Project team involved a group of analysts led by a knowledge engineer. The group consisted of four subgroups:

- A subgroup of analysts includes developers of mind maps.
- A subgroup of ontologists creating the knowledge graph using mind maps in RDF with help of ontology editors.
- A subgroup of domain experts.
- Technical writer-methodologist for the description of internal.

It was needed to differentiate the knowledge engineering part of the team, because several activities should be performed simultaneously, namely, the creation of the mind maps for instances and refinement of the ontology.

According to the NeOn methodology, **reusing of candidate knowledge resources** was performed involving the following activities:

- Search was made for papers describing the domain, standards, and internet resources devoted to the industry (online shops, suppliers' websites).
- The source of the resources was assessed (priority was given to the known providers) and domain experts evaluation.
- Selection was done according to the assess criteria and yielded several resources for re-use.

For the **conceptualization activities (development stage)** mind maps were used as the language of description, and two modelling techniques were applied:

- First, top-down modelling style was applied and ontology mock-up as the mind map was created. It made it possible to identify main properties of a central class and became the main tool for instance description.
- Then, bottom-up style was added. Mind maps were created for the pilot set of instances provided by the customer on the basis of the mockup and semi-structured sources of information about instances and then evaluated by the expert group.

These two processes have been performed iteratively: mind maps for every instance were based on the mockup, and mockup itself was continuously refined according to common patterns identified during instances analysis. Additionally, glossary of terms with definitions was drawn during these activities.

Formalization (development stage) of properties took place and involved spreadsheets, when mind maps were created for the pilot set of instances and ontology mockup was established (i.e. additional instance analysis did not add much to the mockup). Table with properties served as an intermediate stage between mind maps and ontology. The table was mainly aimed at properties hierarchy creation and glossary establishment. Mind maps were transformed to create the table. Transformation technique, similar to the [14, 15] was applied with several additions:

- Instances formed the first row (as a subject in RDF notation).
- Ontology mockup concepts were transformed to the top-level properties, and lower-level concepts were transformed from the instances' mind maps to sub properties. The columns of the table showed it as intended list (as predicates). During transformation, property names were unified according to the chosen naming convention, informal human-readable names were provided for, when labels and description were added.
- The lowest level of the mind map was labeled in the table cells with unit of measurement signs (as objects). All related properties were added to the table together with their dimensions for each instance. For example, if the instance has a property "size" measured in millimeters, the name of property (hasSize) was added in the "Properties" column under relevant group and subgroup of properties and code "mm" was added to the property with AU intersection. In case there was no unit of measurement (for example, qualitative property), plus sign was added.

Table-building involved three activities:

- Initial properties entering according to the mind map ontology mockup.
- Reconstruction of properties structure with respect to their semantic similarity and generalization level.
- Entering of all the instances mockups in the created properties structure, verification and refinement of the structure, adding of labels and descriptions of properties.

Implementation (development stage) followed several steps.

At first, properties were imported. During their transition, joint hierarchy of the properties was subdivided into two separate hierarchies of object properties and data properties with the common upper-level groups and (where possible) subgroups of properties.

Then, class hierarchy was introduced. This hierarchy creation was not a goal of the project, thus, classes were used as storage for different types of values. There are three upper-level classes in the hierarchy:

- CoreEntity for the names of artifacts and organizations.
- QuantitativeValue for the instances describing quantitative values (see further explanation at the nest translation step).
- Reference for the instances describing repeating qualitative values.

Finally, all the entities were imported in the ontology. The entities were one of three types: complex instances (could be split in several parts), their parts and object properties values. Key objects (instances) of analysis were individuals in the CoreEntity subclasses, and Values for object properties – in either Quantitative or Reference subclasses. The values for object properties were linked to the artifacts (exact instances) by the object property. Unique qualitative values like informal descriptions were imported as values for data properties .

3.2 Case Project

The task to create and describe the method for visual collective creation of the ontological knowledge graph arose from the industrial need to develop such graph for an automotive company. Elements of the knowledge graph are assembly units (AUs) of various cars (mainly electric trucks) produced by the company. The created knowledge graph is intended to form the basis for the new corporate knowledge management system, where users (engineer, purchasing manager, economist, etc.) will be able to find an AU they need and relevant information about it. Such system is important for the company since it produces vehicles using mostly AUs available on the market, and it takes engineers much time to search for the information about AUs. The system is supposed to integrate both information about AUs already used in the company and that related to AUs available on the market and contained in the suppliers' catalogs. In future, the knowledge graph may be applied in robotics developed in the company as well. The customer of the method and the knowledge graph is an IT department that performs digitization of all business processes of the enterprise.

Requirements on the method of creating knowledge graph and its deliverable (required AUs' knowledge graph) were specified by the knowledge graph customer (automotive company):

- The required knowledge graph should be focused on a property structure, and a small part of properties is applicable for the top-level class ("AU" or "artifact") and a major part of properties is AU-specific.
- TBox (structure of properties) and ABox (description of AUs/instances) are of equal importance.
- High quality of knowledge graph to use it as a "gold standard data" and for content demonstration.

The team was unequally involved in the development process:

- Analysts (2 persons) were involved in knowledge source search, AUs' mind maps creation, and in mind maps to spreadsheets transformation.
- Ontologists (2 persons) were involved in ontology mockup creation, development of spreadsheet structure, and, mainly, re-use of other knowledge resources and ontology construction and population.
- Domain experts (2 persons) were engaged in the process to check accuracy of the description once new portion of AUs was described, as well as to verify newly-created ontology mockup.
- Methodologist (1 person) consulted the team on the use of methods.

Identified knowledge resources were later used at different stages:

- Core Product Model (CPM) [19] and the product information domains identified in the PLM systems were used as the basis for developing the ontology mockup as a mind map [20].
- ISO 14050:2009(en), ISO 10303-1:1994(en), ISO/IEC 15288:2008, and STEP AP 242 were used for the initial glossary development and terms definition.
- IEC 81346 (the classification of objects according to their purpose or task) and "Unit of measure" list of elements were used as a part of ontology to describe instances at formalization and implementation stages.

Development stage: conceptualization (application of mind maps):

- Top-down modelling style. The initial structure proposed by the customer, as well as the Core Product Model (CPM).The group of analysts created the visual generalized structural mockup of ontology as the mind map; this was done during few iterations and based on discussions with customers. Figure 1 shows the current version of the ontology mockup. It does not contain formalized relationships between the concepts (edges have no labels) as it is dedicated to the presentation of the core concept ("Artifact") aspects of the artifact functioning in the company settings (in nodes).
- Bottom-up style. Figure 2 shows the fragment of the filled ontology mockup for the AU (base-mounted ERC series air compressor). Several important elements are notable in this mind map:
  - New top-level element appeared (marked in gray): "Has connection". This element will further be added into the mockup.
  - Lover-level properties were detailed. For example, "Output" property was detailed by three sub properties: "Noise", "Free air delivery", "Rated pressure".
  - The lowest level values were added. For example, minimum and maximum values in decibels are known for the "Noise" property.

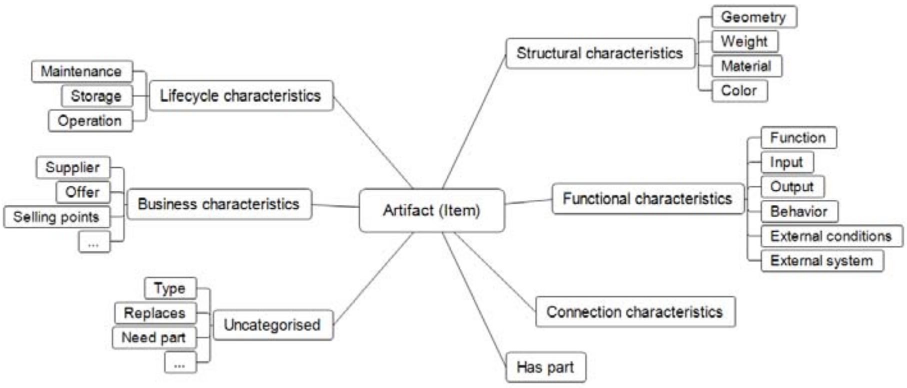


Fig. 1. Current version of the ontology mockup.

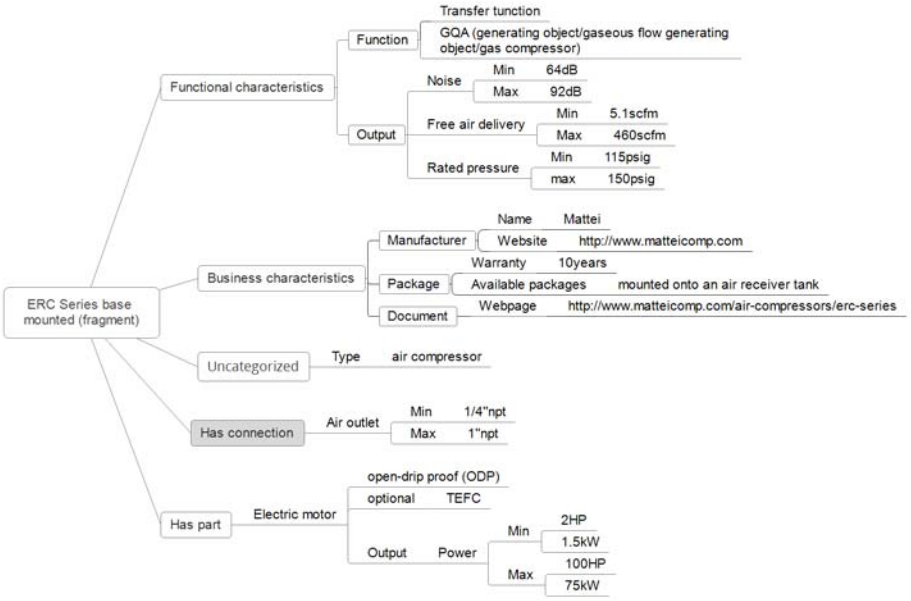


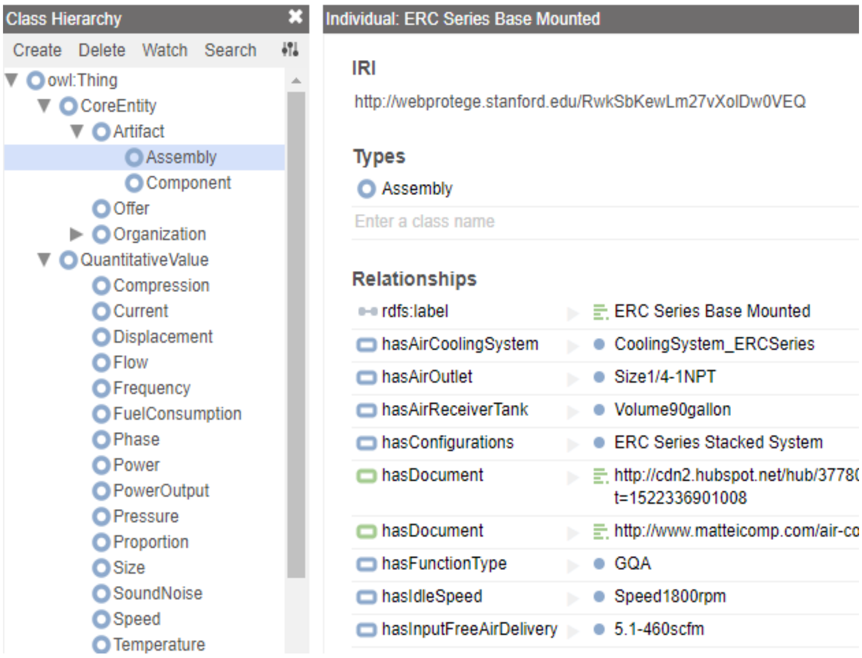
Fig. 2. Fragment of the filled mockup for the AU ERC series air compressor. The new element of the artifact, which should be added to the mockup is marked in gray.

Development stage: formalization (application of spreadsheets) was performed after mind-maps for approximately 50 AUs were created. Mind maps for different AUs contained characteristics being sub properties of the same top-level property (for example, ERC series air compressor has “Free air delivery” output characteristic; this characteristic is air-related output, likewise “Air flow” in the other item). Additionally, the same AUs’ characteristics can be named differently in different supplier catalogues, thus, the table can help to catch the synonyms. Similar uses of the table for ontology constructions were mentioned in [17, 18]. Table 1 represents a fragment of the table for base-mounted ERC series air compressor AU.

Table 1. Fragment of the properties table for the AU ERC series air compressor. Gray shows the high level properties which do not have values (only sub properties)

Property	Label	Description	AUs	
			ERC series	…
hasFunctionalCharacteristics	Functional characteristics	"Attribute or characteristic in the performance and use of a product" (Source: ISO 14050:2009(en), 8.3.3.2)	-	
hasFunctionCharacteristics	Function	Intended behavior of the artifact		
hasType	Item type	Informal type description (i.e., battery, motor)	Q	
hasFunctionType	Function type	Type of function in accordance with IEC 81346	+	
hasOutputCharacteristics	Output	All outputs that can be delivered by the artifact in case of for proper functioning		
hasOutputAirCharacteristics	Air output characteristics	The group of characteristics for air output description		
hasOutputFreeAirDelivery	Free air delivery		scfm	
hasRatedPressure	Rated pressure		psig	
…				

Development stage: implementation (application of ontology editor) followed the steps described in the previous section. Property hierarchy created at the previous step was then transferred into OWL with the help of ontology editor (WEB PROTÉGÉ). Figure 3 represents a fragment of ontological description for the ERC series air compressor. Several Artifacts (AUs) were imported as instances with rdfs:label and rdfs:description (when available) properties during instance creation. All quantitative values were imported as instances with rdfs:label property and properties for values and units of measurement. For example, ERC series air compressor has the “Free air delivery” output property with the value range of 5.1 to 460 scfm. This will be imported as an instance “5.1-460scfm” of the “Flow” class (subclass of “QuantitativeValue” with data properties hasMaxValue (value: 460), hasMinValue (value: 5.1); hasUnitCode (value: scfm).



- owl:Thing
  - CoreEntity
    - Artifact
      - Assembly
      - Component
    - Offer
    - Organization
  - QuantitativeValue
    - Compression
    - Current
    - Displacement
    - Flow
    - Frequency
    - FuelConsumption
    - Phase
    - Power
    - PowerOutput
    - Pressure
    - Proportion
    - Size
    - SoundNoise
    - Speed
    - Temperature

Individual: ERC Series Base Mounted

IRI

http://webprotege.stanford.edu/RwkSbKewLm27vXolDw0VEQ

Types

Assembly

Enter a class name

Relationships

rdfs:label	ERC Series Base Mounted
hasAirCoolingSystem	CoolingSystem_ERCSeries
hasAirOutlet	Size1/4-1NPT
hasAirReceiverTank	Volume90gallon
hasConfigurations	ERC Series Stacked System
hasDocument	http://cdn2.hubspot.net/hub/3778(t=1522336901008
hasDocument	http://www.matteicomp.com/air-co
hasFunctionType	GQA
hasIdleSpeed	Speed1800rpm
hasInputFreeAirDelivery	5.1-460scfm

Fig. 3. Fragment of the ontological description for the AU ERC series air compressor in WEB PROTÉGÉ.

4 Conclusion

The paper describes the new method of creating an enterprise knowledge graph. Although the method has been tested on car assembly units, it may be used in other spheres (for example, development of online stores, medicine, design, management, etc.), where the knowledge graph has a complex property structure associated with one class and its subclasses, and there is a little number of ontology classes. There is often a need for group development and discussion during the development of enterprise knowledge graph. Then the visualization of the graph allows faster and better understanding for the team of developers. One of the main problems is lack of information (or it is not always clear what is important), so it becomes necessary to discuss some fragments of the knowledge graph with domain experts. The visual description helps to focus the expert's attention on puzzling points and makes it easier to find mutual understanding. The proposed methodology helped to create a pilot knowledge graph that includes more than 50 assembly units and describes about 400 properties.

Acknowledgement

The work is partially supported by the Russian Foundation for Basic Research (#17-07-00228).

References

1. B. Villazon-Terrazas, N. Garcia-Santa, Y. Ren, K. Srinivas, M. Rodriguez-Muro, P. Alexopoulos, J. Z. Pan, Construction of Enterprise Knowledge Graphs (I), Exploiting Linked Data and Knowledge Graphs in Large Organisations, Cham: Springer (2017), 87-116.
2. R. Studer, R. Benjamins, D. Fensel, Knowledge Engineering: Principles and Methods, Data and Knowledge Engineering, 25 (1998) 161-197.
3. H. Österle, J. Becker, U. Frank, T. Hess, D. Karagiannis, H. Krcmar, ... , E. J. Sinz, Memorandum on design-oriented information systems research, European Journal of Information Systems, 20 (2011), 7-10.
4. G. Lyu, X. Chu, D. Xue, Product modeling from knowledge, distributed computing and lifecycle perspectives: A literature review, Computers in Industry, 84 (2017), 1-13.
5. M. C. Suárez-Figueroa, A. Gómez-Pérez, M. Fernandez-Lopez. The NeOn Methodology for Ontology Engineering, Ontology Engineering in a Networked World, Cham: Springer (2012), 9-34.
6. M. Fernández-López, A. Gómez-Pérez, and N. Juristo. METHONTOLOGY: From Ontological Art Towards Ontological Engineering, AAAI Technical Report SS-97-06 (1997), 33-40.
7. Y. Sure, S. Staab, and R. Struder.nOn-To-Knowledge Methodology, Handbook on Ontologies, Cham.: Springer (2004), 117-132.
8. A. Ohgren and K. Sandkuhl. Towards a Methodology for Ontology Development in Small and Medium-Sized, IADIS International Conference on Applied Computing (2005), 369-376.
9. M. Uschold, M. Gruninger, M. Uschold, and M. Gruninger. Ontologies: Principles, Methods and Applications, Knowl. Eng. Rev., 2 (1996), 93-136.
10. V. Presutti, E. Daga, A. Gangemi, and E. Blomqvist, eXtreme Design with Content Ontology Design Patterns, Proc. Workshop on Ontology Patterns (2009), 83–97.
11. E. Simperl, M. Luczak-Rösch, Collaborative ontology engineering: a survey, The Knowledge Engineering Review, 29 (2014), 101-131.
12. H. Paulheim. Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods, Semant. Web, 3 (2017), 489-508.
13. B. Zheng, W. Zhang, X. F. B. Feng, A survey of faceted search, Journal of Web engineering, 12 (2013), 41-64.
14. Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke. OntoEdit: Collaborative Ontology Development for the Semantic Web. Semant. Web ISWC 2002, 221-235.
15. P. Křemen, P. Mička, M. Blaško, and M. šmíd. Ontology-driven mindmapping. *Proc. 8th Int. Conf. Semant. Syst. - I-SEMANTICS '12*, September (2014), 125-134.
16. S. Jupp, M. Horridge, L. Iannone, J. Klein, S. Owen, ... & R. Stevens. Populous: a tool for building OWL ontologies from templates, BMC bioinformatics, 1 (2012), S5.
17. D. L. Rubin. Creating and curating a terminology for radiology: Ontology modeling and analysis, J. Digit. Imaging, 4 (2008), 355-362.
18. Z. Xiang, J. Zheng, Y. Lin, and Y. He. Ontorat: Automatic generation of new ontology terms, annotations, and axioms based on ontology design patterns, J. Biomed. Semantics, 1 (2015), 1-10.
19. S. J. Fenves, S. Foufou, C. Bock, R. D. Sriram, CPM2: a core model for product data, Journal of computing and information science in engineering, 8 (2008), 1-6.
20. S. Rachuri, E. Subrahmanian, A. Bouras, S. J. Fenves, S. Foufou, R. D. Sriram, Information sharing and exchange in the context of product lifecycle management: Role of standards, Computer-Aided Design, 40 (2008), 789-800.